Computer Aided Medical Procedures
Prof. Dr. Nassir Navab

Dissertation

Learned 3D Local Features for Rigid Pose Estimation on Point Clouds

Haowen Deng

Fakultät für Informatik
Technische Universität München

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin

Learned 3D Local Features for Rigid Pose Estimation on Point Clouds

Haowen Deng

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende(r): Prof. Dr.-Ing. Matthias Nießner

Prüfer der Dissertation: 1. Priv.-Doz. Dr. Slobodan Ilic

2. Prof. Luigi Di Stefano, Ph.D.

Die Dissertation wurde am 01.07.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 30.08.2020 angenommen.

Haowen Deng
Learned 3D Local Features for Rigid Pose Estimation on Point Clouds
Dissertation, Version 1.0

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin
Boltzmannstraße 3
85748 and Garching bei München

Abstract

Rigid pose estimation is a fundamental task in a wide range of problems such as pairwise registration, 3D reconstruction, 6D object pose estimation, etc. Thanks to the increased deployment of 3D sensors, e.g. laser scans, which produce 3D point clouds, the need for estimating the rigid pose in point clouds is growing significantly. Correspondences, which are usually established between local surface geometries, are key to the robust rigid pose estimation. To facilitate the matching procedure, these geometries need to be described in a unique way, which ideally remains constant across various poses and corruptions such as noises. In this thesis, we developed several deep learning-based methods that generate local features which lead to better matching performances and acquire better rigid pose estimation results based on the correspondences.

We first propose to learn local features from point clouds with full supervision. A novel network architecture, which adopts PointNet-like sub-networks and fuses local and global surface information, is presented. The network is trained to simultaneously optimize all the local descriptors together with their pairwise relationships, specialized for matching purposes. The lightweight of our network, with reduced computation and memory consumption, makes this framework amenable. To suppress the influence of the erroneous labels, we also come up with an autoencoder framework, which aims at distilling the most critical geometric information through the intermediate compact codes in an unsupervised manner. By converting point clouds to pure point pair features, we manage to preserve the sparsity of input data as well as obtain fully rotation-invariant local features. Our learned features exhibit superior performances in local correspondence establishment.

Then we design a network architecture to directly regress the relative poses between correctly matched local patches, which eventually leads to the solution between the pair of holistic point clouds where they are taken from. Our network learns to absorb the pose-related information in the corresponding local features and predicts the poses. We demonstrate that our method could not only speed up the pose estimation procedure but also improve the final registration results, compared with various RANSAC methods.

Finally, we target the notorious uncertainty and ambiguity problems in the pose estimation. Traditional regression-based methods assume only one correct solution exists for each input, which can be easily confused by symmetries. It is also difficult to find out whether the network holds confidence in the outputs. Based on Bingham distribution, which well fits the manifold of quaternions, we put forward a generic framework. Given an input, our Bingham network predicts essential parameters for a multimodal Bingham distribution describing its pose. Different modes which coexist under ambiguity can be well captured by individual components of the predicted mixture distribution. In the meanwhile, uncertainty in the pose prediction can be indicated by the distribution entropy. Rigorous experiments demonstrate the effectiveness of our methods.

Zusammenfassung

Die starre Posenschätzung ist eine grundlegende Aufgabe in einer Vielzahl von Problemen wie der paarweisen Registrierung, der 3D-Rekonstruktion, der 6D-Objektposenschätzung usw. Dank des verstärkten Einsatzes von 3D-Sensoren, wie z.B von Laserscannern, welche 3D-Punktwolken erzeugen, wächst der Bedarf an Schätzungen der starren Pose in Punktwolken erheblich. Punktkorrespondenzen, die normalerweise zwischen lokalen Oberflächengeometrien hergestellt werden, sind der Schlüssel für die robuste Schätzung der starren Pose. Um einen korrekten Punktabgleich zu ermöglichen, müssen diese Geometrien auf eine einzigartige Weise beschrieben werden, welche idealerweise über verschiedene Posen und Verfälschungen wie Rauschen hinweg konstant bleibt. In dieser Arbeit haben wir mehrere auf Deep Learning basierende Methoden entwickelt, die lokale Merkmale erzeugen, welche wiederum zu besseren Übereinstimmungsleistungen führen und auf der Grundlage der Punktkorrespondenzen bessere Ergebnisse für die Schätzung starrer Posen erzielen.

Wir schlagen zunächst vor, lokale Merkmale aus Punktwolken unter vollständiger Aufsicht zu lernen. Es wird eine neuartige Netzwerkarchitektur vorgestellt, die PointNet-ähnliche Teilnetzwerke verwendet und lokale und globale Oberflächeninformationen zusammenführt. Das neuronale Netz wird dafür trainiert, alle lokalen Merkmale zusammen mit ihren paarweisen Beziehungen, spezialisiert für die Suche nach Punktorrespondenzen, gleichzeitig zu optimieren. Die effiziente Struktur unseres Netzwerks mit reduziertem Rechenaufwand und geringerem Speicherverbrauch macht dieses Framework leicht zugänglich. Um den Einfluss von fehlerbehafteten Daten zu lindern, haben wir außerdem ein Autoencoder-Framework entwickelt, das darauf abzielt, die kritischsten geometrischen Informationen automatisch in die kompakten Zwischenmerkmale zu kodieren. Durch die Konvertierung von Punktwolken in reine Punktpaar-Merkmale gelingt es uns, die geringe Dichte der Eingabedaten beizubehalten und zudem vollständig rotationsinvariante lokale Merkmale zu erhalten. Im Vergleich zeigen unsere erlernten Merkmale überlegene Leistungen bei der Erstellung lokaler Punktkorrespondenzen.

Anschließend entwerfen wir eine Netzwerkarchitektur, um die relativen Posen zwischen gefundenen übereinstimmenden lokalen Ausschnitten direkt zu bestimmen, was schließlich zur Lösung zwischen den beiden ganzheitlichen Punktwolken führt, aus denen sie stammen. Unser neuronales Netz lernt, die Poseninformationen in die entsprechenden lokalen Merkmale aufzunehmen und die Posen vorherzusagen. Wir zeigen, dass unsere Methode nicht nur das Posenschätzungsverfahren beschleunigt, sondern auch die endgültigen Registrierungsergebnisse im Vergleich zu verschiedenen RANSAC-Methoden verbessern kann.

Schließlich zielen wir auf die berüchtigten Unsicherheits- und Mehrdeutigkeitsprobleme bei der Posenschätzung ab. Herkömmliche regressionsbasierte Methoden gehen davon aus, dass für jede Eingabe nur eine richtige Lösung vorhanden ist, die leicht durch Symmetrien verwechselt werden kann. Es ist auch schwierig herauszufinden, ob ein neuronales

Netz Vertrauen in seine Aussagen hat. Basierend auf der Bingham-Verteilung, die gut zu dem Gültigkeitsbereich der Quaternionen passt, schlagen wir ein generisches Framework vor. Bei gegebener Eingabe sagt unser Bingham-Netzwerk wesentliche Parameter für eine multimodale Bingham-Verteilung voraus, die ihre Pose beschreibt. Verschiedene Modi, die unter Mehrdeutigkeit existieren, können von einzelnen Komponenten der vorhergesagten Mischungsverteilung gut erfasst werden. Zudem kann die Unsicherheit in der Posenvorhersage durch die Verteilungsentropie angezeigt werden. Umfangreiche Experimente verdeutlichen die Wirksamkeit unserer Methoden.

