Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik
Lehrstuhl für Medientechnik

Methods of Point Cloud Alignment with Applications to 3D Indoor Mapping and Localization

Anas Al-Nuaimi, M.Sc.

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Die Dissertation wurde am 4.10.2016 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 11.3.2017 angenommen.

بسم الله والصلاة والسلام على رسول الله
وَقُل رَّبِّ زِدْنِي عِلْمًا - طه 114

Abstract

Due to numerous technological advancements, 3D sensing is becoming more accurate and affordable. Different types of 3D sensing techniques exist including LiDAR, Time-of-Flight and Multi-View Stereo. While 3D sensors have been relatively expensive and require specialist knowledge to operate, it is the launch of the Microsoft KINECT that made accurate and cheap 3D sensing a reality. Its success was accompanied by many new developments from the computer vision research community which enabled many new applications.

3D sensors capture scans in the shape of collections of 3D points, called point clouds. Just like images of cameras, point clouds are view dependent partial scans of the environment. Unlike images, however, multiple scans can be assembled into more comprehensive 3D maps.

Assembling point clouds into more comprehensive ones requires aligning them along distinctive shapes that are common to the scans. This necessitates uncovering the relative transformation between their local coordinate systems. The research community has developed different techniques to deal with the point cloud registration problem. With the possibility to register multiple scans into more comprehensive 3D models, it is no surprise that part of the research community has focused on developing and using registration techniques to map urban indoor and outdoor spaces in 3D. Such 3D maps are essential to many emerging applications including robot navigation and environment interaction, facility management, cultural heritage archiving, virtual and augmented reality and many more.

This thesis deals with the topic of point cloud registration with special emphasis on the applications of 3D mapping and indoor localization. Four contributions are presented. Contribution 1 deals with the fundamental registration theory using the most popular registration approach, namely using feature matching. Contribution 2 is more high level from the application perspective as it deals with the non-rigid deformations of moving LiDAR scans and its impact on scan matching which is performed to assemble 3D maps from the individual scans. For that the filter presented in contribution 1 is deployed. An algorithm specifically tailored for the registration of 3D maps generated from scan matching such that more comprehensive 3D maps can be assembled from smaller ones is presented in contribution 3. In contribution 4, the comprehensive 3D maps are used as reference point clouds against which object scans obtained with the Kinect are registered for very accurate location retrieval.

Kurzfassung

Große technologische Fortschritte in der Bildgebung und Sensortechnik haben die kostengünstige Erfassung von Strukturen in 3D möglich gemacht. Es existieren verschiedene 3D Messtechniken inklusive LiDAR, Time-of-Flight und Multi-view Stereo. 3D Sensoren erfassen Aufnahmen (Engl. Scans) in Form von "Punktwolken" (Engl. Point Clouds). Ähnlich wie Kamerabilder sind standpunktabhängige Punktwolken Teilaufnahmen einer Umgebung. Entscheidend anders sind sie jedoch darin, dass sie sich nach einer geeigneten Ausrichtung zu einer globalen Aufnahme zusammenfügen lassen.

Das Zusammenfügen mehrerer Punktwolken erfordert, dass sie zueinander ausgerichtet (Engl. Alignmnet) werden. Das wiederum erfordert das Ermitteln der relativen Transformationen zwischen deren lokalen Koordinatensystemen. Die Forschungsgemeinschaft hat mehrere Techniken für das Punktwolkenausrichten entwickelt. Viele der entwickelten Techniken befassen sich vor allem mit dem Ausrichten zum Zweck der 3D Kartierung (Engl. Mapping) von Innen- und Außenbereichen. Die so gewonnenen 3D Karten haben vielerlei Nutzen. Dazu gehören Roboter Navigation und Interaktion mit der Umgebung, *Facility-Management*, Digitalisierung von kulturellem Erbe, *Virtual*- und *Augmented Reality* u.v.m.

Diese Doktorarbeit befasst sich mit dem Ausrichten von Punktwolken mit besonderem Fokus auf die Anwendung der 3D-Kartierung und Lokalisierung. Sie beinhaltet vier Kernbeiträge. Der 1. Beitrag befasst sich mit dem grundlegenden Problem des Ausrichtens aus einem Satz von Punktkorrespondenzen. Im Beitrag werden Mechanismen eingeführt, die einen von uns entwickelten Filter ermöglichen, robust gegen Außreiser zu werden und somit den in der Literatur standardmäig eingesetzten Filter in der Performanz deutlich zu übertreffen. Im 2. Beitrag wird zuerst das Phänomen des sogenannten Scan Skewing, der nicht rigiden Deformation von 3D LiDAR Scans, gründlich untersucht. Danach wird der im 1. Beitrag vorgestellte Filter eingesetzt um Skew-resistentes *Scan Matching*, also das Zusammenfügen nacheinanderfolgenden LiDAR Scans zu einer umfassenden 3D Karte, zu erreichen. Im 3. Beitrag wird einen Ansatz vorgestellt, der es erlaubt mehrere 3D Karten, die als Ausgang vom Scan Matching herausgehen, zu einer umfassenderen 3D Karte zusammenzufügen. Im letzten Beitrag wird ein System zur Ortsbestimmung durch das Ausrichten von Punktwolken kleiner 3D Scans in großen 3D Karten vorgestellt.

Acknowledgements

Having been often intrigued to read the acknowledgments of various thesis and dissertations it seemed to me this rather routine exercise of English gratitude vocabulary is perhaps too classic. Nevertheless, having reached the point of submission I feel more than obliged to acknowledge the contributions and help of those who made this moment reality.

To the following persons I want to say:
Prof. Steinbach, I feel honored for the trust you placed in me. I very much appreciate your calm approach to solving problems and your professionalism.

My dear colleagues, having known you, went for lunch with you and shared endless birthday cakes with you has been a truly rewarding experience. Some of you deserve special thanks: Tashakur Burak (Cizmeci) for jump starting my first project at LMT. Thanks Robert (Huitl), the greatest tech geek I've ever known, for sharing knowledge. Thanks Suat (Gedikli) for being the greatest C++ teacher. Brigado Wilder for being such a great person to work with.

Dear (former) students I had the honor to supervise, I learned a lot from you ... not only technically but also on the personal level. The list of students is as long as there are Mignons on my Doktorhut :) But two of you deserve special mention: Shukreya Jawad for being my first companion in the PhD trip; Grazie Martin (Piccolorovazzi) for being such a pleasant person to supervise.

My dear friends, your advice and your caring helped me to cope with great challenges and overcome the hard times. Special thanks to Majid, Raeed and Shokry for being always there in good and bad times.

Mom, dad, there is nothing like you. Your love and your hard work have helped me come all this way. Sallum and Barhi, I can't thank you enough for your support and encouragement. Words can't describe how much I owe you. But you know ...

Dimah and Mira, you two are my soul, my love ♡ and the light that guided me to the end of the tunnel.

Contents

List of Abbreviations

Abbreviation	Description	Definition
3D	3-dimensional	page 1
LiDAR	Light Detection and Ranging	page 1
SfM	Structure from Motion	page 1
ToF	Time-of-Flight	page 1
DoF	degree-of-freedom	page 2
GA-LMS	Geometric Algebra Least Mean Squares	page 2
NN	nearest-neighbor	page 2
PCD	point cloud	page 6
MSE	mean-squared-error	page 7
SVD	Singular-Value Decomposition	page 7
LS	Least Squares	page 7
GA	Geometric Algebra	page 8
RANSAC	Random Sample Consensus	page 8
PCL	Point Cloud Library	page 9
LRF	local reference frame	page 9
SHOT	Signature of Histograms of OrienTations	page 9
USC	Unique Shape Context	page 9
LCP	Largest Common Point set	page 10
4PCS	4-Point Congruent Sets	page 10
S4PCS	Super4PCS	page 10

Abbreviation	Description	Definition
GMM	Gaussian Mixture Model	page 11
EM	Expectation Maximization	page 11
ICP	Iterative Closest Point	page 13
pt2pt	point-to-point	page 13
pt2pl	point-to-plane	page 13
LMS	least-mean-squares	page 15
AF	Adaptive Filter	page 15
IMU	inertial measurement unit	page 33
PCA	Principal Component Analysis	page 51
PMF	probability mass function	
KinFu	KinectFusion	page 85
ISS	Intrinsic Shape Signature	page 87
SDF	Signed Distance Function	page 88
CLAMS	Calibrating, Localizing, and Mapping, Simultaneously	page 93
MLS	Moving Least Squares	page 94
SOR	sparse outlier removal	page 94
FoV	field of view	page 92
TCR	true correspondence rate	page 96
MCS	Multiple Classifier System	page 100

Chapter 1

Introduction

The digital revolution is rapidly expanding its reach to every aspect of human life. The increased miniaturization of chips and devices and the increased connectedness of digitized entities (*Internet of the Things*) is promising major changes in industry, households and public and private spaces. The digital revolution also manifests itself in the increased automation and the increasing deployment of robotic systems.

With increasing digitization of everyday objects, it becomes important to capture the 3-dimensional (3D) space in which they interact. Also, intelligent robotic systems need accurate spatial knowledge and 3D maps of their surroundings to effectively interact with it and navigate successfully inside it.

The 3D mapping of environments is particularly important for indoor spaces. After all, humans spend most of their time indoors. A similar argument applies to industrial setups where robots are expected to become essential companions of workers. Hence, it is not surprising that the UN Committee of Experts on Global Geospatial Information Management rated indoor positioning and mapping as one of the major emerging trends [8].

3D data of indoor environments can be captured using different 3D sensors and techniques such as Light Detection and Ranging (LiDAR), structured-light sensing (for e.g. Microsoft Kinect [9]), depth from stereo, structure from motion (SfM) [10] and Time-of-Flight (ToF).

3D sensors deliver a partial scan of the 3D environment which depends on many factors including placement, range and field of view. Hence, it is usually important to make multiple scans and register (or *align*) them in 3D to obtain comprehensive 3D maps. Moreover, the registration is important to improve the map accuracy.

This dissertation deals with the topic of alignment (alternatively termed *registration*) of 3D point clouds in 3D space with special emphasis on applications relating to 3D indoor mapping and location retrieval. In Section 1.1 the contributions of this dissertation towards the topic are highlighted. A few important notes on convention and other issues are explained in Section 1.2.

1.1 Contributions of the Thesis

Point cloud registration has multiple applications. Amongst the most popular are 3D mapping, 3D reconstruction and object pose estimation.

There exist a number of methods of point cloud registration. They include: exhaustive voting-based alignment [11], game theory-based alignment [12], probabilistic alignment (example: Coherent Point Drift (CPD) [13]) and feature-based alignment [14].

Arguably, the most prominent approach to point cloud alignment is the feature-based approach [14]. Like feature-based image matching, this approach is characterized by its simplicity and the availability of multiple types of features suitable for different applications.

Feature-based registration of a pair of point clouds requires first setting up point correspondences through feature matching and finally using the correspondences to estimate the 6 degree of freedom (DoF) alignment. The standard solver of the alignment from a set of correspondences can be traced back to the works of B. K. P. Horn [15] and Arun et al. [16]. The solver is widely used, however, it suffers from the lack of resilience towards outliers. In the first contribution we show how the Geometric Algebra Least Mean Squares (GA-LMS) filter, which we had devised to substitute the standard solver [1], can be made outlier resilient by devising mechanisms that exploit its adaptive and sequential nature.

We present a systematic study of a characteristic problem of modern LiDARs called “scan skewing” and deploy a variant of the GA-LMS, again exploiting its adaptive nature, to reduce the impact on the performance of scan matching of 3D LiDAR data in the second contribution.

Feature-based alignment can suffer from efficiency degradation when dealing with large-scale point clouds. The problem is highly related to the fact that computing features requires detecting keypoints followed by descriptor computation. Both require repeated nearest-neighbor (NN) searches in 3D space in the unorganized point clouds[*]. The third contribution approaches this problem by presenting a novel algorithm tailored for the alignment of large-scale indoor models that are the outcome of scan matching. It is shown to be initial alignment insensitive, efficient and supports very low overlap.

In a final contribution a feature-based registration with a geometry-aware Random Sample and Consensus (RANSAC) is used to perform very accurate indoor localization.

From a system perspective, the four contributions cover a chain of successively linked applications whereby the filter presented in contribution 1 is used to reduce the impact of the effect studied in contribution 2. The maps produced in contribution 2 form the input of the algorithm of contribution 3 whose output forms the reference point clouds used in contribution 4. This is further illustrated in Figure 1.1.

[*]Unorganized point clouds is a term adopted from the Point Cloud Library (PCL) [17] and refers to point clouds in which the point index is not related to the spatial relationship. This is a fundamental difference to images where the pixel adjacency defines the neighborhood of any point.

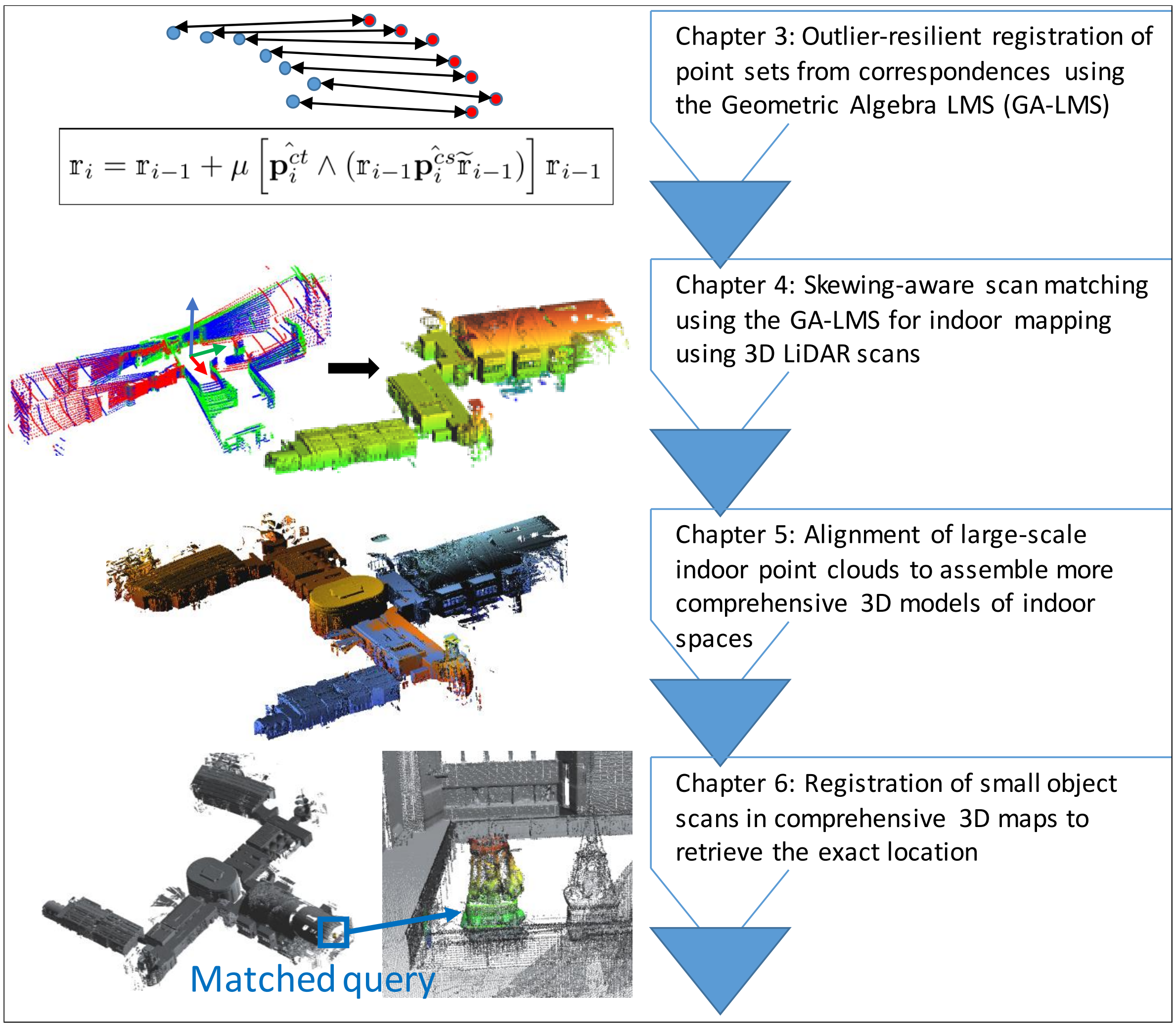

$\mathbb{r}_i = \mathbb{r}_{i-1} + \mu \left[\hat{\mathbf{p}}_i^{ct} \wedge (\mathbb{r}_{i-1} \hat{\mathbf{p}}_i^{cs} \widetilde{\mathbb{r}}_{i-1}) \right] \mathbb{r}_{i-1}$
Chapter 3: Outlier-resilient registration of point sets from correspondences using the Geometric Algebra LMS (GA-LMS)
Chapter 4: Skewing-aware scan matching using the GA-LMS for indoor mapping using 3D LiDAR scans
Chapter 5: Alignment of large-scale indoor point clouds to assemble more comprehensive 3D models of indoor spaces
Chapter 6: Registration of small object scans in comprehensive 3D maps to retrieve the exact location
Matched query

Figure 1.1: The thesis contributions from a system perspective. Contribution 1 is related to the fundamental theory of point set registration on the most basic level. Contribution 2 builds upon contribution 1 and focuses on the application of scan matching: assembling elementary 3D scans into a 3D map. Contribution 3 is concerned with the registration of 3D maps resulting from scan matching into more comprehensive 3D maps. Contribution 4 uses the comprehensive 3D maps for accurate indoor localization.

Following, is a summary of the contributions:

- **Outlier-Resilient Registration from Correspondences using the GA-LMS**: A method for the feature-based alignment of a point cloud pair connected by correspondence pairs is presented. Its novelty is attributed to the fact that it uses an unconventional mathematical formulation, namely Geometric Algebra (GA), to derive an adaptive filter that estimates transformations by processing correspondences individually allowing to control the influence of each correspondence pair on the final outcome. The main contribution, which is presented in Chapter 3, is a description and validation of techniques that exploit the adaptive and sequential nature of the filter to endow it with outlier-resilience.

- **Skew-Resistant Scan Matching for 3D Mapping using the GA-LMS**:
  In Chapter 4 an analysis of the scan skewing problem which is a phenomenon related LiDAR-based 3D scanning is first presented. To the best of our knowledge, this is the first systematic study on the problem. Furthermore, the adaptive and sequential nature of the GA-LMS is exploited to reduce the impact of skewing to produce better 3D maps.

- **Efficient Decoupled Roto-Translation Alignment of Large-Scale Point Clouds**:
  Separate 3D maps are typically acquired for different parts of a large building owing to technical and non-technical limitations. They need to be aligned such that a comprehensive reconstruction of the indoor environment results, which is important for providing a single reference data set for location retrieval purposes. In order to eliminate the need for time-consuming manual alignment, a method for the automatic alignment of large-scale indoor point clouds is presented. The method presented in Chapter 5 is different from the state-of-the-art in that it can handle very small overlaps between the point cloud pair while being very efficient and insensitive to initial alignment.

- **Indoor Localization using Point Cloud Registration**:
  In Chapter 6 a retrieval system is presented that uses only point cloud registration to perform the retrieval of the location. The approach uses a feature-based retrieval system with a robustified geometry-aware estimator and which is shown to achieve localization accuracies in the cm-domain.

Parts of this thesis have appeared in the following papers: [1], [2], [4]–[6].

1.2 Convention

In the following parts of the thesis, a unified mathematical notation is adopted which is outlined in Table 1.1. The $\cdot^T$ operator transposes a matrix or vector. The $|\cdot|$ operator around a set refers to set size. The $\times$ implies vector cross-product. $p(\cdot)$ expresses the probability function. $\langle \cdot \rangle$ represents the Geometric Algebra 0-grade operator.

Table 1.1: Mathematical convention used in the thesis.

Entities	Font type	Example
Scalars	Small letters	a
Vectors	Small bold letters	$\mathbf{a}$
Matrices/Tensors	Capital bold letters	$\mathbf{A}$
GA Multivectors	small "mathbbm" letters	$\mathbb{a}$
Sets & Spaces	Capital "mathcal" letters	$\mathcal{A}$

Chapter 2

Background

2.1 Mapping

The area of 3D reconstruction and mapping has progressed substantially in recent years. Many approaches to the reconstruction of 3D models that rely on accurate range scanners and even cheap hand-held scanners have been presented. Mapping systems include scanners mounted on wheel-based mapping platforms [18], backpack-mounted scanners [19] and hand-held scanners [20] as shown in Figure 2.1. Also, numerous camera-based reconstruction algorithms have been proposed such as [21] which performs urban reconstruction from a video.

Especially LiDAR-based mappers are interesting as they acquire relatively dense point clouds with high accuracy. As opposed to multi-view reconstruction algorithms ([21]), they generate individual 3D scans which have to be registered to one another such that a comprehensive 3D map is generated as illustrated in Figure 2.2. After registration such systems can produce point clouds with impressive level of detail as shown in Figure 2.3. Different techniques used in point cloud registration are surveyed in Section 2.2.

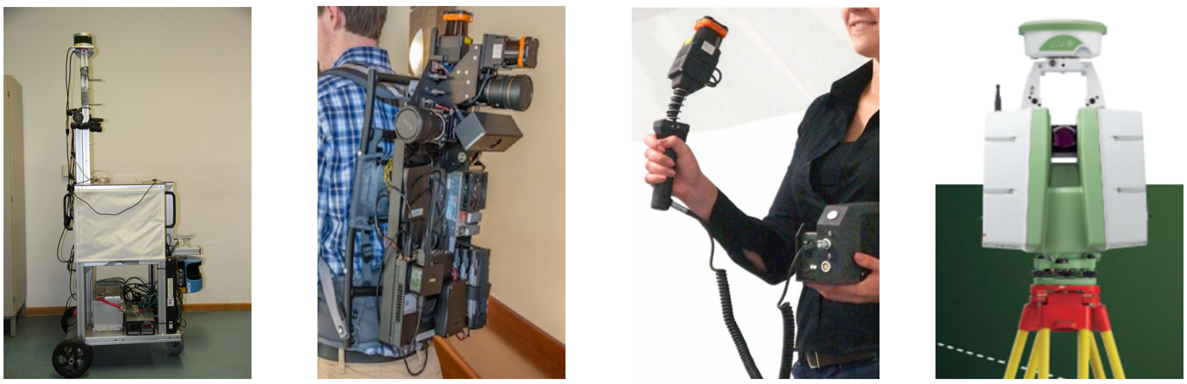


Figure 2.1: Four different types of LiDAR-based mappers. From left to right: LMT trolley-based mapper [18]; backpack-based mapper (image source: [22] – reproduced with permission); ZEB1 hand-held mapper (image source: [23] – reproduced under CC3.0); Leica static ScanStation C10 (image source: [24] – reproduced with permission).

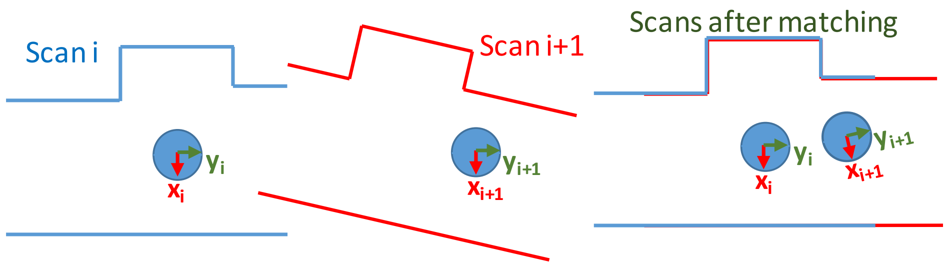


Figure 2.2: 3D mapping with LiDARs. Registering multiple scans (shown here for 2D scans from a top view) using *scan matching* allows retrieving the sensor motion and assembling a map as the one shown in Figure 2.3.

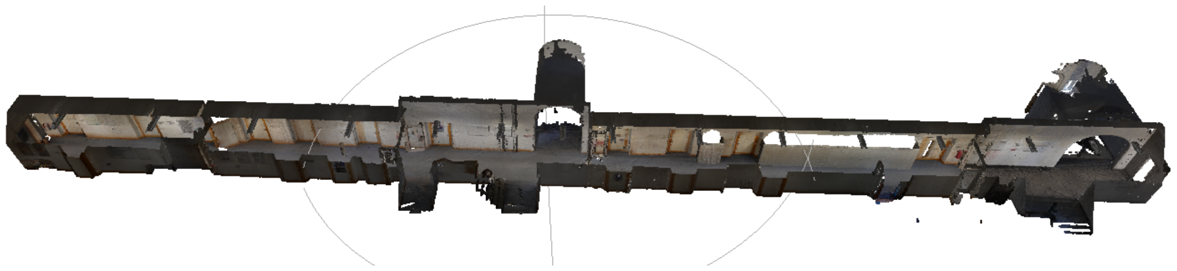


Figure 2.3: A point cloud of a TUM corridor generated using the M3 Mapping Trolley of the NavVis GmbH [25], part of the ceiling and walls have been removed manually.

2.2 Point Cloud Registration

3D-point cloud alignment is the problem of recovering the 6DoF transformation that aligns two overlapping point clouds (PCDs) in a common frame. The alignment problem involves two point sets: the *target* (also called *scene*) which is static and acts as a reference and the *source* (also called *model*) for which the transformation that best aligns it to the target is sought. In the following, the target point cloud (set) shall be mathematically referred to as set $\mathcal{T} = \{\mathbf{p}_1^t, \mathbf{p}_2^t, \ldots, \mathbf{p}_{|\mathcal{T}|}^t\}$ and the source point cloud as set $\mathcal{S} = \{\mathbf{p}_1^s, \mathbf{p}_2^s, \ldots, \mathbf{p}_{|\mathcal{S}|}^s\}$. A point is expressed using the column vector $\mathbf{p} = [\mathbf{p}(x)\ \mathbf{p}(y)\ \mathbf{p}(z)]^T$.

From a theoretical point of view we can distinguish between two cases: 1) A set of correspondences identifying corresponding source-target pairs are known which is addressed in Section 2.2.1; 2) No correspondences are known. In this case, there are two different alignment problems:

- Coarse alignment which does not put restrictions on the initial alignment but is usually less accurate and serves as an initialization for the fine alignment (see Section 2.2.2);
- Fine alignment which is initialization-sensitive (see Section 2.2.3).

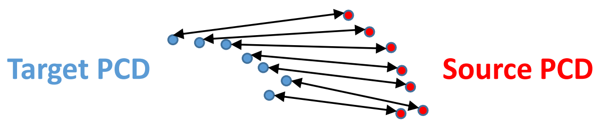


Figure 2.4: Correspondences establish an associative connection between source-target point pairs.

2.2.1 Registration from Correspondences - The Absolute Orientation Problem

Let's assume the existence of a set of correspondences identifying which source point corresponds to which target point as depicted in Figure 2.4. The correspondences can be mathematically expressed as: $\mathcal{C} = \{\{\mathbf{p}_1^{cs}, \mathbf{p}_1^{ct}\}, \{\mathbf{p}_2^{cs}, \mathbf{p}_2^{ct}\}, \ldots, \{\mathbf{p}_N^{cs}, \mathbf{p}_N^{ct}\} | \mathbf{p}_n^{cs} \in \mathcal{S}, \mathbf{p}_n^{ct} \in \mathcal{T}\}$[*]. The registration problem can now be mathematically formulated as the minimization of the mean-squared-error (MSE) cost function:

$$C_{pt2pt}(\mathbf{R}, \mathbf{t}) = \frac{1}{N} \sum_{n=1}^{N} \left\| \mathbf{p}_n^{ct} - \mathbf{R}\mathbf{p}_n^{cs} - \mathbf{t} \right\|_2^2 \tag{2.1}$$

with $\mathbf{R}$ and $\mathbf{t} = [t_x \; t_y \; t_z]^T$ expressing the 3D orthogonal rotation matrix and 3D translation, respectively. At least *three* correspondences are necessary to solve this problem. Typically more correspondences are available making this is a Least Squares (LS) problem.

It can be easily mathematically shown that the minimization of (2.1) breaks down into two steps [26]: compute the translation $\mathbf{t}_{min}$ after first recovering the optimal rotation $\mathbf{R}_{min}$ using the mean-subtracted point clouds by minimizing

$$\frac{1}{N} \sum_{n=1}^{N} \left\| \widehat{\mathbf{p}_n^{ct}} - \mathbf{R}\widehat{\mathbf{p}_n^{cs}} \right\|_2^2, \textbf{ with } \widehat{\mathbf{p}_n^{cs}} = \mathbf{p}_n^{cs} - \overline{\mathbf{p}^{cs}} \;\&\; \widehat{\mathbf{p}_n^{ct}} = \mathbf{p}_n^{ct} - \overline{\mathbf{p}^{ct}} \tag{2.2}$$

where $\overline{\mathbf{p}^{cs}}$ and $\overline{\mathbf{p}^{ct}}$ are the centroids of the source and target point clouds, respectively. The formulation in (2.2) is called the *orthogonal procrustes problem*. Shönemann [27] showed that the rotation minimizing the cost function is obtained using the Singular-Value Decomposition (SVD) of the point scatter matrix $\widehat{\mathbf{P^{cs}}}\widehat{\mathbf{P^{ct}}}^T$ of the matrices $\widehat{\mathbf{P^{cs}}} = [\widehat{\mathbf{p}_1^{cs}} \; \widehat{\mathbf{p}_2^{cs}} \; \ldots \; \widehat{\mathbf{p}_N^{cs}}]$ and $\widehat{\mathbf{P^{ct}}} = [\widehat{\mathbf{p}_1^{ct}} \; \widehat{\mathbf{p}_2^{ct}} \; \ldots \; \widehat{\mathbf{p}_N^{ct}}]$ (Notice, at least three correspondences are necessary to compute the scatter matrix). Hence,

$$\mathbf{R}_{min} = \mathbf{V}\mathbf{U}^T \textbf{ for } \widehat{\mathbf{P^{cs}}}\widehat{\mathbf{P^{ct}}}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.3}$$

The translation can now be computed using the following formula:

$$\mathbf{t}_{min} = \overline{\mathbf{p}^{ct}} - \mathbf{R}_{min}\overline{\mathbf{p}^{cs}}. \tag{2.4}$$

The full derivation can be found in [26]. The original derivation is usually attributed to Arun et al. [16].

[*]Notice that, without loss of generality, the source and target cloud of the correspondences do not necessarily preserve the original order of the points inside their respective point clouds.

Notice, other formulations of (2.2) that use different rotation operators other than orthogonal rotation matrices exist. They include the quaternion-based formulation [15] or the one that uses dual quaternions [28]. All three result in closed-form solutions to the minimization problem and have been found to perform similarly accurate in non-degenerate cases [26].

The accuracy of these LS estimators is degraded by outliers as they treat all points equally. The possibility to incorporate weights explicitly for the individual points exists. However, it is not possible to ad-hoc decide to exclude certain correspondences because they lead to a degradation of the solution or because the filter has converged obliterating the necessity to spend extra computations. Finally, the derivations using these formulas are not straightforward. This applies especially to the quaternion-based estimator. In [1] we derive an alternative filter that uses *rotors* of Geometric Algebra (GA) instead of quaternions. Although rotors are isomorphic to quaternions, the derivations are much more tractable owing to the tools offered by *Geometric Calculus*. In Chapter 3 the adaptive nature of the filter is exploited to endow it with resilience towards outliers.

When not opting for an LS solution, one may compute the transformation using RANSAC [29]. In each RANSAC iteration three correspondences are randomly sampled and used to compute the transformation as explained above. The quality of each transformation hypothesis is given by the number of source points which after transformation using the computed hypothesis are not further away from their corresponding target point than a certain maximal distance. In Chapter 6 a simple trick is used to turn RANSAC into a “geometry-aware” RANSAC that can effectively deal with a high false correspondence rate to result in very accurate registration results.

2.2.2 Coarse Alignment

When registering two point clouds for which no correspondences are known and which have not been roughly pre-aligned, a coarse alignment is necessary. Different categories for coarse alignment exist. They are explained in the following while highlighting their characteristics.

2.2.2.1 Voting-based Alignment

The first category involves voting-based algorithms. These include [11], [30]–[32]. They avoid establishing hard source-target correspondences. Instead, they are based on accumulating votes inside the parameter space cast by different combinations of source-target points.

Typically, three correspondences are needed per vote. Accordingly, the algorithmic complexity is $O(|\mathcal{S}|^3 \cdot |\mathcal{T}|^3)$. This means that slight increases in the number of points lead to large increases in computational burden. Hence, these methods suffer from inefficiency due to their brute force nature. Moreover, with growing dimensions the votes become increasingly sparse requiring many more votes to sufficiently populate the space (curse of dimensionality).

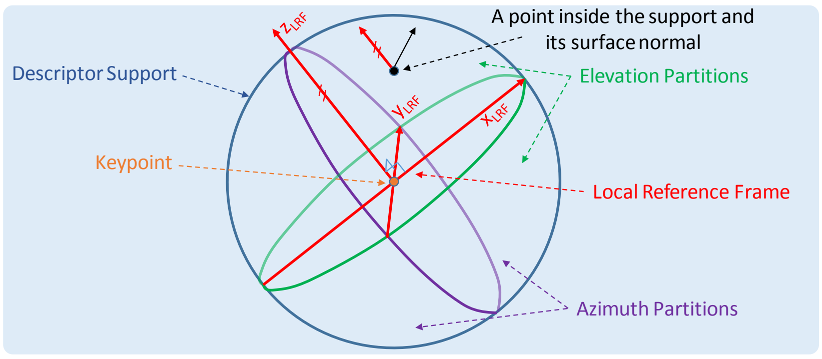


Figure 2.5: SHOT descriptor computation. A local reference frame (LRF) is computed for the keypoint. The dot product of the normal of any point inside the support with the LRF z-axis is computed. The descriptor is made up of 11-bin histograms of the dot product values. One histogram is computed using all points inside one spatial partition of the spherical support around the keypoint. The support is partitioned as follows: 2 elevation partitions (hemispheres), 8 azimuth partitions (only 2 shown here) and 2 radial partitions (not shown in the figure). Hence, the descriptor vector has 352 elements.

In Chapter 5 an algorithm for the alignment of large-scale point clouds is proposed which exploits characteristics of the data to break down the registration problem into multiple problems of reduced complexity. One of them, namely the translation alignment in the ground-plane, is solved using a voting-based approach albeit one that has a substantially reduced complexity. Furthermore, the voting is cast into a probabilistic model (see Section 2.2.2.4) to support low overlap cases.

2.2.2.2 Feature-based Alignment

This category involves methods that require computing point correspondences and using them to estimate the transformation that best maps the source points to their respective target points as explained in Section 2.2.1. The feature matching of local features (such as [33]–[35]), which is essential for establishing correspondences, requires articulated shapes to generate distinctive descriptors. Also, keypoint detection and descriptor computation are usually computationally demanding since each requires repetitive NN searches.

Aldoma at al. [36] compare different local shape descriptors implemented in the Point Cloud Library (PCL) [37] in terms of 3D object retrieval performance. The Signature of Histograms of OrienTations (SHOT) [38], [39] as well as the Unique Shape Context (USC) [40] are shown to be among the best performing in terms of retrieval performance. SHOT is used in this work since its descriptors are significantly smaller than USC descriptors.

The SHOT descriptor comprises concatenated histograms of surface normal dot products w.r.t. to the local reference frame (LRF). The SHOT descriptor computation is explained in Figure 2.5. In Chapter 6 a system for indoor location retrieval based on point cloud registration using feature matching is presented.

2.2.2.3 4-Point Congruent Sets (4PCS)

The relatively new approach of *Four-Point Congruent Sets* (4PCS) [41] also establishes explicit correspondences like the feature-based approaches. However, it is different in that each correspondence is made of a pair of nearly co-planar 4-point sets. Hence, one correspondence is enough to compute a 6DoF transformation. Each 4-point set (basis) correspondence generates a transformation hypothesis which is verified in a RANSAC-like approach.

The 4-point correspondences are not found by feature matching but rather through geometric consistency. The congruent set search for one target 4-point basis with the proximity constraint δ is illustrated in Figure 2.3. Once a congruent set is found, the 6DoF alignment is computed. Since 4PCS poses alignment as a Largest Common Point set (LCP) problem, the hypothesis quality is given by the number of source points which after transformation have a NN that is nearer than δ.

The co-planarity of the bases means that especially objects with large planar surfaces are suitable for alignment with this approach which is a characteristic standard feature-based approaches do not possess as they require articulated shapes for distinctive descriptor generation. The congruent set search is computationally intensive (finding co-planar bases, finding pairs subtending a certain length, predicting intersection points, checking intersection point incidence, see Figure 2.6) making the algorithm relatively slow on large scale point clouds.

Mellado et al. accelerated the 4PCS algorithms creating the Super4PCS (S4PCS) [42]. It is used in our comparisons given its efficiency. Since 4PCS (and S4PCS) pose alignment as an LCP problem, solutions that maximize overlap are favored which is occasionally harmful when registering indoor datasets with large corridors as seen in the results in Section 5.6.5.

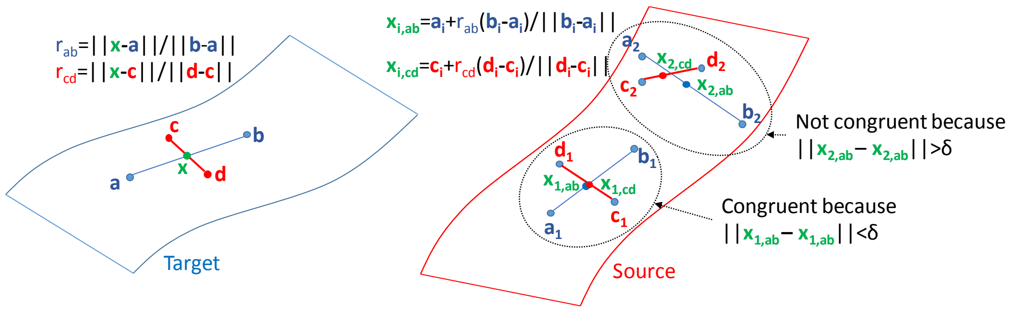


Figure 2.6: 4PCS congruent basis search. The 4PCS uses 4 point bases made of (nearly) co-planar points. The ratios r_{ab} and r_{cd} for the two intersecting lines connecting a randomly chosen co-planar 4-point set $\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}\}$ are computed in the target (left). Then, all point pairs of a length similar to either $||\mathbf{b} - \mathbf{a}||_2$ or $||\mathbf{d} - \mathbf{c}||_2$ are pre-selected in the source ($O(|\mathcal{S}|^2)$) delivering candidate congruent sets. In this example, the following two congruent sets were found: $\{\mathbf{a_1}, \mathbf{b_1}, \mathbf{c_1}, \mathbf{d_1}\}$ and $\{\mathbf{a_2}, \mathbf{b_2}, \mathbf{c_2}, \mathbf{d_2}\}$ (right). For each candidate basis i, the *predicted* intersection points $\mathbf{x_{i,ab}}$ and $\mathbf{x_{i,cd}}$ are computed using the basis points and the ratios r_{ab} and r_{cd}. A set is only accepted as congruent to the target basis if it satisfies $||\mathbf{x_{i,ab}} - \mathbf{x_{i,cd}}||_2 < \delta$. Notice, in practice it is not possible to uniquely assign a strict order to a pair of points (i.e. identifying $\mathbf{a}$ and $\mathbf{b}$ or $\mathbf{c}$ and $\mathbf{d}$) hence all four combinations are tested when predicting the intersection points leading to further computational burden.

2.2.2.4 Probabilistic Alignment

Probabilistic approaches fit probability distributions on the underlying point data and cast these models, after parameterization on the transformation parameters, into minimization problems.

One notable example for probabilistic alignment is that of Jian and Vemuri [43] (henceforth termed *GMMREG*). It fits a Gaussian Mixture Model (GMM) on the source and target clouds and considers the alignment problem as a histogram distance minimization problem when parameterizing one of the GMM models on the transformation parameters. They use the L2 histogram distance and show that the L2 distance between two GMMs has a closed form solution. In essence, the alignment problem amounts to minimizing:

$$\int (gmm(T(\mathcal{S}, \mathbf{T})) - gmm(\mathcal{T}))^2. \tag{2.5}$$

where $gmm(\mathcal{P})$ is a GMM modeled from the point set $\mathcal{P}$ with $|\mathcal{P}|$ mixtures centered on the points. $T(\mathcal{S}, \mathbf{T})$ is a function that transforms the point set $\mathcal{S}$ using transformation $\mathbf{T}$.

The L2 distance between two GMMs (2.5) has a closed form solution which amounts to evaluating $|\mathcal{T}|$ 3D Gaussians at $|\mathcal{S}|$ points. Hence it has the complexity $O(|\mathcal{S}| \cdot |\mathcal{T}|)$. When combined with the iterative solver needed to minimize this L2 distance (2.5) this translates into very slow convergence. Amongst all alignment algorithms tested this is by far the slowest. For our typically large-scale point clouds the computation times are simply prohibitive. Because of the iterative nature of the solver the algorithm is also sensitive towards the initial alignment as shown in Section 5.6.4

Another notable representative of probabilistic alignment is *Coherent Point Drift* [13]. The contribution presented in Chapter 5 is partly inspired by CPD. Like GMMREG, CPD also uses GMMs, however, the (parametrized) GMM model is only fitted on the target cloud where each target point represents a component of the mixture. Each source point is considered a sample out of the GMM model. The transformation parameters are estimated by solving a maximum likelihood (ML) problem using the Expectation Maximization (EM) algorithm. More specifically, in CPD, the probability of sampling source point $\mathbf{p}_i^s \in \mathcal{S}$ out of the target point $\mathbf{p}_j^t \in \mathcal{T}$ after transformation by $\mathbf{T}$ is given as:

$$p(\mathbf{p}_i^s | T(\mathbf{p}_j^t, \mathbf{T})) = \mathcal{N}(\mathbf{p}_i^s; T(\mathbf{p}_j^t, \mathbf{T}), \sigma \mathbf{I}) \tag{2.6}$$

where $\mathbf{I}$ is the identity matrix and where $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ evaluates the Gaussian function centered on $\boldsymbol{\mu}$ and with covariance $\boldsymbol{\Sigma}$ at $\mathbf{x}$. In using GMM modeling it follows that the probability of sampling $\mathbf{p}_i^s$ can be expressed as:

$$p(\mathbf{p}_i^s; \mathbf{T}) = \frac{w}{|\mathcal{T}|} + (1 - w) \sum_{\mathbf{p}_j^t \in \mathcal{T}} p(\mathbf{p}_j^t)\, p(\mathbf{p}_i^s | T(\mathbf{p}_j^t, \mathbf{T})). \tag{2.7}$$

The uniform probability ($w/|\mathcal{T}|$), with $0 \leq w < 1$, is added to the mixture model in CPD to account for parts not in the overlap (we use a similar approach to account for outliers in Section 5.4.7). Here, we have neglected the scaling factor which CPD can also account for. The rotation $\mathbf{R}$ and translation $\mathbf{t}$ are estimated by solving a maximum likelihood (ML) problem which is built on top of the assumption of i.i.d. source point samples, i.e. maximizing

$$p(\mathcal{S};\mathbf{T}) = \bigcap_{\mathbf{p}_i^s \in \mathcal{S}} p(\mathbf{p}_i^s;\mathbf{T}). \tag{2.8}$$

Notice, each time the likelihood is evaluated for a parameter hypothesis $|\mathcal{S}| \cdot |\mathcal{T}|$ 3D Gaussians have to be evaluated. These evaluations have to be repeated as the ML solution is based on the Expectation Maximization (EM) algorithm which is an iterative algorithm. In summary the solution involves iterating over the following main steps:

1. Compute a pair-wise probability $p_{i,j}$ between each source-target point pair $\{\mathbf{p}_i^s, \mathbf{p}_j^t\} \in \mathcal{S} \times \mathcal{T}$. Each probability involves evaluating a Gaussian function;

2. Update model parameters;

until a convergence criterion is reached. The first step represents the bottleneck of computations for CPD. Its complexity is $O(|\mathcal{S}| \cdot |\mathcal{T}|)$ and is repeated in every iteration which for our large-scale point clouds with millions of points translates into very slow convergence. Even when replacing the full Gaussian evaluations by the Fast Gaussian Transform [44] with complexity $O(|\mathcal{S}| + |\mathcal{T}|)$ CPD remains relatively slow as shown in Figure 5.17. Moreover, the used EM algorithm does not guarantee hitting the global optimum [45] which also means that different initial alignments may lead to different results, as further demonstrated in Section 5.6.4. Finally, the point-wise relationship given by the Gaussian function does not involve the local geometry which provides vital clues about how associable two a source-target pair is.

An interesting observation is that CPD, contrary to feature-based approaches and to 4PCS, does not establish hard correspondences but instead establishes a "soft" relationship between point pairs. This is somehow similar to voting-based approaches. However, instead of using every possible pair for casting votes in the transformation space, CPD describes the pair-wise point relationships by probabilities.

CPD and GMMREG have a high degree of flexibility in dealing with different transformations including reflections, affine transformation and scaling. These properties, however, are hardly of interest to our problem since we deal with rigid-body transformation estimation in 3D space. Their iterative nature means that they are sensitive to initial alignment as will be shown in Section 5.6.4. Significant computational complexity is associated with the employed solvers leading to very high computation times especially for large datasets. Nevertheless, they (and especially CPD) remain inspirational as shown in Chapter 5.

2.2.3 Fine Alignment with the Iterative Closest Point

Fine alignment, as the name implies, is concerned with refining the alignment result such that errors inside the overlap region are minimized. It is typically run after a coarse alignment step (see Section 2.2.2).

The *Iterative Closest Point* (ICP) algorithm is still considered the standard solution for the fine alignment problem. Zhang [46] and Besl and McKay [47] are credited with developing the ICP algorithm.

ICP, as its name implies, is an iterative algorithm. It alternates between two processes: correspondence search followed by transformation update. The correspondence computation builds upon the fundamental assumption that corresponding points are in close spatial proximity which is the reason for the requirement of a good initial alignment. Thus it is common to identify correspondences using NN search. The iterations are necessary as the NN search becomes a more reliable correspondence finder as the transformation alignment becomes more accurate which in turn leads to better correspondence estimation.