Acknowledgments

With the accomplishment of this thesis, all kinds of emotions flood into my head. Looking back, the journey towards a PhD degree is such a valuable experience which bears ups and downs and is full of challenges and obstacles. I would never regret this decision, as I met a lot of nice people and learned a lot as well along this way.

First of all, I would like to thank PD Dr. Slobodan Ilic for being my supervisor, leading me into the world of PhD research and guiding me throughout these years, providing me with consistent support and countless discussions. His enthusiasm about life and family also inspires a lot. I am greatly honored for the excellent collaborations with Dr. Tolga Birdal. I am always amazed at his intelligence and deep insights into math and geometry. I am lucky to have Sergey Zakharov and Mai Bui, with whom I started and finished PhD life almost together with. We went through a lot with each other and shared so many beautiful memories.

I would like to thank Prof. Nassir Navab and Dr. Claudio Laloni for providing me with comfortable working environments at Technical University of Munich and Siemens Corporate Technology respectively. I am more than excited about the chances getting involved with their research groups at the Chair for Computer Aided Medical Procedures and the Department for Digital Perception, and getting to know all the lovely colleagues: Mira Slavcheva, Federico Tombari, Chun-Hao Paul Huang, David Tan, Keisuke Tateno, Wadim Kehl, Christian Rupprecht, Iro Laina, Nikolas Brasch, Ivan Shugurov, Agnieszka Tomczak, Adrian Haarbach, Andres Sanchez, Mahdi Hamad, Roman Kaskman, Fabian Bauer, Fu Li, Hao Yu, Yongheng Zhao, Giorgia Pitteri, Yida Wang, Zhongliang Jiang, Yanyan Li, Benjamin Busam, Shun-Cheng Wu, Fabian Manhardt, Johanna Wald, Anees Kazi, Huseyin Coskun, Helisa Dhamo, Dong Wang, Xiao Chen and numerous others. To Prof. Leonidas Guibas for the research stay at Stanford University and those friendly colleagues in his lab: Qingnan Fan, Yueqi Duan, Li Yi, He Wang, Jingwei Huang, Kaichun Mo, Mikaela Angelina Uy, Zhengping Zhou, Yichen Li, Zhangsihao Yang, Zan Gojcic(from ETH), etc. To Prof. Kai Xu and Prof. Yulan Guo for those enlightening chats about research. To Prof. Luigi Di Stefano and Prof. Mathias Niessner for serving on my PhD committee. Further, I am thankful for all those brilliant people met at summer schools, college seminars, academic conferences and so on.

Moreover, I would like to express my gratitude to my family and friends. My parents always give me full support for my decisions. Life in Munich is made much easier and happier with my friends, especially, Xuanwen Bao, Yanfang Wang, Xin Li, Zilu Zhang, Gordan Ristovski etc. Last but not least, I am blessed for having the company and care from my girlfriend Tianli Ma, especially in the hardest time during the pandemic this year.

Contents

Part I

Introduction and Fundamentals

1 Introduction

As one of the essential ways to perceive the world around us, *vision* comes with great importance. The visual information is processed by our brains, providing bases for further knowledge and decisions. The *Industrial Revolution* marked the outset of transition in production from hand methods to machines, and people's ways of work and lives have been massively changed ever since. Most remarkably, labor-intensive work could be given to machines to accomplish. Yet they were not intelligent, and most of them must be operated by workers. "Intelligent" machines are made possible with the invention of computers, which could serve as "digital brains." More complicated instructions can be given by computers. However, if we want to drive the intelligence of machine to another level, reducing human interference and enabling more automatic operations, we have to teach the machines to analyze and understand the information they obtain, instead of purely executing instructions.

Under this background, *Computer Vision* and *Machine Vision*, aiming at providing algorithms for machines to understand what they see, are drawing significant attention. A higher level of automatism could be for machines, further freeing humans from giving repetitious and tedious low-level instructions. Another side benefit of the researches on this thread is that they might, in return, help us human beings better understand how visual signals are processed by our brains, leading to a better self-understanding.

For a long history, 2D images have been the main research objects due to the rather developed capturing devices, i.e. cameras. Bountiful researches have been conducted on applications such as image classification [83, 115], segmentation [6, 127], object detection [66, 158, 159] etc. Most impressive progresses are made in recent years with the advent of deep learning techniques [121]. Deep learning-based algorithms are reaching or even going beyond human levels on several applications , e.g. face verification [182, 185] and image classification [83, 115, 176, 183]. The availability of large-scale annotated datasets [53, 124] play an important role in these breakthroughs. Yet we live in a world of a higher dimension, which makes it more exciting for us to implement those tasks with 3D data. 3D data also contains richer spatial and structural information, which can be extremely beneficial for extracting local features. One of the biggest problems for 2D images is the loss of scale information, which can be easily tackled in 3D. However, in contrast to the success in the 2D domain, the research progress in 3D is relatively slow. The extra dimension brings a lot more challenges to researchers, especially the *Curse of Dimensionality* [160], which causes a surge in data storage as well as computation complexity. Large scale data like ImageNet [53] in 3D is much harder to collect, and it would take more effort to annotate. Luckily, with the development of 3D data capturing devices such as laser scans, lidar sensors, and depth cameras, obtaining 3D data is becoming cheaper and more convenient, and the data quality has been improved along the way. We could foresee a broader deployment of 3D sensors in the future world. These envisions encourage researchers to bravely face the challenges residing in the 3D data.

Local feature extraction is well established in the domain of 2D images. These features are based either on the gradient, texture, color, intensity, etc. A wide range of prominent descriptors such as SIFT [128, 129], SURF [9], HOG [47], etc. are developed. In recent years, with the trend of deep learning, convolutional networks are also used as a useful tool to extract image features [214]. Local features play a critical role in image-based applications such as object detection, face detection, image retrieval and stitching. Similarly, local features are quite essential for 3D data. They can be extremely useful for robustly estimating rigid poses to align partially overlapped point clouds. However, due to the differences in the data modality, with more geometric information and less reliable surface information, a direct transplant does not work out of the box. In this thesis, we explore how to extract powerful and distinctive local features from point clouds with recent deep learning techniques, which well encode the geometric information of 3D local structures. In the meanwhile, we study how to obtain better rigid pose estimation results for point clouds with those features.

1.1 Motivation

Despite the possibility of reconstructing a 3D scene from a set of images captured with a typical monocular RGB camera with certain techniques such as *structure from motion (SfM)* [173], *simultaneous localization and mapping (SLAM)* [57], etc. , they suffer from problems such as illumination change, textureless regions, unknown scale and so on [211]. With the introduction of 3D data, these issues can be easily overcome thanks to the rich geometric information included. However, researches fully utilizing the advantages of 3D data are still far from satisfactory comparing to their 2D peers. Novel challenges come with 3D data, and the optimal data structure to store and represent 3D data remains an open question. Point clouds, as the raw data captured by 3D scanners, are memory efficient thanks to their sparse data format. However, the irregular and non-structured nature of point clouds make it difficult to extend existing vision algorithms in the field of 2D images.