Since good correspondences are essential to obtain a good alignment a number of correspondence refinements, also called correspondence *rejectors*, can be applied. The maximum distance rejector filters out correspondences whose spatial distance is beyond a certain threshold. This is also important to hinder parts that are not overlapping between source and target from contributing to the alignment. It is also possible to reject correspondences whose normals subtend an angle greater than a certain threshold angle. This implies the rotation alignment is largely correct which again explains the requirement for good initial alignment. A number of other rejectors are explained in [14].

At any iteration, once the correspondences have been computed, the transformation can be computed by solving the absolute orientation problem (see Section 2.2.1) which uses the so-called “point-to-point” (pt2pt) metric. However, it has been shown that a reformulation of this problem using the so-called “point-to-plane” (pt2pl) metric leads to faster convergence and more accurate ICP [48].

The pt2pl alternative of the cost function of (2.1) is

$$C_{pt2pl}(\mathbf{R}, \mathbf{t}) = \frac{1}{N} \sum_{n=1}^{N} \left\| (\mathbf{p}_n^{ct} - \mathbf{R}\mathbf{p}_n^{cs} - \mathbf{t})^T \mathbf{n}_n^{ct} \right\|_2^2 \tag{2.9}$$

where $\mathbf{n}_n^{ct}$ is the normalized surface normal vector of the respective target point. This formulation aims at minimizing the perpendicular distances of the source points to the tangents of the respective target points as illustrated in Figure 2.7.

Minimizing the non-linear cost function (2.9) can be done using the Levenberg-Marquardt algorithm. It is common, however, to use the small angle assumption which allows to simplify the derivation. In this case, it can be shown that the transformation can be determined by

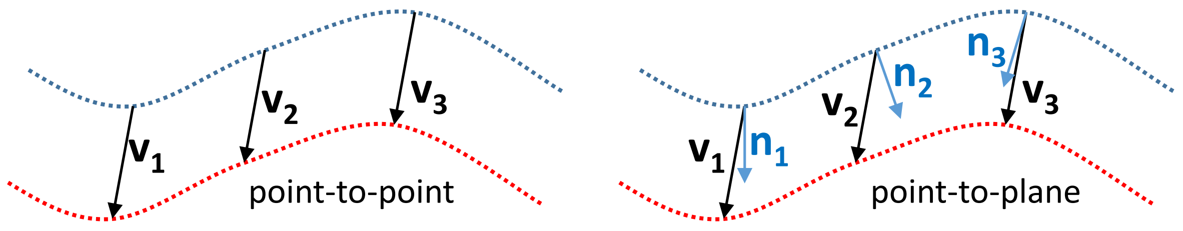


Figure 2.7: The point-to-point metric for surface registration aims at finding the transformation that minimizes the sum distance between correspondences of two surfaces (in this case $\|\mathbf{v_1}\|^2+\|\mathbf{v_2}\|^2+\|\mathbf{v_3}\|^2$, where $\mathbf{v_i}$ are the translation vectors connecting source-target correspondences). The point-to-plane metric aims at minimizing the perpendicular distances of the source points to the target surface tangent of their respective corresponding target points (in this case $(\mathbf{v_1}^T\mathbf{n_1})^2 + (\mathbf{v_2}^T\mathbf{n_2})^2 + (\mathbf{v_3}^T\mathbf{n_3}^2)$, where $\mathbf{n_i}$ are the normalized surface normal vectors). Notice, for flat regions of the surface, the perpendicular distance remains unchanged for a "sliding" movement along the flat region allowing faster convergence by implicitly down weighting the flat region's contribution and up weighting more salient regions with articulated surface.

solving the following equation:

$$\mathbf{G}\mathbf{G}^T[\alpha_x\ \alpha_y\ \alpha_z\ t_x\ t_y\ t_z]^T = \mathbf{G}\mathbf{h}. \tag{2.10}$$

where α_x, α_y and α_z are the rotation angles around the x, y and z axis, respectively, and with

$$\mathbf{G} = \begin{bmatrix} \mathbf{p}_1^{cs} \times \mathbf{n}_1^{ct} & \mathbf{p}_2^{cs} \times \mathbf{n}_2^{ct} & \cdots & \mathbf{p}_N^{cs} \times \mathbf{n}_N^{ct} \\ \mathbf{n}_1^{ct} & \mathbf{n}_2^{ct} & \cdots & \mathbf{n}_N^{ct} \end{bmatrix} \ \textbf{and}\ \mathbf{h} = \begin{bmatrix} (\mathbf{p}_1^{ct} - \mathbf{p}_1^{cs})^T\mathbf{n}_1^{ct} \\ (\mathbf{p}_2^{ct} - \mathbf{p}_2^{cs})^T\mathbf{n}_2^{ct} \\ \cdots \\ (\mathbf{p}_N^{ct} - \mathbf{p}_N^{cs})^T\mathbf{n}_N^{ct} \end{bmatrix}. \tag{2.11}$$

See [49, Appendix A] for the detailed derivation. In (2.10) a difference to the standard pt2pt solution can be observed: The rotation and translation are estimated jointly when using the pt2pl metric. This is fundamentally different to the solution of the pt2pt metric where the solution involves first finding the optimal rotation and then the optimal translation using centroid subtraction (see Section 2.2.1). The results in Chapter 4 show that this leads to a measurable advantage of pt2pl as compared to pt2pt.

It remains to be said that there is a plethora of ICP variants that have been developed over the years. The interested reader may refer to Pomerlau et al.'s extensive survey on the topic [49].

Chapter 3

Point Cloud Registration with the GA-LMS

As explained in Section 2.2.2.2, one principal approach to point cloud alignment requires determining point correspondences between the points of the PCDs to be aligned. Once correspondences have been computed, an LS estimator can used to estimate the MSE transformation (see Section 2.2.1).

In practical systems, the correspondences are usually automatically determined using for instance local-shape descriptor matching (see Section 2.2.2.2). Many correspondences can be false in this case and a standard LS estimator may not lead to satisfactory results.

Recently, we devised a completely new approach for estimating the relative rotation between PCDs from point correspondences [1]. It is based on a least-mean-squares (LMS) Adaptive Filter (AF) that uses Geometric Algebra (GA), a mathematical tool that allows for representing geometric transformations in a very general fashion [50]–[52]. The Geometric Algebra LMS (GA-LMS) is derived using adaptive filtering theory [53], [54] applied on *Geometric Calculus* [50] after re-posing the 3D rotation registration problem as a *rotor* estimation problem [55]–[57]. The GA-based formulation allows for a tractable derivation while emphasizing the geometrical intuition behind rotors, a notion that is less intuitive when using matrix algebra to represent geometric transformations. In our first published work on the topic [1] we used the GA-LMS for pure rotation registration. Our experiments demonstrated that the GA-LMS alignment accuracy to be comparable to that of standard SVD-based solver and thus is a valid replacement thereof.

In this chapter the GA-LMS's adaptive nature is exploited to endow it with error resilience as compared to the standard SVD-based LS estimator.

3.1 Geometric Algebra Background

Geometric (Clifford) Algebra was developed as a mathematical language to unify all the different algebraic systems trying to express geometric relations. Some GA concepts are introduced here. For an in-depth discussion, see [50]–[52].

Given a vector space $\mathcal{A}$, the Geometric Algebra $\mathcal{G}(\mathcal{A})$ is comprised by the so-called *multivectors*, generated by multiplying the vectors in $\mathcal{A}$ via the *geometric product* [52], together with the multiplicative identity (scalar) 1. Thus, whenever dealing with multivectors, the use of the geometric product is implicit. For the case when dealing with "simple" vectors, the geometric multiplication of $\mathbf{a}, \mathbf{b} \in \mathcal{A}$ is defined in terms of the *inner* $(\mathbf{a} \cdot \mathbf{b})$ and *outer* $(\mathbf{a} \wedge \mathbf{b})$ products as $\mathbf{ab} = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \wedge \mathbf{b}$ [50]. GA also defines the *reverse* operator which is analogous to conjugation in complex numbers (in fact, complex numbers form a sub-Algebra of GA). The reverse of multivector $\mathbb{q}$ is expressed as $\widetilde{\mathbb{q}}$. The norm operator $|\cdot|$ in GA generalizes the norm operator $\|.\|$ of linear algebra [52].

It is important to understand that GA is a powerful mathematical language. A GA entity is in general a linear combination of a scalar, the basis vectors and their combinations (the so-called *n-vectors*). Hence, for a 2D space, a multivector can be expressed as:

$$\mathbb{q} = e_0 + e_1 \mathbb{e}_1 + e_2 \mathbb{e}_2 + e_{1,2} \mathbb{e}_{1,2} \tag{3.1}$$

where $\mathbb{e}_1$ and $\mathbb{e}_2$ are the two basis vectors and $\mathbb{e}_{1,2}$ is the only 2-vector (also called *bi-vector*). Anecdotally, it could be said GA allows adding apples and oranges! This property allows expressing different geometrical operations such as rotations, translations and the entities to be transformed using simple compact expressions. This is a fundamental property that we exploited in our work on rotation alignment using the GA-LMS [1].

The reader may have noticed the use of the "mathbbm" font together with small letters to express multivectors (example: $\mathbb{q}$) in line with the introduced notation in Table 1.1. Rotors, which are multivectors, are hence expressed using the same notation. Although vectors in the context of GA are also multivectors, the standard notation (bold small letters) is still used, as they have some special properties such as the one introduced above relating to geometric multiplication.

3.2 The GA-LMS Filter for 6DOF Registration

The GA-LMS acts on a set of point correspondences. In keeping with the notation of Chapter 2 the correspondence set is expressed as $\mathcal{C} = \{\{\mathbf{p}_1^{cs}, \mathbf{p}_1^{ct}\}, \{\mathbf{p}_2^{cs}, \mathbf{p}_2^{ct}\}, \ldots, \{\mathbf{p}_N^{cs}, \mathbf{p}_N^{ct}\} | \mathbf{p}_n^{cs} \in \mathcal{S}, \mathbf{p}_n^{ct} \in \mathcal{T}\}$ where $\mathcal{S}$ and $\mathcal{T}$ are the source and target points, respectively.

To register the source points to the target points in C, the cost function (2.1) has to be minimized. The GA-LMS as we present it in [1] only estimates the rotation alignment.

To derive the GA-LMS, we re-posed the orthogonal Procrustes cost function (2.2) using Geometric Algebra by substituting the matrix $\mathbf{R}$ with the rotor $\mathbb{r}$ and its reversed version $\widetilde{\mathbb{r}}$ in order to represent rotations as follows [1]:

$$C_{GA}(\mathbb{r}) = \frac{1}{N}\sum_{n=1}^{N} |\widehat{\mathbf{p}_n^{ct}} - \mathbb{r}\,\widehat{\mathbf{p}_n^{cs}}\,\widetilde{\mathbb{r}}|^2. \tag{3.2}$$

Remember, $\widehat{\mathbf{p}_n^{cs}}$ and $\widehat{\mathbf{p}_n^{ct}}$ are the corresponding source and target points after subtracting the respective PCD centroids. We applied Geometric Calculus [50], [58] to devise an LMS AF to estimate the best rotor that minimizes (3.2) [1]. Notice the similarity of (3.2) to the equivalent formulation using quaternions. In fact, rotors are isomorphic to quaternions. Nevertheless, you may be aware of the complexity of performing derivations, which are necessary to derive the LMS (or generally gradient decent) filter, when using the standard quaternion formulation.

In [1] we derived the GA-LMS by calculating the gradient $\nabla C_{GA}(\mathbb{r})$ (via Geometric Calculus) and substituting it into the *steepest-descent rule* [53], which recursively provides a new estimate for the rotor $\mathbb{r}$ at each iteration,

$$\mathbb{r}_i = \mathbb{r}_{i-1} - \mu\widetilde{\nabla} C_{GA}(\mathbb{r}_{i-1}), \tag{3.3}$$

where

$$\nabla C_{GA}(\mathbb{r}_{i-1}) = 4\widetilde{\mathbb{r}}_{i-1}\sum_{n=1}^{N}\widehat{\mathbf{p}_n^{ct}} \wedge (\mathbb{r}_{i-1}\widehat{\mathbf{p}_n^{cs}}\widetilde{\mathbb{r}}_{i-1}), \tag{3.4}$$

and $\widetilde{\nabla} C_{GA}(\mathbb{r}_{i-1})$ is the reversed gradient. Note that $\nabla C_{GA}(\mathbb{r}_{i-1})$ is a function of all the correspondence pairs $\{\widehat{\mathbf{p}_n^{ct}}, \widehat{\mathbf{p}_n^{cs}}\}$, $n = 1, \cdots, N$, i.e., it needs all the available correspondences, at each i, to update the estimate for $\mathbb{r}$.

In [1] we approximate the gradient by its *current value* to generate the LMS variant in line with AF theory. Hence, we set $\nabla C_{GA}(\mathbb{r}_{i-1}) \approx 4\widetilde{\mathbb{r}}_{i-1}\left[\widehat{\mathbf{p}_i^{ct}} \wedge (\mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\right]$. This approximation reduces the computational complexity of the algorithm since now it needs only the correspondence pair at the current iteration $\{\widehat{\mathbf{p}_i^{cs}}, \widehat{\mathbf{p}_i^{ct}}\}$. Adopting that in (3.3) results in the GA-LMS update rule,

$$\boxed{\mathbb{r}_i = \mathbb{r}_{i-1} + \mu\left[\widehat{\mathbf{p}_i^{ct}} \wedge (\mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\right]\mathbb{r}_{i-1}}. \tag{3.5}$$

As we show in [1], Equation (3.5) is able to converge using only part of the available correspondences, delivering an estimate as good as the SVD-based method used for comparison.

As Mr Wilder Lopes of the University of Sao Paulo mainly contributed the mathematical derivations, the details are left out here. The interested reader may refer to [1] for the full derivation.

In adopting a similar solution approach to that of the absolute orientation problem (Section 2.2.1), the GA-LMS is simply extended here to compute the translation alignment through the subtraction of the centroids as follows:

$$\mathbf{t}_{min} = \overline{\mathbf{p}^{ct}} - \mathbb{r}_{min}\overline{\mathbf{p}^{cs}}\widetilde{\mathbb{r}}_{min} \tag{3.6}$$

Notice, this is the GA version of (2.4). The results in this chapter will demonstrate that this simple extension suffices to extend the GA-LMS to full 6DOF registration capability.

3.3 Implementation and Robustification against Outliers

3.3.1 Filter Implementation

The GA-LMS was implemented using C++ as part of a shape matching system presented in Chapter 6. It computes point correspondences using shape feature matching. The GA-LMS update rule (3.5) was implemented using the *Geometric Algebra ALgorithms Expression Templates* (Gaalet) [59]. The filter update code is as simple as shown in Listing 3.1.

Listing 3.1: Implementation of (3.5) using Gaalet. ($\wedge$) performs outer product and ($\sim$) computes the reverse of a multivector.

```
//Update rotor
rotAfter = eval(rotBefore+mu*(y^(rotBefore*x*(~rotBefore)))*rotBefore );
//Normalising
rotAfter = normalize_mv(rotAfter);
```

3.3.2 Filter Robustification

Registration systems that establish correspondence pairs are not perfect. Many correspondences can be invalid (false correspondences, also called *outliers* in this chapter). Our experiments have shown that the GA-LMS in its basic form is not outlier-resilient [1]. Its accuracy matches that of standard SVD-based methods and degrades proportionally to an increasing outlier rate. To robustify the filter and achieve outlier resilience a number of improvements that increase the accuracy significantly are presented here.

Adopting established techniques of the adaptive filtering community, as well as simple geometric heuristics to improve the GA-LMS performance, the standard SVD-based registration is outperformed, while not significantly compromising speed. Most proposed techniques exploit the fact that the GA-LMS is an adaptive filter that processes individual sample points (correspondences in our case) separately and iteratively. This enables choosing *how much* (Section 3.3.2.3) and *how often* (Section 3.3.2.2), if at all (Section 3.3.2.1), each correspondence is allowed to contribute towards finding the optimal rotor.

To explain the proposed robustification techniques a source-target pair of the Stanford Bunny dataset [60] is used. Specifically, the Bunny315 and Bunny000 point clouds are used as source and target PCDs, respectively. These two PCDs are shown among others in Figure 3.1.

The figure of merit used to evaluate the AF performance is the mean squared error (MSE). For that we define the *true MSE* which uses *only inliers* to calculate the estimation error. In non-experimental scenarios, where the true correspondents are not known prior-hand, the AF can not compute the *true MSE*. What the AF evaluates is what we call the *filter MSE* which is defined as the MSE that is calculated using all correspondences (inliers and outliers).

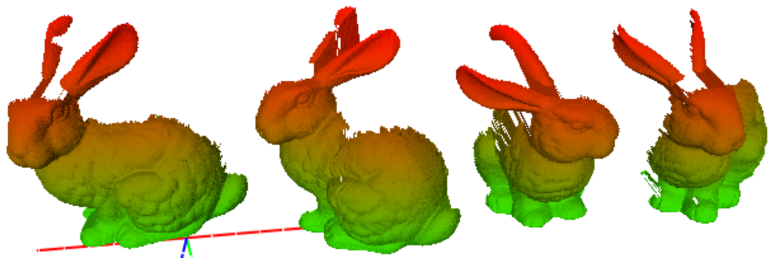


Figure 3.1: The Bunny dataset is composed of 10 scans of a rabbit model obtained from different view points. Four specific scans (Bunny000, Bunny045, Bunny270 and Bunny315) are picked, where the numeric postfix identifies the angle in the ground plane from which the bunny was scanned. Image source: [5] – ©2016 IEEE.

.

3.3.2.1 Iteration Skipping

Having observed that outliers typically lead to an instantaneous increase in the MSE, the idea is to skip the filter update that would lead to such an increase, thus discarding the otherwise harmful contribution of the outlier correspondence to the filter update. The skipping is done based on the filter MSE value.

As depicted in Figure 3.2, the true MSE is always lower when skipping and mostly monotonically decreasing. Moreover, the frequent degradation in the true MSE when not skipping (solid red line in Figure 3.2) no longer appears after skipping (solid green line in Figure 3.2), suggesting that the contribution of the outliers is indeed being discarded through skipping.

As already mentioned in Section 2.2.1, it is not possible to ad-hoc detect and discard potential outliers in standard LS registration algorithms. Iteration skipping is only possible with the GA-LMS due to its iterative nature.

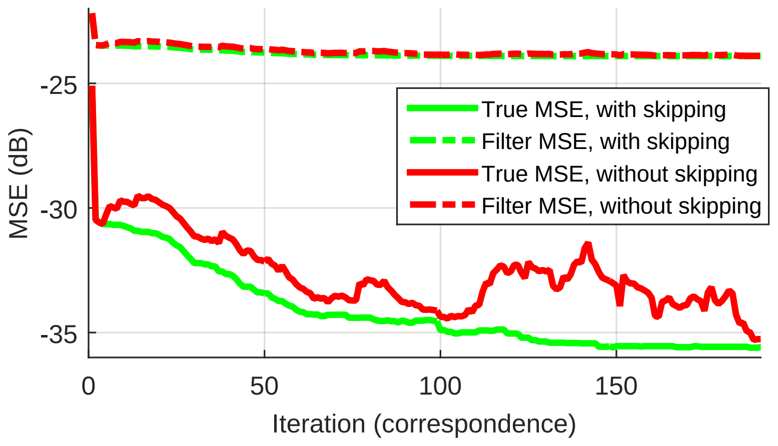


Figure 3.2: Skipping iterations with increasing filter MSE (MSE computed using all correspondences including inliers and outliers) vs. no skipping. When skipping, the true MSE (solid lines) is clearly lower than when not skipping proving that skipping reduces the contributions from outliers leading to better pose estimation. The strong reduction in MSE after the first iteration is due to the subtraction of the centroids (3.6). Image source: [5] – ©2016 IEEE.

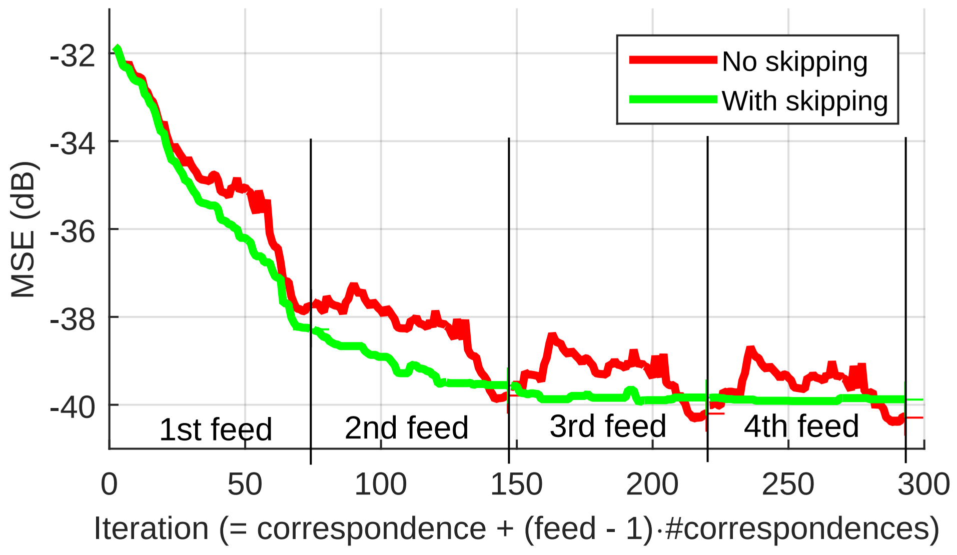


Figure 3.3: Refeeding the same set of correspondences multiple times can extract further information from already processed samples. In this case, the correspondence pairs are fed 4 times into the GA-LMS. The MSE at the end of each cycle is better than that of the previous cycle. The gain in the 3rd and 4th cycles is marginal since the orientation has been already recovered accurately in the 2nd stage. Here we limited the number of correspondences to 33% of those in Figure 3.2 as this technique is particularly useful when a limited number of correspondences are available or the step-size is relatively small. Image source: [5] – ©2016 IEEE.

3.3.2.2 Sample Refeeding

From information theory, we know that data samples (correspondences in our case) that have already been processed can provide new information if processed again at a later stage [53]. The intuitive explanation is that, especially when the step size is relatively small, a sample's residual error, given the current filter state, can lead to another filter update in the opposite direction of the reversed gradient, according to (3.5). Hence, it is common in adaptive filtering literature to refeed (reuse) the same set of data samples to obtain a better final outcome [61]–[64].

This technique can lead to a noticeable improvement in the MSE in the case where the number of correspondences is not enough for the filter to converge, or the step size is not large enough to achieve convergence. The latter might be necessary to enforce filter stability (well-behaved updates).

Figure 3.3 shows the MSE progression for the Bunny pair. However, in this experiment only 33% of the correspondences used in the experiment of Section 3.3.2.1 are used. Clearly, the multiple feeding strategy benefits the alignment. It remains to be said that multiple feeding can not lead to a degradation of the alignment and hence can always be used. The only downside to it is an increase in computational cost. Given that the filter is light-weight (as seen in the results in Section 3.4.2) this extra cost can be easily accommodated.

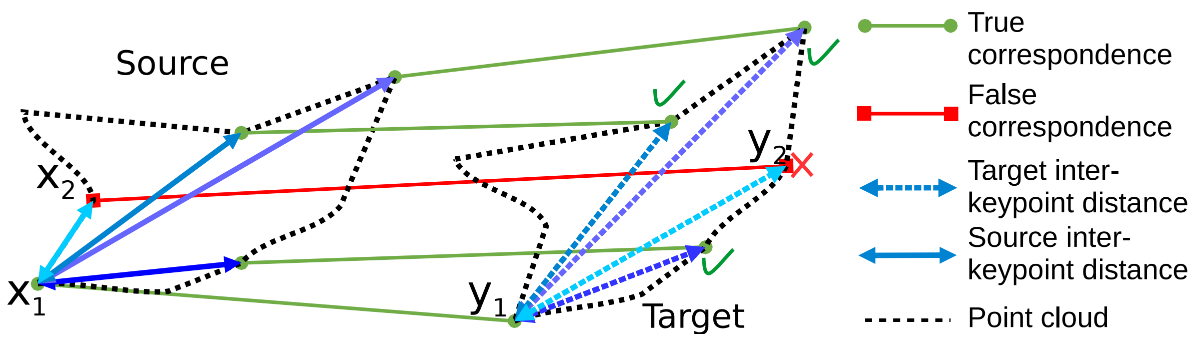


Figure 3.4: The geometric weight of correspondence $\{\mathbf{y}_i, \mathbf{x}_i\}$ is computed by first computing the Euclidean distance of $\mathbf{x}_i$ to all other keypoints in the source cloud (solid blue-shaded lines), $d^s_{i,j}\ \forall j \in \{1,..,N\}\backslash i$, and computing the corresponding distances $d^t_{i,j}\ \forall j \in \{1,..,N\}\backslash i$ on the target side (dashed blue-shaded lines). The number of keypoints which satisfy $|d^s_{i,j} - d^t_{i,j}| < \epsilon$ (green ticks) is a vote for correspondence i being an inlier. In this example, $\{\mathbf{y}_1, \mathbf{x}_1\}$ gets $v_i = 3$ votes. The weight is a function of the votes: $\alpha_i = v_i / \max_j v_j$. Hence, $0 < \alpha_i \leq 1$. Image source: [5] – ©2016 IEEE.

3.3.2.3 Geometric Correspondence Weighting

Here, the adaptive nature of the GA-LMS is exploited as it acts on each correspondence individually. The idea behind this technique is to compute an individual *weight* for each correspondence, which is used to control the correspondence's contribution based on its probability being an outlier.

To compute the outlier probability a property of the Special Euclidean Group $\mathbb{SE}3$ is exploited, namely, that inter-point distances are preserved after transformation. Hence, for each source key point, the number of remaining key points whose distances to it are preserved also on the target side is determined. This number is proportional to the probability of it being an outlier. This is perhaps better understood in the illustration of Figure 3.4. A similar trick is used in the geometry-aware RANSAC proposed in Section 6.4.4.

Once the weight of each correspondence $\alpha_i (0 < \alpha_i \leq 1)$ is computed, (3.5) is slightly modified to incorporate α_i in the step-size computation such that correspondences with a small probability of being an inlier contribute less towards the registration outcome. Hence, by multiplying the geometric weight into the step size the modified filter update becomes:

$$\mathbb{r}_i = \mathbb{r}_{i-1} + \alpha_i \mu \left[\widehat{\mathbf{p}_i^{ct}} \wedge \left(\mathbb{r}_{i-1} \widehat{\mathbf{p}_i^{cs}} \widetilde{\mathbb{r}}_{i-1} \right) \right] \mathbb{r}_{i-1}. \tag{3.7}$$

The results in Table 3.1 suggest that geometric correspondence weighting should not be enabled as the only robustification mechanism. The fact that each correspondence is weighted by a factor smaller than one means that the filter step sizes are generally decreased. Hence, its alignment improvement effect is only observed when combined with refeeding (see Section 3.3.2.5).

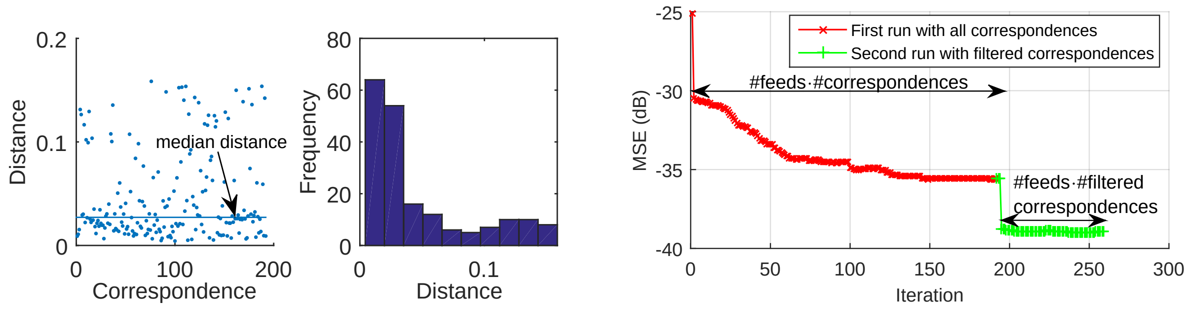


Figure 3.5: Statistical correspondence filter. (Left) Distances between target and aligned source keypoints after the first GA-LMS run and before correspondence filtering. The distances mainly cluster around a certain distance as the filter typically will have recovered the rotation while a residual translation error exists. This is confirmed by the histogram of correspondence distance (middle). (Right) The GA-LMS is first run using all correspondences (red). For the obtained rotor, the residual error is assumed to be mainly a translatory component causing a constant shift in inlier source-target keypoint pairs. The inliers are determined statistically (correspondences with a distance close to the median distance). The filtered correspondences comprising mainly inliers are fed again into the GA-LMS for a 2nd run. The second run converges to a better MSE (green). Image source: [5] – ©2016 IEEE.

3.3.2.4 Statistical Correspondence Filtering

As explained in Section 3.3.2.2, after a sufficient number of correspondence refeeds, no further improvement to the MSE can be obtained. Given the used correspondence set, a lower limit on the MSE will be reached and further improvements might be achievable if a new less outlier-corrupted correspondence set is available. Throughout our experiments, we have observed that once the filter has converged, the recovered orientation is usually accurate up to a few degrees, even in cases with a high outlier ratio. This suggests that the remaining error in the pose is composed mainly of a 3D translation error. Hence, when transforming the source keypoints using the obtained transformation, and subsequently computing the Euclidean distance to their respective target keypoints, the set of inlier correspondences should be identifiable. It is specifically the set of correspondence pairs whose Euclidean distances are roughly similar. The other subset should be composed of outliers.

Indeed, Figure 3.5 clearly shows a concentration of correspondences whose spatial distances, after applying the obtained GA-LMS transformation, are similar. We compute the median distance d and retain only those correspondences whose spatial distances are within $d \pm 0.25\sigma_d$, where σ_d is the standard deviation in the spatial distances. The GA-LMS is then fed with those filtered correspondences, and is initialized with the most recently obtained rotor. Figure 3.5 shows that a significant MSE improvement happens as soon as the second GA-LMS run commences, as the computed centroids coincide more accurately.

Table 3.1: Performance of combinations of the filter robustification techniques explained in Section 3.3.2. Comparing B and C to A it can be seen that skipping and statistical correspondence filtering lead to improvements individually. When combined, they lead to even better alignment (D). Comparing E to A shows that geometric correspondence weighting alone deteriorates performance. However, this is expected since weighting each step by $\alpha_i < 1$ means the step size is reduced and the filter might not converge. Indeed, comparing E to F shows that geometric correspondence weighting is only meaningful once combined with refeeding as it compensates for the reduced step size while benefiting from the selective weighting. Columns G, H and J show that the combination of these mechanisms with increasing refeeding values reduces the angular alignment error. Column I, again shows that geometric correspondence weighting improves results when combined with the other mechanisms. Note, the angle error is computed as explained in Section 3.4.1.

Test	A	B	C	D	E	F	G	H	I	J
Number of feeds Section 3.3.2.2	1	1	1	1	1	2	1	2	2	4
Skipping Section 3.3.2.1	✗	✓	✗	✓	✗	✗	✓	✓	✓	✓
Statistical correspondence filtering Section 3.3.2.4	✗	✗	✓	✓	✗	✗	✓	✓	✓	✓
Geometric correspondence weighting Section 3.3.2.3	✗	✗	✗	✗	✓	✓	✓	✓	✗	✓
Angle error (°) – According to Section 3.4.1	16.7	14.0	10.8	8.4	24.0	14.9	15.1	6.2	7.0	5.4

3.3.2.5 Combining Robustification Mechanisms

The results of the collective contributions of the robustification techniques on the bunny set (introduced in Section 3.3.2) are shown in Table 3.1. It can be seen that while using some of them individually might lead to a deterioration of the alignment, collectively they reinforce one another as each one of them possesses different improvement properties leading to substantially improved alignment results. The experiment in Section 3.4.3 also demonstrates how the different mechanisms collectively lead to a substantial improvement in the alignment accuracy of a different dataset with completely different characteristics.

3.4 Comparative Evaluation

In this section the proposed GA-LMS is compared to the standard SVD-based alignment solver. For the experiments two different variants of the GA-LMS are used: GA-LMS+ is the standard GA-LMS with 4× refeed, statistical correspondence filtering and skipping; GA-LMS++ is GA-LMS+ with additional geometric correspondence weighting. To compute correspondences the feature matching system presented in Chapter 6 is used. However, here we use Harris3D key points instead of Intrinsic Shape Signature (ISS) keypoints.

3.4.1 Computing Alignment Error

The finally computed transformation is checked for correctness. To that end, first the error transformation is computed as follows:

$$\mathbf{T}_e = \begin{pmatrix} \mathbf{R}_e & \mathbf{t}_e \\ 0\ 0\ 0 & 1 \end{pmatrix} = \mathbf{T}^{-1}\hat{\mathbf{T}}. \tag{3.8}$$

where $\mathbf{T}$ and $\hat{\mathbf{T}}$ are the 3D homogeneous transforms of the ground truth and the estimate, respectively. Hence, ideally $\mathbf{T}_e$ should be identity. Any departure from that indicates a residual alignment error. Using the log map, the error rotation matrix $\mathbf{R}_e$ is broken down into a rotation axis and an error rotation angle α_e.

The translation error t_e (not to be confused with $\mathbf{t}_e$) is computed by determining the source's centroid, $\overline{\mathbf{p}^{cs}}$, translation when applying the ground truth transform compared to when applying the estimated transform. So, it is computed as follows:

$$t_e = \left\| T(\overline{\mathbf{p}^{cs}}, \mathbf{T}) - T(\overline{\mathbf{p}^{cs}}, \hat{\mathbf{T}}) \right\|_2 \tag{3.9}$$

where again $T(\mathbf{x}, \mathbf{T})$ is a function that transforms point $\mathbf{x}$ using transform $\mathbf{T}$.

3.4.2 Bunny Dataset

For our experiments we use the Stanford Bunny dataset [60]. The four scans we use (Bunny000, Bunny045, Bunny270 and Bunny315) are shown in Figure 3.1.

After correspondence computation each of the two previously introduced alignment estimators (SVD and GA-LMS) are run to retrieve the alignment.

The results in Figure 3.6 clearly show that the GA-LMS+ and GA-LMS++ are more accurate on average, in terms of both orientation estimation and translation estimation, than the SVD solver. GA-LMS++, which employs the geometric correspondence weighting, is better than GA-LMS+, as expected. However, the gain is small. This is mainly due to the dispersion of the outliers throughout the bunny shape (Section 3.4.3 shows a case where the gain is substantial). The GA-LMS+ is slower than the SVD-based method provided by PCL [65]. Nonetheless, it is typically done in a few milliseconds which makes it suitable for real-time applications. Moreover, we have determined that $> 95\%$ of the time is spent on the MSE computation. This can be spared if no skipping is performed.

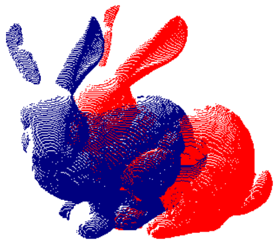
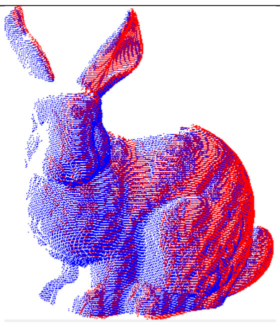
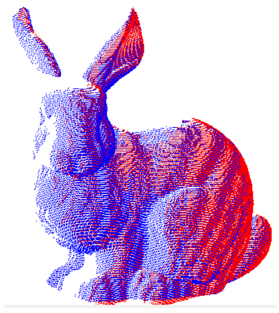
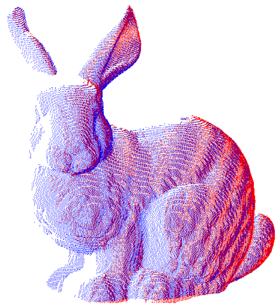
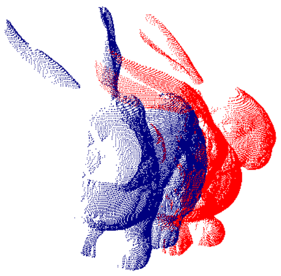
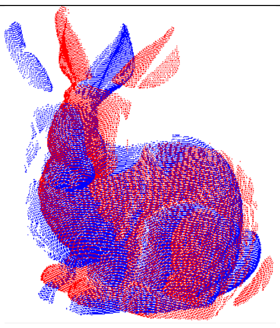
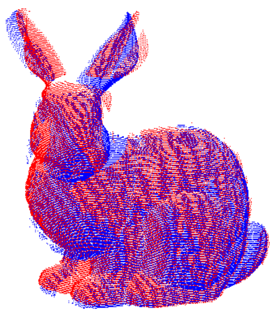
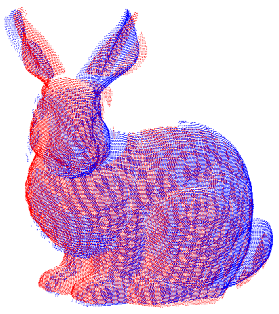
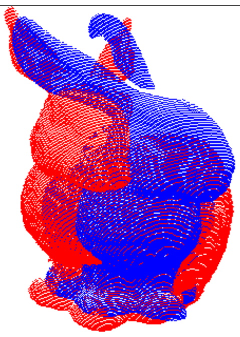
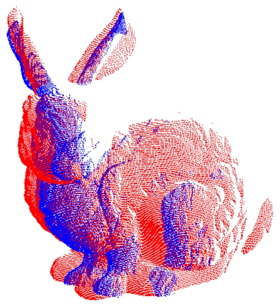
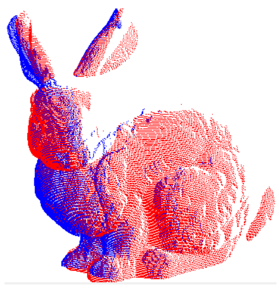
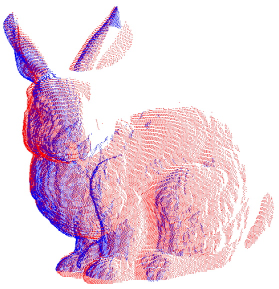
Set	SVD (LS)	GA-LMS+	GA-LMS++
 1	 $2.4°$ $1.8\,\mathrm{mm}$ $0.04\,\mathrm{ms}$	 $0.8°$ $0.3\,\mathrm{mm}$ 23ms	 $0.8°$ $0.3\,\mathrm{mm}$ 23ms
 2	 $9.9°$ $10.7\,\mathrm{mm}$ $0.025\,\mathrm{ms}$	 $6.2°$ $4.0\,\mathrm{mm}$ 8.0ms	 $5.5°$ $3.3\,\mathrm{mm}$ 8.6ms
 3	 $12.9°$ $12.0\,\mathrm{mm}$ $0.02\,\mathrm{ms}$	 $9.5°$ $2.6\,\mathrm{mm}$ 6.4ms	 $6.4°$ $1.7\,\mathrm{mm}$ 7.0ms

Figure 3.6: Experiments using the Stanford Bunny dataset. Three pairs used from the bunny dataset shown in Figure 3.1 (1: bunny045 vs bunny000; 2: bunny315 vs bunny000; 3: bunny315 vs bunny270). The alignment angle error and translation error (see Section 3.4.1) and the required time are shown for each case. These values represent the average values over 10 alignment attempts with different correspondence sets. The true correspondence rate varies from 20% to 80%. One representative visualization is shown for each case. Clearly, both GA-LMS variants are better than the SVD-based approach, in terms of angle and translation accuracy while remaining suitable for real-time applications. Image source: [5] – ©2016 IEEE.

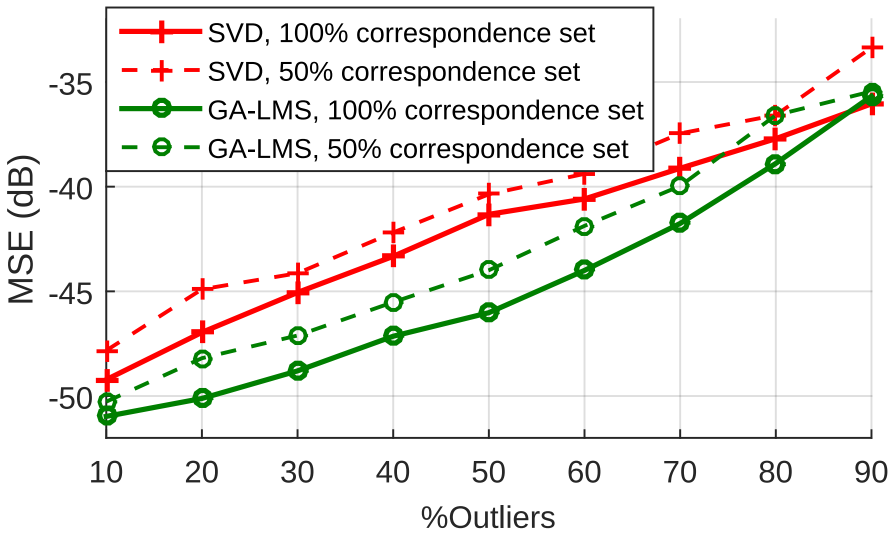
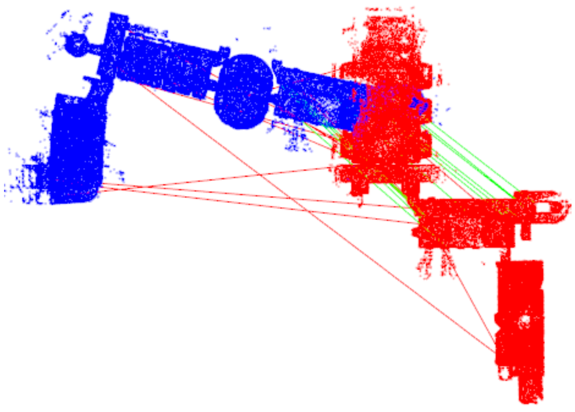


Figure 3.7: Estimation resilience against outliers. A set of randomly chosen outliers is added to confuse the registration algorithms. The curves are averaged over 50 realizations to account for different constellations of outliers. Conducting the experiment using a subset involving only half of the correspondences shows a similar superiority of the GA-LMS w.r.t. SVD. The GA-LMS uses a $4\times$ refeed (Section 3.3.2.2) with geometric correspondence weighting (Section 3.3.2.3). Image source: [5] – ©2016 IEEE.

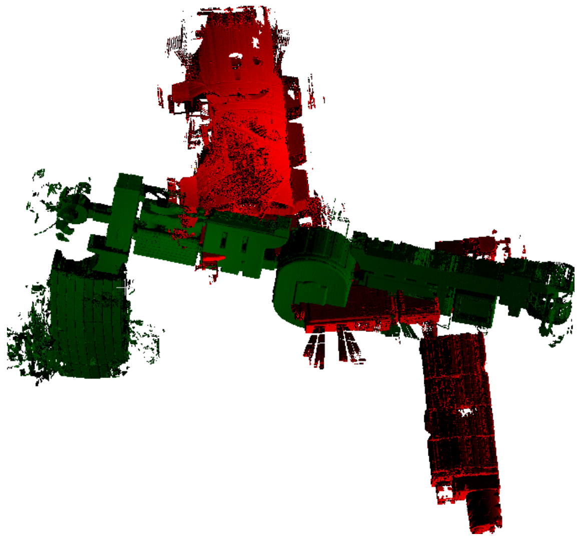
Using the first bunny pair shown in Figure 3.6, the resilience against outliers is shown in Figure 3.7. The outlier correspondences are randomly added to true correspondences to generate tougher alignment scenarios. The results clearly show the GA-LMS superiority as compared to SVD. The same holds true even if half as many correspondences are used. Similar experiments on the two other bunny pairs (not shown) confirmed these conclusions.

3.4.3 Alignment with Few Correspondences

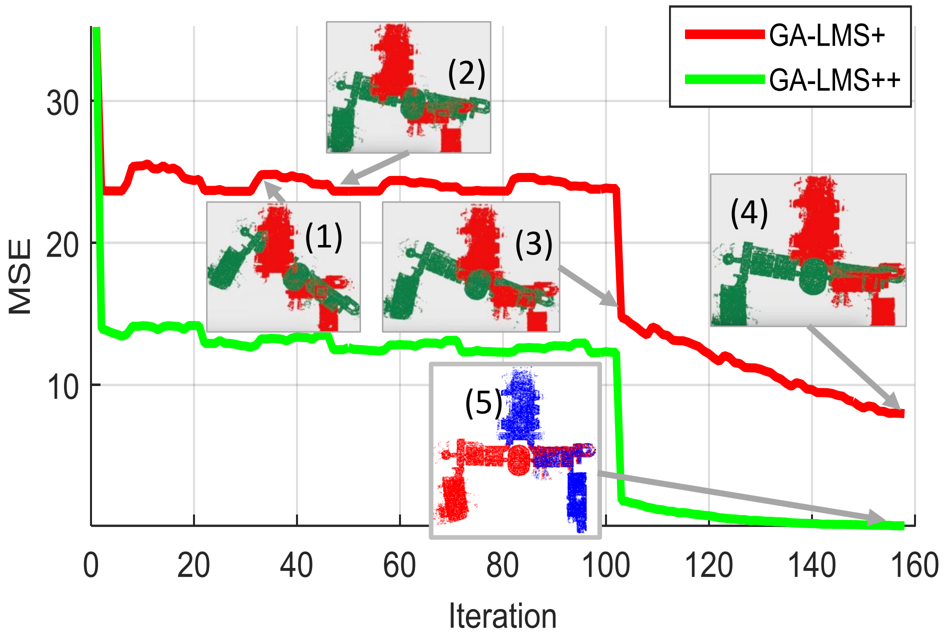
The experiment shown in Figure 3.8 demonstrates the GA-LMS+ capability to deal with cases that have a small number of correspondences, out of which many are false and distributed in an unfavorable geometric fashion. Moreover, it shows how the different mechanisms collectively help to achieve a substantially improved alignment. The GA-LMS+ does not benefit from the multiple feeds before the statistical correspondence filtering is performed. However, as soon as that is done, the multiple feeds are critical to overcome the problem with the limited number of correspondences (see MSE progression after the statistical correspondence filtering in Figure 3.8(c)). The statistical correspondence filtering is seen to significantly reduce the number and ratio of outliers. In effect, the GA-LMS+ retrieves the alignment. The residual orientation error (α_e) is around 7°. The GA-LMS++ shows even better performance. Due to the geometric constellation of the outliers, the geometric correspondence weighting leads to effective downweighting of the outliers contribution. The final outcome has an angle error of $\alpha_e = 0.8°$and nearly optimal translatory alignment. The SVD-based approach, on the other hand, fails completely.