Local features, also known as *local descriptors*, are compact vectors that describe patterns or intrinsic information of local regions. Local feature extraction algorithms have long been studied on 2D images, and a bunch of methods has been well established and widely applied. Most noteworthy breakthroughs in the past several years are made with deep learning techniques, especially convolutional neural networks. With the development of 3D sensors, more and more 3D data are becoming available. It is no doubt that local features for 3D data are of similar importance. Before the deep learning era, human-designed features patterns was the mainstream [77, 99, 166, 170, 188]. However, due to the gaps between clean 3D data and captured noisy data, the repeatability and distinctiveness of 3D features were found to be way below expectations [74, 75, 110]. Performance-wise they are far behind their 2D peers. Rotation-invariance in 3D features is another desired property. Many of those approaches resort to a local reference frame, which is, by its simplest definition, non-unique [76] and often gets cropped by the noises in the read data. Another branch of LRF-free methods such as PPFH [167] and FPFH [166] are built upon the simple yet rotation-invariant point pair features (PPF). They have been applied in many problems to estimate 3D poses or retrieve and recognize objects [166, 196]. In recent years, with the trend of deep learning, studies on harnessing neural networks to obtain better features for 3D data are also observed. As a data-driven method, the success of deep learning relies heavily on the availability of large-scale

data. Thanks to the availability of more and more convenient and accurate 3D data capture devices, especially depth sensors such as Intel RealSense and Microsoft Kinect, obtaining 3D data is becoming much easier for researchers. Relying on the large amount of 3D data at hand, it would be a better idea to harness neural networks to exploit the patterns hiding in the 3D data, thus replicating the success of deep learning in images. 3DMatch [220] is one pioneer work on this thread, which easily beats the traditional handcrafted features by a large margin. However, it adopts a time-consuming SDF computation stage to convert the original point cloud into a structured voxel grid and then applies a 3D Convolutional Network on the top. PointNet [151] proposes a novel architecture that is able to consume point clouds, which opens a new gate for deep learning on 3D point clouds. Inspired by this, we aim to learn robust 3D local features from point clouds, with sophisticated designed networks and training schemes which could well utilize the sparsity of point clouds.

Pose estimation is another aspect of concern in this thesis. It is a widely studied domain due to its fundamentality in many vision-based systems, such as CAD model pose estimation from images [102, 104, 217], object pose estimation from point cloud [150], camera pose estimation [23, 28, 105, 106, 108], pairwise pose alignment [49, 51, 201] etc. For 3D reconstruction from a set of partial scans, the first vital step is to find all the pairwise relative poses, which is also termed as *pairwise registration*. There are mainly two branches of approaches targetting pairwise registration. The first school tries to find an alignment of two point clouds globally. Iterative closest point (ICP) [11] and its transcendents [11, 122, 187, 210] alternatively hypothesize a correspondence set and minimize the 3D registration error optimizing for the rigid pose. Despite its success, making ICP outlier-robust is considered, even today, to be an open problem [31, 58, 96, 195]. Practical applications of ICP also incorporate geometric, photometric or temporal consistency cues [143] or odometry constraints [223], whenever available. ICP is prone to the initialization and is known to tolerate only up to a $15-30^{\circ}$ misalignment [13]. Another family relies on the set of local matches established between two point clouds, and later recover the pose from the set of matches with techniques such as *random sample consensus (RANSAC)* [64] to eliminate wrong matches. The discovered inliers can then be used in a Kabsch-like [101] algorithm to estimate the optimal transformation. This branch of methods is in general more robust, and not subject to drastic pose changes. A notable drawback of RANSAC is the huge amount of trials required, especially when the inlier ratio is low and the expected confidence of finding a correct subset of inliers is high [38]. This encouraged the researchers to propose accelerations to the original framework, and at this time, the literature is filled with an abundance of RANSAC-derived methods [41, 42, 43, 114], unified by the USAC [157]. Also, since the potential correspondence set is generally established by using 3D local features, the quality of adopted features plays a critical role in the final performance as well.

Apart from the correspondence based methods used in those applications for pose estimation [50, 218, 220], direct regression from the input is also quite popular due to its simplicity [49, 108, 207]. Particularly, with the advent of Deep Learning techniques, it is getting much easier to extract powerful features from either image [53, 83] or 3D data [151, 154], which also paves the way for a more accurate direct regression. PoseNet [108] takes a single RGB image as input and directly regresses the camera location and orientation. One of the biggest pitfalls of direct regression over poses is that it does not qualify the predictions, i.e. all the predictions are assumed to be equally correct [49], and no side information of

uncertainty is available to indicate how good or bad the predictions are. Ambiguity is another huge obstacle faced by many researchers caused by symmetric geometric structures of objects or missing parts, which can severely harm the performance of pose estimation algorithms.

1.2 Objectives

Given the aforementioned applications and backgrounds, our first step starts from learning robust 3D local features on point clouds. Handcrafted features not only require too much domain-specific knowledge but also suffer from the difficulties in modeling the corruption in real data. With 3DMatch [220] as a great pioneer work in this direction, our goal is to better utilize the nature of sparsity for point clouds for a more efficient and robust feature learning and extraction. Instead of a Signed Distance Field (SDF) computation in the data pre-process stage to organize point clouds in a regular grid to allow the use of a 3D Convolution Neural Network, we resort to the novel network architecture of *PointNet* [152], which could operate directly on a set of given points, which is more efficient on the aspects of memory usage and computation. We try to design a feature learning network based on PointNet to see whether we can well utilize it to capture the intrinsic patterns of local patches sampled from point clouds. Follow the paradigm set by 3DMatch as other feature-learning counterparts from 2D domains, we will provide pair and non-pair information for the network to learn. The efficiency brought by the operating on point clouds directly means more local patches could be processed at the same time. We can thus provide the network with comprehensive many-to-many pairwise relationships, to aid the training and improve the quality of learned features.

Despite the fact that obtaining correspondences as supervision is alleviated with the ground truth relative poses between point clouds in the training data, this pipeline cannot work when the poses of training data are missing. Also the obtained correspondences tend to be erroneous due to 1) the inaccuracy of the ground truth poses and 2) distance-based metric for deciding pair/non-pair relationship. Inaccurate ground-truth poses cannot make correct alignment between point clouds, thus leading to massive wrong correspondences. Even when the point clouds are correctly aligned, the distance-based metric can still make mistakes as certain local structures can appear repeatedly across the whole point cloud. Simply label them as non-pairs because they are not in the vicinity of each other can confuse the network a lot. Hence, we will investigate the feasibility of training a feature network without the need for ground truth poses of training point clouds in a non-supervised way, i.e. not using correspondences to supervise the training. In the meanwhile, we expect to learn local features which are completely free from the rotations associated with the point clouds, concentrating solely on the geometric information.

Then we take the pose estimation problem into consideration as well. Instead of relying on the commonly used RANSAC process to estimate the pose from the set of correspondences obtained with our learned features, our objective is to train a network to regress the relative poses between matched local patches. It can help reduce the number of hypotheses generated and improve the pose estimation efficiency. For this purpose, pose information should be given to the regression network. We plan to encode each local patch with a pose-invariant feature and a pose-related feature. The former is ideal for matching purpose and the latter could serve the purpose to recover the relative pose between the two point clouds where those

local patches are sampled from. In the meanwhile, we plan to adopt a multi-task training scheme, aiming at improving both the quality of local features and pose prediction results concurrently.

Finally, we dive into the problems of uncertainty and ambiguity which exist commonly in many pose-related applications. Ambiguity can harm the performance of algorithms, especially for those regression-based ones as the same input can be associated with multiple valid poses. Bingham distribution is ideal for modeling a continuous pose distribution in the quaternion space, and the multimodal Bingham distribution can model multiple peaks which stand for different poses coexisting under ambiguity. We will adopt Bingham distribution as the basic tool, trying to implement an end-to-end distribution modeling of the targeting pose using neural networks. As such, the entropy of the predicted unimodal distribution can be taken as a useful indication of uncertainty. For the learned multimodal distribution, the diverse modes which lead to the problem of ambiguity can be well captured by different components. In general, training a *Mxiture Density Network (MDN)*-like network can suffer from problems such as mode collapse and training instability [19], thus failing to capture ambiguity. To facilitate those issues, we will try to explicitly guide the network to capture different modes with a "Winner Takes All" strategy [164].

In a nutshell, our objectives are as follows:

- Robust and rotation-invariant 3D local features for more robust matching performance;
- More efficient registration pipeline for point cloud pairs based on the set of local correspondences;
- Pose predictions with uncertainty information and multiple branches to capture ambiguity where necessary.

1.3 Contributions

To achieve the aforementioned objectives, several novel algorithms are proposed in this thesis. Our main contributions can be summarized as:

- **A complete framework for learning global context aware local features for robust 3D point cloud matching**. For pairs of point clouds, we process local patches utilizing a bunch of weight-sharing PointNet-based mini-networks to extract local information. Global information is summarized by a max-pooling operation across the set of initial local features. We fuse the local and global information into the final set of local features and design a novel many-to-many loss function to enable efficient training of the framework. Point pair features are incorporated with the original point clouds to further strengthen robustness to rotations.
- **A simple yet effective framework for learning rotation-invariant local features from point clouds without supervision**. Regarding the fact that a certain amount of erroneous pair information might harm the performance of the previous super-

vised framework, we further devised a novel unsupervised framework to obtain fully rotation-invariant 3D local features for point clouds. By designing a novel point-based Autoencoder network, we manage to distill critical geometric information from the local neighborhood, while maintaining the fully rotation-invariant property in the extracted local features. It is simple yet efficient, in the meanwhile, demonstrates superior performance in establishing correspondences.

- **End-to-end point cloud pair registration based on matched local features**. Pairs of matched local patches are aligned under the same relative transformation as their parent point clouds. We disentangle the pose information and geometry information embedded in the local features, and then use a network to recover the relative pose between local patches to enable a more efficient point cloud registration, canceling the requirement for a time-consuming RANSAC procedure. We show empirically that our method improves not only the efficiency but also the pose estimation results.

- **End-to-end modeling of unimodal and multimodal Bingham distributions for pose predictions, enabling uncertainty information as well as tackling ambiguity**. In general, for a pose prediction network, the outputs give no information on how confident the network is. We adopt Bingham distribution as it well fits the quaternion space as well as could be used as a tool to provide uncertainty information. Also, we design a framework to infer end-to-end multimodal Bingham distributions, so as to capture diverse modes leading to ambiguity in the posterior.

1.4 Outline

This section provides the structural organization of this thesis, as well as brief overviews of each of the subsequent chapters. Most of the contents provided in this thesis are either previously published, or under submission for a major conference or journal.

Chapter 2 This chapter provides details for the fundamental theories and technologies used across this work. We will first walk through the mainstream 3D data representation methods as well as their pros and cons. Then we provide basic concepts and mathematical basics for rigid transformation. Last but not least, we illustrate recent deep learning techniques on 3D data, especially the networks which can be utilized for our purposes.

Part II: Feature Learning on Point Cloud

Chapter 3 This chapter presents the motivation and related work on the topic of local features, including handcrafted features, deep learning on point clouds, and learned 3d features. In addition, we provide the mathematical formulation of the feature learning task, which inspires the design of our next two algorithms on feature learning.

Chapter 4 We present our first feature learning framework on point clouds, termed as *PPFNet*. It adopts lightweight PointNet as the backend, so it can simultaneously process all the local patches sampled from one point cloud at the same time. It learns to fuse the global and local information across the local patches, and it is trained using a novel N-tuple loss which fully

utilizes all the available pair relationships and explicitly guides the features to be tailored for the matching purpose.

Chapter 5 In this chapter, we continue to present our second feature learning algorithm which is free from both ground-truth poses of point clouds and pairwise relationships between local patches, termed *PPF-FoldNet*. It follows a standard autoencoder architecture, without skip-links between encoder and decoder, thus forcing the intermediate codeword to distill the most critical information in the resulted features. Moreover, we represent the local patches with pure point pair features to realize full rotation invariance.

Part III: From Feature to Poses

Chapter 6 This chapter presents the motivation and related work on the topic of pose estimation, including rigid pose estimation, pairwise registration, dealing with uncertainty and ambiguity.

Chapter 7 Here we build upon the previous PPF-FoldNet and propose a novel architecture that learns to extract two types of local features from local patches for the purpose of correspondence establishment and pose regression directly. A multi-task training scheme is adopted to improve the quality of local features as well as pose predictions.

Chapter 8 We continue to study the problems of uncertainty and ambiguity in pose estimation in this chapter. We propose to use unimodal Bingham distributions to model uncertainty and multimodal Bingham distributions to tackle ambiguity. Two general frameworks termed as *Unimodal Bingham Network (UBN)* and *Multimodal Bingham Network (MBN)* are designed to infer the essential parameters for unimodal and multimodal Bingham distributions to describe the continuous probabilistic states of the target poses. Novel training schemes inspired by the "Winner Takes All"(WTA) strategy are used to facilitate the training of MBN.

Chapter IV: Conclusion

Chapter 9 We summarize our researches and results in this chapter and further analyze the limitations of our current work. In addition, we list the potential directions of future work.

Chapter V: Appendix

A Here we briefly introduced a novel network architecture, termed *3D-PointCapsNet*, to learn more robust and interpretable features on point clouds. It uses *dynamic routing* to group the input point cloud into different *capsules* and further extract part features from those capsules. We show in **Chapter 5** that with this architecture, stronger local features can be obtained as well.

B We provide extended experiments of our UBN and MBN in **Chapter 8** on the application *Camera Relocalization* to show that our frameworks can be easily extended to other pose-related applications, and manage to improve the pose predictions as well.

C A complete list of the authored and co-authored publications is included here.