(a) Correspondences



(b) SVD-based alignment



(c) MSE for GA-LMS-based alignment

Figure 3.8: Aligning two large-scale indoor point clouds obtained inside a museum. The number of correspondences is reduced to 25, out of which 11 are wrong (outliers). (a) shows the correspondences are visualized with green indicating true correspondences as opposed to red. (b) shows the registration result using standard SVD-based LS solver. The SVD-based solver fails to properly register the PCDs. (c) shows the true MSE of different variants of the GA-LMS. The GA-LMS+ initially rises (1). The MSE then reduces as more inliers are encountered (2). Nevertheless the orientation error remains high until the statistical correspondence filtering (see Section 3.3.2.4) is performed (3). This filtering increases the inlier rate and the AF keeps minimizing the MSE. The final alignment (4) is correct in terms of translation and has a relatively small angle error as opposed to the SVD-based approach, which fails completely. Due to the geometric correspondences weighting, GA-LMS++ is able to perform even better, providing nearly optimal translatory and rotational alignment (5). Image source: [5] – ©2016 IEEE.

3.5 Summary

In this chapter the Geometric-Algebra-based LMS adaptive filter (GA-LMS) is extended to estimate the 6DOF alignment between two point clouds, related by 1-to-1 correspondences, without prior alignment. Also, the adaptive-filtering properties of the GA-LMS are exploited to evolve our previous approach [1] by applying a series of techniques that equip the GA-LMS with outlier resilience. The improvements include:

1. Skipping correspondences that lead to an instantaneous increase in the filter MSE. In line with intuition, the results confirm that the true MSE often improves when skipping based on the filter MSE is performed.

2. Sample (correspondence) refeeding: Building upon knowledge from information theory, the correspondences are refed repeatedly to use the residual error in combination with the new filter state to lead to further reduction in the registration error.

3. Geometric weighting: A method that exploits a fundamental property of the Special Euclidean Group is presented which allows to compute the probability that a correspondence is an outlier. The computed probability is used as a weight to the step-size to reduce the contributions from outliers.

4. Statistical correspondence filtering: Having observed that the filter typically recovers the rotation a statistical filter is applied to determine the outlier correspondences identified by being those whose source-target spatial distance is sufficiently different from the median correspondence distance. The filter is then run only with the (presumed) inlier correspondences leading to a more accurate registration.

The results show that the collection of the mechanisms when combined lead to a substantial improvement in registration accuracy clearly outperforming the standard SVD-based LS alignment solver for the tested datasets. The filter has a low computational footprint and is done within a few milliseconds.

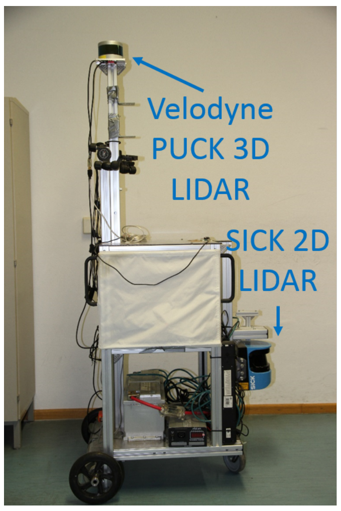
In the next chapter, the GA-LMS's adaptive and nature is exploited to cope with a phenomenon related to the sequential scanning nature of modern LiDARs.

Major parts of this chapter were published in the proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV) 2016 [5].

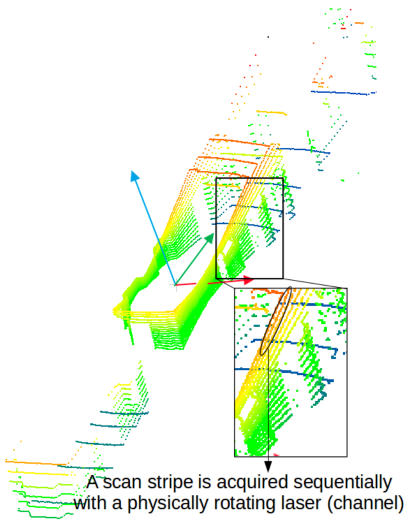
Chapter 4

Analyzing Scan Skewing and its Impact on Scan Matching

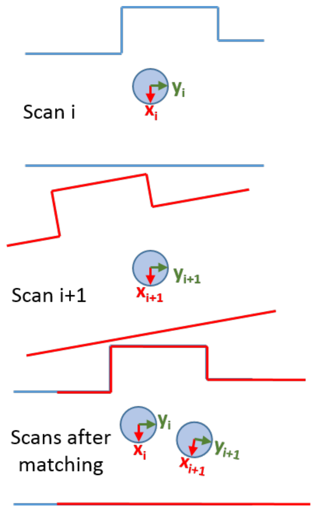
To capture point clouds of indoor environments, LiDAR (Light detection and ranging) sensors can be used. LiDARs are active sensors which emit light rays in known directions and measure the time required to capture the reflected beam which allows inferring the distance of a 3D point along the ray and hence its 3D location w.r.t. the sensor.



(a) LiDAR mapper


A scan stripe is acquired sequentially
with a physically rotating laser (channel)

(b) A LiDAR scan



(c) Scan matching

Figure 4.1: The LMT mapping trolley [18] with a Velodyne Puck LiDAR mounted on top is shown in (a). The LiDAR acquires scans made of 3D points as the one shown in (b), where the height is color-coded. Each tilted stripe of points (point plane) is scanned by one mechanically rotating laser *sequentially*. One scan represents a relatively sparse sampling of the environment. Matching multiple scans as illustrated in (c) is necessary to reconstruct an indoor environment in high quality.

LiDAR sensors typically provide a relatively sparse scan of the environment as shown in Figure 4.1(b). Individual scans[*] of a LiDAR sensor mounted on a moving platform, as that

[*]While LiDARs continuously obtain points using a rotating laser, it is customary to assemble all sequentially scanned points during *one* full rotation into one scan.

shown in 4.1(a), can be matched to estimate the scanner movement and "stitch" the scans (see Figure 4.1(c)) to provide extensive and densely sampled point clouds.

Accurate *Scan matching* is challenging as tiny errors especially in the rotation lead to propagating errors that manifest themselves in trajectory drift. A fundamental problem that can lead to rotation estimation errors in scan matching is that of scan *skewing*. It's a phenomenon that is a direct consequence of platform mobility and the fact that one scan is comprised of *sequentially* scanned points within a certain time frame. This problem is explained in greater detail in Section 4.1 as it is the main focus of this work.

To the best of our knowledge, there exists no published systematic study of the scan skewing phenomenon. Hence, the objectives of the contributions of this chapter are two-fold:

1. Systematically investigate & understand the impact of scan skewing (Sections 4.1, 4.3).

2. Reduce the impact of skewing without the aid of additional sensors (Section 4.4).

In dealing with the second objective, the GA-LMS filter introduced in Chapter 3 is used. Here, however, a modified variant that incorporates the point to plane error metric is derived (Section 4.4.1) and used.

4.1 The Skewing Problem

4.1.1 Scan Distortion

LiDAR's typically have a physically rotating laser emitter and hence points are captured sequentially while completing a full rotation. Hence, a full rotation generates a scan. The Puck, which is used in this work, has 16 such lasers (*channels*) rotating around the vertical axis each tilted by an angle from the horizontal plane resulting in the distinct point arrangement seen in Figure 4.1(b).

Assuming the LiDAR is stationary during one full rotation of the laser and disregarding sensor noise and calibration errors, a geometrically correct sampling of the environment can be achieved. However, as already mentioned, such sensors are typically mounted on mobile platforms. Hence, the assumption of stationarity is mostly violated. Accordingly, the sensor will have a slightly different pose when scanning each point.

The LiDAR captures scans in its local coordinate frame. Since one scan comprises all points scanned in one complete rotation of the laser, the movement of the local coordinate frame when iterating through the points should ideally be accounted for. Unfortunately, the sensor is not capable of doing that as it does not know how it moved. Retrieving the motion, after all, is the ultimate target of scan matching.

The inadvertent outcome of this phenomenon is a distortion in the scan as illustrated in Figure 4.2. A simulation was designed to understand the severity of the problem as a function

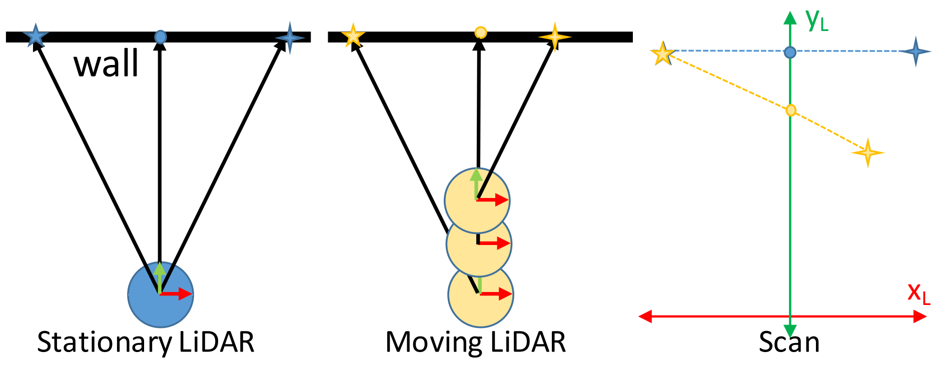


Figure 4.2: Illustration of the scan skewing problem. The LiDAR is depicted from above by the circular shape with two vectors identifying its local coordinate system. Points on a wall are scanned using a rotating laser sequentially in the following order: star, circle and diamond. In blue, a stationary sensor is shown as opposed to the realistic case of a moving sensor shown in yellow. Since the angle between every two consecutive laser beams is fixed, a different set of points are sensed when the LiDAR is moving. Moreover, each new point is sensed from a slightly different pose than the one preceding it. Hence, the movement of the local coordinate system while scanning the points translates into a non-rigid deformation of the geometry when expressing all the scanned points in a common reference frame. Image source: [4] – ©2016 IEEE.

of different types of motion. The results are shown in Figure 4.3. Especially rotational movement is seen to incur noticeable non-rigid distortion affecting mainly points that are far away from the sensor origin. Obviously, the stronger the motion, the stronger the distortions.

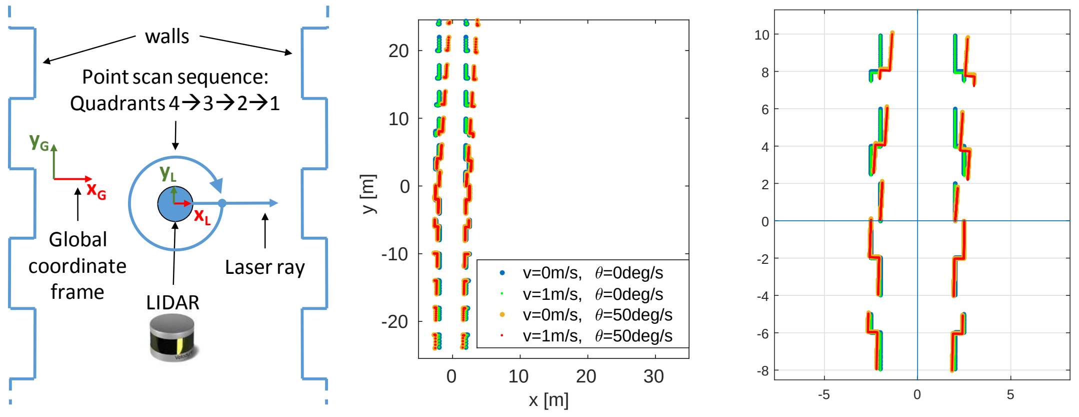
4.1.2 Theoretical Effect on Mapping

Let's start with any point $\mathbf{p}_i$ expressed in the global frame as $\mathbf{p}_i = [\mathbf{p}_i(x)\ \mathbf{p}_i(y)\ \mathbf{p}_i(z)]^T$. In scan j's local coordinate frame it can be expressed as $\mathbf{p}_i^j = \left[\mathbf{p}_i^j(x)\ \mathbf{p}_i^j(y)\ \mathbf{p}_i^j(z)\right]^T$, where the relationship between $\mathbf{p}_i$ and $\mathbf{p}_i^j$ is given by

$$\mathbf{p}_i^j = \mathbf{R_{i(j)}}(\mathbf{R}_0^j\,\mathbf{p}_i + \mathbf{t}_0^j) + \mathbf{t_{i(j)}}, \tag{4.1}$$

where $\mathbf{R_{i(j)}}$ and $\mathbf{t_{i(j)}}$, is the individual rotation and translation of point i due to skewing while capturing scan j. So, $\mathbf{R_{i(j)}}$ and $\mathbf{t_{i(j)}}$ are w.r.t. scan j's local coordinate frame which, without loss of generality, can be defined as the LiDAR pose when scanning the *first* point of the scan. $\mathbf{R}_0^j$ and $\mathbf{t}_0^j$, on the other hand, is the coordinate transformation that expresses points expressed in the global coordinate frame (0) in the local coordinate frame (j).

Assuming a constant velocity and rotation speed motion model, the following approximations can be used: $\mathbf{R}_{i(j)} \approx \mathbf{R}_{i(j+1)}$, $\mathbf{t}_{i(j)} \approx \mathbf{t}_{i(j+1)}$. This means that for consecutive scans the individual deformation for *corresponding* points is almost the same. Indeed, Figure 4.4 shows real scan results that agree with the outcome of this assumption. Intuitively, a like-wise non-rigid transformation of two scans should not impact the matching. Indeed this can be proven by rewriting the point set alignment cost function (2.1) between two consecutive scans



(a) Simulation setup (b) Scan comparison (c) Scan comparison (zoom-in)

Figure 4.3: A 2D scan generated for a synthetic indoor environment made up of planar walls. The setup is shown in (a) from a top view. The LiDAR's local coordinate system is shown as x_L, y_L as opposed to the global coordinate system shown as x_G, y_G. The simulated LiDAR laser rotates around the z-axis clock-wise. Different movement scenarios are simulated involving a translation along y_G by a constant speed v and an angular rotational speed θ around the (invisible) z-axis. The outcome is shown in (b). Especially the rotational speed is seen to incur strong distortions. Also, the farther away the points are from the sensor origin, the larger the distortion. Comparing quadrant 1 (scanned last) to quadrant 4 (scanned first) the distortion is seen, as expected, to increase as the scan proceeds towards later points since the local reference frame will have moved the most. Image source: [4] – ©2016 IEEE.

as follows:

$$
\begin{aligned}
C(j, j+1) &= \sum_i \left\| \mathbf{R}_{i(j+1)}(\mathbf{R}_j^{j+1}\mathbf{p}_i^j + \mathbf{t}_j^{j+1}) + \mathbf{t}_{i(j+1)} - (\mathbf{R}_{i(j)}\mathbf{p}_i^j + \mathbf{t}_{i(j)}) \right\|_2^2 && (4.2)\\
&\approx \sum_i \left\| \mathbf{R}_{i(j+1)}(\mathbf{R}_j^{j+1}\mathbf{p}_i^j + \mathbf{t}_j^{j+1}) - \mathbf{R}_{i(j)}\mathbf{p}_i^j \right\|_2^2 && (4.3)\\
&\approx \sum_i \left\| \mathbf{R}_{i(j+1)}(\mathbf{R}_j^{j+1}\mathbf{p}_i^j + \mathbf{t}_j^{j+1} - \mathbf{p}_i^j) \right\|_2^2 && (4.4)\\
&= \sum_i \left\| \mathbf{R}_j^{j+1}\mathbf{p}_i^j + \mathbf{t}_j^{j+1} - \mathbf{p}_i^j \right\|_2^2 && (4.5)
\end{aligned}
$$

The terms $\mathbf{t}_{i(j+1)}$ and $\mathbf{t}_{i(j)}$ canceled by virtue of the subtraction combined with the above assumption $\mathbf{t}_{i(j)} \approx \mathbf{t}_{i(j+1)}$. Similarly, the assumption $\mathbf{R}_{i(j)} \approx \mathbf{R}_{i(j+1)}$ allows factoring out the deformation rotations. Since a rigid rotation does not change the norm, $\mathbf{R}_{i(j+1)}$ drops out of the equation. What remains is a cost function independent of the skewing and dependent only on the scan-to-scan transformation (expressed by $\mathbf{R}_j^{j+1}$ and $\mathbf{t}_j^{j+1}$) whose estimation is the original objective of the registration.

The preceding derivation might tempt the reader to conclude that the scan skewing problem does not affect the scan matching. However, the derivation is based on the assumption that the point-wise deformation is similar for geometrically corresponding points. As will be shown in Section 4.3 this is not necessarily true as the motion gets more complex. Especially

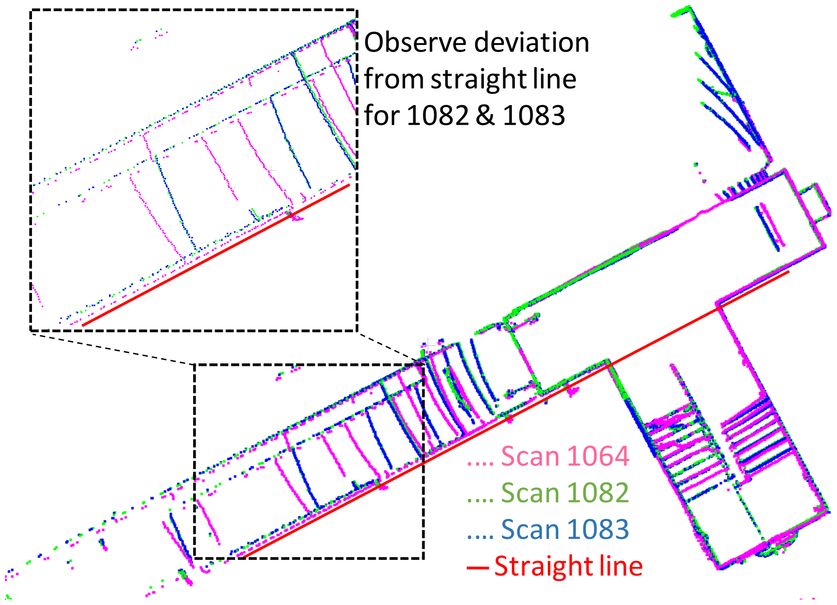


Figure 4.4: Real scans produced by the Puck seen from above after matching. Three scans have been chosen: two consecutive scans (1082th,1083rd) captured as the mapping platform was turning, and one scan (1064th – captured around 2 seconds earlier) captured before the platform started to turn. Noticeable scan skewing is observed for scans 1082 and 1083 seen by the deviation of their points from the ideal straight line. These two scans exhibit a similar skewing seen by their near optimal alignment along all parts. Image source: [4] – ©2016 IEEE.

changes in the rotation speed mean the distortion similarity assumption is violated. Moreover, in scan matching it is typical to run not only matching (localization) but also mapping simultaneously, hence the name simultaneous localization and mapping (SLAM). Accordingly, even if the matching of two consecutive scans were not to be affected by skewing, the mapping will nonetheless be affected as skewed point clouds will be added to the map. Hence, a systematic analysis of this problem is important.

The preceding argumentation should convince the reader that accounting for the skewing is important for accurate scan matching. For mapping platforms equipped with inertial measurement units (IMU) the data can be *de-skewed* by estimating the local coordinate frame transformation between individual point scans and using the estimated transformation to correct the distortion. This is possible because IMUs typically run at a substantially higher sampling rate and their data can be considered reliable for short time frames as those spanned between two consecutive scans. Good IMUs, however, are quite expensive. Moreover, to be usable for de-skewing they require precise calibration w.r.t. the LiDAR. The latter is quite challenging in practice.

Based on the preceding discussions, the objective of this work is to systematically investigate scan skewing and develop a method to reduce its negative impact. For that an ICP-based scan matching framework was built which is explained in greater detail in Section 4.2.

4.2 Scan Matching Framework

A scan matching application based on the ICP algorithm (Section 2.2.3) was built using C++ and the PCL [17]. The ICP algorithm is run on consecutive scans to estimate the scan-to-scan motion and hence recover the motion trajectory.

Each ICP run involves repetitive correspondence estimation with subsequent alignment estimation processes. Our ICP implements three algorithms to estimate the alignment from the correspondences:

1. Standard ICP minimization with the point-to-point (pt2pt) metric [46] using an SVD-based LS filter which was explained in Section 2.2.1. Henceforth termed “ICPpt2pt”.

2. Standard ICP minimization with the linear least-squares (LLS) point-to-plane (pt2pl) metric [66]. Henceforth termed “ICPpt2pl”.

3. Minimization using the Geometric Algebra Adaptive filter we developed (see Chapter 3). Here, however, a new variant of the GA-LMS that uses the point-to-plane metric is used (see Section 4.4.1).

It remains to be said that we use a simple trick to increase the accuracy of scan matcher: Whenever matching a new scan, the scan-to-scan transformation of the previous scan is used to initialize ICP. This not only helps ICP converge faster but also substantially improves the scan-to-scan transformation estimates. The previous transformation is however not used to de-skew the scan data as is employed in [67]. This would involve interpolating a transformation for each point (or group of points) using the previous scan-to-scan transformation and some assumed motion model. Our objective here is to account for the data skewing in the transformation estimation filter instead of explicitly de-skewing the data using some assumed transformations.

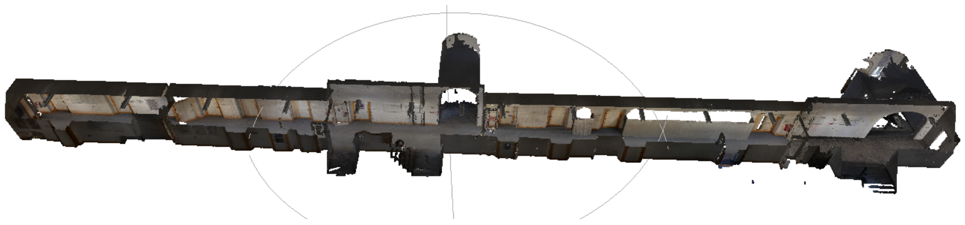
4.3 Experimental Verification of Skewing Impact

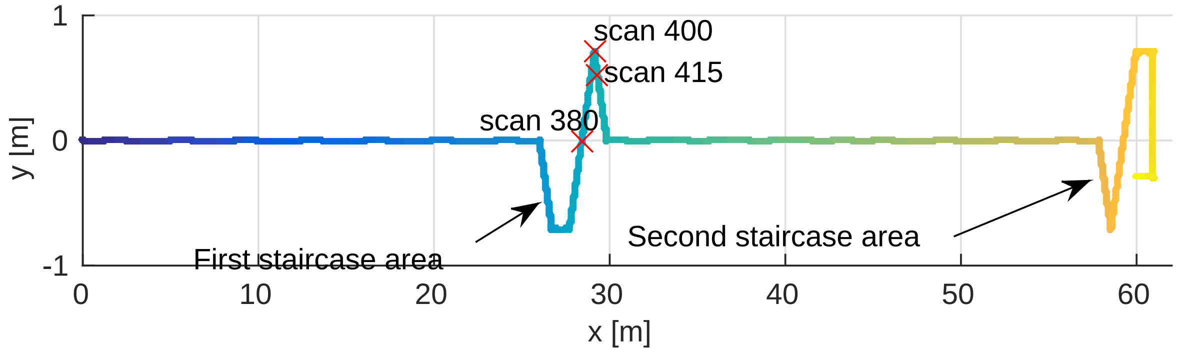
4.3.1 Synthetic Dataset

To systematically investigate the problem of scan matching from the perspective of skewing a synthetic scan dataset was used. It simulates a 3D Puck (also called VLP-16) scanner [68]. The scanner scans points while rotating anti-clockwise around the z-axis from 0°to 360°. Hence the scan proceeds through the quadrants 1, 2, 3 and 4 sequentially (this is important information when designing the weighting function (4.23)).

To make the data as realistic as possible the scanner is moved along a smooth trajectory generated by fitting a spline function on a series of user-defined way points. A similar interpolation is applied on the scanner rotation to generate smooth scanner rotation.

(a) Hallway


(b) Indoor environment mesh



(c) Trajectory

Figure 4.5: Synthetic and real LiDAR data are acquired for the hallway shown in (a). The textured 3D mesh of the hallway used to generate the synthetic data is shown with parts of the ceiling and walls removed in (b). (c) shows the virtual trajectory followed by the simulated scanner. The gradual color change modulates the time index starting with blue and ending with yellow. Three specific scans are marked by their numbers. Image source: [4] – ©2016 IEEE.

Table 4.1: The four synthetic dataset variants with different types of distortions.

variant	1	2	3	4
Simulated noise	no	yes	no	yes
Simulated skewing	no	no	yes	yes

The synthetic scans were generated by moving a virtual scanner along the trajectory shown in Figure 4.5(c) inside the 3D mesh model shown in Figure 4.5(b) which was captured in the hallway shown in Figure 4.5(a). The mesh model has been captured using the NavVis M3 Mapping Trolley [25] which is a commercially available indoor mapper that employs multiple LiDARs and an IMU together with state-of-the-art SLAM to produce accurate 3D maps. Four different dataset variants with the characteristics shown in Table 4.1 were created.

4.3.2 Quantifying the Impact of Skewing on Scan Matching

Figure 4.6 shows the estimated trajectories obtained by running standard ICPpt2pt and ICPpt2pl (Section 4.2) on the datasets introduced in Section 4.3.1. As expected, skewing is seen to noticeably affect the scan matching leading to errors that manifest themselves in the form of trajectory drift.

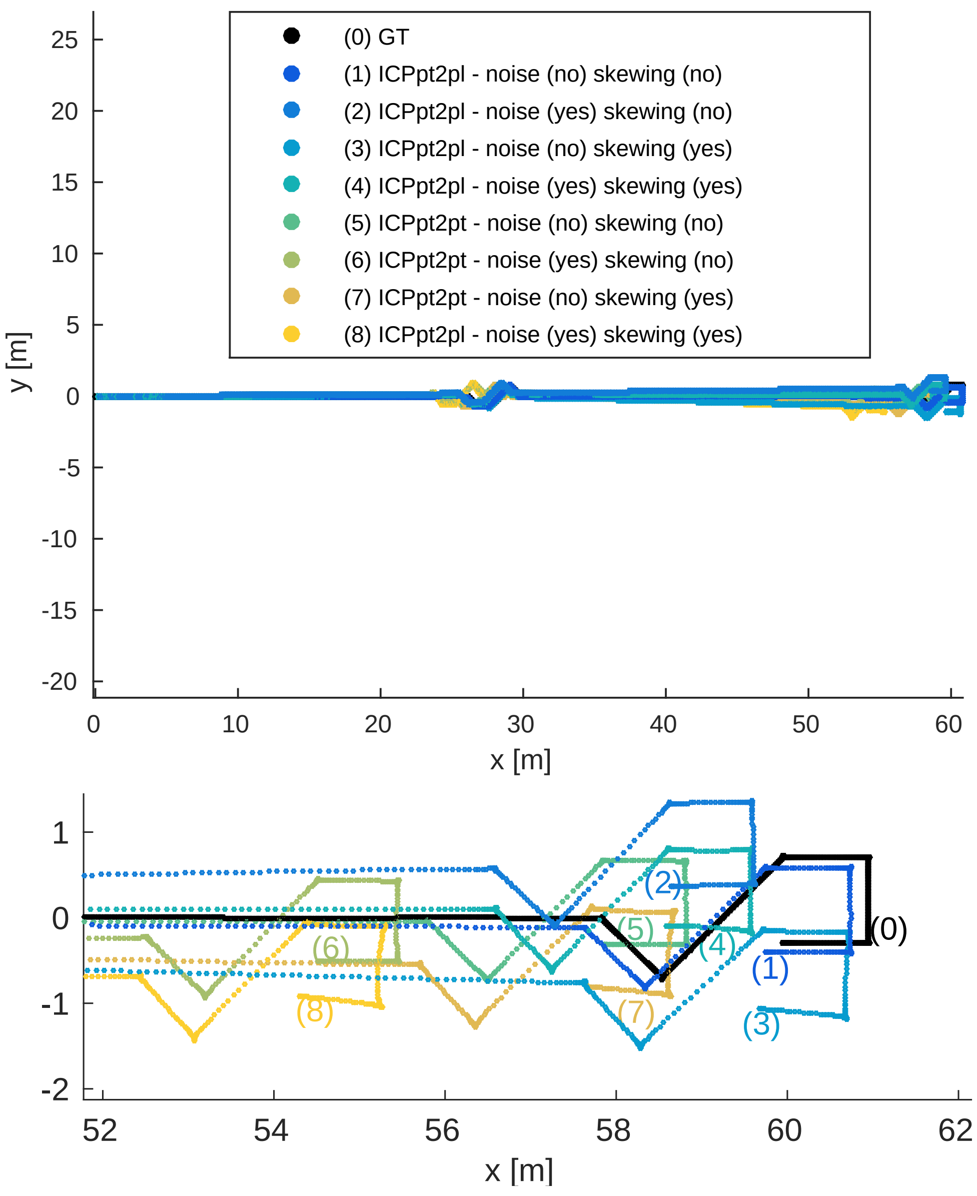


Figure 4.6: Estimated trajectories after scan matching for the synthetic dataset explained in 4.3.1 using standard scan-to-scan ICP (see Section 4.2). The trajectories are seen to coincide with the ground truth (GT) trajectory to a large degree. However, the zoom-in (bottom) reveals that pt2pl scan matching is substantially more accurate than pt2pt scan matching (compare 1 to 5, 2 to 6, 3 to 7 and 4 to 8) in accordance with wide-spread belief of the community (see Section 2.2.3). As expected, the trajectories corresponding to no added distortions (1 & 5) are the closest to the ground truth among their peers that use the same metric. Also, the trajectories with skewed data (3 & 7) have a visible drift. Image source: [4] – ©2016 IEEE.

The apparent compensation of drift from skewing by drift from noise in pt2pl matching (curve 4 in Figure 4.6) is by coincidence. To verify that we ran scan matching starting from scan #415 (see Figure 4.5(c)) and found out that in that case the drifts add up.

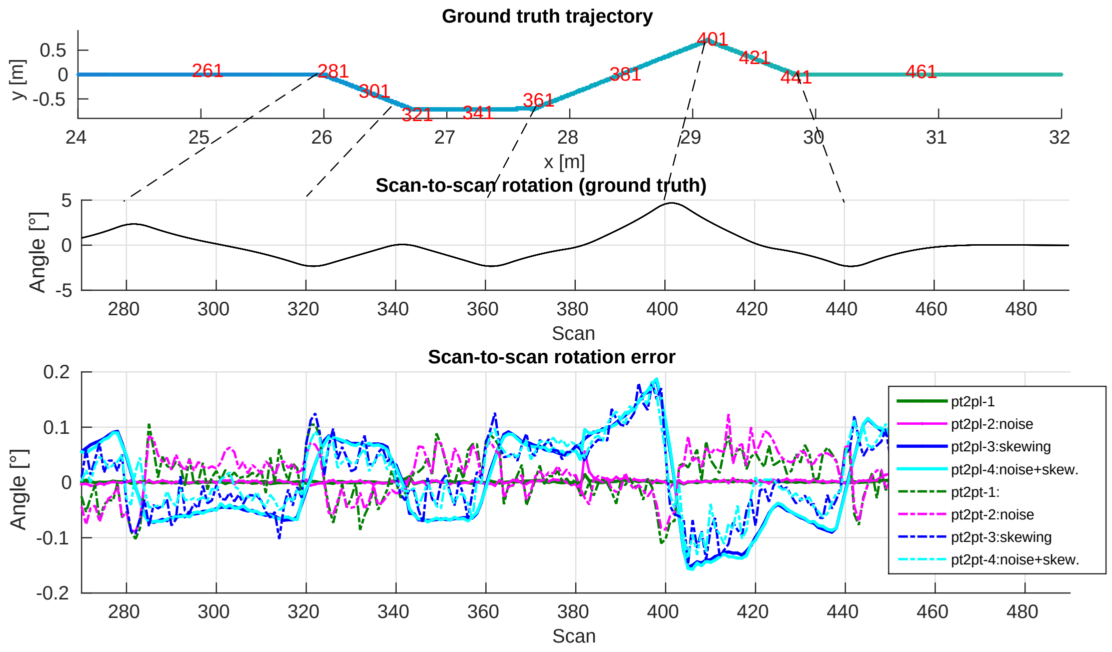


Figure 4.7: Scan-to-scan errors for a selected part of the trajectory. The upper two figures show the ground-truth (GT) trajectory and the GT scan-to-scan rotation around the vertical axis, respectively. Notice, the scanner is not simulated to be rigidly attached to the platform (no kinematic model) hence a straight trajectory does not necessarily imply no scanner rotation. Focusing on the rotation errors (bottom) the following observations can be made: pt2pt matching results in fluctuating and less stable rotation error (compare solid to dashed curves); When adding just noise, pt2pl scan matching, as expected, still results in very accurate rotation estimation (compare solid green and pink curves); Scans with skewing lead to two types of rotation error: +ve error or -ve error (see oscillating pattern of dark blue curves); The type of rotation error, surprisingly, does not depend on the rotation direction only but also on the change in *absolute* rotation speed. For example, in the scan range 320-360 the rotation is always negative, however, we observe -ve rotation error when the absolute rotation speed is increasing and +ve error when the absolute rotation speed is decreasing. This pattern reverses for +ve rotation angles. The rotation error does not necessarily go to zero when the rotation speed goes to zero (300, 380, 420). The rotation error goes to zero only when the rotation speed *change* goes to zero (scans around 280, 320, 340, 360, 400, 440). The observation about when the error goes to zero is in accordance with the derivation of Section 4.1.2 which showed that when the rotation speed is constant, the skewing does not affect the registration. Image source: [4] – ©2016 IEEE.

Having shown that skewing does indeed incur scan matching errors it is time to investigate these errors in more detail. Since we use synthetic data we have accurate ground truth data which we use to analyze the scan-to-scan errors in terms of rotation and translation. These errors are shown in Figure 4.7. As expected, the results show that while the scanner is rotating (in our case around the z-axis) the scan matching *may* exhibit a -ve or +ve error in the rotation estimate around the z-axis (the sign is based on the right-hand-rule of rotation around the axis). What is however unexpected is the observation that the type of error (-ve or +ve) is not a function of rotation direction *only* but also of the change in angular rotation speed (rate). A +ve (anti-clock-wise) rotation leads to a +ve rotation error (equivalent to clock-wise drift) in case of an increasing rotation rate as opposed to a decreasing rotation rate which leads to a -ve rotation error (equivalent to anti-clock-wise drift). Conversely, a -ve (clock-wise) rotation leads to -ve angle error in case of an increasing *absolute*[†] rotation rate as opposed to a decreasing rotation rate in the absolute sense which incurs a +ve error. Another interesting observation is that when the rotation speed stops changing (peaks and minima in the ground-truth scan-to-scan rotation) the rotation error goes to zero. The latter observation verifies the arguments of Section 4.1.2 which suggested that a constant rotation speed incurs similar skewing deformation for corresponding points which was shown to have no impact on the alignment of consecutive scans.

It remains to be said that the previously presented results show that ICPpt2pl consistently outperforms ICPpt2pt. Comparing the trajectories of ICPpt2pt vs. ICPpt2pl shown in Figure 4.6 for the same type of dataset, the ICPpt2pt trajectories appear to be "compressed". This is a direct consequence of the fact that ICPpt2pt uses the pt2pt metric (Section 2.2.1) where the solution involves estimating the translation alignment after the rotation through centroid subtraction. Observing Figure 4.1(b) it can be seen that the centroid is heavily biased towards the scan center irrespective of the scan due to the nature of point acquisition. This means the centroid is not a reliable measure to estimate the translation.

4.4 Skewing-Aware Matching with GA-LMSpt2pl

Having theoretically derived and experimentally verified the impact of skewing the objective now is to reduce this impact. Since we now know which points are affected the most by skewing it is conceivable to tune the contributions of points towards the final estimation from ICP by suitable weighting. For that the Geometric Algebra Alignment filter (GA-LMS) which was presented in Chapter 3 is to be employed. Its iterative nature with the adaptive step-size lends itself to the problem at hand. It is conceivable to weight the step-size based on how much a correspondence is affected by skewing.

Unfortunately, the GA-LMS in its original shape, as presented in Chapter 3, is derived with the pt2pt error metric and hence equivalent to the ICPpt2pt. However, the results in Section 4.3 show that a pt2pl metric results in a more robust trajectory estimation. Hence, the GA-LMS that uses the point-to-plane metric, henceforth termed "GA-LMSpt2pl", is first to be derived.

[†]Theoretically, an increase in clock-wise (-ve) rotation rate in the absolute sense is a decrease in rotation rate. Since a -ve rotation rate, however, implies reversal of rotation direction it may perhaps be more comprehensible to speak about absolute rotation rates coupled with the rotation direction.

4.4.1 Deriving GA-LMSpt2pl

4.4.1.1 Rotation Estimation

The following derivation has been contributed by Mr. Wilder Lopes of the University of Sao Paulo. You may want to revisit Section 3.1 to review the basics of GA and our notation. To derive the GA-LMSpt2pl the Linear-Algebra pt2pl cost function (without translation)

$$C_{pt2pl}(\mathbf{R}) = \frac{1}{N}\sum_{i=1}^{N}\left\|(\widehat{\mathbf{p}_i^{ct}} - \mathbf{R}\widehat{\mathbf{p}_i^{cs}})^T\mathbf{n}_i\right\|^2 \tag{4.6}$$

is replaced by a GA formulation similar to the one in (3.2) as follows

$$C_{pt2pl}(\mathbb{r}) = \frac{1}{N}\sum_{i=1}^{N}\left|(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i\right|^2. \tag{4.7}$$

Remember, from now on vectors (variables in bold small letters) in the context of GA become multivectors. The AF provides an estimate for the bivector $\mathbb{r}$ via a recursive rule of the form,

$$\mathbb{r}_i = \mathbb{r}_{i-1} + \mu\mathbb{g}, \tag{4.8}$$

where i is the (time) iteration, μ is the AF step size, and $\mathbb{g}$ is a multivector-valued update quantity related to the estimation error (analogous to the standard formulation in [53], p.143).

A proper selection of $\mathbb{g}$ is required to enforce $C_{pt2pl}(\mathbb{r}_i) < C_{pt2pl}(\mathbb{r}_{i-1})$ at each iteration. This work adopts the steepest-descent rule [53], [69], in which the AF is designed to follow the opposite direction of the reversed gradient of the cost function, namely $\widetilde{\nabla}C_{pt2pl}(\mathbb{r}_{i-1})$. This way, $\mathbb{g}$ is proportional to $\widetilde{\nabla}C_{pt2pl}(\mathbb{r}_{i-1})$,

$$\mathbb{g} \triangleq -\mathbb{b}\widetilde{\nabla}C_{pt2pl}(\mathbb{r}_{i-1}), \tag{4.9}$$

in which $\mathbb{b}$ is a general multivector, in contrast with the standard case in which $\mathbb{g}$ would be a matrix [53]. Setting $\mathbb{g}$ equal to the identity matrix results in the steepest-descent update rule ([53], Eq. 8-19). In GA though, the multiplicative identity is the multivector (scalar) 1 ([50], p.3), thus $\mathbb{g} = 1$.

Define $C'_{pt2pl}(\mathbb{r}) \triangleq \left|(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i\right|^2$. Using the GA scalar product and the definition of magnitude of a multivector, C_{pt2pl} can be expanded as

$$\begin{aligned} C'_{pt2pl}(\mathbb{r}) &= \left|(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i\right|^2 = \left\langle[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i][(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i]\right\rangle \\ &= [(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i]^2, \end{aligned} \tag{4.10}$$

in which $\widetilde{[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i]} = [(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\cdot\mathbf{n}_i]$ due to the fact that vectors (grade-1 multivectors) are not affected by the reversion operator. Plugging (4.10) back into (4.7), the gradient of $C_{pt2pl}(\mathbb{r})$ can be obtained as follows

$$\nabla_{\mathbb{r}}C_{pt2pl}(\mathbb{r}) = \frac{1}{N}\sum_{i=1}^{N}\nabla_{\mathbb{r}}C'_{pt2pl}(\mathbb{r}), \tag{4.11}$$

where $\nabla_{\mathbb{r}} C'_{pt2pl}(\mathbb{r}) = \nabla_{\mathbb{r}}[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i]^2$. This way,

$$\begin{aligned} C'_{pt2pl}(\mathbb{r}) &= 2[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i]\nabla_{\mathbb{r}}[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i] \\ &= -2[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i]\nabla_{\mathbb{r}}[(\mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i] \\ &= -2[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i]\nabla_{\mathbb{r}}\langle \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}\mathbf{n}_i\rangle. \end{aligned} \tag{4.12}$$

The last term in 4.12 is expanded as

$$\nabla_{\mathbb{r}}\langle \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}\mathbf{n}_i\rangle = \partial_{\mathbb{r}}\langle \dot{r}\mathbb{m}_i\rangle + \partial_{\mathbb{r}}\langle \mathbb{t}_i\dot{\widetilde{\mathbb{r}}}\rangle. \tag{4.13}$$

The terms $\mathbb{m}_i = \widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}\mathbf{n}_i$ and $\mathbb{t}_i = \mathbf{n}_i\mathbb{r}\widehat{\mathbf{p}_i^{cs}}$ are obtained applying the *cyclic reordering property* $\langle \mathbb{a}\mathbb{d}\cdots\mathbb{c}\rangle = \langle \mathbb{d}\cdots\mathbb{c}\mathbb{a}\rangle = \langle \mathbb{c}\mathbb{a}\mathbb{d}\cdots\rangle$ [52]. The first term on the right-hand side of (4.13) is $\partial_r\langle \dot{r}\mathbb{m}_i\rangle{=}\mathbb{m}_i$ ([58], Eq. 7.10), and the second term is $\partial_{\mathbb{r}}\langle \mathbb{t}_i\dot{\widetilde{\mathbb{r}}}\rangle{=}{-}\widetilde{\mathbb{r}}\mathbb{t}_i\widetilde{\mathbb{r}}{=}{-}\widetilde{\mathbb{r}}(\mathbf{n}_i\mathbb{r}\widehat{\mathbf{p}_i^{cs}})\widetilde{\mathbb{r}}$ (see Appendix of [1]). Plugging back into (4.13),

$$\nabla_{\mathbb{r}}\langle \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}\mathbf{n}_i\rangle = 2\widetilde{\mathbb{r}}(\mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}} \wedge \mathbf{n}_i), \tag{4.14}$$

where the relation $\mathbf{ab} - \mathbf{ba} = 2(\mathbf{a} \wedge \mathbf{b})$ was used ([51], p.39).

Substituting (4.14) in (4.12) results in

$$C'_{pt2pl}(\mathbb{r}) = -4[(\widehat{\mathbf{p}_i^{ct}} - \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}) \cdot \mathbf{n}_i]\widetilde{\mathbb{r}}(\mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}} \wedge \mathbf{n}_i) = 4[\mathbf{e}_i \cdot \mathbf{n}_i]\widetilde{\mathbb{r}}(\mathbf{n}_i \wedge \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}), \tag{4.15}$$

which allows us to finally obtain the gradient of $C_{pt2pl}(\mathbb{r})$

$$\nabla_{\mathbb{r}} C_{pt2pl}(\mathbb{r}) = \frac{4}{N}\sum_{i=1}^{N}(\mathbf{e}_i \cdot \mathbf{n}_i)\widetilde{\mathbb{r}}(\mathbf{n}_i \wedge \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}). \tag{4.16}$$

Substituting (4.16) into (4.9) (with $\mathbb{b} = 1$, as aforementioned) and explicitly showing the term $1/N$ results in

$$\mathbb{g} = \frac{4}{N}\sum_{i=1}^{N}(\mathbf{e}_i \cdot \mathbf{n}_i)(\mathbf{n}_i \wedge \mathbb{r}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}})\mathbb{r}, \tag{4.17}$$

which upon plugging into (4.8) and explicitly writing the time index yields

$$\mathbb{r}_i = \mathbb{r}_{i-1} + \mu\frac{4}{N}\sum_{i=1}^{N}(\mathbf{e}_i \cdot \mathbf{n}_i)(\mathbf{n}_i \wedge \mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\mathbb{r}_{i-1}, \tag{4.18}$$

Here again we approximate the gradient by its current value to get the LMS filter [53] (similar to what we did to get (3.5)) i.e.

$$\frac{4}{N}\sum_{i=1}^{N}(\mathbf{e}_i \cdot \mathbf{n}_i)(\mathbf{n}_i \wedge \mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\mathbb{r}{\approx}4(\mathbf{e}_i \cdot \mathbf{n}_i)(\mathbf{n}_i \wedge \mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\mathbb{r}_{i-1}, \tag{4.19}$$

resulting in the GA-LMSpt2pl update rule,

$$\boxed{\mathbb{r}_i = \mathbb{r}_{i-1} + \mu(i)(\mathbf{n}_i \wedge \mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\mathbb{r}_{i-1}}, \tag{4.20}$$

in which $\mu(i) = \mu\,\mathbf{e}_i \cdot \mathbf{n}_i$ where the factor 4 was absorbed by μ. Note that the step size μ is weighted by the inner product between the error vector $\mathbf{e}_i$ and the normal $\mathbf{n}_i$. Thus, the net step size $\mu(i)$ becomes *time-dependent*, having its value down-weighted as long as $\mathbf{e}_i$ has its norm decreased at each AF iteration.

4.4.1.2 Estimating Translation

The standard Linear-Allgebra cost function for the pt2pl metric [66] incorporates an operator that accounts for rotation and translation instead of just the rotation matrix $\mathbf{R}$ of (4.6). Replacing this operator using GA is not straight-forward. Hence, a similar strategy as the one we presented in [5] (also explained in Section 3.2) is used to estimate the translation: After estimating the rotation with the centroid-subtracted source and target clouds ((4.6),(4.20)), the translation is computed by subtracting the rotationally aligned source centroid from the target centroid as in (3.6). This decoupled approach, although standard for pt2pt metrics, is inferior to pt2pl which can estimate rotation and translation jointly. As already explained in Section 4.3.2, this is especially problematic for the LiDAR data used here as there is a very high density of points near the sensor origin (see Figure 4.1(b)) making the centroids an unreliable measure of translation when these points happen to lie on walls. To compensate for this we propose using curvature weights (4.24) to compute weighted centroids instead increasing the contribution of points with high curvature in the centroid computation which represent well localizable entities in the point clouds that are reliable for translation estimation.

4.4.2 Skewing-Aware GA-LMSpt2pl

The GA-LMSpt2pl, derived in Section 4.4.1, is used to reduce the impact of scan skewing. More specifically, (4.20) is modified to incorporate a skewing weight $w(i)$ as follows:

$$\mathbb{r}_i = \widetilde{\mathbb{r}}_{i-1} + w(i)\mu(i)(\mathbf{n_i} \wedge \mathbb{r}_{i-1}\widehat{\mathbf{p}_i^{cs}}\widetilde{\mathbb{r}}_{i-1})\mathbb{r}_{i-1} \tag{4.21}$$

where

$$w(i) = f_w(w_c(i), w_s(i)) \tag{4.22}$$

with $f_w()$ being a function that combines the two weights. The skewing weight of correspondence i, $w_s(i)$, is calculated as follows:

$$w_s(i) = cos(\theta_z(i)/4) \tag{4.23}$$

where $\theta_z(i)$ is the xy-plane angle of the vector connecting the source scan origin to point i and is limited to $[0, 360)°$(i.e. $\theta_z(i) = atan(\mathbf{p}_i^s(y)/\mathbf{p}_i^s(x))$. The weight for the first scanned

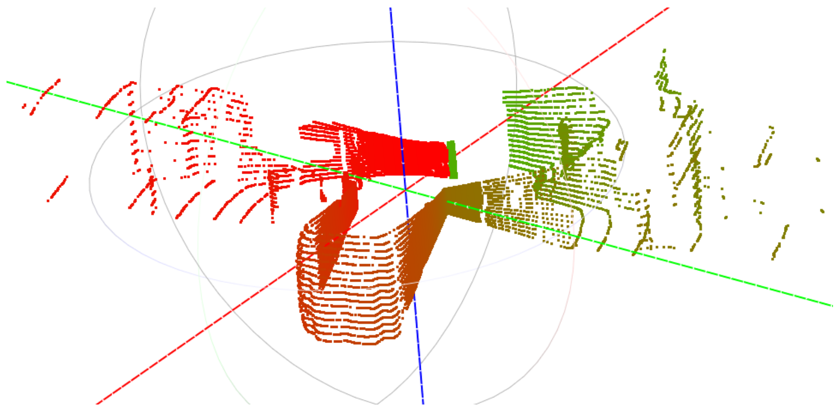


Figure 4.8: The skewing iteration weights computed according to (4.23) for scan #405. The weights are color-coded ($[0, 1] - >$[Green,Red]). Image source: [4] – ©2016 IEEE.

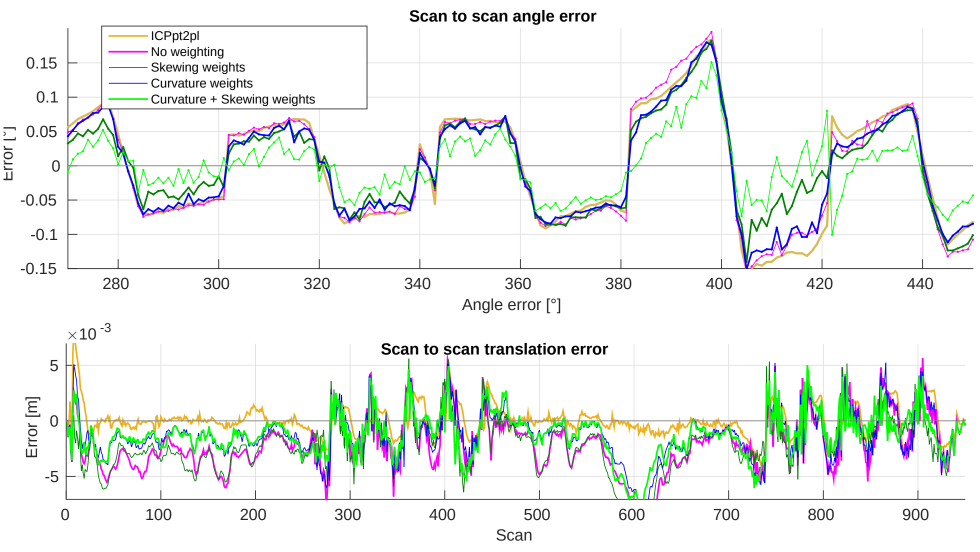
points ($\theta_z \approx 0$) is almost 1 whereas points scanned towards the end ($\theta_z \approx 360$) have a weight that is almost zero as seen in Figure 4.8. Without loss of generality this function is designed to incorporate the property of the simulated scanner whose first scanned point has $\theta_z(1) = 0$ and which scans in clock-wise direction (see Section 4.3.1). A scanner rotating in a different direction or starting from a different angle (as the one used to get the real data used in Section 4.5) requires a simple and obvious adjustment of the weighting function.

The curvature weight w_c, on the other hand, accounts for the importance of a point geometrically as points with a higher curvature typically lie on edges and thus are more distinctive for the alignment process. It is calculated using:

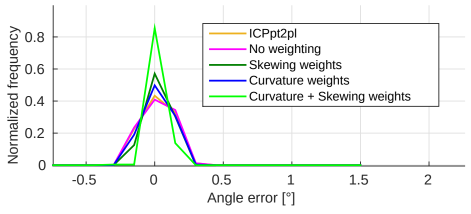
$$w_c(i) = max(0.25, min(\mathbf{p}_i^s(c)/c_r, 1.0)) \tag{4.24}$$

where $\mathbf{p}_i^s(c)$ is the curvature value of source point i and c_r is a reference curvature value indicating points lying on a highly curved surface. These weights are also used for the centroid computation which is critical for the translation estimation (see Section 4.4.1.2).

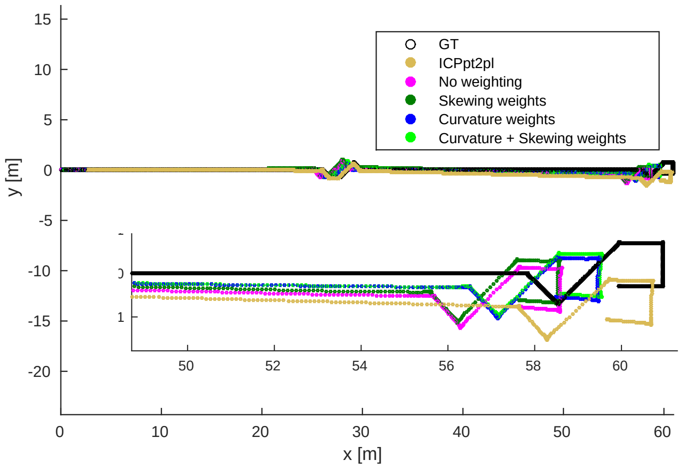
Figure 4.9 compares scan matching between different variants of GA-LMSpt2pl that are obtained when turning on and off skewing weighting and curvature weighting. The GA-LMSpt2pl (Section 4.4.1) delivers a similar rotation estimation accuracy as the standard ICP pt2pl proving it to be a credible substitute. More importantly, as seen in Figure 4.9(a), combining skewing and curvature weighting substantially reduces the rotation estimation errors in the scan regions involving rotations. Indeed, Figure 4.9(b) shows a significantly improved rotation error distribution with errors being more frequently around zero degrees. Also, while the standard GA-LMSpt2pl consistently underestimates the translation due to the unreliability of the centroid estimation (see Section 4.4.1.2), the curvature weight effectively reduces this problem leading to a more accurate trajectory that is less "compressed" as seen in Figure 4.9(c).



(a) Scan-to-scan errors



(b) Scan-to-scan rotation error histogram



(c) Estimated trajectories (with zoom-in)

Figure 4.9: Scan matching is run on the third synthetic dataset variant involving only skewing (Table 4.1). The scan matching is conducted using ICPpt2pl and different variants of the GA-LMSpt2pl (Section 4.4.1) involving no weighting (“No weighting”), skewing weights (4.23) (“Skewing weights”), curvature weights (4.24) (“Curvature weights) and a combination of both weights (“Curvature + Skewing weights”). The angle error curves in (a) show that a combination of skewing and curvature weights according to (4.22) with $f_w(w_s, w_c) = w_s \cdot w_c$ reduces orientation estimation errors (compare green and orange curves) in the regions with strong rotations which is furthermore confirmed by the rotation error histogram in (b). The translation error curves show that the curvature weight also helps in reducing the underestimation of the translation in the GA-LMSpt2pl (compare blue and green curves to pink curve). This leads to a more accurate trajectory that is less “compressed” as seen in (c). Image source: [4] – ©2016 IEEE.

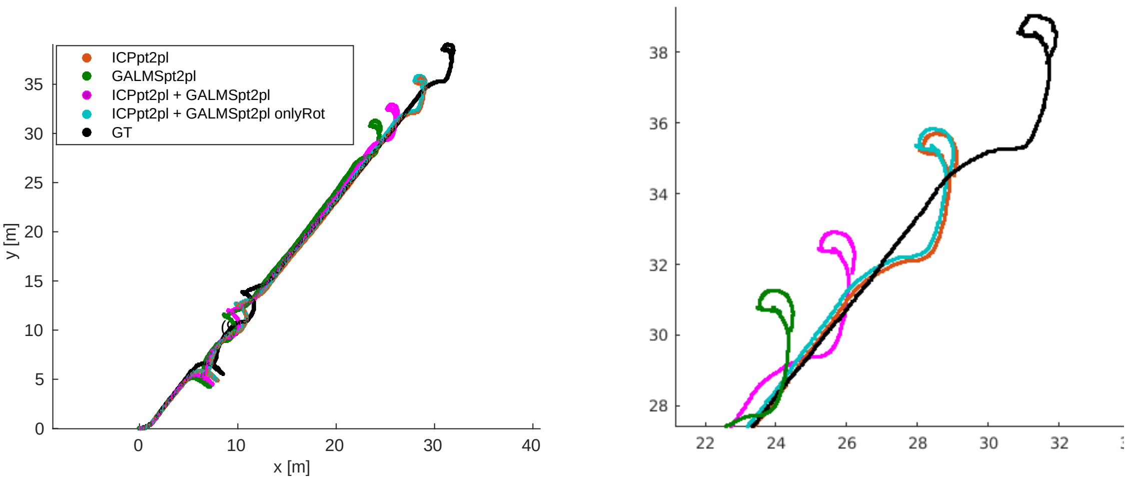
4.5 Evaluation with Real Data

Real data corresponding to the synthetic scan data (see Section 4.3.1) were obtained using the LMT trolley-mounted Velodyne Puck mapper shown in Figure 4.1(a). The GT motion data was computed using Hector SLAM [70] running on SICK 2D LiDAR data.

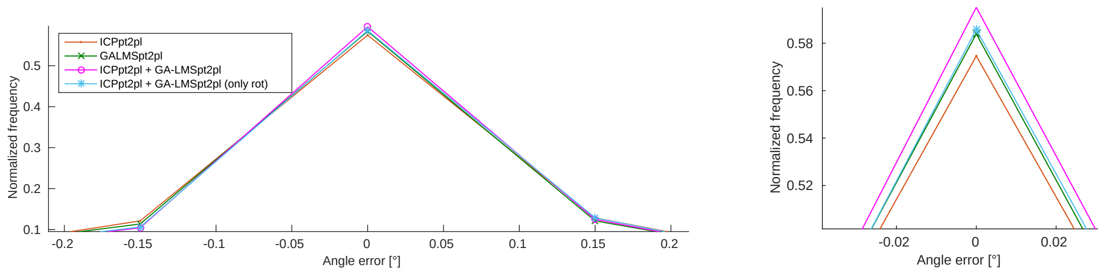
Scan matching using ICPpt2pl and GA-LMSpt2pl with skewing and curvature weighting was performed on the data. The GA-LMSpt2pl histogram of scan-to-scan angle estimation error, shown in Figure 4.10(b) shows that the GA-LMSpt2pl with curvature and skewing weights is only marginally better compared to the ICPpt2pl as opposed to the results with synthetic data shown in Figure 4.7. Nevertheless, there is an improvement. However, here again the trajectory "compression" due to separated rotation and translation alignment is observed.

These results call upon the combination of GA-LMSpt2pl with ICPpt2pl to benefit from the improved rotation estimation accuracy of the GA-LMSpt2pl while benefiting from the superior translation estimation capability of the ICPpt2pl. The results in Figure 4.10 show that the combination indeed results in the smallest scan-to-scan errors. Also the trajectory compression is significantly reduced. However, the centroid estimation seems to be undoing part of the translation estimation obtained from ICPpt2pl. Hence, in a final variant the ICPpt2pl is combined with a GA-LMSpt2pl that only refines the rotation. This combination has a better scan-to-scan angle error distribution compared to pure ICPpt2pl while having a trajectory that does not have the same compression problems as those of pure GA-LMSpt2pl.

The reason why the improvements here are not as significant as those observed in Figure 4.9 may relate to measurement noise. The results of Figure 4.9 were produced with the synthetic dataset with only skewing (no noise). Here, however, we are dealing with a real dataset that also includes real noise. While we verified that noticeable improvements are achieved in the synthetic dataset with noise and skewing, we note that the real sensor noise is not strictly Gaussian. Moreover, its variance is a function of the point distance and the ray incidence angle. Hence, the noise is stronger for far away points and for points with unfavorable surface normal. Accordingly, accounting for skewing may not deliver the expected benefit as the noise component may be more dominant especially for the points that are supposed to gain the most from skew-aware scan matching.



(a) Estimated trajectories (with zoom-in).



(b) Scan-to-scan rotation error histogram (with zoom-in).

Figure 4.10: Evaluation with real data. ICPpt2pl + GA-LMSpt2pl implies running ICPpt2pl and then refining the outcome with GA-LMSpt2pl. In the case of "only rot" the GA-LMSpt2pl only refines the rotation. The GA-LMSpt2pl includes curvature and skewing weights with $f_w(w_s, w_c) = max(w_s, w_c)$. (b) shows that the GA-LMSpt2pl in general has a better scan-to-scan rotation error distribution. (a) shows that a combination of the ICPpt2pl with a GA-LMSpt2pl that only refines the rotation alignment does not suffer from the trajectory compression problem. This combination has a better angle error distribution than plain ICPpt2pl as shown in (b). Notice, that improvements in rotation estimation do not necessarily translate to more accurate trajectories as the symmetric error histograms imply that errors cancel over time. Image source: [4] – ©2016 IEEE.

4.6 Summary

This chapter presents a study about the LiDAR scan skewing problem. To the best of our knowledge this is the first systematic study on this phenomenon. In the study the source of this phenomenon is first analyzed and its impact on scan matching is derived. The impact is then quantified using synthetic scan data with controlled distortions to verify the validity of the derivations and explanations. Finally, a new variant of the Geometric Algebra-LMS (Chapter 3) that incorporates the point-to-plane metric (GA-LMSpt2pl) was derived and used in conjunction with a skewing-aware iteration update scheme to reduce scan matching estimation errors due to skewing. Significant gains in terms of rotation estimation accuracy on synthetic data were achieved. Also experiments with real data were conducted. The results with real data show smaller improvements as compared to the synthetic data hinting there maybe other factors to consider in a further study.

Future developments could focus on developing the GA-LMSpt2pl to jointly estimate rotation and translation to reduce the underestimation of translation. Also, the sequential nature of the filter could be exploited to perform estimation using subsets of points employing convergence criteria and thus save unnecessary computations especially when it comes to correspondence estimation. This would be a major step towards what we call “Iterative-Iterative Closest Point”.

Major parts of this chapter were published in the proceedings of the Indoor Positioning and Indoor Navigation Conference 2016 (IPIN2016) [4].

Chapter 5

Decoupled Roto-Translation Alignment of Large-Scale Indoor Point Clouds

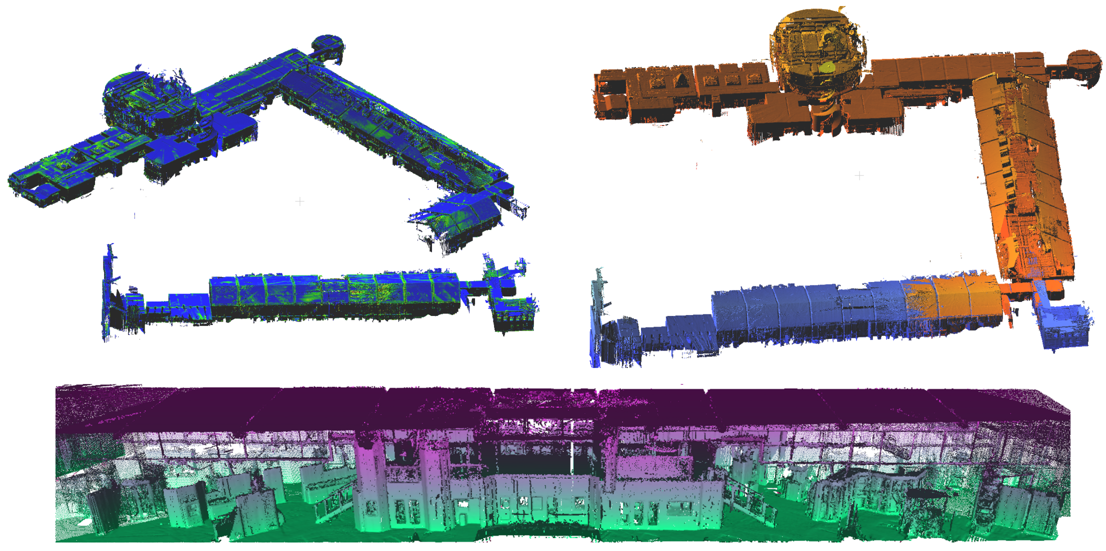


Figure 5.1: Top left: two large scale indoor 3D models with surface normal and curvature data and very small overlap (the curvature is color-coded). Bottom: cross-sectional view depicting the level of detail of one of the indoor models shown above. Upper right: the alignment computed using the proposed algorithm which is specifically designed to align indoor models and can handle very small regions of overlap. Image source: [2] – ©2016 Elsevier.

The digitization of indoor spaces in the form of 3D maps has progressed dramatically in recent years due to the development of accurate 3D range scanners and 3D sensors and the conception of efficient algorithms for on-the-fly stitching of individual scans into comprehensive 3D models [20], [67], [70], [71]. Many applications benefit from the ability to capture accurate 3D models in short time. They include indoor navigation (as we show in [7]), assistive robotics, facility management and building information management. Hence, especially large indoor spaces such as malls, museums and large company buildings are attractive for 3D mapping.

It is not customary to acquire one complete 3D model for a huge building. Limitations often require reconstructing multiple models that *overlap* at intersections such that they can be later aligned into one comprehensive model. Technical limitations include drift (see Chapter 4), memory and processing capacity or mobility constraints of the scanning arrangement (for example, wheel-based mapping platforms, as the one shown in Figure 2.1, cannot overcome steps). Non-technical limitations include building accessibility which manifests itself in intermittent scanning times. Also, scan updates may be necessary and it would be extremely cumbersome if entire buildings have to be remapped for that purpose.

There exists a body of research on the alignment of 3D models (see survey in Chapter 2). Existing approaches have been developed for general alignment problems of different 3D shapes. Different approaches excel at handling different alignment problems. However, their effectiveness quickly degrades when used for point cloud alignment of indoor models with small overlap. Moreover, huge point clouds quickly render many algorithms unusable unless a dramatic subsampling is performed which is undesirable as shape features may be lost. This motivates the development of an approach which exploits characteristics of indoor environments while remaining light-weight.

Starting with large-scale indoor 3D models this contribution presents an alignment approach that distinguishes itself by:

- handling very low overlap scenarios such as that shown in Figure 5.1,
- being invariant to initial alignment in 3D space,
- handling large-scale point clouds with millions of points efficiently.

To that end, prior knowledge about building interiors is exploited in the algorithmic design which allows loosening the coupling between rotation and translation estimation such that the high-dimensional estimation problem is broken down into multiple lower dimensional problems. Each is solved using strong cues that are inherent to the data. As shown in Figure 5.2, the point clouds are first aligned to gravity (represented by the z-axis throughout the chapter) using PCA (Section 5.2.1). Then their floors are aligned such that the translation along the z-axis is recovered (Section 5.3.1). Subsequently, the point clouds are aligned rotationally in the ground-plane using histogram signatures of surface normal directions (Section 5.2.2). Finally, the translation in the ground-plane is recovered using what we call *Curvature-Pair Shift Histograms* (CPSH) which we show can be interpreted as probability distributions over the translation space (Section 5.3.2). In Section 5.4, we explain the probabilistic framework built up from CPSHs computed on segment pairs of point clouds to explicitly handle cases of very low overlap. The results in Section 5.6 demonstrate the superiority of this decoupled sequential approach.

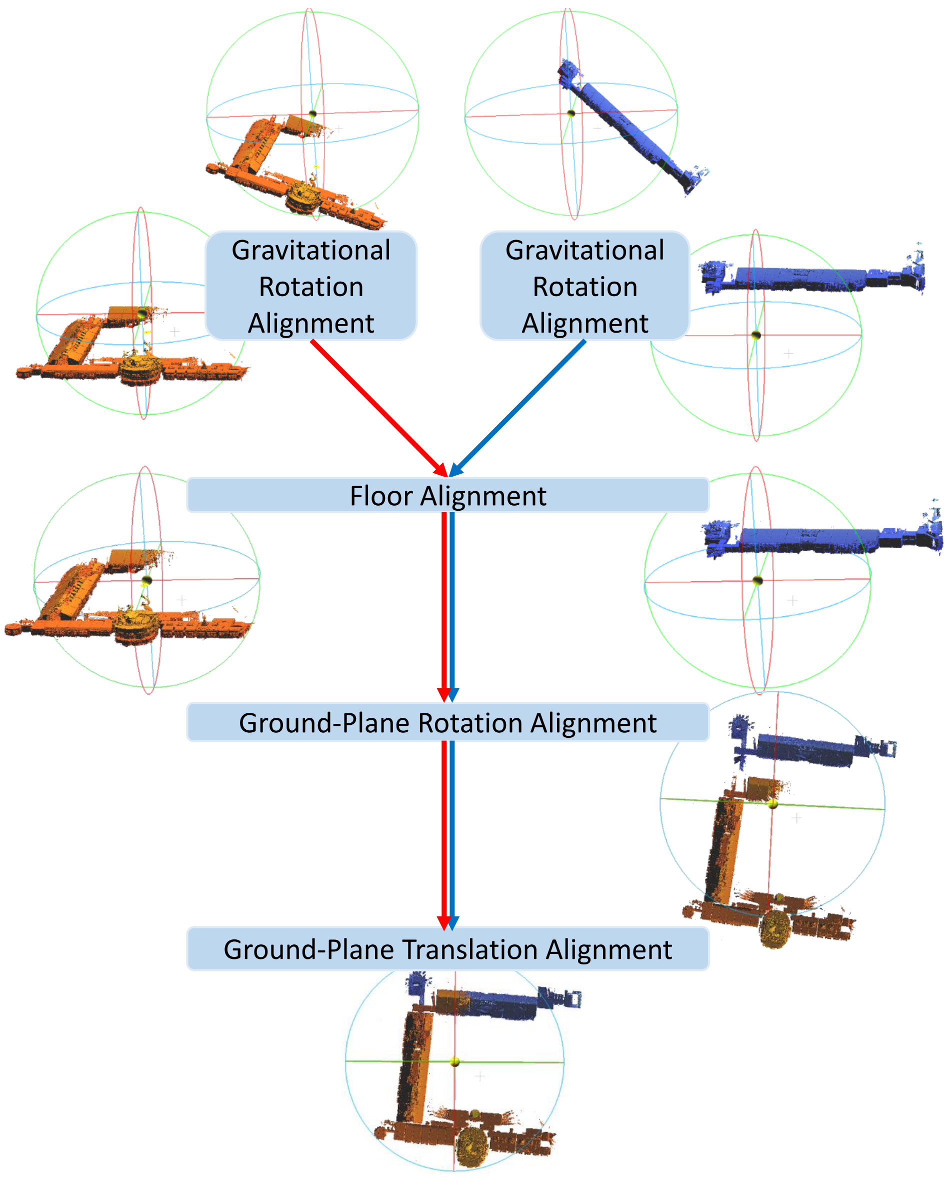

Gravitational Rotation Alignment
Gravitational Rotation Alignment
Floor Alignment
Ground-Plane Rotation Alignment
Ground-Plane Translation Alignment

Figure 5.2: Proposed large-scale point indoor point cloud alignment system. The alignment is broken down into simplified problems which can be solved using strong features inherent to the data. An algorithmic summary in the form of pseudo-code is provided in Algorithm 1 (p.70). Image source: [2] – ©2016 Elsevier.

5.1 Background

Only coarse registration methods are valid candidates for the problem at hand as no requirements are posed on the initial alignment.

As already explained in Chapter 2, different categories of coarse point cloud alignment methods exist. One category includes voting-based algorithms, including [11], [30]–[32], in which the 6DOF parameter space is quantized and exhaustive point-triplet alignment is used to vote in the parameter space. These methods suffer from inefficiency due to their brute force nature. Moreover, with growing dimensions the votes become increasingly sparse requiring many more votes to sufficiently populate the space (curse of dimensionality). The second category involves computing point correspondences and using them to estimate the transformation that best maps the source points to their respective target points as explained in [14] (see "coarse alignment"). The feature matching of local features (such as [33]–[35]), which is essential for establishing correspondences, requires articulated shapes to generate distinctive descriptors. Also, keypoint detection and descriptor computation are usually computationally demanding since each requires repetitive nearest-neighbor (NN) searches leading to registration times in the order of 10s of seconds for point clouds with millions of points with an optimized C++ implementation (See Figure 5.18). This is due to the fact that any point's index inside the point cloud does not convey any information about its proximity to other points.

The relatively new approach of *Four-Point Congruent Sets* (4PCS) [41] approaches the alignment problem from the perspective of the Largest Common Point set (LCP) problem. The alignment seeks the solution for which the largest subset of the source cloud is maximally δ units away from the target after transformation. It is solved by establishing correspondences of sets of four points (bases) (see Section 2.2.2.3). The correspondence computation is relatively slow when dealing with large-scale point clouds. Mellado et al. accelerated the 4PCS algorithms creating the Super4PCS (S4PCS) [42]. It is used in the comparisons given its efficiency and because indoor models have walls which are suitable for generating 4-point congruent sets. The results in Section 5.6.5 show that the latter property isn't always beneficial especially when dealing with datasets of small overlap and large perpendicular corridors as is common in human made structures.

Alignment using probabilistic frameworks constitutes yet another category. Probabilistic approaches avoid establishing hard correspondences and rather establish a "soft" relationship between point pairs described by probabilities. Two notable examples for probabilistic alignment were presented in Section 2.2.2.4, namely GMMREG [43] and CPD [13]. As already explained, probabilistic algorithms have a high degree of flexibility in dealing with different transformations including reflections, affine transformation and scaling. These properties, however, are hardly of interest to the problem at hand since we deal with rigid-body transformation estimation in 3D space. Their iterative nature means that they are sensitive to initial alignment as opposed to ours as will be shown in Section 5.6.4. Significant computational complexity is associated with the employed solvers leading to very high computation times especially for large datasets as demonstrated by the results in Section 5.6.6. Nevertheless, they remain inspirational as explained in Section 5.4.

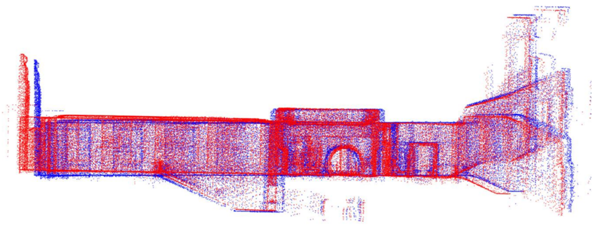
The mentioned works present algorithms suitable for general purpose alignment problems by cleverly including all degrees-of-freedom in one estimation problem. However, they frequently fail on our large-scale datasets and/or are not very efficient as seen in Section 5.6. The combination of large-scale data and small overlap while requiring insensitivity to initial alignment makes joint 6DOF estimation error-prone and computationally demanding. Meanwhile, the availability of strong cues that help in solving individual estimation problems, as shown in this contribution, calls upon breaking down the estimation problem into separate estimations of reduced complexity with smaller parameter spaces and high success probability. Hence, like the works of [72]–[74], we focus on prior knowledge related to the application. We design an approach that partly decouples the rotation and translation estimation by strongly exploiting the characteristics of indoor spaces. For the ground-plane translation estimation inspiration is drawn from CPD while avoiding its drawbacks which are lack of efficiency, sensitivity to initial alignment and limited robustness towards overlap. The proposed algorithm is particularly robust on Manhattan datasets, which is a common assumption in 3D mapping and reconstruction (see [74]–[77]). Yet when and how it also works on non-Manhattan cases is explained and demonstrated with an example.

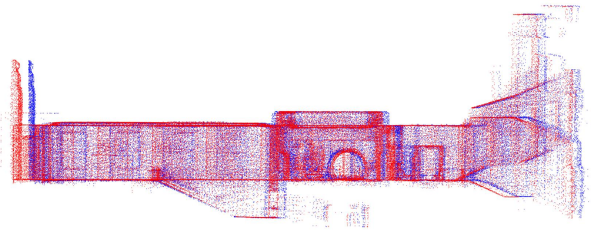
5.2 Rotation Alignment

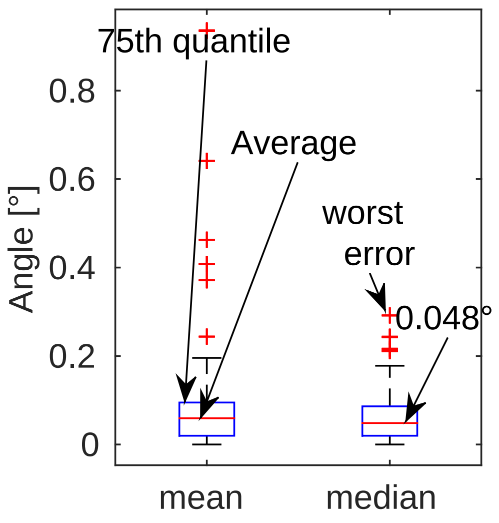
We propose a decoupled approach to 3D rotation alignment which solves the rotation in two steps. We first propose a method to align indoor building point clouds to gravity using Principal Component Analysis (Section 5.2.1). We then introduce a method to compute in-ground rotation signatures from surface normals to solve the ground-plane alignment (Section 5.2.2).

5.2.1 Gravitational Rotation Alignment

In a first step, the source and target point clouds are rotationally aligned to gravity such that walls become parallel to the z-axis. Note here that the coordinate variation for large flat buildings is smallest along the vertical direction of the walls. Accordingly, Principal Component Analysis (PCA) is first performed to align the third principal component to the global z-axis. This simple approach is very reliable and fast. Since after this step there is a small residual error left (see Figure 5.3(a)), we choose all surface normals which have an angle $< \phi_{n_z}$ w.r.t. to the global z-axis (i.e. the floor normals) and compute their median vector. The latter is used to compute a further refinement of the alignment such that the median normal is aligned to the global z-axis (see Figure 5.3(b)). This approach results in very accurate rotation alignment w.r.t. to gravity. The results in Figure 5.3(c) show that the mean angle error is $\approx 0.05°$while the worst error is $< 0.3°$.


(a) After PCA


(b) After refinement



(c) Error statistics

Figure 5.3: Gravitational rotation alignment. (a) shows the alignment by simply computing the PCA of the point cloud and aligning the third principal component to z. (b) is the outcome after the refinement achieved by choosing the surface normals that belong to the floor and aligning the median vector to the z-axis. (c) shows statistics of the angle between the global z-axis and the building's (local) z-axis after alignment. The statistics are computed by pre-rotating a gravitationally aligned building point cloud with a random rotation around the three axis before attempting gravitational alignment. 10 random rotations per point cloud were simulated. 27 different point clouds were involved (270 data points). Using the median vector instead of the mean vector in the refinement step results in more accurate gravitational alignment with less than 0.05°average angle error. Image source: [2] – ©2016 Elsevier.

5.2.2 Ground-plane Rotation Estimation

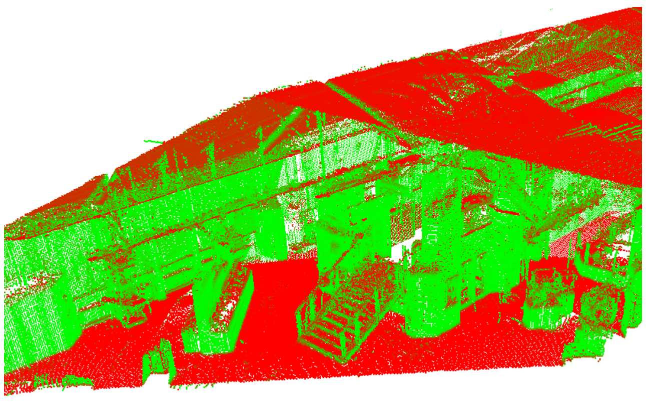
Starting with two point clouds that are rotationally gravitationally aligned (see Section 5.2.1) we can rely on a primitive point-wise feature: the surface normal. We observe that the walls and the dominant structures often deliver very reliable references for ground-plane rotation alignment. Hence, we propose to compute histograms of normal directions arguing that the resulting signature is similar up to a cyclic shift. Only points whose surface normals are nearly parallel to the ground plane (Figure 5.4(a)) are considered. For each considered surface normal $\mathbf{n}_k = [\mathbf{n}_k(x)\ \mathbf{n}_k(y)\ \mathbf{n}_k(z)]^T$ ($\|\mathbf{n}_k\| = 1$), the ground-plane angle ϕ_k is computed as:

$$\phi_k = \text{atan}(\frac{\mathbf{n}_k(y)}{\mathbf{n}_k(x)}). \tag{5.1}$$

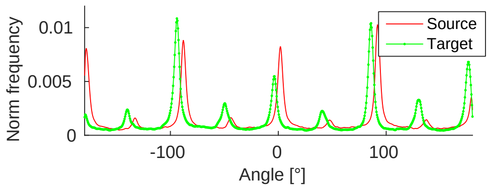
Then, a rotation signature is computed by forming the histogram of these angles $\mathbf{h}_s$ with v equidistant bin centers using Kernel Density Estimation (KDE) (Figure 5.4(b)). This process is repeated for the target cloud producing $\mathbf{h}_t$. Finally, a cyclic cross-correlation is performed using the signatures to estimate the rotation $\hat{\theta}$ in the x-y plane (Figure 5.4(c)). Hence,

$$\hat{\theta} = \underset{l}{\operatorname{argmax}}\, C_{xcorr}(l; \mathbf{h}_s, \mathbf{h}_t) \cdot 360/v \,. \tag{5.2}$$

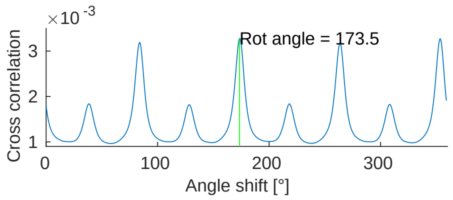
C_{xcorr} is the *cyclic* cross-correlation which is a function of the shift parameter l $(0 \leq l < v)$.



(a) Contributing points (green)



(b) Rotation signatures



(c) Cross-correlation

Figure 5.4: Using the normal direction histogram signature to recover the rotation for the example in Figure 5.1. In (a), the subset of points which are chosen to contribute to the signature computation of (5.1) are shown. They constitute the subset of points whose normals are nearly parallel to the ground (x-y) plane. In (b), the signature resulting from (5.1) is shown. In (c), the cyclic cross-correlation of the source and target signatures is plotted. The peak determines the rotation alignment. The resulting rotation alignment is correct as seen in Figure 5.10(c), however, a slightly different parameter setting might have produced a stronger peak at one of the other *dominant* directions which are 90°apart. Resolving this ambiguity is explained in Section 5.3.3 and Section 5.4.5. Image source: [2] – ©2016 Elsevier.

Ground-plane rotation alignment works even in the case of very low overlap shown in Figure 5.1. It is easy to see that for Manhattan datasets (which we *don't* assume) it can work even when there is no overlap at all. However, from Figure 5.4(c), the reader might rightfully conclude that an ambiguity by multiples of 90°exists since typically 4 large peaks exist in the cross-correlation due to the fact that most buildings have walls that are perpendicular to one another covering 4 dominant directions. The ambiguity is resolved in combination with the subsequent ground-plane translation alignment using CPSHs as will be explained in Section 5.3.3 and Section 5.4.5. When dealing with non-Manhattan datasets, on the other hand, this ambiguity does not occur. Non-Manhattan datasets can be supported since cross-correlation is used instead of explicit detection of dominant directions (see [78]). As an alternative one may do this explicit detection of dominant walls in the context of a "mixture of

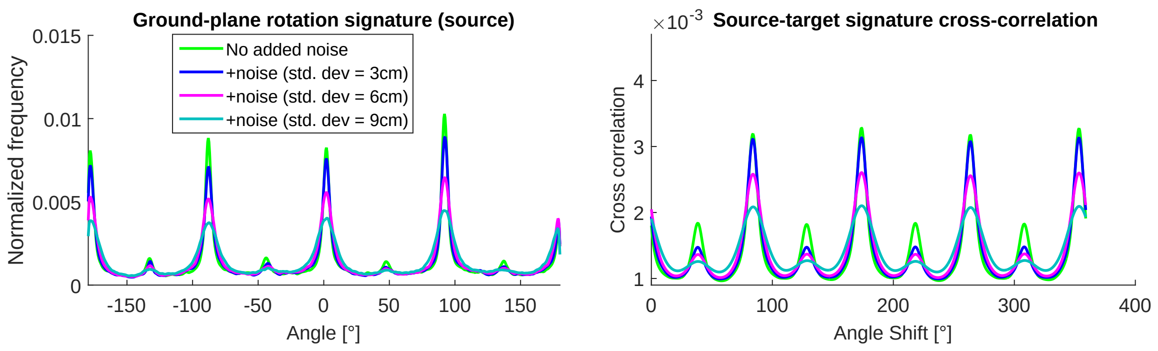


Figure 5.5: Experimental evaluation of the robustness of the surface normal direction histogram signature used in the ground-plane rotation alignment. Random Gaussian noise is added to the source-target pair shown in Figure 5.1. Subsequently, the normals are re-estimated and finally the ground-plane rotation alignment is reattempted. The rotation signature exhibits peaks in the same bins irrespective of the noise intensity. Accordingly, the cross-correlation attains peaks at the same rotation shifts. Image source: [2] – ©2016 Elsevier.

Manhattan frames" [79]. However, one would need to establish Manhattan-frame correspondences to solve the rotation. It remains to be said that when dealing with non-Manhattan datasets our proposed approach needs good overlap to guarantee that the signatures have similar distinctive parts matchable by cross-correlation as seen in Figure 5.19.

Ground-plane rotation alignment using the normal direction histogram signature is very robust because it depends on dominant structures found in all buildings, namely the walls. Since the normals can be computed reliably on planar surfaces, the resulting signature is very robust towards noise. To verify that, the following experiment is performed: Gaussian noise is added to the source-target point cloud pair shown in Figure 5.4 first. Then the normals are re-estimated. Finally, the ground-plane rotation alignment is performed. The results in Figure 5.5 clearly show that adding noise with a standard deviation as high as 9 cm can be handled. Even though the surface normals become noisy (seen through the widening of the peaks) the histogram effectively averages out the errors and maintains the peak locations.

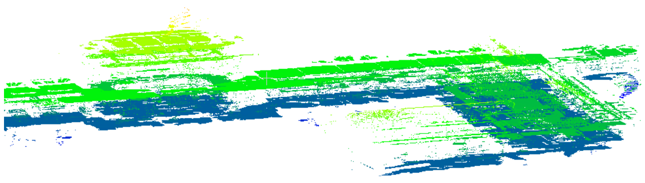
5.3 Translation Alignment

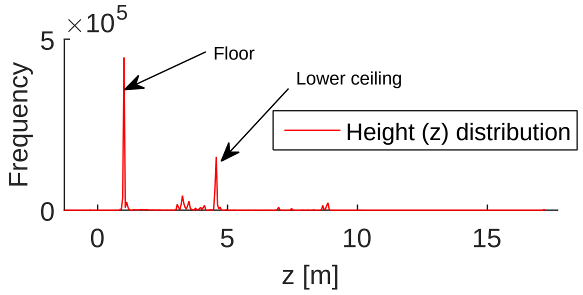
Once the source and target have been aligned in terms of rotation in 3D, translation alignment commences. Again, we decouple alignment along gravity from the ground-plane alignment. For the alignment along gravity we propose a method that exploits point distributions in building interiors (Section 5.3.1). Following which translation alignment in the ground-plane is performed using our newly introduced Curvature Pair Shift Histograms (Section 5.3.2).

5.3.1 Floor Alignment

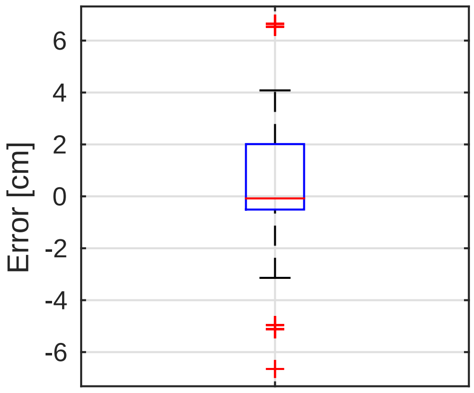
Here again, we propose to use a very reliable feature which is a dominant characteristic of indoor point clouds: the floor and the ceiling. More specifically, the histogram of z values (Figure 5.6(b)) for points with normals nearly parallel with the z-axis (Figure 5.6(a)) is computed. We cross-correlate the signatures to retrieve the shift along z. Notice this is more robust than simply taking the location of the strongest peak as sometimes datasets may have multiple floors/ceilings (see example in Figure 5.20). Since the histograms are invariant towards rotation in the ground-plane, this step can already be performed before the ground-plane rotation alignment - as suggested in Figure 5.2).

The results in Figure 5.6(c) show that the alignment along gravity error evaluated on many point clouds averages around 0 cm and is maximally 7 cm.


(a) Contributing points (colored by height)



(b) Height signature



(c) Error statistics

Figure 5.6: The floor of rotationally aligned PCDs can be identified through a histogram analysis of the z-values of their points. First, only the subset of points with normals nearly parallel to gravity (z) are extracted as shown in (a). Subsequently, the histogram of their z-values is computed. The histogram signatures of a source and target pair are cross-correlated to detect the common floor which is identified in the signature shown in (b). This is more robust than simply taking the dominant peak with the least z-value as some datasets have multiple floors and ceilings that may confuse the detection. To analyze the accuracy of this approach an experiment similar to the one in Figure 5.3(c) is performed wherein a point cloud is pre-rotated and translated along z randomly and then gravitational rotational alignment followed by floor estimation is conducted (here we simply take the dominant peak with the smallest z-value to be the floor). This is repeated 10 times per point cloud and performed on 27 different point clouds. The mean and maximum errors, as shown in (c), are around 0 cm and 7 cm, respectively. Notice gravitational rotation alignment is also involved in this experiment as its success is a pre-requisite for a successful floor alignment. Image source: [2] – ©2016 Elsevier.

5.3.2 Ground-plane Translation Alignment

To estimate the ground-plane translation we propose the Curvature-Pair Shift Histogram (CPSH). The idea and motivation for using CPSHs is first provided in Section 5.3.2.1. Section 5.3.2.2 introduces a series of techniques for discriminative voting that we propose to make CPSHs far more effective and fast. Finally, we provide some insights and further thoughts that help the reader understand what inspired us to propose CPSHs for ground-plane translation alignment in Section 5.3.2.3.

5.3.2.1 Proposed Curvature Pair Shift Histogram (CPSH) - Motivation and Idea

To compute the translation in the ground-plane it is possible to compute complex shape features, such as [35], and match them to establish correspondences. However, this is intentionally avoided due to its computational requirements and lack of robustness when structures are ambiguous (such as repetitive structures which are characteristic of indoor spaces).

Since the rotation has been estimated (assuming the explained ground-plane rotation ambiguity has been resolved), the difference in coordinate value along all dimensions for *corresponding* point pairs is constant. However, we do not know the point correspondences. Instead of computing and matching features, we propose to compute the x and y translations from each source point to each target point. Then a bi-variate translation histogram of the x-y translation values is computed. The correct translation should get as many hits as there are truly corresponding source-target point pairs. All other false correspondences, on the other hand, will distribute their votes on the entire translation space ($\mathcal{D}$) with a low probability of voting highly for a *certain* wrong translation. This is illustrated in Figure 5.7.

5.3.2.2 Discriminative Voting using Primitive Features

Computing the CPSH for a source-target cloud pair has complexity $O(|\mathcal{S}| \cdot |\mathcal{T}|)$ where $\mathcal{S}$ and $\mathcal{T}$ are again the source and target point sets, respectively. For large-scale indoor point clouds this can quickly lead to high computational times. Moreover, taking every source-target point pair means more wrong translations are binned in the histogram increasing the possibility that a wrong translation gets a higher probability value than the true one. Hence, we seek to minimize the computational burden while also increasing the distinctiveness of the correspondence choices.