A significant part of this thesis, including methods, texts, and other materials are previously published or under submission to a major conference or journal. To make it easy for readers of interest to refer to the original publications and clarify the copyright notice, we list the related publications here with the specific copyright owners. For chapters in **Part II** and **Appendix A**, the related publications are:

- Haowen Deng, Tolga Birdal, Slobodan Ilic, "*PPFNet: Global Context Aware Local Features for Robust 3D Point Matching*", In ©IEEE Computer Vision and Pattern Recognition (CVPR), Salt Lake City, United States, June 2018

- Haowen Deng, Tolga Birdal, Slobodan Ilic, "*PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors*", In ©Springer European Conference On Computer Vision (ECCV), Munich, Germany, September 2018

- Yongheng Zhao, Tolga Birdal, Haowen Deng, Federico Tombari, "*3D Point Capsule Networks*", In ©IEEE Computer Vision and Pattern Recognition (CVPR), Long Beach, United States, June 2019

For chapters in **Part III** and **Appendix B**, the related publications are:

- Haowen Deng, Tolga Birdal, Slobodan Ilic, "*3D Local Features for Direct Pairwise Registration*", In ©IEEE Computer Vision and Pattern Recognition (CVPR), Long Beach, United States, June 2019

- Mai Bui, Tolga Birdal, Haowen Deng, Shadi Albarqouni, Leonidas Guibas, Slobodan Ilic, Nassir Navab, "*6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference*", In European Conference On Computer Vision (ECCV), Glasgow, United Kingdom, August 2020

- Haowen Deng, Mai Bui, Yongheng Zhao, Leonidas Guibas, Slobodan Ilic, Tolga Birdal, "*Deep Bingham Networks: Dealing with Uncertainty and Ambiguity in Pose Estimation*", 2020, [under submission]

All of the rights belonging to the publications that appeared prior to the submission of this thesis are transferred to IEEE/Springer under the relevant copyrights. The figures, texts, and other materials are the author's original published work and are used here with appropriate permissions.

2 Fundamentals

This chapter provides some basic concepts, theories, and techniques used in this thesis. Readers who are experienced in this domain can safely skip forward.

2.1 3D Data Representation

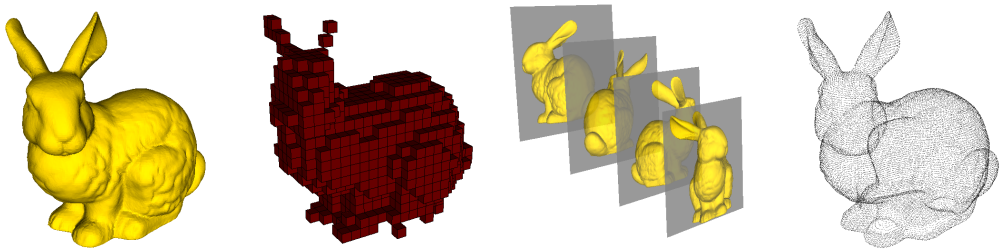
Images are commonly represented as 2D grids. A natural extension of this tradition is to store 3D data in 3D grids. However, this simple idea is confronted with the largely increased amount of storage due to the additional dimension. Different representations for 3D data are proposed, and distinct processings are applied accordingly. Here we cover the most common ones in this section, including *3D volume*, *multi-view* and, most importantly, *point cloud*, which serves as the main structure that we use to manage our data in this thesis.

2.1.1 3D Volume

One of the most popular representations for 3D data is *3D Volume*, which partitions the data space into a regular 3D grid. Each cell of the grid is called a *voxel* and it stores the associated local information of the data. Depending on the specific information stored in the voxel, the volumetric representation can be further divided into different subtypes. For *Occupancy Grid* [137, 206], each cell indicates whether it contains data or not, as in Fig. 2.1b. *Signed Distance Field (SDF)* [220] otherwise stores the distances of the voxel centers to the 3D shape surface. When viewport information is needed, the cells can be further labeled as visible or occluded.

As a natural expansion of images in the 3D domain, 3D Volume is favored as it has the advantage of structure regularity. Typical 2D image operations such as convolution and correlation can be easily extended and applied, and certain experience in 2D can be conveniently generalized as well. Yet, there is a limit on storage resources. One more dimension can lead to a memory explosion. As a compromise, the resolutions used in practice are generally low, which causes information loss. The problem becomes extremely noticeable when it comes to large-scale 3D data. In the meanwhile, 3D data are in general sparse, which means most of the cells do not store any useful information, which leads to a significant amount of unnecessary storage as well as computation.

Octree-based voxelization [162] is one of these efforts to improve storage efficiency. It adopts a hierarchical partition scheme. Space with higher point density is divided with finer-scale voxels, whereas an empty region can be represented with one big voxel. This way, 3D shape details are better preserved and the total number of voxels is decreased as well. However,



(a) Mesh surface (b) Voxel grid (c) Multiview projections (d) Point cloud

Fig. 2.1. A visual illustration of some common ways to represent a complete 3D mesh surface, including voxel grid, multiview projections, and point cloud.

this practice also breaks the regularity property of the grid, which means regular operations stemmed from 2D would not work and sophisticated operations need to be designed for processing these octree-based volumes [161, 162, 186, 198].

2.1.2 Multiview

3D data can also be represented as an ensemble of 2D projections onto different planes using *parallel projection* or *perspetive projective* model, as in Fig. 2.1c. The parallel projection assumes all the rays are parallel to each other during projection. When the rays are perpendicular to the projection plane, it is also called *orthgraphic projection*. Perspective projection can be described with a *pinhole camera model*. Considering a pinhole camera with an intrinsic matrix $\mathbf{K}$:

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

f_x and f_y stand for focal length along x- and y-axis respectively, while (c_x, c_y) is the *principal point* where the optical axis intersects with the image plane.

A 3D point $\mathbf{X} = (x, y, z)$ in the camera coordinate world is associated with a pixel $\mathbf{x}$ located at (p_x, p_y) on the image plane as follows:

$$p_x = \frac{x}{z} f_x + c_x \tag{2.2}$$

$$p_y = \frac{y}{z} f_y + c_y \tag{2.3}$$

Each pixel (p_x, p_y) on the projected 2D images can be used to store distinct information such as textures, distances, occupancy, silhouette, etc.

Multiview allows easy adoption of a wide range of mature techniques developed for 2D data, and it partially avoids the storage overloads as it is controlled by the number of views configured. However, it is still difficult to decide, how many projections are needed to achieve satisfactory performance without too much information loss. When the number of projections

is huge, it falls into the same pitfall as volumetric representation, consuming a significant amount of storage and computation resources. It proves to be quite effective for single 3D objects [153, 181]. But it can be complicated to represent large 3D scenes as too many views would be required.

Among those 2D projections, *depth image* is special as each pixel records the distance of the camera center to the surface along the principal axis. When camera intrinsic parameters are known, A partial point cloud can be obtained by projecting the depth image back into 3D space. This inverse procedure is called *back-projection*. For a pixel located at (p_x, p_y), depth information z and camera intrinsic matrix $\mathbf{K}$, the corresponding 3D point $\mathbf{X}$ is

$$\left(\frac{(p_x - c_x)z}{f_x}, \frac{(p_y - c_y)z}{f_y}, z\right) \tag{2.4}$$

2.1.3 Point Cloud

Point cloud, as indicated by its name, refers to a set of points, as shown in Fig. 2.1d. In general, each point is represented by its cartesian coordinate (x, y, z). Extra per-point information such as normal, curvature, color, etc. can be appended as well. Compared with previous representations, the biggest advantage of the point cloud is its sparsity. For a point cloud, points only appear in the space where there are objects. As such, the storage demand is much smaller to represent a large-scale scene with fine details.