At this point we note that, in contrast to ground-plane rotation estimation (Section 5.2.2), points on corners are more important for the estimation than those on planes (consider the ambiguity introduced by a source point pairing with target points lying on a wall). We propose to use the readily available curvature values (which are a byproduct of surface normal estimation) to consider only points with a curvature value above a certain threshold, thus

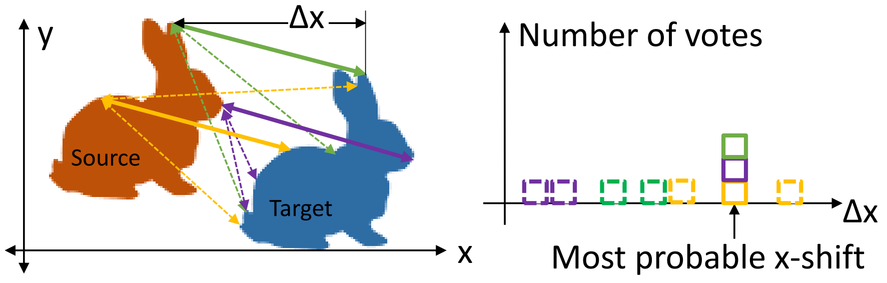


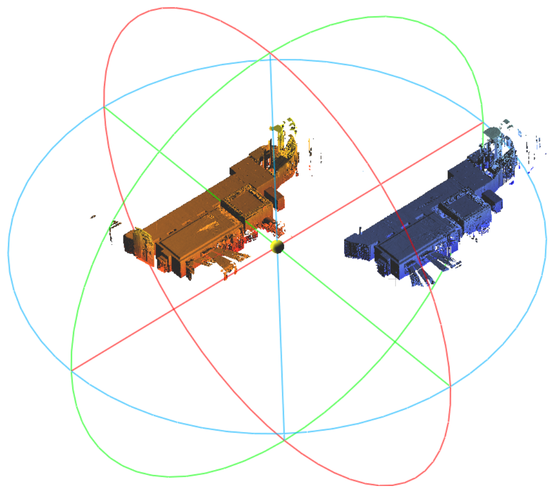
Figure 5.7: Curvature-Pair Shift Histogram (CPSH) for ground-plane translation estimation. Starting with rotationally aligned point clouds the translations of any source point (with high curvature) to different target points (with high curvatures) are computed and binned in a histogram. This process is repeated for different source points (different colors). The translations between truly corresponding point pairs (connected by solid lines) will be binned to the same bin and thus accumulate. The translations between non-corresponding points (dashed lines) distribute over the translation space. A normalized CPSH can be considered a probability mass function (PMF). Accordingly, the strength of each bin can be considered a probability that the respective translation is the alignment translation. Notice, the example shows the CPSH just for the x-axis. In practice we compute the bi-variate CPSH for x and y jointly as a 1D histogram is equivalent to a marginalized 2D histogram with lost information. After all there is valuable information in the co-occurrence of shifts. Image source: [2] – ©2016 Elsevier.

the name “curvature-pair”. This already dramatically decreases the number of points that contribute to the CPSH.

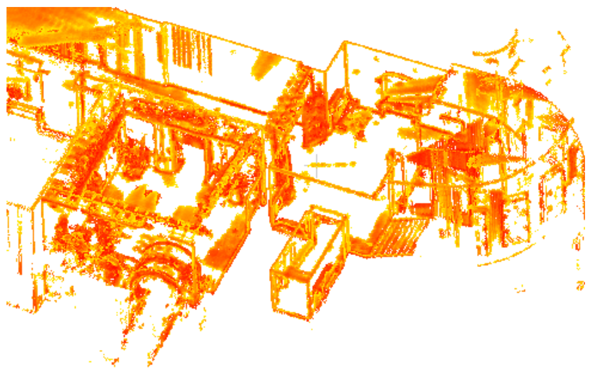
The voting process is furthermore improved by using the surface normal direction as yet another primitive feature. It’s exploited by only allowing source-target point pair $\{\mathbf{p}_i^s, \mathbf{p}_j^t\}$ to vote if their normals have a small dihedral angle, i.e. $(\mathbf{n}_i^{cs})^T \mathbf{n}_j^{ct} < cos(\epsilon_\phi)$ (the superscripts s and t refer to source and target, respectively). Hence, only point pairs whose normals point in almost the same direction are formed. Figure 5.8(d) shows how this selection mechanism leads to a dramatic improvement in the CPSH, significantly amplifying the true translation peak.

To make the voting process for the CPSH even more discriminative, another primitive point-wise feature can be exploited: The height value (z-coordinate in our data). Since at this point the point clouds will have been aligned along the height dimension, corresponding points will have almost the same height. A certain threshold has yet to be used to consider small misalignment errors (see Figure 5.6(c)). Using this approach, $\{\mathbf{p}_i^s, \mathbf{p}_j^t\}$ can form a pair only if $|\mathbf{p}_i^s(z) - \mathbf{p}_j^t(z)| < \epsilon_z$. Figure 5.8(e) shows further significant improvement in the CPSH once incorporating this selection mechanism as well.

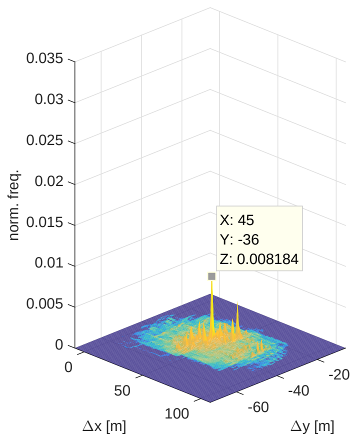
The collection of these selection mechanisms not only improves the CPSH quality dramatically but also reduces the computation time by more than 95% as the number of voting point pairs is significantly reduced.



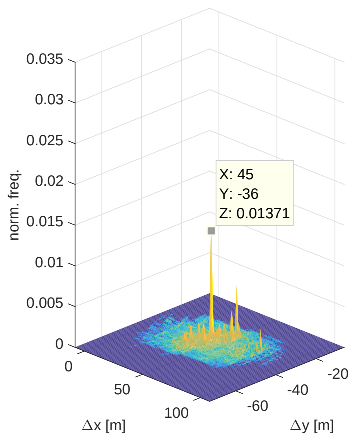
(a)



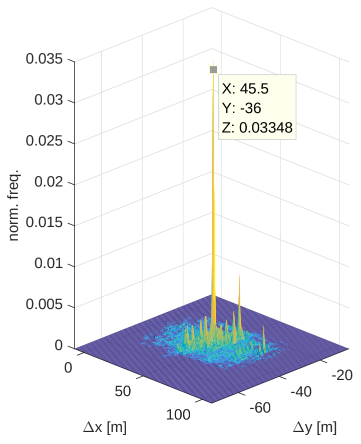
(b)



(c)



(d)



(e)

Figure 5.8: Enabling discriminative voting to compute more discriminative CPSHs and more efficiently. In 5.8(a) a source and target cloud pair with high overlap is shown after rotation alignment and floor alignment. In (b) the subset of source cloud points with high curvature values are shown. Different normalized CPSHs are shown in (c), (d) and (e). (c) shows the normalized CPSH obtained when letting every point pair with high curvature values vote in the CPSH. The maximum probability occurs at the right translation. In (d) the bin strength at the translation is significantly amplified compared to spurious peaks when allowing only source-target point pairs with a similar normal to vote ($\epsilon_\phi = 30°$). In (e) the peak at the right translation is even more dominant by adding a height constraint allowing only points with a height value that is equal up to a threshold $\epsilon_z = 10\,\text{cm}$ (set based on the height alignment error statistics of Figure 5.6(c)) to vote. Image source: [2] – ©2016 Elsevier.

5.3.2.3 CPSH Insights

The choice for the CPSH bin size affects the alignment accuracy as well as the robustness to noise. Coarser bins are more memory efficient and make the CPSHs more robust to noise while reducing the alignment accuracy.

Using CPSHs is similar to a voting-based alignment scheme (see Section 2.2.2.1). However, the decoupling of rotation and translation means that no longer three point pairs are required to cast a single vote as is the case in joint roto-translation estimation. Hence, the voting complexity sharply reduces from $O(|\mathcal{S}|^3 \cdot |\mathcal{T}|^3)$ to just $O(|\mathcal{S}| \cdot |\mathcal{T}|)$ where $\mathcal{S}$ and $\mathcal{T}$ are the source and target point sets, respectively. This is very beneficial given the large-scale point clouds the algorithm is designed for.

The CPSH effectively uses *soft correspondences* (see Section 5.1). The coincidence of a certain shift obtained from different point pairs re-enforces the belief that a particular ground-plane translation is the true translation.

Normalized versions of the CPSHs whose area under the surface is 1 can be considered bivariate probability mass functions for the translation. Whenever speaking about probabilities it shall implicitly refer to normalized CPSHs. Normalized CPSHs interpreted as probability mass functions and computed from segment pairs of the source and target clouds are used in Section 5.4 to pose the translation estimation problem as a maximum likelihood problem.

5.3.3 Resolving Ground-Plane Rotation Ambiguity I

In Section 5.2 the problem of ground-plane rotation ambiguity by multiples of 90°was discussed. Considering CPSHs, it is easy to see that using the wrong rotation would disallow true correspondences from voting at the same bin. Accordingly, it is expected that the CPSHs for the three other dominant directions other than the correct one are likely not to have the same dominant peak. Indeed, Figure 5.9 shows that the CPSHs of the example of Figure 5.8 evaluated at the three other (wrong) rotations are less peaky than the one obtained at the correct rotation.

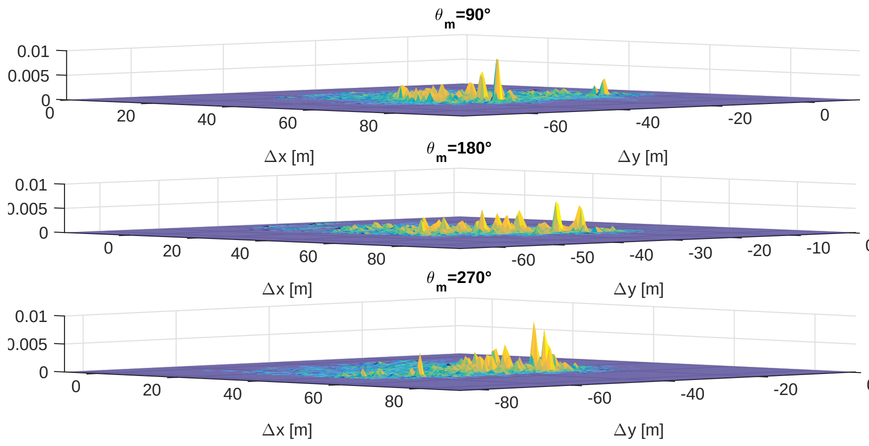


Figure 5.9: The normalized CPSHs for the example of Figure 5.8 evaluated at the three other dominant rotations (see Section 5.2) with the same parameters used for 5.8(e). Each of the dominant rotations is obtained by adding θ_m to the correct ground-plane rotation. Clearly, none contains the same dominant peak as Figure 5.8(e). Compare also the maximum probabilities to the one of Figure 5.8(e). The wrong ground-plane rotation means true correspondences will not be binned similarly. Image source: [2] – ©2016 Elsevier.

5.4 Segment-Based Alignment

The previously explained methods for rotation alignment and floor alignment are highly robust towards small overlap. The same argument does not necessarily apply to the ground-plane translation alignment as it needs enough overlap to generate an unambiguous CPSH. In this section we explain how to use CPSHs in a segment-based ground-plane translation alignment approach to handle low overlap cases. The underlying idea is to segment the source and the target into multiple point clouds such that certain (unknown) segments will have significant overlap and thus produce good CPSHs with clear identification of the true translation. In the simplest case, the alignment with the highest score can be chosen. However, we present here an approach where different alignment hypothesis are fused such that geometric consistencies help to reinforce the true hypothesis and invalidate false hypotheses.

The proposed approach uses a probabilistic formulation (Section 5.4.3) that allows posing the translation estimation problem as a Maximum Likelihood (ML) problem (Section 5.4.4) which is extended to explicitly account for non-overlaps (Section 5.4.7) and applied on an example case in Section 5.4.8. First, however, the motivation to consider a segment-based translation alignment is provided (Section 5.4.1) together with a brief review of the Coherent Point Drift (CPD) algorithm (Section 5.4.2) from which we draw some inspiration.

5.4.1 Motivation

The ground-plane translation alignment using the CPSHs explained in Section 5.3 exhibits some robustness towards varying degrees of overlap. The translation alignment example of Figure 5.10(a) results in a CPSH whose maximum unambiguously occurs at the true translation despite significant non-overlapping regions. This is mainly attributed to the discriminative voting explained in Section 5.3.2.2. The example shown in Figure 5.10(c) is far more challenging as the overlap between the source and target clouds is much smaller. Outlier peaks dominate the CPSH. Accordingly, using the translation value obtained at the maximum CPSH point would lead to an erroneous result. In the remainder of this section we show how this problem can be circumvented using a segment-based alignment approach with an underlying probabilistic modeling that borrows ideas from CPD.

5.4.2 Connection to Coherent Point Drift (CPD)

Given that CPSHs represent probability mass functions it appears logical to draw inspiration from probabilistic registration algorithms. Recall from Section 2.2.2.4 that CPD is a probabilistic registration algorithm. It considers the source point cloud as a set of points that have been drawn from a GMM distribution fitted on the target point cloud. The alignment is recovered by maximizing the likelihood of the source points conditioned on the parametrized

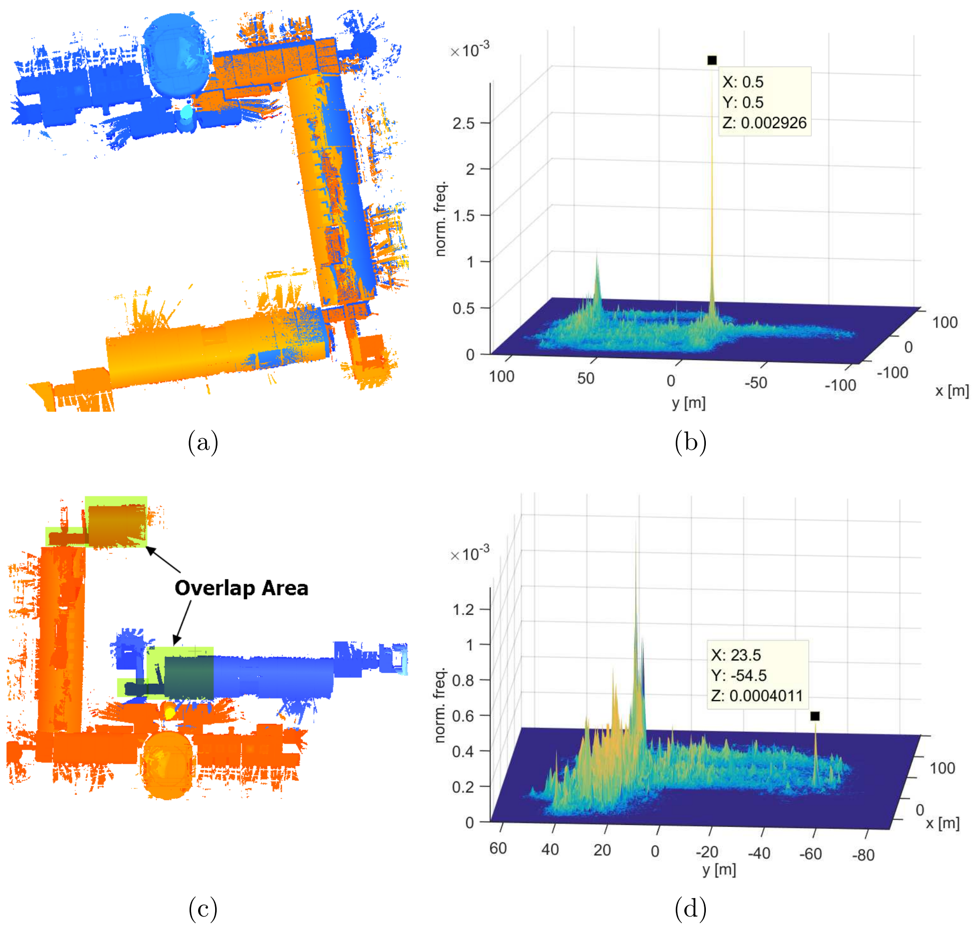


Figure 5.10: The source and target cloud pair (rotationally aligned as explained in Section 5.2) and the corresponding CPSH are shown for an example with substantial overlap and one with a very small overlap in figures (a)-(b) and (c)-(d), respectively. For the large overlap case the peak in the CPSH is unambiguous and at the true translation (seen by the fact that the translation misalignment is minimal and hence a peak near $[0\ 0]^T$ is expected). For the very small overlap case, the bin strength at the true translation (identified by the data tip) is significantly lower than other outlier peaks. Regions not overlapping contribute point pairs which vote for outlier translations thus increasing false peaks. In the very low overlap case this leads to false translation hypotheses dominating the CPSH. Image source: [2] – ©2016 Elsevier.

GMM. The solution involves iterating over the following main steps: 1)Evaluate a Gaussian function for each source-target point pair $\{\mathbf{p}_i^s, \mathbf{p}_j^t\} \in \mathcal{S} \times \mathcal{T}$; 2)Update the transformation parameters. The first step represents the bottleneck of computations for CPD. It involves evaluating $|\mathcal{S}| \cdot |\mathcal{T}|$ Gaussians and is repeated in every iteration which for our large-scale point clouds with millions of points translates into slow convergence. The used EM algorithm does not guarantee hitting the global optimum [45] which means CPD is sensitive to the initial alignment, as we further demonstrate in Section 5.6.4. Finally, the point-wise relationship given by the Gaussian function does not involve the local geometry which provides vital clues about how associable two points in the source and target are.

Consider now drawing a source segment (instead of a source point) from a target segment (instead of a target point) and shifting it by a certain translation. The CPSH value at the translation can be interpreted as the probability analogous to that of drawing a source point from a target point in CPD. However, now the probability value will have been computed from a collection of points and is thus based on more shape information than the simple point-to-point relationship of CPD. In replacing the Gaussian function in (2.8) with the CPSHs we can fuse multiple CPSHs (as opposed to greedily choosing the best) to support low overlap cases while remaining efficient and initial alignment insensitive. Like CPD, our approach uses a mixture probability model as a basis for ML filtering but is fundamentally different in defining the components of the mixture model which are the segments. This means that there are far less mixture components in the mixture leading to a substantial reduction in complexity. Also the probabilities are pre-computed over the entire transformation space in non-parametric mass functions which, as we show later, mean that no longer an iterative and initialization-sensitive solver is required. The details are explained in Section 5.4.3.

5.4.3 Probabilistic Formulation

Assuming we can segment the source cloud $\mathcal{S}$ and the target cloud $\mathcal{T}$ into (possibly overlapping) segment sets $\mathcal{P}_\mathcal{S}$ and $\mathcal{P}_\mathcal{T}$, respectively, such that

$$\mathcal{P}_\mathcal{S} \quad = \{\mathcal{S}_1, \mathcal{S}_1, .., \mathcal{S}_N\}, \qquad \mathcal{P}_\mathcal{T} \quad = \{\mathcal{T}_1, \mathcal{T}_2, .., \mathcal{T}_M\}, \tag{5.3}$$

where the point cloud segments $\mathcal{S}_i$ and $\mathcal{T}_j$ satisfy

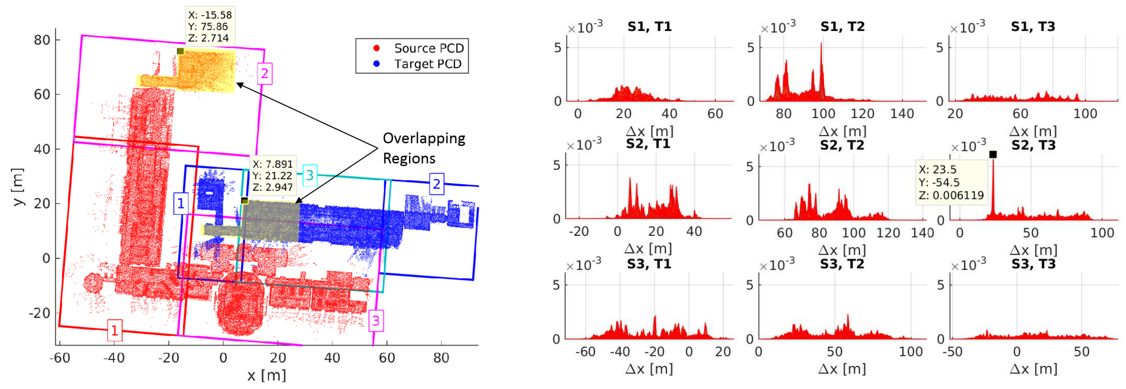
$$\mathcal{S} = \bigcup_{\mathcal{S}_i \in \mathcal{P}_\mathcal{S}} \mathcal{S}_i, \qquad \mathcal{T} = \bigcup_{\mathcal{T}_j \in \mathcal{P}_\mathcal{T}} \mathcal{T}_j\,. \tag{5.4}$$

We can now compute the CPSH (see Section 5.3) for each source-target segment pair $\{\mathcal{S}_i,\mathcal{T}_j\}$ which results in the CPSH function $f_{i,j}(\mathbf{t})$. An example segmentation with corresponding CPSHs is shown in Figure 5.11.

Now, consider $f_{i,j}(\mathbf{t})$ to be the probability of observing source segment $\mathcal{S}_i$ at location $\mathbf{t} \in \mathcal{D}$ (remember, $\mathcal{D}$ is the quantized translation space) given it was drawn from target segment $\mathcal{T}_j$, i.e.:

$$p(\mathcal{S}_i, \mathbf{t}|\mathcal{T}_j) \triangleq f_{i,j}(\mathbf{t}). \tag{5.5}$$

In using the analogy with CPD, $p(\mathcal{S}_i, \mathbf{t}|\mathcal{T}_j)$ can be understood as the probability that an observed source segment i in reality (perhaps partially) overlaps the target segment j and is translated by $\mathbf{t}$. One might argue that since the *observed* $\mathcal{S}_i$ is assumed to be sampled from $\mathcal{T}_j$ then $\mathbf{t}$ can be directly inferred. However, since it is not known out of which part of $\mathcal{T}_j$ $\mathcal{S}_i$ was cutout, $\mathbf{t}$ remains to be estimated.



(a) Segmentation of rot. aligned clouds. (b) Segment pair CPSHs (vertical axis = norm. frequency).

Figure 5.11: CPSHs computed from source-target segments. The segments of the point clouds shown in Figure 5.10(c) have been computed using the algorithm described in [2, Appendix B] and are identified by the bounding boxes in (a). The two data tips identify a corresponding point pair illustrating that the correct translation is $\mathbf{t} = [23.5 \quad -54.7]^T$. The resulting CPSHs for each source target pair are shown in (b). For illustration purposes, the 2D normalized histograms are shown from a side view. The CPSH for source segment 2 (S2) with target segment 3 (T3) exhibits the highest probability at the true translation (as seen through the data tip) and the peak far dominates all others in the histogram (what matters is the *actual* point cloud overlap and not the bounding-box overlap). None of the other CPSHs exhibits a similarly dominant peak since all other segment pairs, apart from S2-T1, have an overlap and hence the mismatch in structures disallows building an unambiguous CPSH. Judging based on bounding boxes, S2-T1 in a first glance seems to have a large overlap. However, notice that S2 and T1 do not actually share many points and hence fail to identify the true translation. Image source: [2] – ©2016 Elsevier.

Inspired by CPD, we now describe $p(\mathcal{S}_i, \mathbf{t})$ by a mixture model which accounts for $\mathcal{S}_i$'s relation to each target segment $\mathcal{T}_j$ as follows:

$$p(\mathcal{S}_i, \mathbf{t}) = \sum_{\mathcal{T}_j \in \mathcal{P}_\mathcal{T}} p(\mathcal{S}_i, \mathbf{t} | \mathcal{T}_j)\ \pi_j = \sum_{j \in [1,M]} f_{i,j}(\mathbf{t})\ \pi_j, \tag{5.6}$$

where $p(\mathcal{T}_j) \triangleq \pi_j$ are typically called the "mixture coefficients" and have to adhere to the following constraint:

$$\sum_{j \in [1,M]} \pi_j = 1. \tag{5.7}$$

The mixture coefficients can be understood in light of mixture modeling as "responsibility" probabilities identifying the probability that a certain target segment is used to sample a source segment. Hence, it is conceivable to compute π_j based on certain properties of the segments such as their size since the larger a segment the more likely it has also been mapped in the source. As will be shown, however, we choose to treat them as latent variables, as is common in mixture modeling, that are estimated as well.

We note again that, in contrast to CPD, (5.6) does not use a Gaussian Mixture Model (GMM), but rather once-computed probability mass functions represented by the CPSHs.

This is advantageous since the CPSHs model the probabilities using a collection of points which implicitly embeds their local geometric relationship.

Our ultimate goal is to estimate $\mathbf{t}$. We opt for a maximum likelihood (ML) estimation whereby we seek to maximize $L_{normal} = p(\mathcal{P_S}|\mathbf{t})$. Assuming i.i.d. source segments $\mathcal{S}_i$, $p(\mathcal{P_S}|\mathbf{t})$ can be written as:

$$L_{normal} = p(\mathcal{P_S}|\mathbf{t}) = \prod_{\mathcal{S}_i \in \mathcal{P_S}} p(\mathcal{S}_i|\mathbf{t}). \tag{5.8}$$

The highly valuable geometric relationship between individual segments is not lost in the i.i.d. assumption. The individual segment pairs represented by the CPSHs are coupled through the translation parameter $\mathbf{t}$. Hence, translation hypotheses supported by the observed geometric arrangement of segments are more likely to be chosen. Nevertheless, it is conceivable to use a more complex joint probability model which for example considers the fact that mapped areas are contiguous owing to the nature of the scanning process and hence the segments can be more tightly coupled.

Now, we need to derive $p(\mathcal{S}_i|\mathbf{t})$. We use Bayes' rule to rewrite $p(\mathcal{S}_i, \mathbf{t})$ as follows:

$$p(\mathcal{S}_i, \mathbf{t}) = p(\mathcal{S}_i|\mathbf{t})\ p(\mathbf{t}). \tag{5.9}$$

Since we want the algorithm to work irrespective of the initial alignment, we model $p(\mathbf{t})$ as a uniform distribution over $\mathcal{D}$ making the following proportionality valid:

$$p(\mathcal{S}_i|\mathbf{t}) \propto p(\mathcal{S}_i, \mathbf{t}). \tag{5.10}$$

Now, we can expand the likelihood function of (5.8) using (5.10) and (5.6) to obtain

$$L_{normal}(\mathbf{t}, \pi_1, \cdots, \pi_M) \propto \prod_{\mathcal{S}_i \in \mathcal{P_S}} p(\mathcal{S}_i, \mathbf{t}) = \prod_{i \in [1,N]} \sum_{j \in [1,M]} f_{i,j}(\mathbf{t})\ \pi_j. \tag{5.11}$$

Notice, the likelihood is a function of $\mathbf{t} = [t_x\ t_y]^T$ and $\pi_1, \cdots, \pi_M$. In essence, the estimation problem *can* now include the *hidden* mixture coefficients in addition to the translation.

5.4.4 The Maximum Likelihood (ML) Estimation Problem

Here, we derive the maximum likelihood (ML) solution using the proposed CPSHs (henceforth termed "normal" as opposed to "exponential" CPSHs which will be introduced shortly) and subsequently show that transforming the CPSHs using an exponential function leads to a more beneficial solution with an intuitive interpretability.

5.4.4.1 ML with normal CPSH

The estimation problem can be formulated as follows:

$$\begin{aligned} \mathbf{t}_{normal}, \pi_{1,normal}, \cdots, \pi_{M,normal} &= \underset{\mathbf{t},\pi_1,\cdots,\pi_M}{\operatorname{argmax}}\ L_{normal}(\mathbf{t}, \pi_1, \cdots, \pi_M) \\ &= \underset{\mathbf{t},\pi_1,\cdots,\pi_M}{\operatorname{argmax}} \prod_{i \in [1,N]} \sum_{j \in [1,M]} f_{i,j}(\mathbf{t})\ \pi_j, \end{aligned} \tag{5.12}$$

where $\mathbf{t}_{normal}$ and $\pi_{1,normal}, \cdots, \pi_{M,normal}$ are the estimated translation and mixture coefficients, respectively. Using log-likelihoods instead, we obtain:

$$\boxed{\mathbf{t}_{normal}, \pi_{1,normal}, \cdots, \pi_{M,normal} = \underset{\mathbf{t},\pi_1,\cdots,\pi_M}{\operatorname{argmax}} \sum_{i\in[1,N]} \log \sum_{j\in[1,M]} f_{i,j}(\mathbf{t})\, \pi_j} \tag{5.13}$$

Contemplating (5.13) it can be seen that multiple segment pairs that encompass the overlapping parts and whose CPSHs peak at the true translation will reinforce the hypothesis corresponding to the true translation. This is the mechanism by which multiple segment pairs contribute towards the final solution as opposed to a hard-decision approach where only the segment pair with the best quality determines the final outcome.

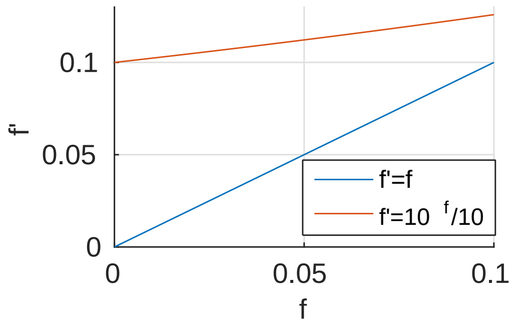
(5.13) has to be solved numerically as $f_{i,j}$'s are non-parametric mass functions. But even when $f_{i,j}$ are Gaussians (CPD), no closed form solution exists. In our case we note the following characteristics for our problem:

- The likelihood's elements are the CPSHs which are higher-level abstractions of collections of points implicitly involving the inter-point geometric relationships.
- The number of segments can be limited to keep the optimization problem tractable and the complexity is not a function of the number of points.
- The conditional probabilities $f_{i,j}(\mathbf{t})$ already embed the translation parameter and are precomputed *once* and do not need updates.
- We have decoupled the translation estimation from the rotation estimation where the rotation can already be estimated before commencing with translation estimation. Accordingly, the estimation problem has a significantly reduced parameter space covered by 2D histograms (or PMFs).

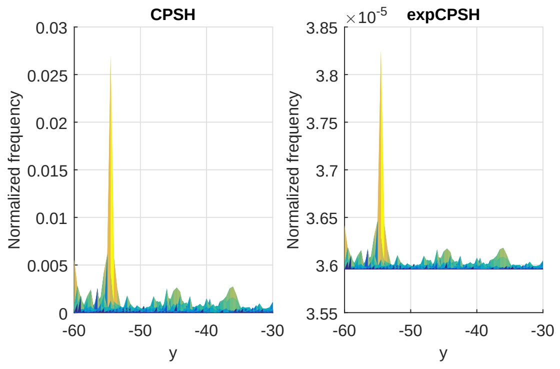
In achieving these characteristics we exploit fundamental properties of the application and data. Needless to say, for more general alignment problems, CPD provides a more flexible framework with the capability of handling different kinds of transformations.

Given the points above, we consider a grid-search-based maximization approach, whereby a set of *distinct* mixture probability *values* is considered such that $\pi_j \in \mathcal{U}$. *All* permutations of M (the number of target segments) mixture probability values sampled from $\mathcal{U}$ that add up to 1 (i.e. satisfying (5.7)) result in a *set* of mixture coefficient *sets* $\mathcal{K}$. Each mixture coefficient set $\in \mathcal{K}$ represents a mixture coefficient set hypothesis. Each hypothesis requires computing the sum of the log of the weighted average CPSH for each source segment.

Notice that $|\mathcal{K}| << |\mathcal{U}|^M$ due to the fact that only mixture sets which add up to 1 (5.7) can be considered. Hence, by choosing an appropriately small set of possible mixture probability values $\mathcal{U}$ and keeping the number of segments low, such grid-search-based maximization can be ensured to achieve a target time limit. Also, the CPSH bin size can be tuned to ensure fast computation.



(a)



(b)

Figure 5.12: Computing the expCPSH. In (a), the mapping of equation (5.14) is seen (without normalization by $c_{i,j}$ and normalized to keep probability values between 0 and 1) for the probability interval that is commonly encountered in CPSHs. In this interval (5.14) represents a linear mapping with a linear factor < 1. In (b), a CPSH and the corresponding expCPSH obtained using (5.14) are shown. As expected, the CPSH shape is maintained in the expCPSH. Image source: [2] – ©2016 Elsevier.

5.4.4.2 ML with exponential CPSH (expCPSH)

We have observed the likelihood surfaces resulting from 5.13 to be "noisy" and highly ambiguous as seen in Figure 5.13(a). The obvious reason is the logarithm which reduces the ratio between strong and weak hypotheses. Dropping the log from (5.13) has constantly delivered better results. However, we need to understand what implications such a formula change on the CPSHs has.

The fundamental question is what kind of transformation do the CPSHs have to undergo such that the likelihood function does not include the log? One might be tempted to think of a transformation which involves an exponential function since the log of an exponential reduces down to the exponent. Consider the transformation of CPSH $f_{i,j}$ to the expCPSH:

$$\hat{f}_{i,j}(\mathbf{t}) = \frac{e^{f_{i,j}(\mathbf{t})}}{c_{i,j}} \tag{5.14}$$

where $c_{i,j}$ is the normalization factor ensuring $\hat{f}_{i,j}$ can be considered a probability density. As Figure 5.12(a) shows, this transformation maps probabilities (in the probability interval commonly encountered in CPSHs) essentially in a linear manner. This means that the CPSH surface is maintained and the peak relationships are not altered. Hence, there is no intra-CPSH distortion. Despite being effectively a linear transformation there exists the possibility that the linear parameters are different between different CPSHs due to the normalization constants $c_{i,j}$ and hence the inter-CPSH relationships being distorted. However, in Appendix A we show that the normalization coefficient $c_{i,j}$ is almost equal irrespective of the CPSH.

While the expCPSHs maintain the inter and intra-CPSH hypotheses relationships, they present a functional benefit and a mathematical convenience which become apparent when inserting (5.14) instead of the usual $f_{i,j}$ in (5.12) resulting in the "exponential" likelihood:

$$L_{exp}(\mathbf{t}, \pi_1, \cdots, \pi_M) = \sum_{i\in[1,N]} \log_e\Big(\sum_{j\in[1,M]} \frac{e^{f_{i,j}(\mathbf{t})}}{c_{i,j}}\, \pi_j\Big). \tag{5.15}$$

Noting again that $c_{i,j} \approx c \;\forall\{\mathcal{S}_i, \mathcal{T}_j\} \in \mathcal{P}_{\mathcal{S}} \times \mathcal{P}_{\mathcal{T}}$ (see Appendix A), $c_{i,j}$ is irrelevant for the maximization and is hence dropped delivering

$$L'_{exp}(\mathbf{t}, \pi_1, \cdots, \pi_M) \triangleq \sum_{i\in[1,N]} \log_e\Big(\sum_{j\in[1,M]} e^{f_{i,j}(\mathbf{t})}\, \pi_j\Big). \tag{5.16}$$

At this moment the intended simplification of a log on an exponential is not achievable as the log precedes a summation. However, since the mixture coefficients add up to one (see (5.7)), Jensen's inequality can be used to move the log inside the sum obtaining a lower bound

$$\begin{aligned} L'_{exp}(\mathbf{t}, \pi_1, \cdots, \pi_M) \geq \sum_{i\in[1,N]} \sum_{j\in[1,M]} \log_e(e^{f_{i,j}(\mathbf{t})})\, \pi_j = \\ \sum_{i\in[1,N]} \sum_{j\in[1,M]} f_{i,j}(\mathbf{t})\, \pi_j \triangleq L''_{exp}(\mathbf{t}, \pi_1, \cdots, \pi_M). \end{aligned} \tag{5.17}$$

As will be shown in Section 5.4.8, the approximation of L'_{exp} by L''_{exp} is very accurate while simplifying the full exponential likelihood of (5.15) and facilitating an intuitive interpretation of the results. The maximum *approximate exponential likelihood* can now be written as:

$$\boxed{\mathbf{t}_{exp}, \pi_{1,exp}, \cdots, \pi_{M,exp} = \underset{\mathbf{t},\pi_1,\cdots,\pi_M}{\operatorname{argmax}}\; L''_{exp} = \sum_{i\in[1,N]} \sum_{j\in[1,M]} f_{i,j}(\mathbf{t})\, \pi_j} \tag{5.18}$$

The approximate exponential likelihood amounts to "simple" weighted mean computation of the CPSHs for different weighting coefficient sets (mixture coefficients sets). The functional benefit lies in the absence of the log term which no longer leads to a companding of the CPSHs. Hence, only fewer dominant hypotheses are considered while insignificant candidates are effectively neglected. Dominant outlier hypothesis are not likely to vote for the same wrong translation, whereas the hypotheses corresponding to the right translation (contributed by multiple truly overlapping source-target segment pairs) re-enforce one another. The mathematical convenience lies in the fact that no actual transformation has to be performed requiring no further computations. The transformation manifests itself in the likelihood function change from (5.13) to (5.18).

The reader is invited to contemplate the observation that computing the mean CPSH is an intuitive approach to fusing different CPSHs contributed by the source-target pairs. With this derivation, intuition is supported by mathematics whereby we prove that the mean CPSH is essentially the maximum likelihood solution when considering the CPSHs to be probability mass functions which are part of a mixture probability model.

5.4.5 Resolving Ground-Plane Rotation Ambiguity II

The same argument used in Section 5.3.3 to address the issue with rotation ambiguity (Section 5.2) applies here: We compute how ambiguously the maximum likelihood *value* $L''_{exp}(\mathbf{t}_{exp}, \pi_{1,exp}, \cdots, \pi_{M,exp})$ is identified in the likelihood *function* $L''_{exp}(\mathbf{t}, \pi_{1,exp}, \cdots, \pi_{M,exp})$ as a quality for each of the four dominant rotation directions $\theta_r = \hat{\theta} + (r-1) \cdot 90$ (remember $\hat{\theta}$ is the estimated ground-plane rotation before disambiguation according to (5.2)) where $r = \{1, 2, 3, 4\}$. Hence, the quality value is computed as:

$$q_r = \frac{L''_{exp}(\mathbf{t}_{exp}, \pi_{1,exp}, \cdots, \pi_{M,exp}) - min_{\mathbf{t}}(L''_{exp}(\mathbf{t}, \pi_{1,exp}, \cdots, \pi_{M,exp}))}{\sum_{\mathbf{t}}[L''_{exp}(\mathbf{t}, \pi_{1,exp}, \cdots, \pi_{M,exp}) - min_{\mathbf{t}}(L''_{exp}(\mathbf{t}, \pi_{1,exp}, \cdots, \pi_{M,exp}))]} \tag{5.19}$$

Notice this formula is equally valid when no segmentation (the entire cloud is a segment) is used where $L''_{exp}(\mathbf{t})$ is essentially the (only) normalized CPSH. In the results we will refer to the CPSH (or equivalently likelihood function) with the highest quality as the "winning" one.

5.4.6 Determining Overlap as Byproduct

This segment-based approach has a useful byproduct: By evaluating the posterior probabilities

$$p(\mathcal{T}_j|\mathcal{S}_i, \mathbf{t}_{exp}) = \frac{p(\mathcal{S}_i, \mathbf{t}_{exp}|\mathcal{T}_j)\,\pi_{j,exp}}{p(\mathcal{S}_i, \mathbf{t}_{exp})} = \frac{f_{i,j}(\mathbf{t}_{exp})\,\pi_{j,exp}}{\sum_{j\in[1,M]} f_{i,j}(\mathbf{t}_{exp})\pi_{j,exp}}, \tag{5.20}$$

the *correspondence probability* between any source-target segment pair can be evaluated. By suitable adaptation of the segmentation, as will be shown in Section 5.4.7, this makes it possible to identify which segments have an overlap and which ones do not.

5.4.7 Accounting for Non-Overlap

We want to specifically handle cases where the overlap between the source and target may be low. In that case, for many source segments there might be no corresponding target segment. Hence, we propose to add another *non-physical* target segment, $\mathcal{T}_0$. The conditional probability $p(\mathcal{S}_i, \mathbf{t}|\mathcal{T}_0)$ is modeled as a uniform distribution, hence:

$$p(\mathcal{S}_i, \mathbf{t}|\mathcal{T}_0) = \frac{1}{|\mathcal{D}|}. \tag{5.21}$$

Remember, $\mathcal{D}$ is the set of all possible translations. CPD also uses a uniform distribution term to account for outliers (see Section 5.4.2). Our approach is different in that the probability component responsible for outliers is part of the mixture and associated with a (latent) mixture coefficient (π_0) whose value is estimated, whereas in CPD the outlier-accounting parameter (w in 2.8) has to be preset by the user. Accordingly, the estimated mixture coefficients including π_0 allow us to identify which source segments are actually outliers as will be demonstrated in Section 5.4.8.

5.4.8 Demonstration

The segment-based alignment is applied on the very challenging example shown in Figures 5.1 after it has been rotationally aligned as shown in Figure 5.10(c). The used segmentation and the resulting CPSHs are shown in Figure 5.11. The different likelihood functions evaluated on the CPSHs are shown Figure 5.13. The figures illustrate how critical the exponential mapping is to getting likelihood functions with unambiguous results. It can also be seen that using the exponential likelihood or approximate exponential likelihood results in a correct translation estimate despite the very limited overlap between the source and target. Also, the approximation of the exponential likelihood (L_{exp}) by the approximate exponential likelihood (L''_{exp}) does not introduce any visible distortions.

The mixture coefficient set that maximizes the approx. exp. likelihood (Figure 5.13(c)) is found to be: $\{\pi_1, \pi_2, \pi_3, \pi_0\} = \{0.1, 0.1, 0.7, 0.1\}$. So, the *third* physical target segment is identified as being the most important which agrees with the fact that only T3 has a significant overlap with any source segment (see Figure 5.11(a)). Notice the target has only three segments. The fourth mixture probability value (π_0) corresponds to the non-physical segment that accounts for outlier segments (non-overlapping segments).

The posterior correspondence probabilities for the example in Figure 5.13, as evaluated using (5.20), are as follows:

	T1	T2	T3	T0 (non-physical)
S1	0.01	0.01	0.07	**0.92**
S2	0.04	0.00	**0.97**	0.00
S3	0.01	0.01	0.07	**0.92**

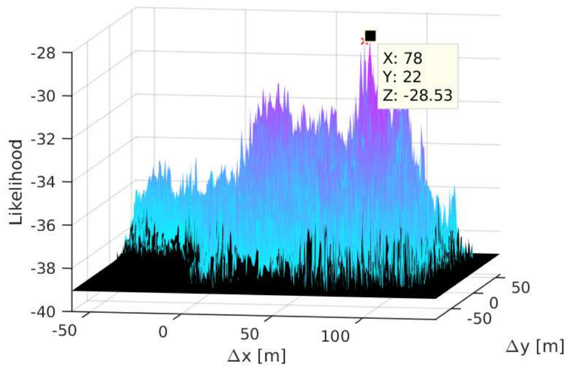
The correspondence probability for S2-T3 is almost 1.0 owing to the unambiguous peak in the corresponding CPSH. S1 and S3, on the other hand, exhibit their highest correspondence probabilities w.r.t. T0. They are effectively labeled as segments with no overlap with the target.

5.5 The Algorithm in Summary

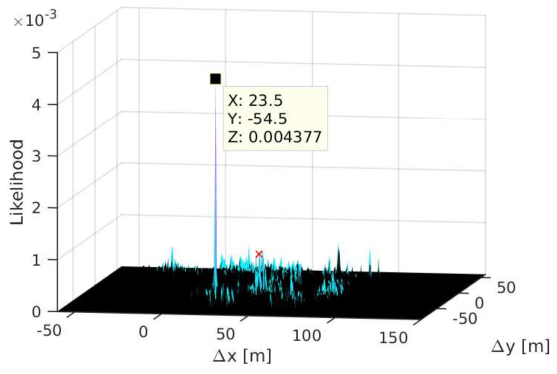
Now that the individual pieces of the algorithm have been developed and before delving into the evaluation and results, we present here a summary of the algorithm in pseudo-code in Algorithm 1.

Algorithm 1: The proposed alignment algorithm pseudo-code. The function-section relationships are as follows: RotAlignToGravity-5.2.1, RotAlignGrndPlane-5.2.2, RunExpML-5.4.4.2 (q refers to quality as computed using (5.19)), ComputeCPSH-5.3.2, CreateCloudSegments-[2, Appendix B]. RotateSegmentsXYPlane rotates the points of all segments by a given angle around the z-axis. $\mathbf{R_z}(\theta)$ builds a 3D rotation matrix that represents rotation around the z-axis by θ.

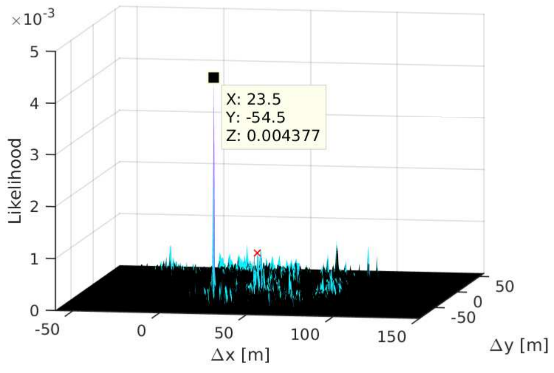
```
Data: Source ($\mathcal{S}$) and target ($\mathcal{T}$) point clouds.
Result: 3D Homog. transf. matrix to align $\mathcal{S}$ to $\mathcal{T}$.

/* ——Grav. Rotation Alignment—— */
$[\mathcal{S}_{grav}, \mathbf{R_{g,s}}] \leftarrow$ RotAlignToGravity $(\mathcal{S},\phi_{n_z})$;
$[\mathcal{T}_{grav}, \mathbf{R_{g,t}}] \leftarrow$ RotAlignToGravity $(\mathcal{T},\phi_{n_z})$;

/* ——Grav. Transl. Alignment—— */
$[\mathcal{S}_h, t_{z,s}, \mathcal{T}_h, t_{z,t}] \leftarrow$ AlignFloor $(\mathcal{S}_{grav},\mathcal{T}_{grav})$;

/* ——Ground-Plane Rotation Alignment—— */
$\hat{\theta} \leftarrow$ RotAlignGrndPlane $(\mathcal{S}_h,\mathcal{T}_h)$;

/* ——Ground-Plane Translation Alignment—— */
if segMode then
    $\mathcal{P}_\mathcal{S} \leftarrow \{\mathcal{S}_1, \mathcal{S}_1, .., \mathcal{S}_N\} \leftarrow$CreateCloudSegments $(\mathcal{S}_h)$;
    $\mathcal{P}_\mathcal{T} \leftarrow \{\mathcal{T}_1, \mathcal{T}_2, .., \mathcal{T}_M\} \leftarrow$CreateCloudSegments $(\mathcal{T}_h)$;
else
    $\mathcal{P}_\mathcal{S} \leftarrow \mathcal{S}_1 \leftarrow \mathcal{S}$;
    $\mathcal{P}_\mathcal{T} \leftarrow \mathcal{T}_1 \leftarrow \mathcal{T}$;
/*Consider ground-plane rotation ambiguity*/
for $r \leftarrow 1$ to 4 do
    $\theta(r) \leftarrow \hat{\theta} + (r-1) \cdot 90$;
    $\tilde{\mathcal{P}_\mathcal{S}} \leftarrow [\tilde{\mathcal{S}_1}, .., \tilde{\mathcal{S}_N}] \leftarrow$RotateSegmentsXY$(\mathcal{P}_\mathcal{S},\theta(r))$;
    /*All combinations of source target segments*/
    for $i \leftarrow 1$ to $N$ do
        for $j \leftarrow 1$ to $M$ do
            $f_{i,j}(\mathbf{t}) \leftarrow$ ComputeCPSH $(\tilde{\mathcal{S}_i},\tilde{\mathcal{T}_j},\epsilon_\phi,\epsilon_z)$
    $\mathbf{t}(r), L''_{exp}, q(r) \leftarrow$ RunExpML $(f_{1,1},...,f_{N,M})$;
/*Take result with highest quality*/
$r_f \leftarrow \text{argmax}_r\, q(r)$;
$\theta_{z,f} \leftarrow \theta(r_f)$;
$\mathbf{t}_f = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \leftarrow \mathbf{t}(r_f)$;

/* ——Assemble alignment transformation —— */
$\hat{\mathbf{T}} \leftarrow \begin{bmatrix} [\mathbf{R_{g,t}}^T \mathbf{R_z}(\theta_{z,f}) \mathbf{R_{g,s}}] & \mathbf{R_{g,t}}^T \begin{bmatrix} \mathbf{t_f} \\ t_{z,t} - t_{z,s} \end{bmatrix} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$;
```



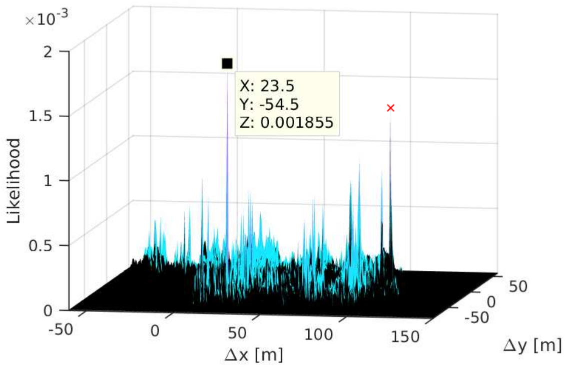
(a) $L(\mathbf{t}, \pi_{1,normal}, \cdots, \pi_{M,normal})$



(b) $L_{exp}(\mathbf{t}, \pi_{1,exp}, \cdots, \pi_{M,exp})$



(c) $L''_{exp}(\mathbf{t}, \pi_{1,exp}, \cdots, \pi_{M,exp})$



(d) $L''_{exp}(\mathbf{t}, \pi_j = 1/M \forall j)$

Figure 5.13: Different likelihood functions evaluated on the CPSHs shown in Figure 5.11(b). In (a) the normal likelihood of (5.13), in (b) the exponential likelihood of (5.15) and in (c) the approximate exponential likelihood of (5.18). All three likelihoods are shown as functions of $\mathbf{t}$ evaluated at the respective estimated optimal mixture coefficient set ($\pi_{1,normal}, \cdots, \pi_{M,normal}$ in case of the normal likelihood and $\pi_{1,exp}, \cdots, \pi_{M,exp}$ in the case of the exponential likelihoods). In (d) the approximate exponential likelihood function is evaluated using a uniform mixture probability set, i.e. $\pi_j = 1/M$. The strongest peak of each function is identified by a data tip and the second strongest is identified by an x mark. Clearly, the normal ML is very "noisy" and, as seen by the data tip, does not correctly identify the true translation (the true translation alignment is $\mathbf{t} = [22.3\ 54.8]^T$ m, see Figure 5.11) since the logarithm operation down-weights the otherwise unambiguous peak of the only overlapping segment pair (see Figure 5.11). Once replacing the CPSHs with expCPSHs, the translation is correctly and unambiguously estimated. Comparing (b) to (c) it becomes evident that the approximation using Jensen's inequality introduced to get to the approximate exponential likelihood preserves the likelihood function shape and the inter-hypothesis relationships. Comparing (c) to (d) it can be seen that involving the mixture probabilities provides an added advantage as the true hypothesis is amplified as compared to all other translations. Image source: [2] – ©2016 Elsevier.

5.6 Performance Evaluation

In this section we present results highlighting aspects relating to the strengths of the proposed algorithm. We start out by presenting the parameter values used for the different algorithms followed by a brief explanation of our evaluation methodology in Section 5.6.2. Then results regarding algorithm resilience towards sampling issues are presented in Section 5.6.3

followed by results relating to algorithm sensitivity towards initial alignment in Section 5.6.4 and overlap sensitivity in Section 5.6.5. Comparative results on different datasets are presented in Section 5.6.6. We also evaluate the algorithm on a strictly non-Manhattan dataset (Section 5.6.7) and an external dataset (Section 5.6.8).

5.6.1 Parameters and Evaluation Environment

For our proposed approach we use a Matlab implementation. For gravity rotation we set $\phi_{n_z} = 15°$. For the ground-plane rotation signatures we set $v = 720$ resulting in 0.5° bin size. For the ground-plane translation ϵ_ϕ is set to 30° (see [80] for typical angle errors in surface normal estimation) and the height threshold is set as $\epsilon_z = 10\,\text{cm}$ given the maximum error encountered in the results presented in Figure 5.6(c). The CPSH bin size is set to 0.5 m along x and y which we found to provide a good compromise delivering alignment accuracies comparable to those of other working methods (see Figure 5.17 & 5.18) while accommodating the sensing inaccuracies.

We use the reference implementation of S4PCS provided online by the author [81] for comparison purposes. The number of samples n and the LCP distance δ have been tuned to provide good alignment results for the datasets shown in Figure 5.17. More specifically, combinations of $n = \{10, 50, 200, 800, 1200, 2000, 2500, 4000, 5000\}$ (see [41], Table 1) and $\delta = \{0.1, 0.125, 0.15, 0.2, 0.4\}$m were considered. The values were finally set to $n = 2500$ and $\delta = 0.15\,\text{m}$. The maximum run time was set to 150 s. The overlap rate was provided manually for each dataset as we also do with CPD and GMMREG for which we also use the reference implementations of the authors ([82], [83]). As in S4PCS, the overlap ratio is provided manually to CPD. The value "normalize" is set to true.

We also compare against a feature-based approach. We use the feature-based alignment system presented in Chapter 6 (without the pre-processor and without ICP), however, with Harris3D key points instead of ISS and with Progressive Sample Consensus (PROSAC) [84] instead of RANSAC for faster robust estimation of the transformation. The key point and descriptor support radii are 0.3 m and 1.2 m, respectively. The algorithm is written in C++ and uses the PCL [17] library which already implements a parallelized version of SHOT that uses all processor cores to perform repetitive point-wise operations (such as computing the NNs of a point) in parallel. We furthermore accelerated the keypoint detection and parallelized feature matching to run on all cores using OpenMP.

Time measurements are conducted using a PC equipped with an Intel i7-4770 quad-core processor with 3.5GHz clock rate.

5.6.2 Evaluation Methodology

Similar to the approach explained in Section 3.4.1, we compute the error transformation $\mathbf{T}_e$ and use it to compute the error angle α_e and the translation error t_e. The rotation alignment is considered successful if $\alpha_e \leq 1°$, while a translation is considered successful if $t_e \leq 1\,\mathrm{m}$. Note that the translation error also depends on the angle error. When not specifying whether we speak about rotation or translation alignment success, then we require both conditions to be satisfied. Whenever suitable we also report the mean squared error (MSE).

5.6.3 Sensitivity to Subsampling

Although the point clouds of indoor models commonly are obtained with the same sensor, they may differ in their point densities in certain regions. This is mainly because of the spatial relationship between the sensor and the sensed points. Hence, we want to investigate the sensitivity of our proposed algorithm towards differing point densities.

We performed the following experiments: While keeping the target point cloud set fixed, the number of source points is reduced by the subsampling factor o, i.e., keeping one out of every o points. The experiment is run on two datasets with 100% and 70% overlap.

Table 5.1: Robustness towards point density difference. The alignment is performed after subsampling the source cloud by the factor o. The alignment is repeated 100 times for each subsampling factor, each time dropping a different randomly chosen subset of points to obtain statistically reliable results. For each attempt, the alignment success is determined (see Section 5.6.2). The experiment is conducted for the high overlap example source-target set seen in Figure 5.8 (named "100%") and the one with around 70% overlap shown in Figure 5.10(a). "NoSeg" refers to alignment without segmentation as opposed to "WithSeg".

	100% NoSeg	70% NoSeg	100% WithSeg
$o = 5$	100	100	99
$o = 10$	100	100	99
$o = 15$	100	100	95
$o = 20$	100	100	99
$o = 30$	100	100	90

Table 5.1 shows the percentage of successful alignment attempts remains above 90% despite severe down-sampling. It can also be seen that subsampling only adversely affects alignment when using segmentation. Segments have much fewer points to participate in the voting. Combined with the further reduction due to down-sampling this can result in having too few votes to sufficiently populate the translation space resulting in non-conclusive CPSHs. Indeed, after investigation of the cases that failed it turns out the rotation error is not due to the ground-plane rotation alignment step but rather due to the rotation disambiguation step explained in Section 5.4.5.

It remains to be said that a subsampling rate of 30 is very severe. Moreover, sampling densities may vary for small regions. In this experiment, however, we changed the point density throughout the entire source cloud.

5.6.4 Sensitivity towards Initial Alignment

Here, we examine the robustness properties towards initial alignment. For this we use the overlap example of Figure 5.8. However, first the source cloud is perfectly aligned to the target cloud. Then, a misalignment in the shape of an in-ground plane rotation by θ_m is applied and the alignment is attempted using our approach (without segmentation) as well as using CPD and GMMREG. This process is repeated for different angular misalignments.

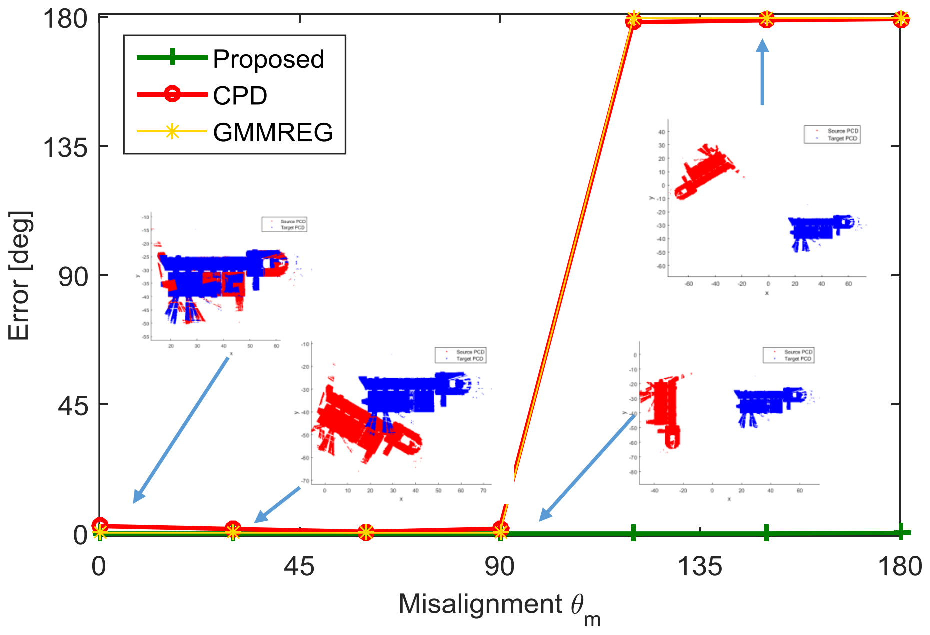


Figure 5.14: Rotation alignment error (as defined in Section 5.6.2) as a function of initial alignment. The alignment is attempted after applying an angular misalignment of θ_m in the ground plane on the perfectly aligned source-target cloud pair of Figure 5.8. The small images depict the source-target cloud pair to be aligned, seen from a top-down view. Image source: [2] – ©2016 Elsevier.

The results in Figure 5.14 show that our proposed alignment algorithm has an angular alignment error (see Section 5.6.2) of around 0° irrespective of the initial alignment. The probabilistic approaches of CPD and GMMREG, on the other hand, completely fail at certain initial alignments. As soon as the initial ground-plane rotation is $> 90°$, the final alignment error approaches 180°. This is because the CPD and GMMREG's iterative optimizers converge "in the other direction" hitting a local maximum of the utility function (the likelihood in case of CPD and the L2 distance in case of GMMREG).

5.6.5 Sensitivity towards Overlap

Low overlap poses a challenge to all alignment algorithms. Here, we compare our proposed algorithm with CPD, GMMREG, S4PCS and feature-based (Features) in terms of robustness towards overlap when considering large scale indoor 3D models.

To understand the overlap robustness properties we use the pair of point clouds of Figure 5.10(a) after first aligning them perfectly and subsequently introducing a modest translational misalignment. Now, we align the point clouds after cropping out, in steps, parts of the source and target clouds such that the overlap is reduced further. Three different cropping steps are performed.

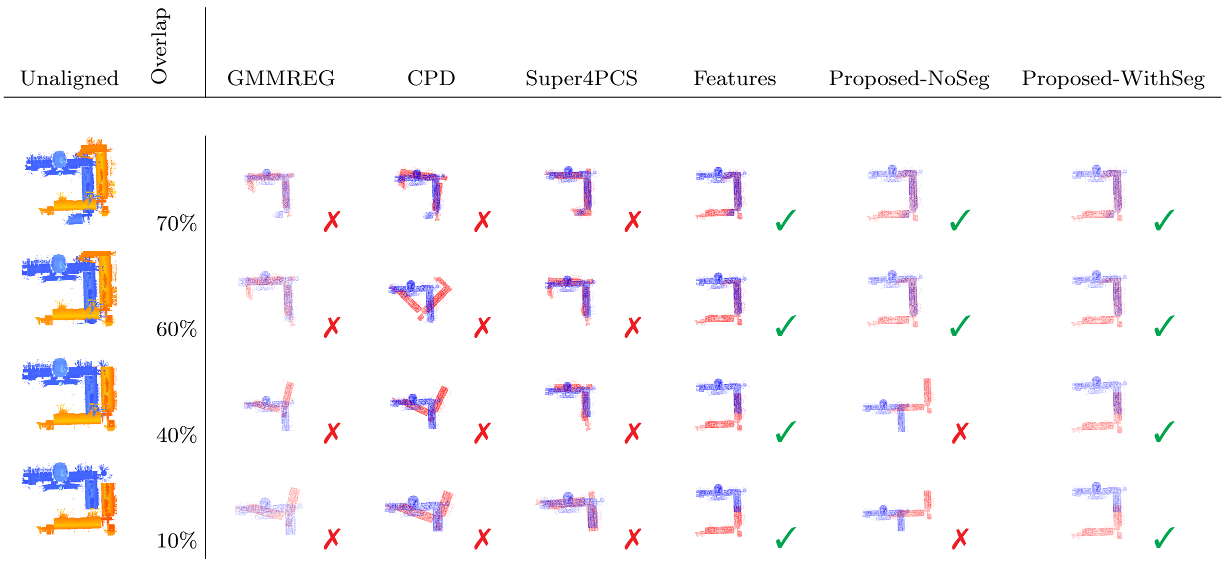
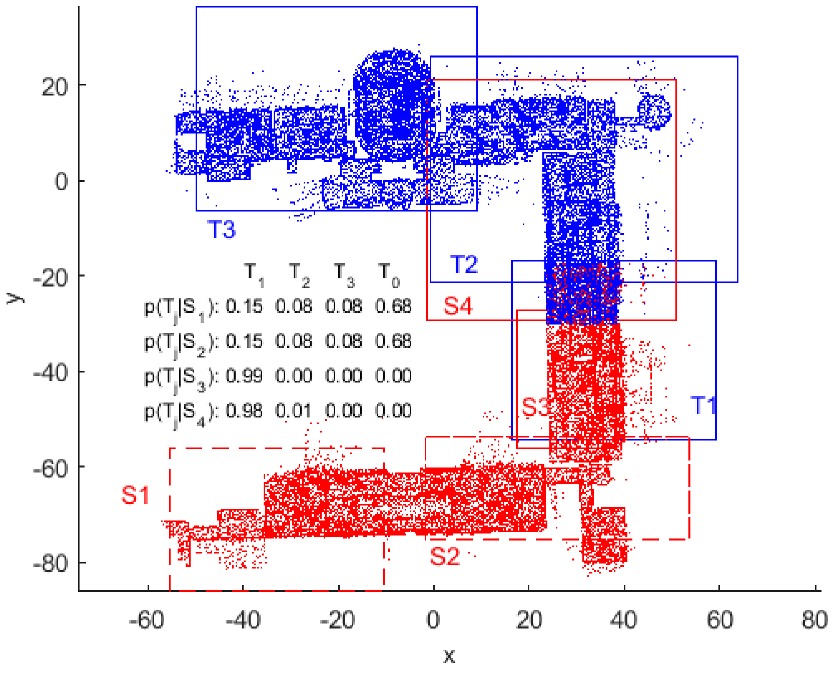


Figure 5.15: Sensitivity towards overlap. Starting with perfect rotation alignment and limited translation misalignment, the overlap between the source and target clouds of Figure 5.10(a) is decreased in steps by cropping out parts of the source and target clouds that form part of the overlap. The alignment is attempted using GMMREG, CPD, S4PCS, Features and using our proposed approach with and without segmentation. The overlap value used in CPD and S4PCS is indicated for each pair. The final alignment is shown from a top down view for each tested overlap scenario and respective alignment algorithm. A ✓ means the alignment was successful, as opposed to a ✗. Image source: [2] – ©2016 Elsevier.

The results in Figure 5.15 show that S4PCS fails to recover the alignment even in the highest overlap case. It consistently aligns the large corridors to one another as it tries to maximize the common point set (see Section 2.2.2.3). Also GMMREG fails to find the proper alignment irrespective of the overlap as it gets stuck in local minima. The outcome of two cases resembles that of S4PCS hinting again that a maximization of overlap is being sought. CPD also fails in all instances. Interestingly, the final outcome is occasionally very similar to that of GMMREG owing to their similar nature. Our proposed algorithm, which is specifically tailored for aligning indoor point clouds, succeeds in the alignment in two out of four cases.

It can be observed that the alignment fails when the overlap drops below 30% for each point cloud. Using the segment-based approach, however, allows it to handle the two low overlap cases as well. The feature-based approach succeeds in all overlap scenarios demonstrating the robustness of our implementation and the proper tuning of the parameters.

The alignment result with our proposed approach using segments (Proposed-WithSeg) for the case with 10% overlap is shown enlarged in Figure 5.16. It becomes evident how challenging this case is as the overlap makes up a tiny fraction of each point cloud. Yet, the small overlap between S3-T1 and S4-T1 is enough to overcome the challenge. The maximization in (5.18) puts more emphasis on the CPSHs which have peaks that re-enforce one another helping to suppress the false peaks.



	T_1	T_2	T_3	T_0
$p(T_j\|S_1)$:	0.15	0.08	0.08	0.68
$p(T_j\|S_2)$:	0.15	0.08	0.08	0.68
$p(T_j\|S_3)$:	0.99	0.00	0.00	0.00
$p(T_j\|S_4)$:	0.98	0.01	0.00	0.00

Figure 5.16: The result obtained with the segment-based alignment on the smallest overlap case, which is shown in Figure 5.15, is shown enlarged here. Also the used segmentation is shown superimposed on the point clouds with source segments without any overlap with target segments having dashed bounding boxes. The posterior correspondence probabilities for the ground-plane translation alignment evaluated according to (5.20) are shown in the figure as well. The source segments S1 and S2, identified by the dashed boxes, lead to the highest correspondence probabilities with T0, which is the non-physical segment that accounts for non-overlap. Hence, S1 and S2 have been identified as not overlapping with the target which they truly are. Noting the posterior probability for S4-T2 we draw the reader's attention not to be confused by the bounding boxes when understanding the posterior probabilities. What matters is the overlap region in terms of points in the point clouds. Since S4 indeed does not share many points with T2 it is no wonder the respective posterior is soo low. Image source: [2] – ©2016 Elsevier.

5.6.6 Comparative Evaluation - Multiple Datasets

Here, we compare our proposed algorithm to S4PCS, CPD and the feature-based algorithm on a number of datasets we obtained using our mapper [18]. GMMREG is not considered because it is extremely slow.

Figure 5.17 shows the results obtained with point clouds captured at the “Alte Pinakothek”, one of the world’s oldest galleries. The point clouds span dimensions of tens of meters and comprise hundreds of thousands of points. Three pairs are chosen such that the alignment problem is made increasingly challenging with decreasing overlap.

CPD fails in all three cases. Besides, the required run times are beyond any practical deployment. S4PCS and feature-based alignment on the other hand, manage to align all three cases correctly and relatively efficiently. Similarly, our proposed approach correctly aligns all three cases even without having to use the segment-based approach. Also, the required run times for our approach are comparable to those of S4PCS and the feature-based system. In fact, without segmentation our approach is the fastest. These are very encouraging results as we use a Matlab implementation whereas S4PCS uses C++ implementation and is an algorithm designed to be very efficient. A similar argument applies to the feature-based system which we highly optimized for speed (see Section 5.6.2).

Figure 5.18 shows the results obtained with point clouds captured at the Deutsches Museum, the world’s largest technical museum. This represents a more challenging data set compared to the Alte Pinakothek data set. The point clouds span dimensions that can exceed 100m and comprise multiple million points. The overlap regions are smaller. Also shape similarities in the form of large perpendicular corridors are particularly challenging for the S4PCS approach which relies on solving the LCP problem. Again, three pairs are chosen such that the alignment problem is made increasingly challenging.

The results in Figure 5.18 show that our proposed approach without segmentation can handle the two pairs where the overlap remains above $\approx 30\%$ for each point cloud. Once segmentation is turned on, even the very low overlap pair with around only 10% overlap can be handled. The segmentation is seen to increase the required computational time between 2 and 3 times. This is attributed to the fact that the segments are overlapping and not disjoint. The likelihood maximization incurs no significant cost as a uniform set of mixture coefficients is chosen (the latent variables are not estimated in this experiment). CPD and S4PCS fail to find the correct alignment for all three cases. Similar to the results in Figure 5.15, S4PCS is seen to maximize the overlap between the point cloud pairs as it attempts to maximize the common point set. The only competing approach that works for all three cases is the feature-based approach. However, we outperform it notably in terms of efficiency despite using a Matlab implementation.

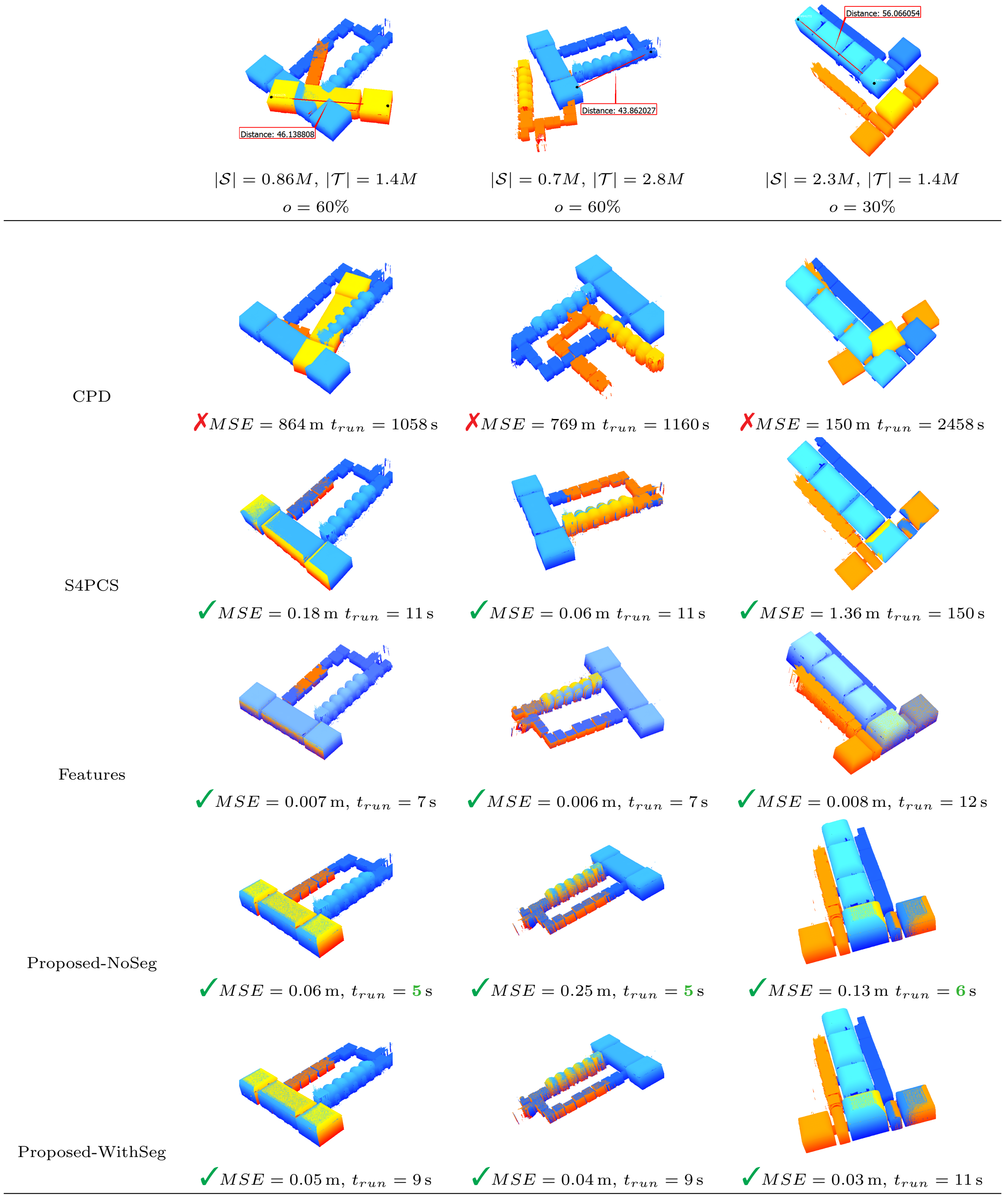

Distance: 46.138808
Distance: 43.862027
Distance: 56.066054
$|\mathcal{S}| = 0.86M,\ |\mathcal{T}| = 1.4M$ $o = 60\%$
$|\mathcal{S}| = 0.7M,\ |\mathcal{T}| = 2.8M$ $o = 60\%$
$|\mathcal{S}| = 2.3M,\ |\mathcal{T}| = 1.4M$ $o = 30\%$
CPD
✗ $MSE = 864\,\mathrm{m}\ t_{run} = 1058\,\mathrm{s}$
✗ $MSE = 769\,\mathrm{m}\ t_{run} = 1160\,\mathrm{s}$
✗ $MSE = 150\,\mathrm{m}\ t_{run} = 2458\,\mathrm{s}$
S4PCS
✓ $MSE = 0.18\,\mathrm{m}\ t_{run} = 11\,\mathrm{s}$
✓ $MSE = 0.06\,\mathrm{m}\ t_{run} = 11\,\mathrm{s}$
✓ $MSE = 1.36\,\mathrm{m}\ t_{run} = 150\,\mathrm{s}$
Features
✓ $MSE = 0.007\,\mathrm{m},\ t_{run} = 7\,\mathrm{s}$
✓ $MSE = 0.006\,\mathrm{m},\ t_{run} = 7\,\mathrm{s}$
✓ $MSE = 0.008\,\mathrm{m},\ t_{run} = 12\,\mathrm{s}$
Proposed-NoSeg
✓ $MSE = 0.06\,\mathrm{m},\ t_{run} = \mathbf{5}\,\mathrm{s}$
✓ $MSE = 0.25\,\mathrm{m},\ t_{run} = \mathbf{5}\,\mathrm{s}$
✓ $MSE = 0.13\,\mathrm{m}\ t_{run} = \mathbf{6}\,\mathrm{s}$
Proposed-WithSeg
✓ $MSE = 0.05\,\mathrm{m},\ t_{run} = 9\,\mathrm{s}$
✓ $MSE = 0.04\,\mathrm{m},\ t_{run} = 9\,\mathrm{s}$
✓ $MSE = 0.03\,\mathrm{m},\ t_{run} = 11\,\mathrm{s}$

Figure 5.17: Comparative test with Alte Pinakothek dataset. Each column represents a different source-target cloud pair shown unaligned in the first row. The number of source points ($|\mathcal{S}|$) and target points ($|\mathcal{T}|$) as well as the overlap value used in CPD and Super4PCS (o) is shown below each pair. The alignment outcome obtained with the different approaches are shown in the rows below where Proposed-NoSeg refers to segment-less ground-plane translation alignment as opposed to Proposed-WithSeg. The run times include that required to resolve the rotation ambiguity (see algorithm in Section 5.5). Successful alignment is marked by ✓ and failure by ✗. The method with the lowest run time among all successful ones for a dataset is highlighted in green bold font. Image source: [2] – ©2016 Elsevier.

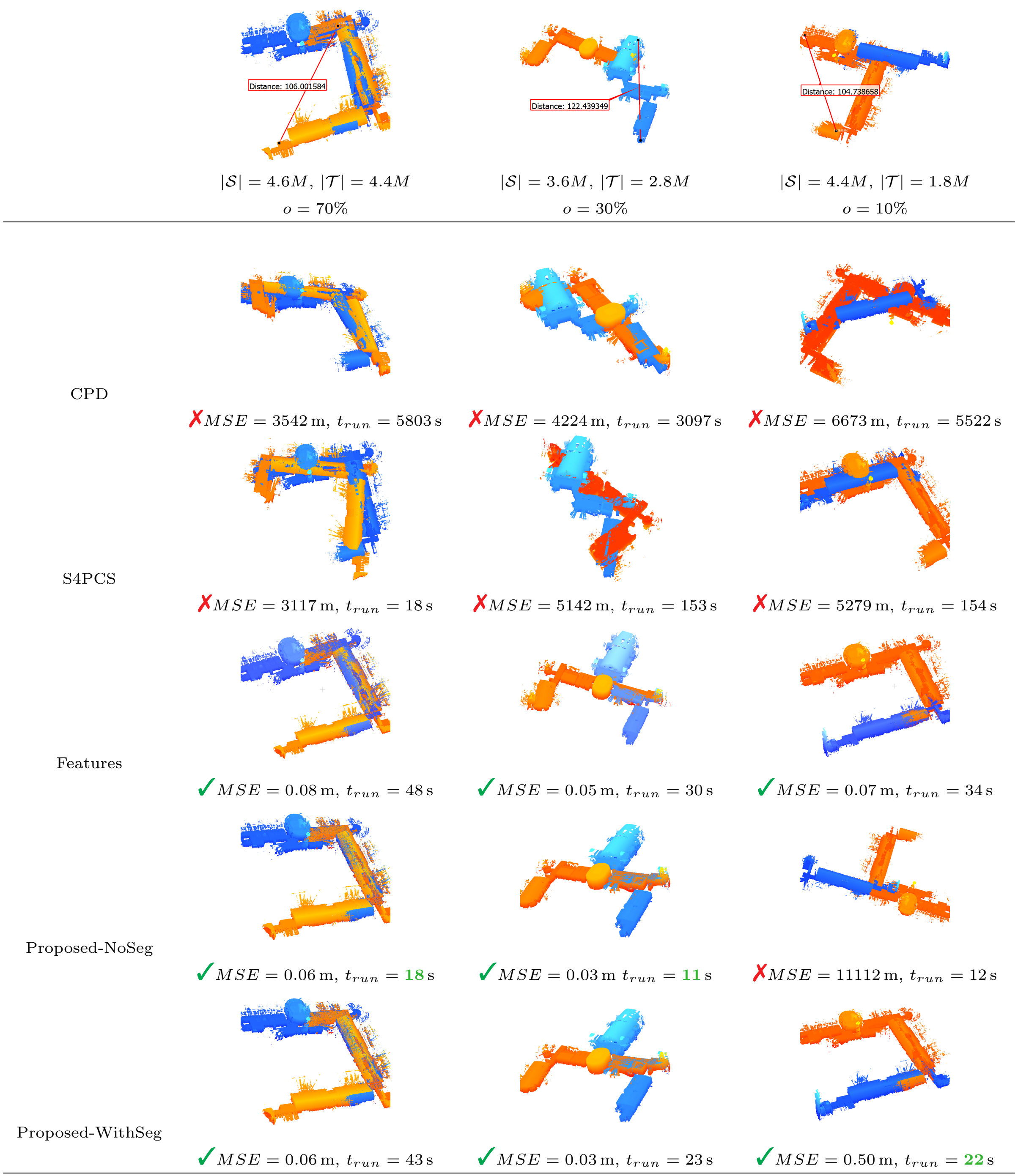

Distance: 106.001584
Distance: 122.439349
Distance: 104.738658
$|\mathcal{S}| = 4.6M,\ |\mathcal{T}| = 4.4M$
$o = 70\%$
$|\mathcal{S}| = 3.6M,\ |\mathcal{T}| = 2.8M$
$o = 30\%$
$|\mathcal{S}| = 4.4M,\ |\mathcal{T}| = 1.8M$
$o = 10\%$
CPD
✗ $MSE = 3542\,\mathrm{m},\ t_{run} = 5803\,\mathrm{s}$
✗ $MSE = 4224\,\mathrm{m},\ t_{run} = 3097\,\mathrm{s}$
✗ $MSE = 6673\,\mathrm{m},\ t_{run} = 5522\,\mathrm{s}$
S4PCS
✗ $MSE = 3117\,\mathrm{m},\ t_{run} = 18\,\mathrm{s}$
✗ $MSE = 5142\,\mathrm{m},\ t_{run} = 153\,\mathrm{s}$
✗ $MSE = 5279\,\mathrm{m},\ t_{run} = 154\,\mathrm{s}$
Features
✓ $MSE = 0.08\,\mathrm{m},\ t_{run} = 48\,\mathrm{s}$
✓ $MSE = 0.05\,\mathrm{m},\ t_{run} = 30\,\mathrm{s}$
✓ $MSE = 0.07\,\mathrm{m},\ t_{run} = 34\,\mathrm{s}$
Proposed-NoSeg
✓ $MSE = 0.06\,\mathrm{m},\ t_{run} = \mathbf{18}\,\mathrm{s}$
✓ $MSE = 0.03\,\mathrm{m}\ t_{run} = \mathbf{11}\,\mathrm{s}$
✗ $MSE = 11112\,\mathrm{m},\ t_{run} = 12\,\mathrm{s}$
Proposed-WithSeg
✓ $MSE = 0.06\,\mathrm{m},\ t_{run} = 43\,\mathrm{s}$
✓ $MSE = 0.03\,\mathrm{m},\ t_{run} = 23\,\mathrm{s}$
✓ $MSE = 0.50\,\mathrm{m},\ t_{run} = \mathbf{22}\,\mathrm{s}$

Figure 5.18: Comparative test with Deutsches Museum (DM) dataset. Each column represents a different source-target cloud pair shown unaligned in the first row. The number of source points ($|\mathcal{S}|$) and target points ($|\mathcal{T}|$) as well as the overlap value used in CPD and Super4PCS (o) is shown below each pair. The alignment outcome obtained with the different approaches are shown in the rows below where Proposed-NoSeg refers to segment-less ground-plane translation alignment as opposed to Proposed-WithSeg. The run times include that required to resolve the rotation ambiguity (see algorithm in Section 5.5). Successful alignment is marked by ✓ and failure by ✗. The method with the lowest run time among all successful ones for a dataset is highlighted in green bold font. The dataset is downloadable at `https://www.lmt.ei.tum.de/downloads/CVIU_DM_Dataset`. Image source: [2] – ©2016 Elsevier.

5.6.7 Non-Manhattan Experiment

The proposed alignment algorithm is tested on a dataset that has strong non-Manhattan characteristics and is shown in Figure 5.19. As expected, the four dominant peaks in the ground-plane rotation signature no longer appear. The subsequent step which uses cross-correlation, which doesn't explicitly require estimation of dominant directions like those of [78], [79], still recovers the true alignment as it exploits the entire distribution of directions of surface normals. A test with a smaller overlap, as expected, failed as the signatures grow increasingly different. Hence, there is a limit to the robustness of this approach when dealing with the not frequently encountered non-Manhattan datasets.

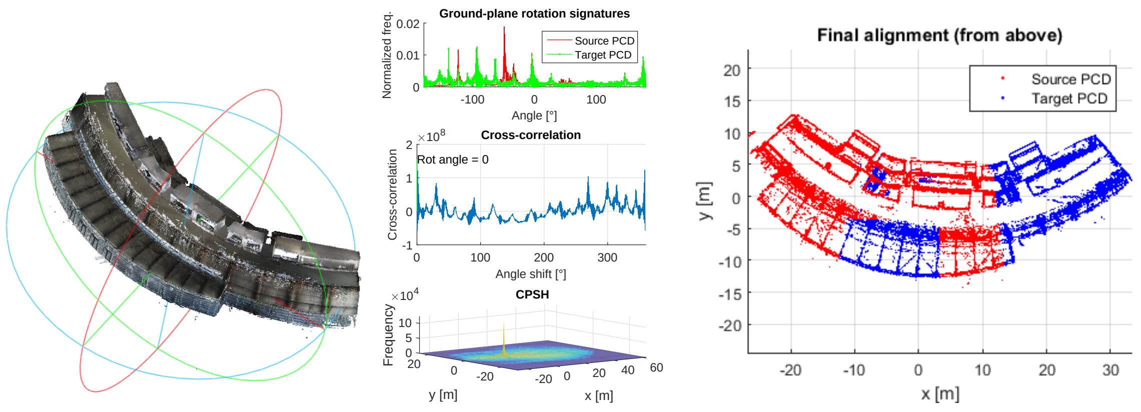


Figure 5.19: Alignment test with point cloud data from a non-Manhattan building (shown left). Alignment conducted with the proposed approach without segmentation. The ground-plane rotation signatures no longer show the dominant four peaks (c.f. Figure 5.4). The "winning" CPSH unambiguously determines the correct ground-plane translation. The final alignment (shown using subsets of the clouds that highlight the contours) is seen to be correct and accurate. Image source: [2] – ©2016 Elsevier.

5.6.8 External Dataset

We apply the algorithm on a 3rd party dataset of Oeseau et al. [77] shown in Figure 5.20. We use the variant provided by Monszpart et al. [85] which already includes normals computed using their RAPter algorithm. We segment the point cloud in two overlapping parts and estimate the curvatures while maintaining the original normals. Despite the errors in the normal estimation [80], the ground-plane rotation alignment is successful. As long as the surface normals do not have systematic errors, the averaging effect of the histograms facilitates successful ground-plane rotation alignment. The final alignment seen in Figure 5.20 is seen to be correct and provides a very good initialization for fine alignment using ICP.

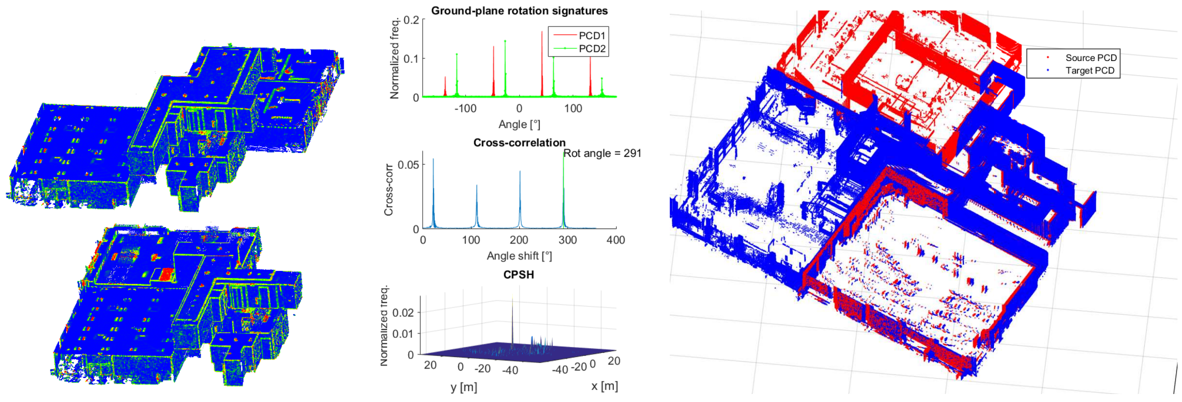


Figure 5.20: Alignment test with the "Euler" dataset of [77]. Data acquired by static mapper (Leica ScanStation P20) put at multiple locations in the environment (different to ours which is obtained by a mobile platform - see Figure 2.1). Alignment conducted with the proposed approach without segmentation using the supplied surface normals without alteration. The point cloud is segmented into a source and target that have ≈ 50% overlap (shown on the left with color-coded curvature values). The ground-plane rotation signatures have very dominant peaks leading to strong cross-correlation at the true ground-plane rotation. The "winning" CPSH out of the four variants computed for the four ambiguous rotations unambiguously determines the correct ground-plane translation. The final alignment (shown using subsets of the point clouds that highlight the contours) is seen to be correct and provides a good initialization for fine alignment. Notice the floor alignment along the z-axis (Section 5.3.1) is successful despite having multiple floors and ceilings as the cross-correlation correctly captures the floor-ceiling relationships. Image source: [2] – ©2016 Elsevier.

Figure 5.21: An example with extremely small overlap. The absence of salient features (objects with corners) means that no reliable CPSH can be computed even if appropriate segments are available. Image source: [2] – ©2016 Elsevier.

5.6.9 Limitations of the Algorithm

In Figure 5.21 a point cloud pair is shown which can not be aligned with our approach. The overlap area makes up a very small fraction of either point cloud. More problematic is the fact that the overlap area has not been even properly mapped (scan artifacts are seen as the scanner never entered the corridor). Also, it is composed mainly of walls without any salient shapes. Hence, even when using segmentation, the CPSH corresponding the segments of the overlap area can not reliably determine the translation. Also the algorithm has a limited robustness towards non-Manhattan datasets as the ground-plane rotation alignment becomes less robust (see Section 5.6.7).

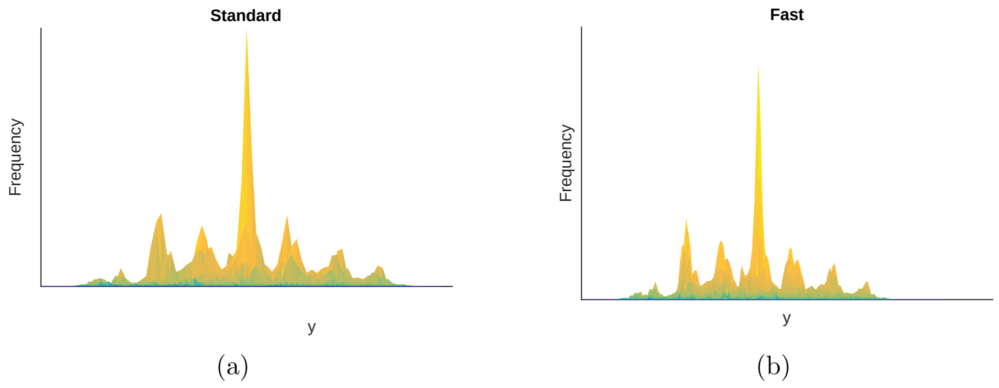


Figure 5.22: Fast histogram computation using histogram convolution of the PMFs computed individually for the source and target. The "standard" CPSH (computed as explained in Section 5.3.2) is seen to be similar to the one computed by simply convolving the PMFs computed individually for the source and target which is much faster (thus the name "fast").

5.7 Fast Histogram Computation Using PMF Convolution

Despite outperforming even efficient algorithms in terms of speed we show here that there is greater potential for efficiency. We address the computationally heavy histogram computation necessary for the CPSH. Since we need to compute four different CPSHs for each source-target segment pair we try to answer the question whether it is possible to circumvent this overhead in computation which is necessary to resolve the ambiguity in ground-plane rotation alignment.

Ideally, the computationally heavy steps are done once and a less demanding computational operation can be used to assemble the CPSH for each of the four different ground-plane rotations. To solve this problem we are served by probability theory which says that the PMF of a subtraction of two independent random variables is simply the convolution of their individual PMFs (with one being flipped). Hence we can easily compute the CPSH as follows

$$f_{i,j}(\mathbf{t}) = p_{T,j}(\mathbf{t}) * p_{S,i}(-\mathbf{t}) \tag{5.22}$$

where $*$ is the 2D convolution and p_T and p_S are the target and source PMFs, respectively. Figure 5.22 shows that the CPSH computed this way is very similar to the one computed using the standard approach (without discriminative voting).

The beauty of this approach lies in the fact that no intermediate subtractions have to be computed while computing the CPSHs which dramatically reduces the computationally burden. Just as important is that fact that to obtain the CPSH corresponding to a different in-ground rotation the source segment PMF has to be simply rotated accordingly before recomputing the convolution.

The CPSH computation is accelerated $\approx 100\times$ using this method. The downside is that this approach in its current shape cannot support discriminative voting using similar normals (see Section 5.3.2.2).

5.8 Summary

In this chapter, a method for the 6DOF alignment of large-scale indoor point clouds is presented. First the point clouds are rotationally aligned to gravity using PCA and then aligned in translation along gravity by matching histograms of point distributions. The gravitationally-aligned PCDs are subsequently aligned in terms of rotation and translation in the ground-plane using a loosely coupled sequential two-step process which involves: computing histograms of surface normal directions from horizontal normals and using cross-correlation to compute the rotation; computing histograms of shift values between pairs of points with high curvature values to estimate the translation, the so-called Curvature-Pair Shift Histograms (CPSH). An extension is presented that involves computing the CPSHs on pairs of point cloud segments and using a maximum likelihood (ML) estimator to fuse multiple hypotheses such that alignment problems with very small overlap can be supported. We show that the ML solution amounts to a simple weighted averaging of the CPSHs. The results show that alignment of point clouds involving multiple million points and with an overlap as low as 10% is achievable with the algorithm irrespective of the initial alignment. We are at least as fast as the most efficient algorithms we compared to despite using a Matlab implementation. We show that the algorithm also works on non-Manhattan datasets.

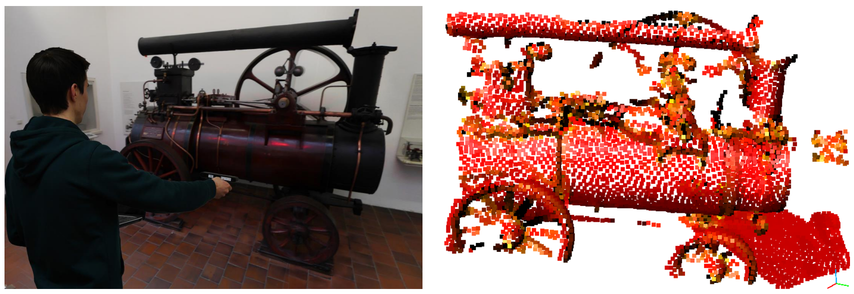
Point clouds aligned with this algorithm can serve as reference clouds for the shape matching-based localization presented in Chapter 6.