Normal computation

Raw point clouds captured by scanners or converted from depth sensors miss the normal information. To compute a normal, the tangent plane to each local neighborhood is approximated by the least-square fitting as proposed in [89]. Computing the equation of the plane then boils down to a principal component analysis of a covariance matrix created from the nearest neighbors:

$$\mathbf{C} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \tag{2.5}$$

where $\bar{\mathbf{x}}$ denotes the mean, or the center of the local patch. The equation of the plane is then computed from the eigenvectors of $\mathbf{C}$. Due to the sign ambiguity in eigenvector analysis, the direction of the resulting normal is unknown. Thus, we use the convention where each surface normal is made to point towards the camera by ensuring that the dot product between the viewpoint vector and surface normal is acute:

$$\mathbf{p} \cdot \mathbf{n} > 0 \tag{2.6}$$

Point pair feature (PPF)

Point pair features are antisymmetric 4D descriptors, describing the surface of a pair of oriented 3D points $\mathbf{x}_1$ and $\mathbf{x}_2$, constructed as:

$$\boldsymbol{\psi}_{12} = (\|\mathbf{d}\|_2, \angle(\mathbf{n}_1, \mathbf{d}), \angle(\mathbf{n}_2, \mathbf{d}), \angle(\mathbf{n}_1, \mathbf{n}_2)) \tag{2.7}$$

where $\mathbf{d}$ denotes the difference vector between points, $\mathbf{n}_1$ and $\mathbf{n}_2$ are the surface normals at $\mathbf{x}_1$ and $\mathbf{x}_2$. $\|\cdot\|_2$ is the Euclidean distance and $\angle$ is the angle operator computed in a numerically robust manner as in [16]:

$$\angle(\mathbf{v}_1, \mathbf{v}_2) = \text{atan2}\left(\|\mathbf{v}_1 \times \mathbf{v}_2\|_2, \mathbf{v}_1 \cdot \mathbf{v}_2\right) \tag{2.8}$$

$\angle(\mathbf{v}_1, \mathbf{v}_2)$ is guaranteed to lie in the range $[0, \pi)$. By construction, this feature is invariant under Euclidean transformations and reflections as the distances and angles are preserved between every pair of points. It builds upon point clouds and is quite easy and efficient to compute.

2.2 Rigid Transformation

A *rigid transformation* is a geometric operation which alters the orientation and location of an object while preserving the Euclidean distance between any two points from the object. It can be also called as *rigid pose* when describing a static pose status of a rigid object in the Euclidean space, or *rigid motion* when used to emphasize the spatial mobilization. In general, it can be decomposed as a *rotation* and a *translation*, each has 3 degrees of freedom in the 3D Euclidean space.

2.2.1 Rotation matrix

A rotation in 3D can be expressed with a 3×3 orthogonal matrix which satisfies

$$\mathbf{R}^T\mathbf{R} = \mathbf{I} \tag{2.9}$$

and

$$det(\mathbf{R}) = 1. \tag{2.10}$$

The group of matrices which satisfy the aforementioned properties form a complete rotation space *Special Orthogonal Group* $SO(3)$. The inverse of a rotation $\mathbf{R}$ can be conveniently obtained by transposition due to its orthogonality.

$$\mathbf{R}^{-1} = \mathbf{R}^T \tag{2.11}$$

A rotation $\mathbf{R}$ applied on a 3D point $\mathbf{x}$ can thus be expressed as a matrix multiplication

$$\mathbf{x}' = \mathbf{R}\mathbf{x} \tag{2.12}$$

A translation $\mathbf{t}$ can be simply represented as a 3-d vector $(t_x, t_y, t_z) \in \mathbb{R}^3$. So a transformed $\mathbf{x}'$ is

$$\mathbf{x}' = \mathbf{R}\mathbf{x} + \mathbf{t} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} \tag{2.13}$$

As such, a transformation $\mathbf{T}$ can be expressed as a 4×4 matrix. The group of such matrices also form a *Special Euclidean Group* $SE(3)$. Also, an inverse transformation can be expressed as

$$\mathbf{T}^{-1} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \tag{2.14}$$

Despite its convenience in computation, the rotation matrix notation demands 9 values for 3DOF, which suffers from over-parameterization.

2.2.2 Quaternion

As an extended complex number system in Hamilton algebra $\mathbb{H}$, quaternions can be represented in the form

$$\mathbf{q} = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} \tag{2.15}$$

with

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1. \tag{2.16}$$

The norm of a quaternion is

$$||q|| = \sqrt{a^2 + b^2 + c^2 + d^2} \tag{2.17}$$

and its conjugation $\bar{\mathbf{q}}$ is given by

$$\bar{\mathbf{q}} = a - b\mathbf{i} - c\mathbf{j} - d\mathbf{k}. \tag{2.18}$$

A point $\mathbf{x} \in \mathbb{R}^3$ with Cartesian coordinate (x, y, z) can be expressed as a quaternion $\mathbf{p} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ where the real part is set as 0.

A rotation around unit vector $\mathbf{v} = (v_1, v_2, v_3) \in \mathbb{R}^3$ for angle θ can be expressed with a quaternion

$$\mathbf{q} = \cos(\theta/2) + \sin(\theta/2)\,(v_1\mathbf{i} + v_2\mathbf{j} + v_3\mathbf{k}). \tag{2.19}$$

It is easy to verify that $\mathbf{q}$ is a *unit quaternion*.

$$||\mathbf{q}|| = \sqrt{\cos^2(\theta/2) + \sin^2(\theta/2)||\mathbf{v}||} = 1. \tag{2.20}$$

The inverse of the rotation quaternion $\mathbf{q}$ equals its conjugation

$$\mathbf{q}^{-1} = \bar{\mathbf{q}}. \tag{2.21}$$

Rotated point $\mathbf{p}'$ can be computed by

$$\mathbf{p}' = \mathbf{qpq}^{-1}. \tag{2.22}$$

2.2.3 Representation Conversion

Given a 3×3 rotation matrix

$$\mathbf{R} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \tag{2.23}$$

and a 4-d unit quaternion

$$\mathbf{q} = (q_w, q_x, q_y, q_z) \tag{2.24}$$

denoting the same rotation, a bijection mapping can be established by fixing the sign of the first element in $\mathbf{q}$.