Major parts of this chapter have been accepted for publication in the Elsevier Journal on Computer Vision and Image Understanding – Special Issue on Large-Scale 3D Modeling of Urban Indoor or Outdoor Scenes from Images and Range Scans [2].

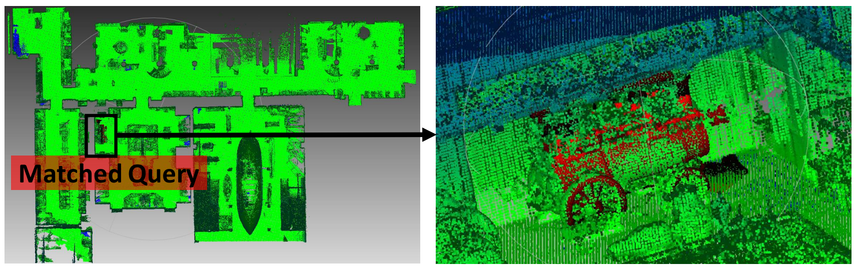
Chapter 6

Indoor Location Retrieval using 3D Shape Matching

In the previous two chapters we dealt with obtaining extensive 3D maps of indoor spaces in the form of point clouds. Meanwhile, the advent of the Microsoft KINECT and similar cheap hand-held 3D sensors has made 3D shape sensing of the local environment easily possible. The KinectFusion algorithm (KinFu) [86] can stitch multiple Kinect depth frames into a more extensive surface scan allowing for the scanning of an object beyond single-view occlusions. Registering such small object models inside large-scale point clouds would allow us to retrieve the accurate location of the sensor and subsequently the person holding it. This contribution



(a) Scanning an object (SteamLocomotive) using a Kinect and KinFu [86] (left) to produce its 3D query scan (right).



(b) Matching the query scan to its respective point cloud using the system explained in Section 6.1 retrieves the Kinect's 6DoF pose identifying the person's location.

Figure 6.1: Pose retrieval of a KinFu scan in a large-scale indoor point cloud to retrieve the location of a person. Image source: [6] – ©2015 Eurographics.

demonstrates that the 6DOF pose of local 3D shape scans obtained with a Kinect-like sensor and KinFu (as shown in Figure 6.1(a)) can be retrieved in a large scale indoor point cloud to accurately retrieve the indoor location of a user (as shown in Figure 6.1(b)). Feature-based registration is used for that purpose (Section 2.2.2.2). Compared to the established camera-based localization schemes [7], [87], [3], [88] which are based on content-based image retrieval it has some fundamental advantages: First, the accuracy is no longer a function of the spatial density of the recorded reference views. Second, the local shape of an object is not affected by the lighting conditions. Third, by using KinFu, a view-independent and largely occlusion-free query can be generated. Finally, the sensor's 6DoF pose can be retrieved achieving superior accuracy.

The used retrieval system (Section 6.1) computes descriptors of 3D keypoints of the KinFu scan and the target point cloud. Good descriptor quality is crucial for successful retrieval. The relatively large scans generated with KinFu in this work, as opposed to the standard table-top scans, exhibit strong distortions in the form of bent surfaces and amplified noise that adversely impact the descriptor quality.

One contribution lies in the analysis of these KinFu scanning artifacts, which arise due to the largeness of the query scans which articulate 3D sensing distortions typical for Kinect-like sensors (Section 6.2). Moreover, the effective reduction of these artifacts through proper 3D sensor calibration using the CLAMS technique [89] together with pre-processing of the final KinFu scan is explained (Section 6.3). The effectiveness of the location retrieval system is demonstrated in Section 6.4 using real data obtained at the *Deutsches Museum* in Munich achieving near-cm-level accurate localization. Finally, an extension based on a Multi-Classifier System is presented that accelerates the location retrieval ten-fold while significantly improving the retrieval performance.

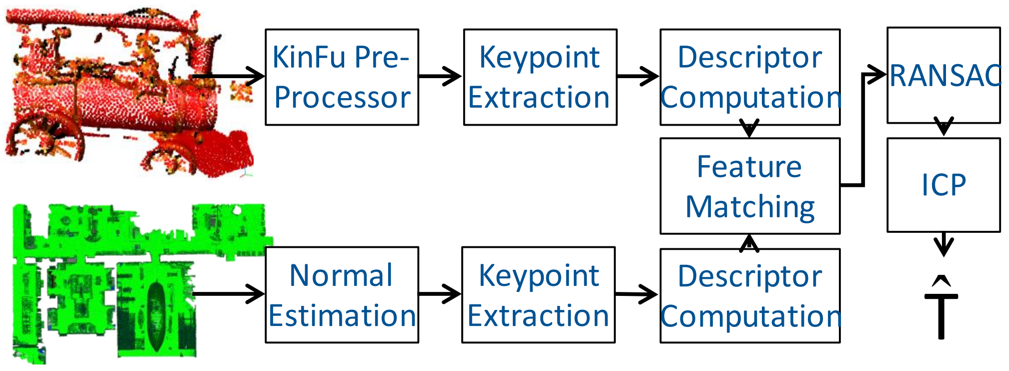


Figure 6.2: Retrieval system: The KinFu query is pre-processed (see Section 6.3). Keypoints are computed for the KinFu query scan and the target point cloud. A descriptor is computed for each keypoint. The descriptors are matched to determine point correspondences between the query and the target. A tentative alignment is computed using RANSAC and refined using ICP producing the 6DoF homogeneous alignment transform $\hat{\mathbf{T}}$. Image source: [6] – ©2015 Eurographics.

6.1 Retrieval System

A point cloud registration system that performs initialization-insensitive *part-in-whole* shape matching (as defined by Tangelder and Veltkamp in [90]) was built. It is used to retrieve the 6DoF pose of the KinFu scan (henceforth called *query*) in the indoor point cloud (henceforth called *target*). Figure 6.2 shows that the KinFu scans are first pre-processed – as explained in detail in Section 6.3 – to handle scanning distortions and produce reliable surface normals. The normals for the target are also computed. As explained in Section 2.2.2.2, 3D keypoints are first detected around which descriptors are computed. The descriptors of the query are matched to the target descriptors to establish query-target point correspondences. A random sample consensus (RANSAC) estimator is used to validate the correspondences and estimate the 6DoF transformation that aligns the query to the target. The iteration with the highest amount of *inliers* delivers the used transformation hypothesis. Finally, the Iterative Closest Point (ICP) algorithm (Section 2.2.3) runs to retrieve the pose more accurately expressed as the 6DoF homogeneous transformation $\hat{\mathbf{T}}$.

The only difference to the feature-based registration system used in Section 5.6 is the used keypoint type. Here we use the Intrinsic Shape Signature (ISS) of Zhong [91] which has been shown to outperform many standard detectors in terms of relative repeatability under various distortions and transformations [92].

The results of Section 5.6 showed that feature-based registration compares favorably in terms of speed and registration success when used with large-scale point clouds. Here, we will show that it performs quite well also for this extreme case of part-in-whole matching. Notice, the feature-based system is particularly relevant here due to the fact that we are dealing with articulated objects. This is a distinct difference to 4PCS and Super4PCS [42] which are more suited for cases with dominant semi-planar surfaces. In fact, several tests with S4PCS failed to deliver appropriate results.

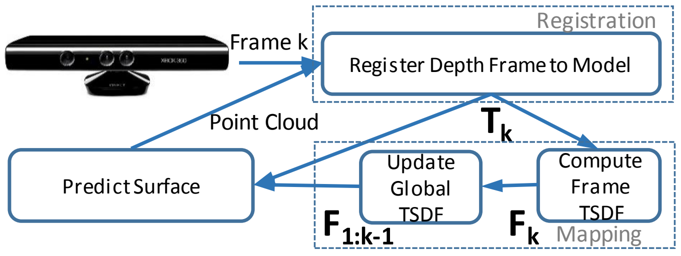


Figure 6.3: KinFu overview. An incoming depth frame is aligned to the most recently predicted surface to produce the transformation matrix $\mathbf{T}_k$. The Signed Distance Function (SDF) of the registered frame $\mathbf{F}_k$ is computed and fused with the cumulative SDF $\mathbf{F}_{1:k-1}$ to produce $\mathbf{F}_{1:k}$. A new surface is predicted from the viewpoint $\mathbf{T}_k$ to be used in the alignment of the next depth frame. Image source: [6] – ©2015 Eurographics.

6.2 KinFu Scan Issues

The KinFu scans suffer from distortions that can be attributed to two main sources: the sensor data and the KinFu reconstruction algorithm. The distortions are explained in detail in Section 6.2.2 preceded by a brief explanation of KinFu in Section 6.2.1 to aid in understanding the distortions.

6.2.1 KinFu Algorithm

As shown in Figure 6.3, KinFu has two main processing functions: registration and mapping. These processing functions are interdependent whereby the outcome of registration is used during mapping and the outcome of mapping is used for the registration of a newly incoming depth frame.

During registration, a new incoming depth frame is registered to the local scene to retrieve the 6DoF pose of the 3D sensor. ICP with the point-to-plane metric (Section 2.2.3) is used for this purpose. In KinFu an incoming depth frame is registered against the most recently updated 3D shape model of the scene obtained through mapping resulting in highly accurate registration. This, however, requires updating the scene's 3D scene model at frame rate.

To compute the 3D scene model at frame rate a volumetric scene representation based on the truncated Signed Distance Function (SDF) [93] is used. The SDF captures for each point in a cubic volume encompassing the scene the minimum distance to the 3D surface. Two SDFs are maintained: One that accumulates the knowledge about the surface over multiple frames and another computed only using the sensed surface in the current registered depth frame.

KinFu uses a discretized lattice of 3D points as an approximation of a continuous SDF. So a cubic volume of side length l is subdivided into *voxels* of side length (l/m). l is adapted to the largeness of the scene ($l = 300\,\text{cm}$ in our case). m is usually limited by the graphics card memory (we use $m = 512$ as in the original KinFu paper). The ratio l/m determines the granularity with which the surface is mapped.

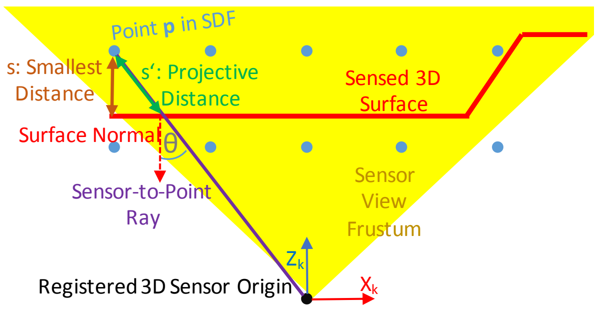


Figure 6.4: KinFu uses the "projective distance" in the SDF computation which is an approximation of the true smallest distance to the sensed surface. The projective distance s' for point $\mathbf{p}$ in the SDF is always an overestimation of the true distance s. The incurred error increases with increasing θ. Image source: [6] – ©2015 Eurographics.

The SDF value at any point $\mathbf{p}$ in the lattice should be the smallest distance from the point to the sensed surface. KinFu, however, approximates this distance, as shown in Figure 6.4, by computing the projective distance. It is obtained along the ray connecting $\mathbf{p}$ to the sensor origin. It is argued that this approximation still leads to good mapping results while allowing computing the SDF at high frame rates.

For any depth frame k, the SDF volume $\mathbf{F}_k$ is computed. and subsequently fused with the cumulative volume $\mathbf{F}_{1:k-1}$ using a per-voxel simple running average update rule. Specifically, the new SDF value at point $\mathbf{p}$ is computed as follows:

$$\mathbf{F}_k(\mathbf{p}) = \frac{\mathbf{W}_{1:k-1}(\mathbf{p})\mathbf{F}_{1:k-1}(\mathbf{p}) + \mathbf{W}_k(\mathbf{p})\mathbf{F}_k(\mathbf{p})}{\mathbf{W}_{1:k-1}(\mathbf{p}) + \mathbf{W}_k(\mathbf{p})} \tag{6.1}$$

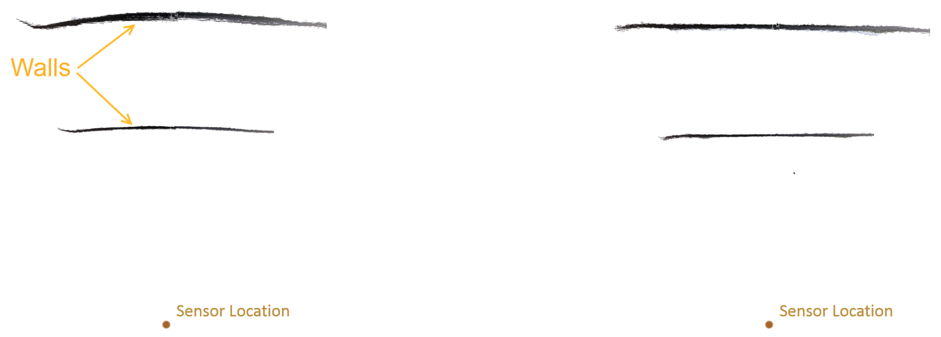
with $\mathbf{W}_{1:k-1}$ being the accumulated-weight volume, i.e. $\mathbf{W}_{1:k} = \mathbf{W}_{1:k-1} + \mathbf{W}_k$*.

Once the current mapping iteration is done, the surface is partially predicted from the perspective of the currently registered frame. This is used to provide a reference surface to be used in the registration of the next incoming depth frame. Hence, abrupt trajectory changes and movements can lead to ICP failure. At the end of the scanning, the most recently obtained cumulative SDF is used to produce a 3D mesh. The zero-crossings inside the SDF represent the surface.

6.2.2 KinFu Query Scan Distortions

Surface bending. One fundamental issue we have faced is related to bent planar surfaces as shown in Figure 6.6. This issue can be mainly attributed to the raw Kinect data. We have observed that the raw 3D data suffers from non-linear distortions as shown in Figure 6.5(a). Critically, planar surfaces appear curved and the curvature increases with increasing distance from the scene. Teichman et al. [89] show that this is especially true for Kinect-like sensors

*The multiplication in (6.1) is an element-wise multiplication.



(a) Without CLAMS calibration. (b) With CLAMS calibration.

Figure 6.5: 3D sensor raw data (Asus Xtion Pro Live) of a wall scanned from two distances (1.5 m & 2.5 m) shown as a point cloud from above. Despite IR camera calibration a bending of the wall is observed. The curvature of the bending increases with increasing distance from the sensor. Calibration with CLAMS [89] effectively reduces the bending.

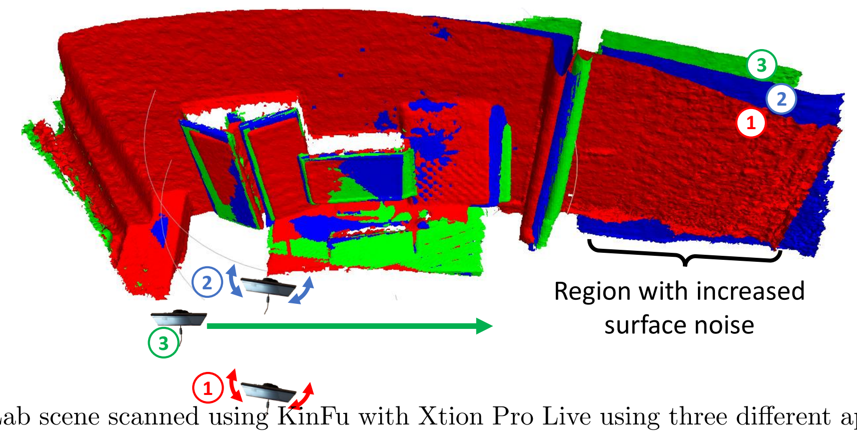


Figure 6.6: Lab scene scanned using KinFu with Xtion Pro Live using three different approaches: (1) Standing still; (2) Same as 1 but *closer* to the wall; (3) Scanning sideways while remaining close to the wall. The generated point clouds prove that surface bending is less in (2) than in (1) due to the decreased curvature of the raw 3D data at lower distances as shown in Figure 6.5. Maintaining a close distance to the surface as in (3) further reduces the bending. Image source: [6] – ©2015 Eurographics.

(Microsoft Kinect, Asus Xtion, etc.). The Xtion is particularly interesting because it can be carried around and thus lends itself for our application. In our case we use the Xtion Pro Live.

Considering that KinFu essentially runs mapping and registration on each frame in succession, the bending of the raw 3D data is particularly harmful. Initial surfaces exhibiting the bending will cause new depth frames to be registered slightly wrong. This in turn results in a wrong mapping update in the cumulative SDF which in turn affects future registrations. As a result the error propagates and with increasingly larger scans the bending effect is accentuated.

To prove this the following experiment was performed: KinFu is used to scan a scene at our lab which includes a large wall as well as some articulated objects to ensure accurate registration. A number of different scans were performed. In the first, identified by label (1) in Figure 6.6 the Kinect is held far from the wall and is panned left and right. In the second, a similar scan is performed, however at a close proximity from the wall. The generated point clouds, shown in the same figure, clearly exhibit bending which increases towards the edges. However and as expected, the bending of scan (2) is notably less than that in (1). In a third experiment the wall is scanned while moving the Kinect parallel to it. As can be seen in Figure 6.6, experiment (3) exhibits a substantially reduced bending compared to (1) and (2) albeit at the cost of making the scene scanning complicated. The bending problem was less severe when we used a Microsoft KINECT which, however, is less portable than the XTION.

It remains to be said that the errors in depth impact also all coordinates of the scan points computed from the depth image as these are computed using the pin-hole camera model. Indeed, Figure 6.6 shows that the less the wall is bent, the larger is also the extent of the scene and the scan dimensions are closer to reality which is important for the descriptors.

Sensor noise. Besides surface bending another issue we have to deal with is surface noise which affects surface normal estimation. The used SHOT descriptor as well as all other mentioned descriptors in [36] rely on surface normals.

Point $\mathbf{p}_i$'s surface normal can be typically computed by first computing the covariance matrix of the points in $\mathbf{p}_i$'s neighborhood as follows:

$$\mathbf{C}_i = \sum_{\mathbf{p_j} \in \mathcal{N}_i} (\mathbf{p}_i - \mathbf{p}_j)(\mathbf{p}_i - \mathbf{p}_j)^T \tag{6.2}$$

where $\mathcal{N}_i$ is the set of points within radius r_n of $\mathbf{p}_i$. The eigenvector corresponding to the smallest eigenvalue of $\mathbf{C}_i$ is deemed as the normal vector.

Surface noise and surface distortions can have significant impact on the covariance matrix $\mathbf{C}$ and the computed eigenvectors [94]. This is especially critical when r_n is small. Unfortunately, this is the case with our museum KinFu scans which have articulated small shapes which require r_n to be small (14 cm). Figure 6.10 shows the surface curvature of an example query scan. The unprocessed scan surface is noisy and results in noisy normals.

A fundamental noise source is again the 3D sensor. Assuming a Gaussian error model, Khoshelham and Elbrink [95] showed that the standard deviation in the measured depth of a point by Kinect can be given as:

$$\sigma_d = \sigma_\delta d^2 \alpha \tag{6.3}$$

where d is the depth of the point, σ_δ is the standard deviation of the measured disparity, and α is a constant that depends on the Kinect camera parameters. So in essence, the Kinect depth error increases quadratically with increasing depth. KinFu implicitly runs a maximum likelihood estimation over multiple measurements from multiple frames by averaging new SDFs into the cumulative SDF using (6.1). This results in substantially smoother surfaces when comparing to the raw point clouds delivered in individual frames. Nevertheless, the noise will inevitably increase with increasing distance from the scene as the variance in the estimate itself increases.

Due to the use of the projective distance when computing the SDF, see Figure 6.4, the noise can furthermore be amplified for points with unfavorable scanning conditions (large θ in Figure 6.4). This is because the projective distance will always overestimate the true smallest distance to the sensed surface. For the case shown in Figure 6.4 the projective distance s' for the point $\mathbf{p}$ can be computed as a function of the true signed distance s as:

$$s' = s/cos(\theta) \tag{6.4}$$

The error in Kinect depth measurement leads to an error in the signed distance function that can, for the example shown in Figure 6.4, also be described by a Gaussian model. However, the standard deviation is amplified and can be computed as:

$$\sigma_{SDF} = \sigma_d/cos(\theta) \tag{6.5}$$

Hence, the larger the angle between the surface normal and the sensor-to-SDF-point ray the larger is the variance in the projective signed distance. Since the Kinect and the Xtion both have a horizontal field of view (FoV) of around 60°, the largest value for θ for the case shown in Figure 6.4 is $60/2 = 30°$. In this case the standard deviation according to Equation 6.5 increases by 15%. For surfaces that are not perpendicular to the sensor's z-axis (z_k in Figure 6.4) even higher amplifications can occur.

Surface points lying on the fringe of a large KinFu scan are worse off in terms of surface noise compared to other points. They are affected more by the variance amplification explained above. Moreover, they are sensed by relatively few frames and thus will not benefit as much from the SDF averaging. This explains the visible increase in surface noise seen in the right part of the scans in Figure 6.6.

It is important to note that these two fundamental KinFu scan distortions, the surface bending and the surface noise, arise particularly due to the largeness of the scans as opposed to the relatively small scenes presented in the KinFu paper [86]. Hence, they deserve special attention and proper processing to ensure good location retrieval performance.

6.3 Sensor Calibration and KinFu Scan Pre-Processing

Sensor calibration. One of the main issues with our relatively large KinFu query scans is surface bending. Experiment (3) shown in Figure 6.6 showed that it is principally possible to mitigate the bending issue in the raw data by scanning a scene from close proximity which requires large translations to cover the entire scene. This may not be practical as the close proximity greatly increases the probability of ICP failure as less salient objects are seen.

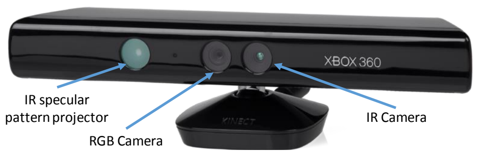


Figure 6.7: Kinect-like sensors emit a specular pattern using an IR projector whose reflection is captured by an IR camera. The projector-camera pair acts as a stereo camera used to recover the 3D shape of the surface on which the specular pattern is projected. The images returned by the RGB camera, which is calibrated w.r.t. the IR camera, can also be used to color the resulting point cloud.

Since Kinect-like 3D sensing is based on stereo-camera techniques (see Figure 6.7) one might think of camera calibration (of the IR camera used to capture the specular pattern reflection) as an obvious solution to the bending issue. Indeed, radial distortion will definitely affect the disparity computation and the severity of its impact changes as a function of pixel location. Note that the farther away the pixels from the center, the larger the distortion which may explain why the depth values are less accurate towards the edges of the depth map.

While a standard camera calibration of the IR camera can accurately compute the focal length, principal point and radial distortion coefficients, these parameters cannot be uploaded onto the Kinect's chip that performs the disparity computation (the Kinect computes the disparity and subsequently depth values internally). The computed calibration can be at best used to produce better point clouds from the depth maps that the Kinect finally delivers. We performed standard IR camera calibration and as expected the improvements to the point clouds were insufficient and the bending artifacts remained severe.

Teichman et al. [89] investigated 3D sensor calibration of Kinect-like sensors and showed that such devices are essentially *myopic* in terms of their distortion characteristics. Like us, they observed that depth images (and their derived point clouds) exhibit a bending that increases with distance. As a solution they propose using mutliplicative depth compensation factors that are learned as a function of depth and pixel region. They are learned by Calibrating, Localizing, and Mapping, Simultaneously (CLAMS).

CLAMS essentially runs simultaneous localization and mapping (SLAM) on the RGBD data of a Kinect-like sensor. The sensor trajectory is estimated. This is used to build a 3D model of the scene, however, only using reliable depth data (depth $< 2\,\mathrm{m}$). Finally, all depth data from all pixels of each frame are used to compute the depth error at different pixel regions and different depth levels to compute the compensating multiplicative factors.

We used the CLAMS technique to calibrate our Xtion RGBD sensor. The learned *depth multiplier images* are shown in Figure 6.8. Applying the learned model on the raw 3D sensor data leads to visible improvements as shown in Figure 6.5(b) and the bending is largely removed. Using the undistorted depth images KinFu can produce scans without bending artifacts as shown in Figure 6.9(a). If the scan is relatively small, however, and scanned from a close distance no visible improvement can be observed as seen in Figure 6.9(b).

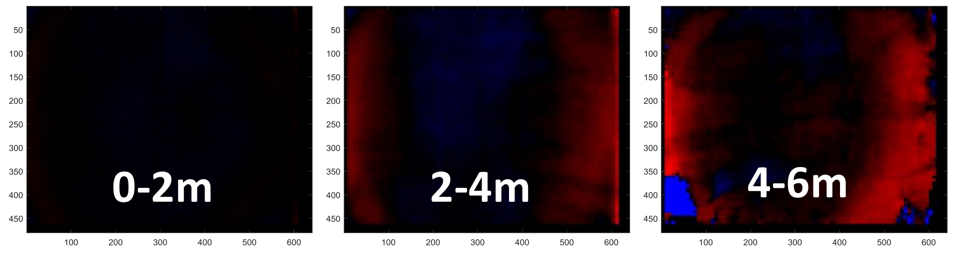
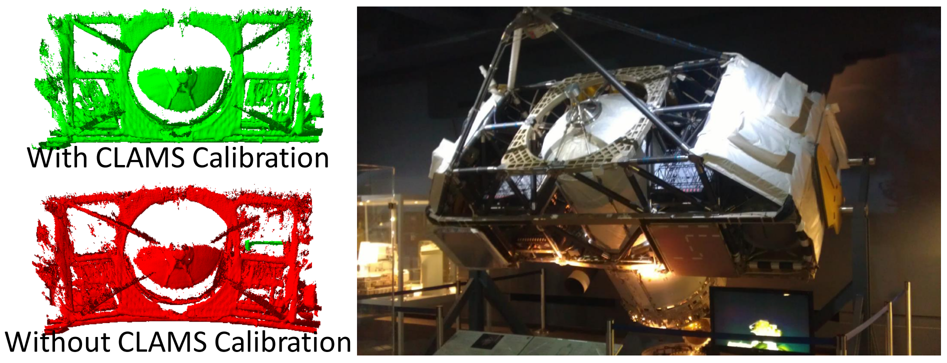
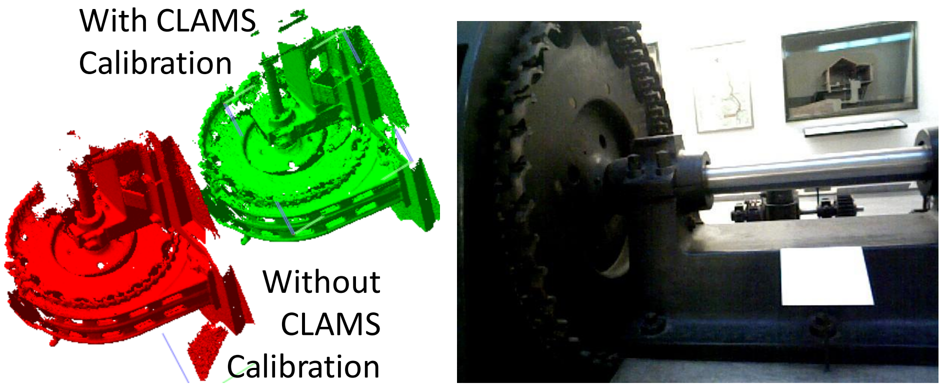


Figure 6.8: Learned depth multiplier images using CLAMS for three depth levels. Red regions imply multiplicative factors that lead to depth value decrease. The color intensity is directly related to the amount of applied correction. Clearly, at higher depth larger corrections are needed to compensate the depth errors. Also, more compensation is necessary when deviating away from the principal point to offset the bending. Image source: [6] – ©2015 Eurographics.



(a) Astro-Spas KinFu Query Scan.



(b) Generator3 KinFu Query Scan.

Figure 6.9: Two query scans generated once with prior CLAMS sensor calibration and once without. Especially large scans benefit from the calibration and do not exhibit the bending artifact. Image source: [6] – ©2015 Eurographics.

KinFu surface pre-processing. Having obtained unbent KinFu scans we address the remaining issues highlighted in Section 6.2. First, we reduce surface noise using a moving least squares (MLS) filter. Spurious points and remaining noise that does not fit with the local surface point statistics are treated using a sparse outlier removal (SOR) filter.

The MLS filter is a projection-based procedure that approximates surfaces locally by polynomial functions [96]. This leads to a shape-aware smoothing of the point cloud in which

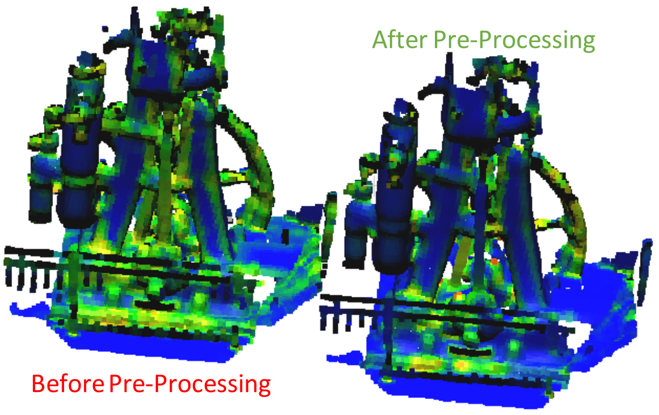


Figure 6.10: KinFu query pre-processing shown for SteamEngine. The color represents the surface curvature. The greener, the higher the curvature value. Prior to retrieval MLS and SOR filters (see Section 6.3) are used to reduce surface noise and eliminate spurious points. Especially flat surfaces as well as edges benefit from the filtering. Image source: [6] – ©2015 Eurographics.

sharp discontinuities are preserved while effectively reducing noise.

To deal with the shadow surface problem as well as spurious points and remaining noise an SOR filter [97] is used. The SOR filter learns the global point statistics to detect outliers and filter them out. It computes the mean and the variance of the avearge distance to the r-NNs of each point. The learned statistics are used to filter out points whose average r-NN distance is substantially different from the learnt mean distance.

The combined effect of MLS and SOR filtering are less noisy surfaces as shown in Figure 6.10. Once filtering is finished, we estimate surface normals and disambiguate them to a consistent orientation that agrees with that of the respective part in the target point cloud.

6.4 Evaluation

The used query scans and target clouds are introduced in Section 6.4.1 followed by an explanation of the evaluation metrics in Section 6.4.2, the used retrieval parameters in Section 6.4.3, the obtained results in Section 6.4.4 and finally the analysis in Section 6.4.5.

6.4.1 KinFu Queries and Reference Point Clouds

We recorded a set of 9 queries in the *Deutsches Museum* (DM) with an Xtion Pro Live. The same set of depth images is fed to KinFu once undistorted with the learned CLAMS model and once without distortion compensation. Once the query scans have been generated they are pre-processed as explained in Section 6.3.

The target clouds have been recorded using the LMT mapping trolley [18]. The target clouds cover up to 3500 m^2 of floor space and encompass multiple exhibition areas. Figure 6.11 shows the target cloud for the queries GF200-Plane, Generator1 and Generator3.

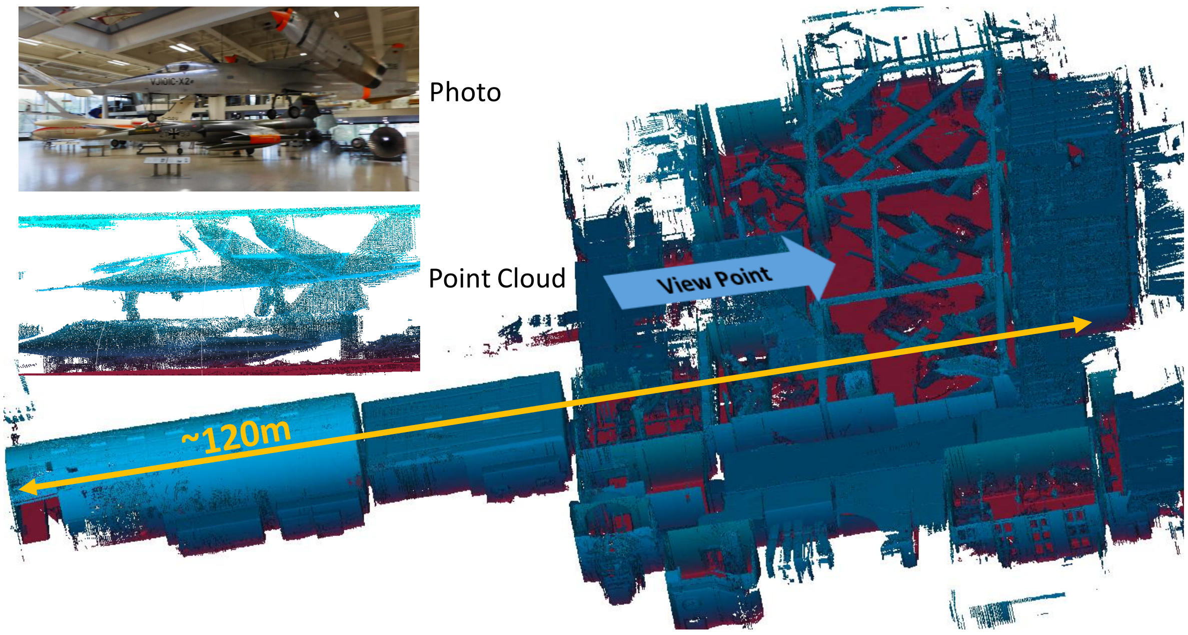


Figure 6.11: One of the used target clouds including the airplanes exhibition. A photo and the corresponding part in the point cloud are shown from the indicated view point. Image source: [6] – ©2015 Eurographics.

6.4.2 Evaluation Metrics

While retrieving any query we measure the true correspondence rate (TCR). This is the fraction of correspondences that adhere to the ground transformation $\mathbf{T}$. Also, the finally computed transformation using RANSAC and ICP is checked for correctness. This process is repeated 100 times to give reliable results as RANSAC is random. The percentage of successful retrievals from within the 100 runs defines the *precision* of retrieval. For each successful retrieval we measure the accuracy of retrieval. For that, again, the error transformation $\mathbf{T}_e$ is first computed and used to derive the error angle α_e and the translation error t_e as explained in Section 3.4.1. The accuracy values are averaged over all successful runs of the respective query.

6.4.3 System Parameters

The keypoint and descriptor radii have been tuned to 10 cm and 1 m, respectively. The order of the polynomial for the MLS filter is 4. The MLS search radius is 5 times the mesh resolution. For the SOR filter, $r = 60$ nearest neighbors is used for the statistics.

6.4.4 Evaluation Results

Comparing the TCR values in Table 6.1 it can be seen that the CLAMS calibration leads to an increase in TCR in 6 out of 9 queries. For the remaining three queries the decrease

Table 6.1: Retrieval results using the evaluation metrics introduced in Section 6.4.2. The results are shown in terms of the true correspondence rate (TCR) and the retrieval precision for two cases: No CLAMS calibration and with CLAMS calibration. The retrieval accuracy in terms of error angle (α_e) and error translation (t_e) averaged over all successful retrievals (thus "weighted" average) is shown for the case with CLAMS. The evaluation metric are explained Section 6.4.2.

Query	TCR (%)		Precision (%)		t_e (cm)	α_e (°)
	No CLAMS	With CLAMS	No CLAMS	With CLAMS		
FrancisTurbine	25.0	31.9	100	100	3	0.8
GirardTurbine	14.3	20.8	86	100	6	1.9
Astro-Spas	27.0	49.4	100	100	0	0.0
GF200-Plane	13.6	12.9	78	96	9	2.1
SteamLocomotive	7.5	14.4	15	92	5	5.7
SteamEngine	20.5	13.8	0	100	5	2.4
Balloon	16.2	23.7	0	70	4	0.5
Generator1	20.2	23.4	100	100	4	1.9
Generator3	7.8	7.4	74	65	17	9.3
Weighted average	16.9	22.0	61	91	6	2.5

in TCR is notable only in one query (SteamEngine) while it is less than 1% in the other two. The increase in TCR can reach up to 22.4% and averages 5.1%. Nevertheless, it can be seen that even after pre-processing, the TCR is relatively low averaging 22% which is a direct consequence of the extreme case of part-in-whole matching that this application is.

The increase in TCR is seen to have a large impact on the retrieval precision which rises from 61% to reach 91%.

Table 6.1 also shows that a true correspondence rate as low as 12.9% is sometimes enough to lead to a 96% precision (GF200-Plane).

The average error in the retrieved orientation is 2.5°and the average location accuracy is 6 cm. These accuracy values are beyond any of those obtainable with a standard visual localization system [98].

6.4.5 Analysis

The results in Section 6.4.4 show that proper calibration for our relatively large KinFu scans, as opposed to simple IR camera calibration, leads to a significant improvement in retrieval results. Especially large query scans such as FrancisTurbine, GirardTurbine, Astro-Spas and SteamLocomotive benefit greatly from the calibration either in terms of true correspondence rate (TCR) or precision or both.

The precision values for the queries SteamEngine and Balloon rise, as can be seen in Table 6.1, from 0% to 100% after CLAMS calibration. While this is easily justifiable in

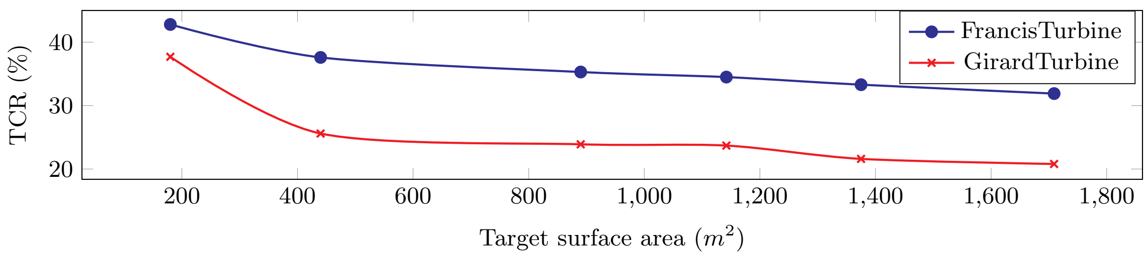


Figure 6.12: The true correspondence rate (TCR) of GirardTurbine reduces substantially with increasing target cloud floor size as opposed to FrancisTurbine hinting that the query itself is less distinctive. Image source: [6] – ©2015 Eurographics.

the case of Balloon through the increase in TCR, it seems counter intuitive in the case of SteamEngine whose TCR decreases after CLAMS calibration. A deeper inspection shows that while the TCR decreases, the actual absolute number of true correspondences increases by 33%. In fact, the absolute number of true correspondences, not shown in Table 6.1, increases for all nine queries after CLAMS calibration. This increase is effectively exploited by our RANSAC implementation which includes a built-in false correspondence rejector that will be explained later.

The TCR of the GirardTurbine is 10% less than that of FrancisTurbine which is located beside it in the museum. We argue that the problem is related to the lack of intrinsic distinctiveness of the shape itself. To prove this we compare the reduction in TCR of both scans as we match each one of them to increasingly larger cutouts of their common target cloud. We argue that a distinctive query exhibits a stable TCR irrespective of the target size. The curves in Figure 6.12 indeed show a large decrease in the TCR of GirardTurbine as the target cloud increases as opposed to FrancisTurbine whose TCR decreases at a far less rate. The GirardTurbine query scans from multiple matching runs without CLAMS calibration are visualized after alignment in red in Figure 6.13. It can be seen that occasionally the query gets matched to the nearby turbines. This problem is not observed in the case of CLAMS calibration. All 100 retrieval attempts succeed in that case.

An interesting case is the GF200-Plane query which does not benefit from the calibration despite being relatively large. The plane body has few distinctive local features. Moreover the target cloud includes many planes (see Figure 6.11) which are similar in shape. Accordingly, the CLAMS calibration does not lead to an improvement. Despite the low true correspondence rate, our RANSAC implementation still manages to retrieve the location correctly in 96/100 runs.

The results in Table 6.1 show that the average true correspondence rate is generally low even after CLAMS calibration. This is mainly due to the fact that the queries make up a tiny fraction of the large-scale target clouds. Despite the low true correspondence rates, the final retrieval is very precise on average. This is a testimony to the robustness of RANSAC and the used parameters. One fundamental feature of the RANSAC we implemented is a

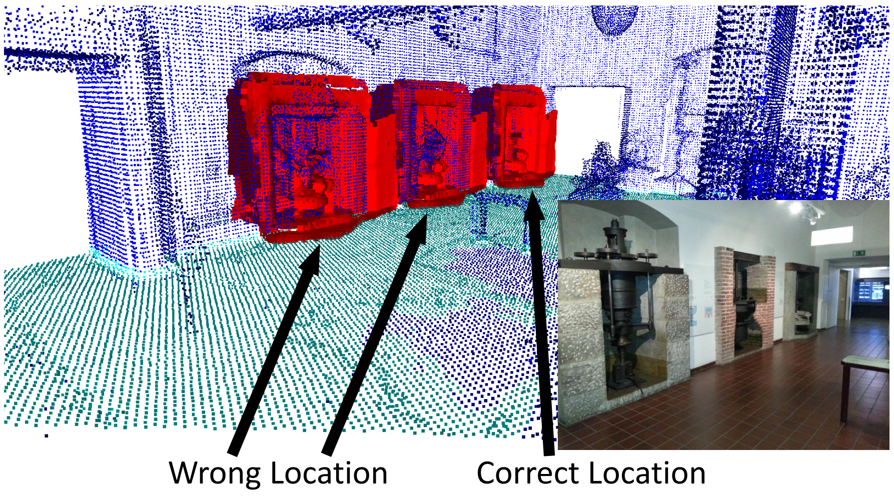


Figure 6.13: The results of multiple retrievals of the GirardTurbine query (red) are shown overlapped on the same target cloud (blue). Some retrieval attempts match the query to neighboring turbines displaying the issue of distinctiveness. Image source: [6] – ©2015 Eurographics.

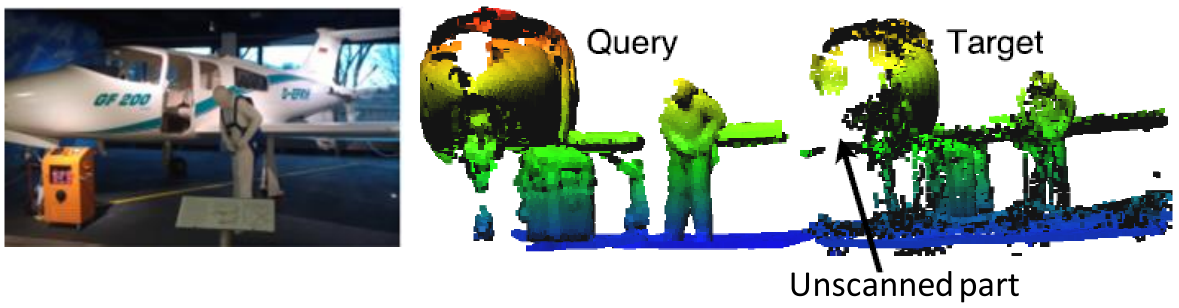


Figure 6.14: Occlusion in the target cloud due to exhibition restriction leading to a low true correspondence rate for the GF200-Plane query.

built-in false correspondence rejector. The rejector validates any sampled correspondence with already pre-sampled correspondences in the same iteration. The validation is achieved by checking whether the spatial distances to the other samples on the query side are preserved on the target side, exploiting a fundamental property of the Special Euclidean Group $\mathbb{SE}3$ as illustrated in Figure 6.15. RANSAC's success depends on the probability of finding a subset of points which are inliers. The probability of such can be given as

$$p_v = p(s_1, s_2, s_3) = p(s_3|s_2, s_1)p(s_2|s_1)p(s_1). \tag{6.6}$$

with s_i meaning that sample i is an inlier. Given that $p(s_i) = p_s$, when sampling randomly p_v becomes $p_{v,rand} = p_s^3$. If however, the geometry aware sampling is used p_v becomes

$$p_{v,geom} = p(s_3|s_2, s_1)p(s_2|s_1)p(s_1) = p(s_3|s_1)p(s_2|s_1)p(s_1) \approx p(s_1) = p_s \tag{6.7}$$

as we assume that geometrically consistent samples most likely are inliers given that the first sampled correspondence is an inlier. Hence, the probability of picking an outlier-free set is significantly increased. This also has a direct consequence on the number of required iterations, N, which is a function of p_v and the required success probability p_S:

$$N \geq log(1 - p_S)/log(1 - p_v). \tag{6.8}$$

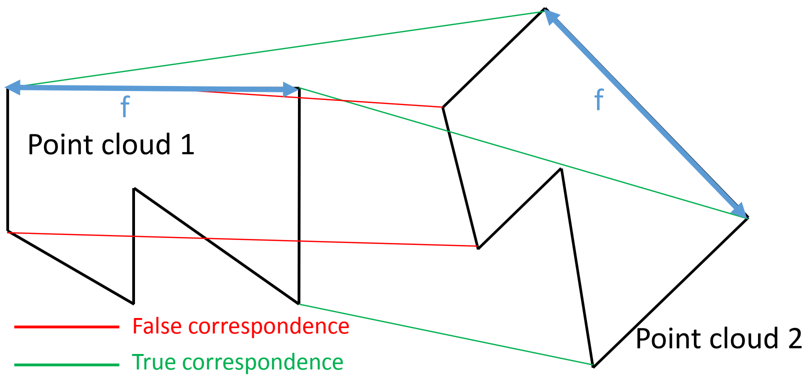


Figure 6.15: Geometry-aware RANSAC. Two point clouds to be registered are shown connected by true and false correspondences. First a correspondence is randomly sampled. Another correspondence is sampled next. Only if the distance of its point in PCD 1 to correspondence 1's point in PCD 1, f, is also preserved in PCD 2 is this correspondence added proceeding to sampling the final correspondence needed for RANSAC which is chosen similarly.

6.5 Retrieval Acceleration

Here, we deal with a fundamental problem of the proposed location retrieval system, namely retrieval time. In this section an extension of the retrieval system is presented which accelerates the retrieval by an order of magnitude while also improving the retrieval performance.

The main problem with the retrieval system shown in Figure 6.2 is that it requires matching keypoints in the query with all keypoints found inside the large-scale reference point cloud to establish correspondences. This not only leads to high computational times but also decreases the accuracy as there are many more false correspondence candidates per query keypoint.