Rotation matrix to quaternion

A rotation matrix can be converted to a unit quaternion via the following formula:

$$q_w = \frac{\sqrt{1 + R_{11} + R_{22} + R_{33}}}{2} \tag{2.25}$$

$$q_x = \frac{R_{32} - R_{23}}{4q_w} \tag{2.26}$$

$$q_y = \frac{R_{13} - R_{31}}{4q_w} \tag{2.27}$$

$$q_z = \frac{R_{21} - R_{12}}{4q_w} \tag{2.28}$$

Quaternion to rotation matrix

A rotation matrix can be constructed from a unit quaternion using the following formula:

$$R_{11} = 1 - 2q_y^2 - 2q_z^2 \tag{2.29}$$

$$R_{12} = 2q_xq_y - 2q_zq_w \tag{2.30}$$

$$R_{13} = 2q_xq_z + 2q_yq_w \tag{2.31}$$

$$R_{21} = 2q_xq_y + 2q_zq_w \tag{2.32}$$

$$R_{22} = 1 - 2q_x^2 - 2q_z^2 \tag{2.33}$$

$$R_{23} = 2q_yq_z - 2q_xq_w \tag{2.34}$$

$$R_{31} = 2q_xq_z - 2q_yq_w \tag{2.35}$$

$$R_{32} = 2q_yq_z - 2q_xq_w \tag{2.36}$$

$$R_{33} = 1 - 2q_x^2 - 2q_y^2 \tag{2.37}$$

2.2.4 Estimation from Correspondences

Given a set of matched pairs $\mathbf{P} = \{\mathbf{p}_1, \dots \mathbf{p}_i, \dots \mathbf{p}_N\}$ and $\mathbf{Q} = \{\mathbf{q}_1, \dots, \mathbf{q}_i, \dots, \mathbf{q}_N\}$ from two point clouds, we need to find out the rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$, which minimizes the sum of l_2 distances between the matches after alignment

$$\mathcal{E} = \sum_{i=1}^{N} ||\mathbf{q}_i - (\mathbf{R}\mathbf{p}_i + \mathbf{t})||_2 \tag{2.38}$$

Assume the centroid for $\mathbf{P}$ and $\mathbf{Q}$ are $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ respectively, the centered points $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{q}}_i$ can be written as

$$\hat{\mathbf{p}}_i = \mathbf{p}_i - \bar{\mathbf{p}} \tag{2.39}$$

$$\hat{\mathbf{q}}_i = \mathbf{q}_i - \bar{\mathbf{q}} \tag{2.40}$$

Thus, Eq. (2.38) can be rewritten as

$$\mathcal{E} = \sum_{i=1}^{N} ||\hat{\mathbf{q}}_i + \bar{\mathbf{q}} - \mathbf{R}(\hat{\mathbf{p}}_i + \bar{\mathbf{p}}) - \mathbf{t}||_2 \tag{2.41}$$

$$= \sum_{i=1}^{N} ||\hat{\mathbf{q}}_i - \mathbf{R}\hat{\mathbf{p}}_i + (\bar{\mathbf{q}} - \mathbf{R}\bar{\mathbf{p}} - \mathbf{t})||_2 \tag{2.42}$$

Note the best $\mathbf{t}$ which minimizes $\mathcal{E}$ satisfies

$$\frac{\partial \mathcal{E}}{\partial \mathbf{t}} = -2 \sum_{i=1}^{N} (\mathbf{q}_i - \mathbf{R}\mathbf{p}_i - \mathbf{t}) = 0 \tag{2.43}$$

so

$$\mathbf{t} = \frac{1}{N} (\sum_{i=1}^{N} \mathbf{q}_i - \mathbf{R}\mathbf{p}_i) = \bar{\mathbf{q}} - \mathbf{R}\bar{\mathbf{p}} \tag{2.44}$$

It can be understood that once rotation is known, the translation can be easily computed by comparing the centroids of the two matching sets.

Combining Eq. (2.42) and Eq. (2.44), we have

$$\mathcal{E} = \sum_{i=1}^{N} ||\hat{\mathbf{q}}_i - \mathbf{R}\hat{\mathbf{p}}_i||_2 \tag{2.45}$$

So the main problem now is to find the optimal rotation that minimizes $\mathcal{E}$. Kabsch algorithm [100, 101] provides a matrix-based closed-form solution to this problem, using Lagrange multipliers to find the optimal rotation which minimizes the error $\mathcal{E}$. A more intuitive solution which utilizes *singular value decomposition (SVD)* is later proposed by Arun et al. [4]. There are several similar algorithms targeting the same problem [91, 92, 191, 197], but

they are found having no difference in the robustness of the final solutions in the real-world cases [59].

Expanding Eq. (2.45), we have

$$\mathcal{E} = \sum_{i=1}^{N} (\hat{\mathbf{q}}_i^T \hat{\mathbf{q}}_i + \hat{\mathbf{p}}_i^T \hat{\mathbf{p}}_i) - \sum_{i=1}^{N} 2\hat{\mathbf{q}}_i^T \mathbf{R} \hat{\mathbf{p}}_i. \tag{2.46}$$

$\mathcal{E}$ is minimized when the last term is maximized, which equals maximizing Trace($\mathbf{RH}$), where $\mathbf{H}$ is the covariance matrix:

$$\mathbf{H} = \sum_{i=1}^{N} \hat{\mathbf{p}}_i \hat{\mathbf{q}}_i^T \tag{2.47}$$

$$= \mathbf{USV}^T \tag{2.48}$$

The optimal $\mathbf{R}$ can be computed as

$$\mathbf{R} = \mathbf{VU}^T \tag{2.49}$$

In certain cases, the determinant of the resulted $\mathbf{R}$ can be -1, which does not constitute a legal rotation matrix in the *right handed system*. This problem can be handled by constraining the determinant in the final solution, so

$$\mathbf{R} = \mathbf{V} \begin{pmatrix} 1 & & \\ & 1 & \\ & & \det(\mathbf{UV}^T) \end{pmatrix} \mathbf{U}^T \tag{2.50}$$

However, this algorithm assumes all the matches are correct, otherwise outliers would degenerate the results. This assumption is hard to fulfill in reality. As a result, it is commonly used to generate hypotheses on minimal random subsets of matches in the *Random Sample Consensus (RANSAC)* pipeline [63].

2.3 Deep Learning for 3D Data

Convolutional neural network

Also known as *CNN* or *ConvNet*, it is an important class of networks in deep learning. It is named after the special mathematical operation adopted in the network – *convolution*, which significantly decreases the number of parameters as well as imports *shift invariance*, as opposed to the networks composed of fully connected layers.

CNN has been widely used in image-based applications and achieves remarkable results. An important observation is that, image features from CNNs trained on large datasets such as ImageNet could generalize well to new domains, and performance could be further boosted

with domain-specific data [53, 165]. MVCNN [181] takes advantage of this property of CNN, extracting features from multiple rendered 2D images of a 3D model and aggregates them as the shape descriptor, which is proven to be effective.

3D convolutional network

As indicated by its name, *3D convolutional network* is quite similar to *CNN* only that its kernels have one more dimension, which enables it to operate directly on regular 3D gridded data. It also inherits good properties such as *shift invariance* and *weights sharing* from CNN.

It was initially applied in the field of analyzing sequential images [103, 189], so the extra dimension is mainly for temporal information. 3DShapeNet [206] and VoxNet [137] are two early work, which bring volumetric convolutional architecture to handle pure spatial 3D data. However, it accepts only data stored in structured grids instead of raw point clouds. As a result, a time-consuming data preprocessing stage is necessary, also called *voxelization*. Also, this data format allocates too much memory to store empty space, considering the sparse nature of point clouds.

PointNet

PointNet [152] is an inspiring pioneer work in deep learning addressing the issue of consuming point clouds within a network architecture. The core components include the stacking of independent MLPs anchored on points up until the last layers and a max-pooling layer which aggregates the per-point information as well as makes the network inconsiderate of the input order. Interested readers are encouraged to refer to the original paper for more details [152]. PointNet enables fast and efficient learning from sparse point cloud representation, and serves as the basic deep learning block for the proposed algorithms across this thesis, as our main research objects are point clouds.