In the extension we propose to make the query-reference correspondence search faster and more accurate by choosing a subset of the reference point cloud which is more likely to encompass the object also captured by the query.

Choosing a subset of the reference point cloud is based on two fundamental steps: 1)Segmentation of the reference point cloud into semantically meaningful objects; 2)The usage of a Multiple Classifier System (MCS) for identifying which objects are more likely to correspond to the query scan. This is illustrated in Figure 6.16. To segment the reference point cloud into semantically meaningful objects we first run keypoint detection on the entire reference point cloud and compute a histogram of keypoint location using Kernel Density Estimation (KDE). The peaks of the histogram identify distinct objects. This is based on the observation

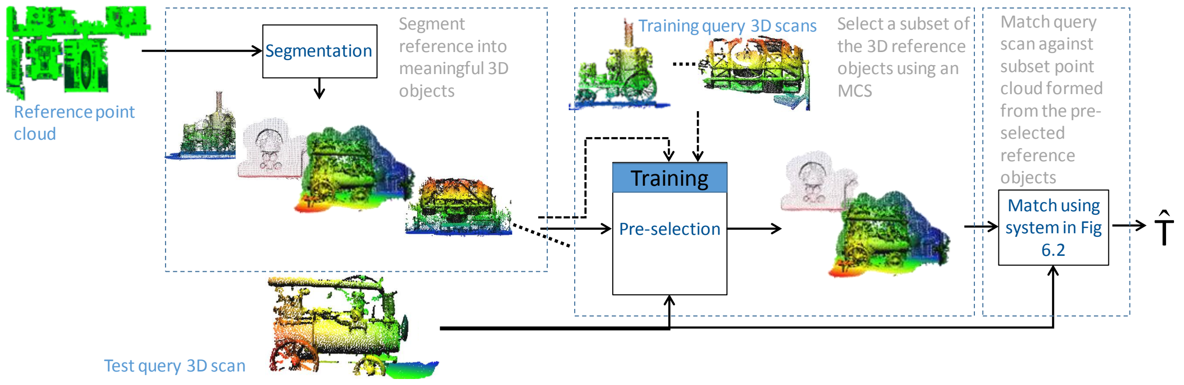

Reference point cloud
Segmentation
Segment reference into meaningful 3D objects
Training query 3D scans
Select a subset of the 3D reference objects using an MCS
Training
Pre-selection
Match query scan against subset point cloud formed from the pre-selected reference objects
Match using system in Fig 6.2
T̂
Test query 3D scan

Figure 6.16: Accelerating the location retrieval using an MCS. The reference point cloud is segmented into 3D reference objects. An MCS is trained to pre-select a few relevant reference objects when presented with a query scan. The location retrieval as explained in Section 6.1 is run only on the subset point cloud assembled from the pre-selected reference objects.

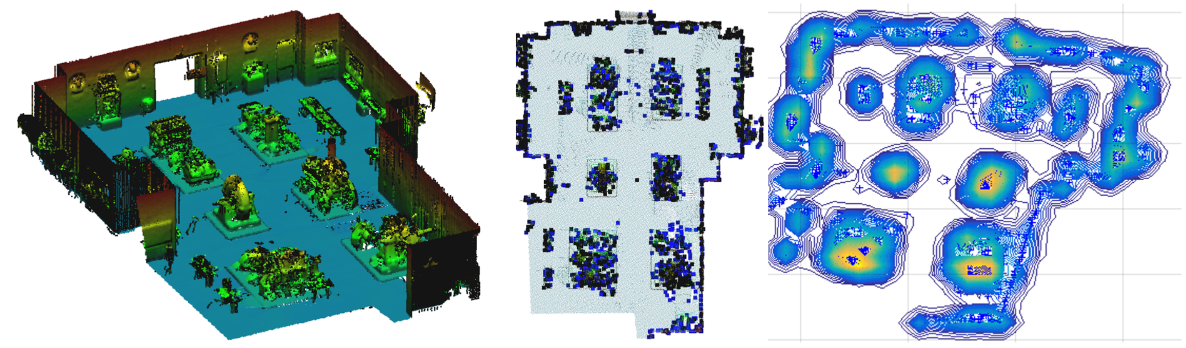


Figure 6.17: Reference point cloud segmentation into semantically meaningful objects. Left: A point cloud of an exhibition hall of the Deutsches Museum. Middle: Keypoints extracted from the point cloud shown from above. Evidently, the keypoints concentrate around the interesting objects. Right: Keypoint density histogram, shown using a contour plot. The peaks in the density plot (yellow regions) are used to identify distinct reference objects. Image source: [99] – reproduced with permission.

that "interesting" objects have articulated shapes which set them apart from the surrounding environment which is typically made up of walls and accordingly higher densities of keypoints tend to occur around the objects as shown in Figure 6.17. Finally, a region growing algorithm is used to associate points in the reference point cloud to the different objects identified by the peaks in the histogram. The algorithm is explained in detail in [99]. Figure 6.18 shows a few examples of segmented objects and corresponding query scans. The segmented objects are seen to encompass the physical objects properly.

At this point we observe that different objects can vary significantly in size. Accordingly, we postulate that simple features that measure size can already serve as good pre-selection mechanisms when searching for the match of any query. Observing that segmented objects can include or miss significant parts of point clouds available in the respective queries it becomes apparent that the size features should not be computed from the point clouds directly but rather from their keypoints. For instance, the "SteamLocomotive" reference object seen in

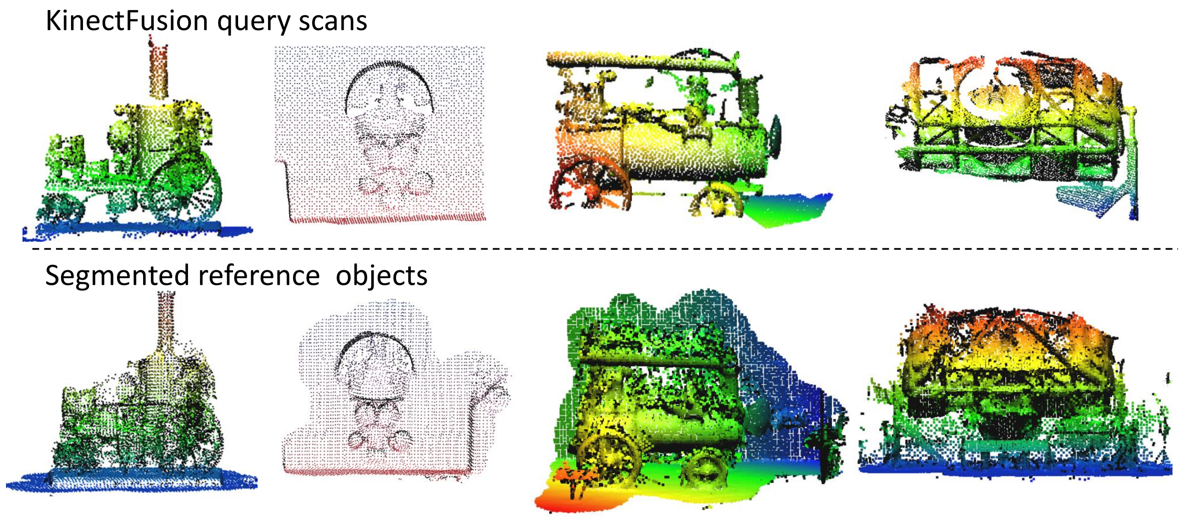


Figure 6.18: Examples of reference objects segmented from a reference point cloud of the Deutsches Museum and corresponding KinectFusion query scans. From left to right: "SteamEngine4", "FrancisTurbine", "SteamLocomotive" and "Astro-Spas". Image source: [99] – reproduced with permission.

Figure 6.18 has many more points than its respective query scan. These points, however, lie mainly on the nearby wall and would accordingly not produce any keypoints. A similar argument applies to "FrancisTurbine".

The keypoint spreads allow to filter objects based on size. However, they have a limited discriminative power beyond that. The SHOT feature, on the other hand, captures more interesting information about the 3D shape. However, the problem with SHOT is that it needs a repeatable keypoint and local reference frame (see Section 2.2.2.2) to produce similar feature vectors. We present an extensive investigation of this problem in [99].

Given the preceding arguments about pre-selection our objective is to use simple features about size to already filter out many reference objects and then choose the most likely corresponding objects using SHOT. Instead of building a system that performs such a two-stage selection we opt to use a technique that automatically combines the classification results from multiple features, a so-called Multiple Classifier System (MCS). We specifically implement MCS using the intersection method of Ho et al. [100].

Using a training set of query scans and the segmented reference objects we train an MCS that uses simple features derived from a 3D scan to learn how to effectively identify the corresponding reference object when presented with any query scan. The MCS is trained by deriving the following features from each reference object and each training query scan: 1)One SHOT feature with a 1.5 m radius centered on the keypoint density peak; 2)The keypoint density spread in the x-y plane capturing the horizontal object size 3)The keypoint density spread along the z-axis capturing the object size vertically. The latter two features are illustrated in Figure 6.19. The features are explained in greater detail in [99]. What is important to mention here is that the density spread features can be computed very easily

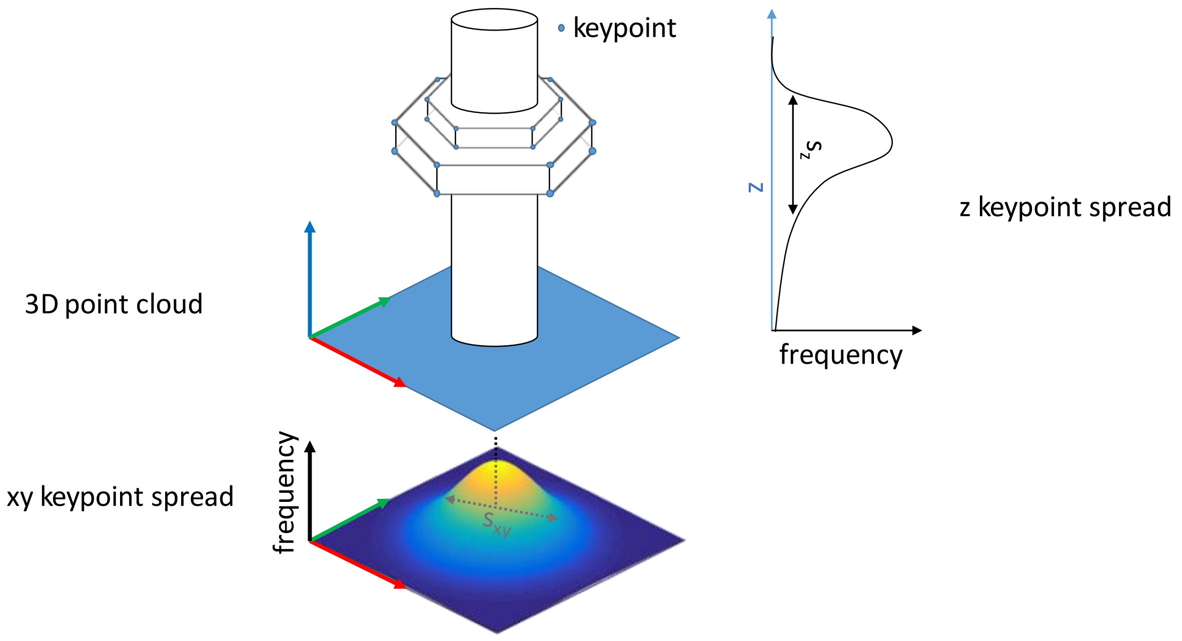


Figure 6.19: The keypoint density spread in the x-y plane and along the z-axis are computed and used as primitive features for quick pre-selection of relevant reference objects. In the example a 3D object with identified keypoints are shown. The histogram of keypoints along the z-axis and in the x-y plane are computed separately From each, a spread value is calculated.

and fast. In fact the keypoint spread in the x-y plane is a by-product of the segmentation process. Also the single SHOT can be computed very quickly. Hence, this feature selection is intentional as the objective is to provide a quick and reliable pre-selection mechanism.

The training step learns the thresholds that determine how many of the top retrievals to consider from each separate feature. Once a query object is now presented to the system the most similar reference objects are retrieved using each feature separately and only the top results up to the learned threshold are maintained. Finally, only those reference objects that appear in the short list of all three features are kept as potential matching candidates, hence the name "intersection method". For more details refer to [100].

The results in Table 6.2 show that the pre-selection mechanism succeeds in identifying a short list of reference object candidates in 8/10 test queries. The short list of reference objects makes up only 7% of the total list of reference objects on average leading to a dramatic reduction in required matching time as well as reducing the potential for wrong matches in the subsequent feature matching step. Indeed, this is confirmed by the results in Table 6.3 show that many query scans that the TCR for many test query scans rises from 0% to above 90% owing to the reduction of search space. Simultaneously, the retrieval time drops from 43 s on average to just 1.4 s. It remains to be mentioned that there is a computational overhead associated with computing the features that amounts to about 2 s. Nevertheless, in total a 10 fold speed-up is achieved while dramatically improving the retrieval success.

Table 6.2: Pre-selection results using the MCS. The *MCS status* column indicates whether or not the suggested reference objects short list contains the correct one, i.e. the one corresponding to the query scan. The column *# candidates* indicates how many reference objects are suggested as a potential candidate. The column *% of database* is the % of reference objects the short list of candidates makes up of the total number of reference objects that would have otherwise been necessary to consider as potential candidates if there were no pre-selection. The upper group of queries (delimited by the horizontal line) marks the training set as opposed to the lower group which presents the testing set.

Query	MCS status	# candidates	% of database
GirardTurbine	✓	4	2.9
FrancisTurbine	✓	8	5.8
Plane	✓	15	9.7
SteamLocomotive	✓	5	3.6
SteamEngine	✓	8	5.8
Generator1	✓	15	9.7
Generator3	✓	13	8.4
SteamEngine4	✓	10	7.2
SteamEngine5	✓	13	9.4
Engine1	✓	17	11.0
Turbine3	✓	14	10.1
Astro1	X	-	-
Astro6	✓	5	2.8
Astro8	✓	8	4.5
Generator7	✓	16	10.3
Generator8	✓	5	3.2
Werkzeug	X	-	-
Average	-	10.4	7.0

Table 6.3: Results in terms of true-correspondence rate (TCR). The column *TCR no-MCS* indicates the TCR results when matching against the full reference point cloud, as opposed to *TCR MCS* which are TCR values when matching only against the subset point cloud formed from the short list of reference objects obtained using the MCS pre-selection. Similarly, the retrieval times for the MCS and no-MCS cases are presented. The MCS system is seen to increase the TCR on average from 56% to 91% while simultaneously reducing the retrieval time to 1.4 s from 43 s on average. The upper group of queries (delimited by the horizontal line) marks the training set as opposed to the lower group which presents the testing set.

Query	TCR MCS [%]	TCR no-MCS [%]	Time MCS [s]	Time no-MCS [s]
FrancisTurbine	100	100	0.6	27
GirardTurbine	100	100	0.6	29
Plane	100	96	2.0	68
SteamLocomotive	100	92	1.0	26
SteamEngine	100	100	1.4	27
Generator1	100	100	2.0	71
Generator3	100	65	2.1	70
SteamEngine4	100	0	1.3	28
SteamEngine5	87	0	1.7	29
Engine1	0	0	1.9	26
Turbine3	100	0	1.4	27
Astro6	100	93.5	1.1	42
Astro8	100	0	0.6	38
Generator7	95	96.5	2.8	72
Generator8	75	3.23	1.0	72
Average	90.5	56.3	1.4	43

6.6 Summary

This chapter presents an indoor localization system that uses point cloud registration to achieve highly accurate location retrieval inside a large-scale museum environment. From a theoretical point of view, the system handles an extreme case of part-in-whole shape matching problem.

The query scans, scanned by persons, are obtained through 3D scanning with the Kinect 3D sensor in combination with KinectFusion reconstruction algorithm. The query scans are registered to large-scale indoor point clouds using a feature-based matching system with a geometry-aware RANSAC that deals effectively with the inadvertent low true correspondence rate.

The contribution also highlights a problem connected to 3D reconstruction of large objects using Kinect-like sensors, which manifests itself in the shape of surface bending, and analyses its impact on the resulting 3D scans.

The system is evaluated with data obtained from the Deutsches Musem. The queries are matched against reference point clouds of up to 3500m^2 floor space. Accurate retrieval results with an average translation error of only 6cm and an average rotation error of less than 3° are demonstrated. The results furthermore quantify the impact of proper sensor calibration using advanced techniques, such as CLAMS, on the retrieval results.

In a follow-up work, we developed a method to accelerate the system by pre-segmenting the reference PCD into sementically meaningful objects (each object is a small point cloud) using the Kernel Density Estimation Technique. A Multi-Classifier System (MCS) then pre-selects a few of the candidate objects and assembles them into a point cloud with far less points than the original one. Finally, the feature-based retrieval system is used to accurately retrieve the location accurately. We demonstrated a speedup of around $10\times$ with a substantial improvement in the retrieval performance.

Major parts of this work appeared in the proceedings of the Eurographics Workshop on 3D Object Retrieval (3DOR) 2015 [6]. The follow-up work on accelerating the retrieval is presented in detail in [99].

Chapter 7

Conclusions and Outlook for Future Research

7.1 Conclusions

This dissertation deals with the topic of 3D mapping of indoor spaces. The main technical problem treated in the thesis is that of point cloud alignment which is essential to 3D mapping. The thesis presents four major contributions that relate to alignment for mapping from different levels starting out with the basic mathematical theory and going up all the way to applications benefiting from 3D mapping. Each contribution builds upon the previous one and hence in collection a comprehensive view on the topic is provided. The four contributions and their inter-linkage are summarized in the following:

- **Chapter 3 - Outlier-Resilient Alignment using the Geometric Algebra LMS**: This contribution relates to the fundamental theory of point cloud alignment. Starting with a set of correspondences the problem is to determine the optimal rotation and translation that bring the two point clouds in alignment. Arun et al. showed the existence of a closed-form solution based on the Singular Value Decomposition (SVD) back in 1987 [16]. The SVD is used as a standard solution for alignment from correspondences up till today. Nevertheless, it suffers from its inherent sensitivity to outliers. In reformulating the alignment problem using Geometric Algebra (GA) we managed to derive an iterative adaptive filter (AF), the GA-LMS, that provides an alternative means to solving the *rotation* alignment from correspondences [1]. In Chapter 3 an extension of the GA-LMS to solve rotation and translation alignment is provided. Also, a series of mechanisms that exploit the adaptive and sequential nature of the GA-LMS are presented which are shown to endow the filter with outlier-resilience. The light-weight nature of the GA-LMS in combination with the presented mechanisms make it a credible substitute for the long used SVD-based solution.

The GA-LMS in its current shape isn't inherently outlier resilient. Rather, the proposed mechanisms help to make it more accurate as compared to standard SVD-based alignment. This is due to the nature of formulation whereby the alignment problem is posed, as in the standard Linear Algebraic formulation, as a least mean squares problem. It would be interesting to investigate new GA-based formulations that do not use the LMS criterion but rather some outlier-resistant criterion as demonstrated by Bouaziz et al. [101]. Now, equipped with the new GA-based formulation which comes along with the powerful GA Calculus, derivations may prove to be far less cumbersome and an iterative solution may yet again be obtainable which can be again exploited to endow the filter with certain desirable characteristics such as the capability to do online processing of correspondences as they are determined.

- **Chapter 4 - Skew-aware Scan Matching using the GA-LMS**: Two fundamental sub-contributions are presented. In the first, a study of the so-called "skewing" problem of scans of modern LiDARs is presented. Scan skewing is the rigid deformation of 3D scans (point clouds obtained from one full scan sequence of the LiDAR) due to sensor mobility combined with the sequential scanning nature of the LiDAR. The impact of skewing on scan matching, the process with which consecutive LiDAR scans are assembled into a comprehensive 3D map (point cloud) while simultaneously recovering the sensor motion trajectory, is theoretically derived and experimentally verified using a set of synthetic datasets with controlled distortion artifacts and accurate ground-truth. This, to the best of our knowledge, is the first systematic study of the problem of skewing. In the second sub-contribution the GA-LMS presented in Chapter 3 is re-derived for the point-to-plane metric (GA-LMSpt2pl) and employed to reduce the impact of skewing on scan matching. Specifically, we show how the adaptive nature of the GA-LMS can be exploited to account for the severity of the skew for each point such that more accurate scan matching with less trajectory drift can be achieved. Results with real LiDAR data (VLP-16) are also presented.

  Although the GA-LMSpt2pl clearly helps in reducing the rotation estimation errors during scan matching it suffers from solving the rotation and translation in two separate steps. Owing to the nature of the LiDAR scans and the fact that point cloud centroids are used to estimate the translation, small translation errors accumulate into trajectory compression which harms the overall scan matching outcome. It is expected that replacing the rotor operator in the GA-LMSpt2pl formulation with a *motor* operator that performs rotation and translation can help in deriving a new family of GA-based alignment filters that can perform joint rotation-translation alignment and thus avoid the compression problem while retaining the capability to effectively deal with skewing.

- **Chapter 5 - Alignment of large-scale indoor point clouds**: While Chapter 4 addresses the problem of acquiring 3D maps from scan matching, in this contribution we show how to automatically align multiple such 3D maps into more comprehensive 3D maps. We are motivated by the fact that it is more beneficial to reconstruct a large building from multiple maps as many technical and non-technical challenges are circumvented such as trajectory drift which is addressed in Chapter 4. We present an automatic 6DOF alignment method that is insensitive to initial alignment and is particularly efficient despite the huge amount of points per 3D model. To achieve that, we show that it is possible to decouple the estimation of the alignment for the different degrees of freedom by exploiting inherent characteristics to indoor environments. In a first step the point clouds are aligned to and along gravity using PCA and point distribution histograms. Subsequently, the ground-plane rotation is recovered up to a degree of ambiguity using histograms of surface normal angles. Finally, the ground-plane translation is computed by determining the peak of a bi-variate shift histogram built from selected source-target point pairs, the CPSH. The method is further extended to deal with very small overlap between the 3D models by re-interpreting CPSHs as probability density functions which are fused using Maximum Likelihood estimation. The superiority of the approach in terms of efficiency as well as insensitivity to initial alignment and overlap size is demonstrated through experiments with real 3D maps acquired inside large museum environments.

  A refinement of the proposed algorithm may focus on the problem of needing to compute four different CPSHs per segment pair to resolve the ambiguity in the ground-plane rotation estimation. We presented a method for significantly accelerating the CPSH computation in Section 5.7. This needs to be extended to enable all types discriminative voting which have been found to greatly improve the CPSH quality.

- **Chapter 6 - Indoor Location Retrieval using 3D Shape Matching**: In Chapter 5 we show how to obtain extensive 3D maps of indoor environments. They can be used for a plethora of applications. In a final contribution we show how such maps can be used for the application of indoor location retrieval. This remains to date one of the big challenges as the highly popular Global Positioning System (GPS) does not work indoors. The idea is to use point cloud alignment to match a locally acquired 3D shape scan inside a large scale indoor point cloud. This is not only motivated by the ability to acquire large scale point clouds using scan matching and alignment of large-scale point clouds but also by recent developments in 3D sensing that have brought about the highly celebrated Kinect sensor (and similarly working structured-light sensors) as well as algorithms for real-time stitching of multiple Kinect scans into one refined 3D model. Perhaps the best known of these algorithms is KinectFusion (KinFu). In this contribution we demonstrate location retrieval inside a museum environment by matching KinectFusion 3D models inside large-scale 3D maps. We explain the challenges relating to sensing accuracies and distortions of Kinect-like sensors and show how to design a matching system that can handle this extreme case of *part-in-whole* matching. Our results, obtained with real data, show that location accuracies in the cm range are possible. We also present an extension based on pre-selection of candidate parts of the 3D model using a Multi-Classifier System (MCS) whereby the system is accelerated by an order of magnitude while simultaneously increasing the accuracy.

The presented system may not be very practical to use as it requires a person to perform a lengthy scan of an object using a Kinect-like sensor before matching can be triggered. Interested researchers may investigate means for faster query acquisition.

Appendices

Appendix A

Exponential Mapping Normalization Constant $c_{i,j}$

Here we show that the normalization constant of the exponential mapping of (5.14) is almost the same for any CPSH. The proof is done by induction: Starting with a uniform CPSH of v bins, i.e. $f_{i,j} = 1/v$, the exponential mapping would require a normalization factor:

$$c_1 = v \cdot b^{1/v} = \frac{v}{2}(b^{1/v} + b^{1/v}) \tag{A.1}$$

If the bin strength is now decreased for half of all CPSH bins by $1/v$ and necessarily the strength of the remaining bins is increased by $1/v$, then the normalization factor would become

$$c_2 = \frac{v}{2}b^0 + \frac{v}{2}b^{2/v} = \frac{v}{2}(1 + b^{2/v}) \tag{A.2}$$

Since the exponential function is convex $b^a + b^a \leq b^0 + b^{2a}$ for $0 \leq a$ it follows that:

$$c_1 = v \cdot b^{1/v} = \frac{v}{2}(b^{1/v} + b^{1/v}) < \frac{v}{2}(1 + b^{2/v}) = c_2. \tag{A.3}$$

By recursively applying the same procedure of setting half of the non-zero bins to 0 and necessarily doubling the other half it can be seen that the normalization constant monotonically increases until only two non-zero bins remain with 0.5 strength each. Here again, the normalization value increases when one of the two is set to zero and the other becomes 1. Hence, the maximum normalization factor is needed for the case with the peakiest distribution i.e. the Dirac distribution. Vice versa the distribution with the smallest normalization factor is the uniform distribution.

Evaluating the normalization factor for the two limiting cases for a typical CPSH which may have a grid of 200×200 and for $b = 10$ would result in $c_{min} = 40002$ and $c_{max} = 40009$. Obviously the ratio of these two values is almost 1.0 and hence the normalization factor is almost equal for different CPSHs.

Bibliography

Publications by the author

Journal publications

[1] W. B. Lopes, A. Al-Nuaimi, and C. G. Lopes, "Geometric-algebra lms adaptive filter and its application to rotation estimation," *IEEE Signal Processing Letters*, vol. 23, no. 6, pp. 858–862, 2016, ISSN: 1070-9908.

[2] A. Al-Nuaimi, S. Hilsenbeck, A. Garcea, and E. Steinbach, "6dof decoupled roto-translation alignment of large-scale indoor point clouds," *Computer Vision and Image Understanding*, vol. 157, pp. 72 –89, 2017, Large-Scale 3D Modeling of Urban Indoor or Outdoor Scenes from Images and Range Scans, ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2016.08.004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077314216301126.

[3] G. Schroth, R. Huitl, D. Chen, M. Abu-Alqumsan, A. Al-Nuaimi, and E. Steinbach, "Mobile visual location recognition," *Signal Processing Magazine, IEEE*, vol. 28, no. 4, pp. 77–89, 2011.

Conference publications

[4] A. Al-Nuaimi, W. Lopes, P. Zeller, A. Garcea, C. Lopes, and E. Steinbach, "Analyzing liDAR scan skewing and its impact on scan matching," in *Seventh Int. Conf. on Indoor Positioning and Indoor Navigation*, Madrid, Spain, 2016.

[5] A. Al-Nuaimi, E. Steinbach, W. B. Lopes, and C. G. Lopes, "6dof point cloud alignment using geometric algebra-based adaptive filtering," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2016, pp. 1–9.

[6] A. Al-Nuaimi, M. Piccolrovazzi, S. Gedikli, E. Steinbach, and G. Schroth, "Indoor location retrieval using shape matching of kinectfusion scans to large-scale indoor point clouds," in *Proceedings of the 2015 Eurographics Workshop on 3D Object Retrieval*, ser. 3DOR, Zurich, Switzerland: Eurographics Association, 2015, pp. 31–38, ISBN: 978-3-

905674-78-1. DOI: 10.2312/3dor.20151052. [Online]. Available: http://dx.doi.org/10.2312/3dor.20151052.

[7] A. Al-Nuaimi, R. Huitl, S. Taifour, S. Sarin, X. Song, Y. Gu, E. Steinbach, and M. Fahrmair, "Towards location recognition using range images," in *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, IEEE, 2013, pp. 1–6.

General publications

[8] J. Carpenter and J. Snell, *Future trends in geospatial information management: The five to ten year vision.* 2013, ISBN: 9780319087923.

[9] J. Smisek, M. Jancosek, and T. Pajdla, "3d with kinect," in *Consumer Depth Cameras for Computer Vision*, ser. Advances in Computer Vision and Pattern Recognition, A Fossati, J. Gall, H. Grabner, X. Ren, and K. Konolige, Eds., Springer London, 2013, pp. 3–25. [Online]. Available: http://dx.doi.org/10.1007/978-1-4471-4640-7_1.

[10] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010.

[11] H. J. Wolfson and I. Rigoutsos, "Geometric hashing: an overview," *IEEE Comput. Sci. Eng.*, vol. 4, no. 4, pp. 10–21, Oct. 1997, ISSN: 1070-9924. DOI: 10.1109/99.641604. [Online]. Available: http://dx.doi.org/10.1109/99.641604.

[12] A. Albarelli, E. Rodolà, and A. Torsello, "Fast and accurate surface alignment through an isometry-enforcing game," *Pattern Recogn.*, vol. 48, no. 7, pp. 2209–2226, Jul. 2015, ISSN: 0031-3203. DOI: 10.1016/j.patcog.2015.01.020. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2015.01.020.

[13] A. Myronenko and X. Song, "Point set registration: coherent point drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[14] D. Holz, A.-E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke, "A modular framework for aligning 3D point clouds - Registration with the Point Cloud Library," *IEEE Robotics and Automation Magazine*, 2015.

[15] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987.

[16] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5, pp. 698–700, 1987, ISSN: 0162-8828. DOI: 10.1109/TPAMI.1987.4767965.

[17] R. B. Rusu and S. Cousins, “3d is here: point cloud library (pcl),” in *IEEE Intern. Conf. on Robotics and Automation*, 2011, pp. 1–4. DOI: 10.1109/ICRA.2011.5980567.

[18] R. Huitl, G. Schroth, S. Hilsenbeck, F. Schweiger, and E. Steinbach, “Tumindoor: an extensive image and point cloud dataset for visual indoor localization and mapping,” in *IEEE ICIP*, 2012, pp. 1773–1776.

[19] T. Liu, M. Carlberg, G. Chen, J. Chen, J. Kua, and A. Zakhor, “Indoor localization and visualization using a human-operated backpack system,” in *2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010, pp. 1–10. DOI: 10.1109/IPIN.2010.5646820.

[20] T. Whelan, M. Kaess, J. J. Leonard, and J. McDonald, “Deformation-based loop closure for large scale dense rgb-d slam,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2013, pp. 548–555.

[21] A. Akbarzadeh, J.-M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, and M. Pollefeys, “Towards Urban 3d Reconstruction from Video,” in *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, Jun. 2006, pp. 1–8. DOI: 10.1109/3DPVT.2006.141.

[22] *IndoorReality - press coverage*, http://indoorreality.com/about/news/, [Online; accessed 9-June-2016], 2016.

[23] B. Sirmacek, Y. Shen, R. Lindenbergh, S. Zlatanova, and A. Diakite, “Comparison of zeb1 and leica c10 indoor laser scanning point clouds.,” *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 3, no. 1, 2016.

[24] L. Geosystems, *Leica ScanStation C10*, http://leica-geosystems.com/-/media/files/products/brochures/leica_scanstation_c10_bro_en.ashx?la=en, [Online; accessed 21-June-2016], 2016.

[25] *Navvis m3 mapping trolley*, http://www.navvis.com/products/m3-trolley/, [Online; accessed 24-August-2016].

[26] D. W. Eggert, A. Lorusso, and R. B. Fisher, “Estimating 3-d rigid body transformations: a comparison of four major algorithms,” *Machine Vision and Applications*, vol. 9, no. 5-6, pp. 272–290, 1997.

[27] P. H. Schönemann, “A generalized solution of the orthogonal procrustes problem,” *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966.

[28] M. W. Walker, L. Shao, and R. A. Volz, “Estimating 3-d location parameters using dual number quaternions,” *CVGIP: Image Underst.*, vol. 54, no. 3, pp. 358–367, Oct. 1991, ISSN: 1049-9660. DOI: 10.1016/1049-9660(91)90036-O. [Online]. Available: http://dx.doi.org/10.1016/1049-9660(91)90036-O.

[29] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981, ISSN: 0001-0782. DOI: 10.1145/358669.358692. [Online]. Available: http://doi.acm.org/10.1145/358669.358692.

[30] Y. Hecker and R. Bolle, “On geometric hashing and the generalized hough transform,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 9, pp. 1328–1338, 1994, ISSN: 0018-9472. DOI: 10.1109/21.310509.

[31] D. Huttenlocher and S. Ullman, “Recognizing solid objects by alignment with an image,” English, *International Journal of Computer Vision*, vol. 5, no. 2, pp. 195–212, 1990, ISSN: 0920-5691. DOI: 10.1007/BF00054921. [Online]. Available: http://dx.doi.org/10.1007/BF00054921.

[32] G. Stockman, “Object recognition and localization via pose clustering,” *Comput. Vision Graph. Image Process.*, vol. 40, no. 3, pp. 361–387, Dec. 1987, ISSN: 0734-189X. DOI: 10.1016/S0734-189X(87)80147-0. [Online]. Available: http://dx.doi.org/10.1016/S0734-189X(87)80147-0.

[33] A. E. Johnson and M. Hebert, “Using spin images for efficient object recognition in cluttered 3d scenes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 433–449, 1999.

[34] R. B. Rusu, N. Blodow, and M. Beetz, “Fast point feature histograms (fpfh) for 3d registration,” in *IEEE International Conference on Robotics and Automation, 2009. ICRA’09*, IEEE, 2009, pp. 3212–3217.

[35] S. Salti, F. Tombari, and L. Di Stefano, “Shot: unique signatures of histograms for surface and texture description,” *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014.

[36] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. Rusu, S. Gedikli, and M. Vincze, “Tutorial: point cloud library: three-dimensional object recognition and 6 dof pose estimation,” *IEEE Robotics & Automation Mag.*, vol. 19, no. 3, pp. 80–91, 2012, ISSN: 1070-9932.

[37] R. Rusu and S. Cousins, “3d is here: point cloud library (pcl),” in *IEEE 2011 Internat. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 1–4.

[38] F. Tombari, S. Salti, and L. Di Stefano, “Unique signatures of histograms for local surface description,” in *Proc. of the 11th European Conference on Computer Vision Conference on Computer Vision: Part III*, Heraklion, Crete, Greece: Springer-Verlag, 2010, pp. 356–369.

[39] S. Salti, F. Tombari, and L. D. Stefano, “Shot: unique signatures of histograms for surface and texture description,” *Computer Vision and Image Understanding*, vol. 125, pp. 251 –264, 2014, ISSN: 1077-3142. DOI: http://dx.doi.org/10.1016/j.cviu.2014.04.011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077314214000988.

[40] F. Tombari, S. Salti, and L. Di Stefano, “Unique shape context for 3d data description,” in *ACM Workshop on 3D Object Retrieval*, Firenze, Italy, 2010, pp. 57–62.

[41] D. Aiger, N. J. Mitra, and D. Cohen-Or, “4-points congruent sets for robust surface registration,” *ACM Transactions on Graphics*, vol. 27, no. 3, #85, 1–10, 2008.

[42] N. Mellado, D. Aiger, and N. J. Mitra, “Super 4pcs fast global pointcloud registration via smart indexing,” *Computer Graphics Forum*, vol. 33, no. 5, pp. 205–215, 2014, ISSN: 1467-8659.

[43] B. Jian and B. C. Vemuri, “Robust point set registration using gaussian mixture models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1633–1645, 2011.

[44] L. Greengard and J. Strain, “The fast gauss transform,” *SIAM Journal on Scientific and Statistical Computing*, vol. 12, no. 1, pp. 79–94, 1991. DOI: 10.1137/0912004. [Online]. Available: http://dx.doi.org/10.1137/0912004.

[45] S. J. Prince, *Computer vision: models, learning, and inference*. Cambridge University Press, 2012.

[46] Z. Zhang, “Iterative point matching for registration of free-form curves and surfaces,” *Int. J. Comput. Vision*, vol. 13, no. 2, Oct. 1994, ISSN: 0920-5691. DOI: 10.1007/BF01427149. [Online]. Available: http://dx.doi.org/10.1007/BF01427149.

[47] P. Besl and N. D. McKay, “A method for registration of 3-d shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992, ISSN: 0162-8828. DOI: 10.1109/34.121791.

[48] S. Rusinkiewicz and M. Levoy, “Efficient variants of the icp algorithm,” in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, IEEE, 2001, pp. 145–152.

[49] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Found. Trends Robot*, vol. 4, no. 1, pp. 1–104, May 2015, ISSN: 1935-8253. DOI: 10.1561/2300000035. [Online]. Available: http://dx.doi.org/10.1561/2300000035.

[50] D. Hestenes and G. Sobczyk, *Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics*, ser. Fundamental Theories of Physics. Springer Netherlands, 1987, ISBN: 9789027725615. [Online]. Available: http://books.google.de/books?id=yyjLeKEdt20C.

[51] D. Hestenes, *New Foundations for Classical Mechanics*, ser. Fundamental Theories of Physics. Springer, 1999, ISBN: 9780792355144. [Online]. Available: http://books.google.de/books?id=N9ZEXX-ypSQC.

[52] E. Hitzer, "Introduction to Clifford's Geometric Algebra," *Journal of the Society of Instrument and Control Engineers*, vol. 51, no. 4, pp. 338–350, 2012.

[53] A. Sayed, *Adaptive filters*. Wiley-IEEE Press, 2008.

[54] P. S. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*, 2nd ed. Norwell, MA, USA: Kluwer Academic Publishers, 2002, ISBN: 1402071256.

[55] L. Dorst, D. Fontijne, and S. Mann, *Geometric Algebra for Computer Science: An Object-Oriented Approach to Geometry (The Morgan Kaufmann Series in Computer Graphics)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, ISBN: 0123694655.

[56] R. Valkenburg and L. Dorst, "Estimating motors from a variety of geometric data in 3d conformal geometric algebra," English, in *Guide to Geometric Algebra in Practice*, L. Dorst and J. Lasenby, Eds., Springer London, 2011, pp. 25–45, ISBN: 978-0-85729-810-2. DOI: 10.1007/978-0-85729-811-9_2. [Online]. Available: http://dx.doi.org/10.1007/978-0-85729-811-9_2.

[57] M. J. Stanway and J. C. Kinsey, "Rotation identification in geometric algebra: theory and application to the navigation of underwater robots in the field," *Journal of Field Robotics*, 2015, ISSN: 1556-4967. DOI: 10.1002/rob.21572. [Online]. Available: http://dx.doi.org/10.1002/rob.21572.

[58] E. Hitzer, "Multivector differential calculus," English, *Advances in Applied Clifford Algebras*, vol. 12, no. 2, pp. 135–182, 2002, ISSN: 0188-7009. DOI: 10.1007/BF03161244. [Online]. Available: http://dx.doi.org/10.1007/BF03161244.

[59] F. Seybold and U Wössner, "Gaalet-a c++ expression template library for implementing geometric algebra," in *6th High-End Visualization Workshop*, 2010.

[60] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '94, New York, NY, USA: ACM, 1994, pp. 311–318, ISBN: 0-89791-667-0. DOI: 10.1145/192161.192241. [Online]. Available: http://doi.acm.org/10.1145/192161.192241.

[61] W. B. Lopes and C. G. Lopes, "Incremental-cooperative strategies in combination of adaptive filters," in *Int. Conf. in Acoust. Speech and Signal Process.*, IEEE, 2011, pp. 4132 –4135.

[62] L. Chamon, W. Lopes, and C. Lopes, "Combination of adaptive filters with coefficients feedback," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 3785 –3788. DOI: 10.1109/ICASSP.2012.6288741.

[63] W. B. Lopes and C. G. Lopes, "Incremental combination of rls and lms adaptive filters in nonstationary scenarios," in *Int. Conf. in Acoust. Speech and Signal Process.*, IEEE, 2013.

[64] L. F. O. Chamon and C. G. Lopes, "There's plenty of room at the bottom: incremental combinations of sign-error lms filters.," in *ICASSP*, 2014, pp. 7248–7252.

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

[66] K.-L. Low, "Linear least-squares optimization for point-to-plane icp surface registration," *Chapel Hill, Univ. of North Carolina*, vol. 4, 2004.

[67] F. Moosmann and C. Stiller, "Velodyne slam," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2011, pp. 393–398.

[68] *Velodyne VLP-16 Puck*, http://velodynelidar.com/vlp-16.html, [Online; accessed 29-May-2016].

[69] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*, 4th ed. Springer US, 2013, ISBN: 978-1-4614-4105-2.

[70] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, IEEE, 2011, pp. 155–160.

[71] J. Zhang and S. Singh, "Loam: lidar odometry and mapping in real-time," in *Robotics: Science and Systems Conference (RSS)*, 2014.

[72] A. Bódis-Szomorú, H. Riemenschneider, and L. Van Gool, "Efficient volumetric fusion of airborne and street-side data for urban reconstruction," *arXiv preprint arXiv:1609.01345*, 2016.

[73] A. Cohen, C. Zach, S. Sinha, and M. Pollefeys, "Discovering and exploiting 3d symmetries in structure from motion," in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2012, pp. 1514–1521. DOI: 10.1109/CVPR.2012.6247841.

[74] A. Cohen, T. Sattler, and M. Pollefeys, "Merging the unmatchable: stitching visually disconnected SfM models," IEEE/CVF ICCV 2015, 2015.

[75] D. C. Lee, M. Hebert, and T. Kanade, "Geometric reasoning for single image structure recovery," in *IEEE Conference on Computer Vision and Pattern Recognition, 2009*, 2009, pp. 2136–2143. DOI: 10.1109/CVPR.2009.5206872.

[76] R. Cabral and Y. Furukawa, "Piecewise planar and compact floorplan reconstruction from images," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 628–635. DOI: 10.1109/CVPR.2014.546.

[77] S. Oesau, F. Lafarge, and P. Alliez, "Indoor Scene Reconstruction using Feature Sensitive Primitive Extraction and Graph-cut," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 90, pp. 68–82, Mar. 2014. DOI: 10.1016/j.isprsjprs.2014.02.004. [Online]. Available: https://hal.inria.fr/hal-00980804.

[78] C. Chao and I. Stamos, "Semi-automatic range to range registration: a feature-based method," in *Fifth International Conference on 3-D Digital Imaging and Modeling, 2005. 3DIM 2005*, Jun. 2005, pp. 254–261. DOI: 10.1109/3DIM.2005.72.

[79] J. Straub, G. Rosman, O. Freifeld, J. J. Leonard, and J. W. Fisher, "A mixture of manhattan frames: beyond the manhattan world," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3770–3777. DOI: 10.1109/CVPR.2014.488.

[80] A. Monszpart, N. Mellado, G. Brostow, and N. Mitra, "RAPter: rebuilding man-made scenes with regular arrangements of planes," *ACM SIGGRAPH 2015*, 2015.

[81] N. Mellado, *Super4PCS on github*, https://github.com/nmellado/Super4PCS, [Online; accessed 30-Nov-2015].

[82] B. Jian, *Implementations of the robust point set registration algorithm described in the paper "Robust Point Set Registration Using Gaussian Mixture Models"*, https://github.com/bing-jian/gmmreg, [Online; accessed 30-Nov-2015].

[83] A. Myronenko, *Coherent Point Drift implementation*, https://sites.google.com/site/myronenko/research/cpd, [Online; accessed 30-Nov-2015].

[84] O. Chum and J. Matas, "Matching with prosac - progressive sample consensus," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, 220–226 vol. 1. DOI: 10.1109/CVPR.2005.221.

[85] *Euler dataset download*, http://geometry.cs.ucl.ac.uk/projects/2015/regular-arrangements-of-planes/. [Online]. Available: http://geometry.cs.ucl.ac.uk/projects/2015/regular-arrangements-of-planes/.

[86] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: real-time dense surface mapping and tracking," in *Proceedings of the IEEE 10th International Symposium on Mixed and Augmented Reality*, 2011, pp. 127–136.

[87] G. Schroth, A. Al-Nuaimi, R. Huitl, F. Schweiger, and E. Steinbach, "Rapid image retrieval for mobile location recognition," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 2320–2323. DOI: 10.1109/ICASSP.2011.5946947.

[88] R. Huitl, G. Schroth, S. Hilsenbeck, F. Schweiger, and E. Steinbach, "Virtual reference view generation for cbir-based visual pose estimation," in *Proceedings of the 20th ACM international conference on Multimedia*, ACM, 2012, pp. 993–996.

[89] A. Teichman, S. Miller, and S. Thrun, "Unsupervised intrinsic calibration of depth sensors via slam," in *Proceedings of Robotics: Science and Systems*, Berlin, 2013.

[90] J. Tangelder and R. Veltkamp, "A survey of content based 3d shape retrieval methods," in *Proc. of 2004 Internat. Conf. on Shape Modeling Applications*, 2004, pp. 145–156.

[91] Y. Zhong, "Intrinsic shape signatures: a shape descriptor for 3d object recognition," in *Proc. of the IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, 2009, pp. 689–696.

[92] S. Filipe and L. A. Alexandre, "A comparative eval. of 3d keypoint detectors in a rgb-d object dataset," in *9th Intern. Conf. on Computer Vision Theory and Applications*, 2014.

[93] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proc. of the ACM 23rd Annual Conf. on Computer Graphics and Interactive Techniques*, 1996, pp. 303–312.

[94] N. J. Mitra and A. Nguyen, "Estimating surface normals in noisy point cloud data," in *Proceedings of the ACM 19th Annual Symposium on Computational Geometry*, San Diego, California, USA, 2003, pp. 322–328.

[95] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012, ISSN: 1424-8220.

[96] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 3–15, Jan. 2003, ISSN: 1077-2626.

[97] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3d point cloud based object maps for household environments," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927–941, 2008.

[98] G. G. Schroth, "Mobile visual location recognition," Dissertation, Technische Universitaet Muenchen, Muenchen, 2013.

[99] A. A.-N. Martin Piccolorovazzin, "Fast indoor location retrieval using shape matching," Chair of Media Technology, `www.lmt.ei.tum.de`, Research Internship Report, 2016.

[100] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 1, pp. 66–75, 1994.

[101] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point," in *Computer graphics forum*, Wiley Online Library, vol. 32, 2013, pp. 113–123.

List of Figures

List of Tables