Part II

Feature Learning on Point Cloud

3 Introduction

This chapter presents the motivation and related work on the topic of local features, including handcrafted features, deep learning on point clouds, and learned 3d features. In addition, we provide the mathematical formulation of the feature learning task, which inspires the design of our next two algorithms on feature learning.

3.1 Motivation

Local features are one of the most essential tools widely used in computer vision as it precedes fundamental tasks such as correspondence estimation, matching, registration, object detection, image or shape retrieval [129, 140]. Such wide application makes the local features amenable for use in robotics [36], navigation (SLAM) [169] and scene reconstruction for creation of VR contents and digitalization. Developing such a general-purpose tool motivated scholars to hand-craft their 3D features for decades [166, 167, 170]. Unfortunately, we notice that this quest has not been very fruitful in generating the desired repeatable and discriminative local descriptors for 3D point cloud data, especially when the input is partial or noisy [75, 110].

Already in 2D, learned features significantly outperform their engineered counterparts [142, 214], showing promise to enhance multitudes of vision solutions. Thus, it is natural for scholars to tackle the task of 3D local feature extraction employing similar approaches. But the latest works either base the representation on a handcrafted input encoding [109] or try to naively extend the networks from the 2D domain to 3D [220]. Both approaches are sub-optimal, as they do not address learning of the raw data, point sets.

Based on the recent progress in the community, our goal is to develop efficient deep learning-based methods to extract powerful local features for point clouds, which are both discriminative and robust to data incompleteness and noises.

3.2 Related Work

3.2.1 Handcrafted Features

Before the deep learning era, manually handcrafting is the most common way of obtaining local features. There has been a wide range of well-established image features such as SIFT [129], SURF [9], ORB [140]. Similar to its 2D counterpart, extracting meaningful and robust local descriptors from 3D data has been keeping the 3D computer vision researchers busy for a long period of time [77, 99, 166, 170, 188]. Unfortunately, contrary to 2D, the

repeatability and distinctiveness of 3D features were found to be way below expectations [74, 75, 110]. In general, handcrafting features involves manual design and selection of the feature patterns and properties, and requires expert domain knowledge about the data. Moreover, the various types of noises, clutter, missing parts, etc. existing commonly in the real data captured by devices are hard to be modeled, which in return would harm the performance by a large margin in real-life applications.

Among existing approaches, many of them try to discover a local reference frame (LRF) to rotate the local patch surrounding a keypoint back to its canonical form, to acquire rotation invariance. However, by its simplest definition, LRF can be non-unique [76] and none of the existing LRF methods can provide perfect repeatability. Also, the computation of LRFs is based on the eigenvectors of the covariance matrix of the set of points in the sample, which is sensitive to the data corruptions. This shifts the attention to LRF-free methods such as the rotation invariant point pair features (PPF) to be used as the basis for creating powerful descriptors like PPFH [167] and FPFH [166]. PPFs are also made semi-global to perform reasonably well under difficult scenarios, such as clutter and occlusions [13, 16, 56, 86]. Thus, they have been applied in many applications to estimate 3D poses or retrieve and recognize objects [166, 196]. Nevertheless, A comprehensive evaluation conducted by Guo et al. [74] indicates that, there is still a huge space for improvement for handcrafted local features, especially for being more resilient to those problems in the real data.

3.2.2 Learning on Point Clouds

The commonly used convolutional operations on 2D images cannot be easily extended to point clouds due to the requirement of the structured data format. Thus, early works choose to first convert the point clouds into partitioned 3D grids, and then give it to a 3D convolutional network to process [137, 220]. On the other hand, point clouds can also be treated as graphs by associating edges among neighbors. This paves the way to the appliance of graph convolutional networks [134]. Wang et. al. [200] tackle the segmentation tasks on point sets via graph convolutions networks (GCNs), while Qi et. al. [156] apply GCNs to RGB-D semantic segmentation.

Direct consumption of unstructured point input in the form of a set within deep networks started with PointNet. Qi et al. [151] proposed to use a point-wise multi-layer perceptron (MLP) and aggregated individual feature maps into a global one by a permutation invariant max pooling. Agnostic to the input order, PointNet can generate per-point local descriptors as well as a global one, which can be combined to solve different problems such as keypoint extraction, 3D segmentation, or classification. While not being the most powerful network, it clearly sets out a successful architecture giving rise to many successive works [2, 154, 155, 174].

While PointNet can work on point clouds, it is still a supervised architecture, and constructing unsupervised extensions like an auto-encoder on points is non-trivial as the upsampling step is required to interpolate sets [154, 215]. Yang et. al. bring a different perspective and instead of resorting to costly voxelizations, propose *folding* [212], as a strong decoder alternative. Folding warps an underlying low-dimensional grid towards the desired set, specifically a 3D

point cloud. Compared to other unsupervised methods, including GANs [205], FoldingNet achieves superior performance in common tasks such as classification, and therefore, we show it later in our PPF-FoldNet that we benefit from its decoder structure, though slightly altered.

3.2.3 Learned 3D Features

With the advent of deep learning, efforts on addressing problems like 3D retrieval, object classification, segmentation, as well as feature learning have been witnessed using 3D data [178, 206]. In general, different methods with dedicated network architectures are designed with regard to the specific way of representing the sparse and unstructured 3D data.

Early works exploited the apparent dense voxels [137, 153], usually in form of a binary-occupancy grid. This idea is quickly extended to more informative encoding such as TDF [179], TSDF [220] (Truncated Signed Distance Field), multi-label occupancy [206] and other different ones [95, 213]. Since mainly used in the context of 3D retrieval, entire 3D objects were represented with small voxel grids 30^3 limited by the maximal size of 3D convolutions kernels [137]. These representations have also been used for describing the local neighbors in the context of 3D descriptor learning. One such contemporary work is 3DMatch [220]. It is based on a robust volumetric TSDF encoding with a contrastive loss to learn correspondences. Albeit straightforward, 3DMatch ignores the raw nature of the input: sparsity and unstructured-ness. It uses dense local grids and 3D CNNs to learn the descriptor and thus can fall short in training/testing performance and recall.

A parallel track of works follow a view based scheme [60, 94], where the sub-spaces of 3D information in the form of projections or depth map are learned with well studied 2D networks. Promising potential, these methods do not cover for sparse point sets. Another spectrum of research exploits graph networks [48, 113] also to represent point sets [156]. This new direction prospers in other domains but graph neural networks are not really suited to point clouds, as they require edges, not naturally arising from point sets. Khoury et. al. [109] try to overcome the local representation problem by a hand-crafted approach and use a deep-network only for a dimensionality reduction. Their algorithm also computes the non-unique LRF and taking such a path deviates from the efforts of end-to-end learning. Yet, their method involves too much handcraft effort and fails to fully utilize the power of the deep network.

3.3 Problem Formulation

Before diving into our methods on learned features, let us map out the setting which shapes the cornerstones of our architectures.

Consider two 3D point sets $\mathbf{X} \in \mathbb{R}^{n\times 3}$ and $\mathbf{Y} \in \mathbb{R}^{n\times 3}$. Let $\mathbf{x}_i$ and $\mathbf{y}_i$ denote the coordinates of the i^{th} points in the two sets, respectively. Momentarily, we assume that for each $\mathbf{x}_i$, there exists a corresponding $\mathbf{y}_i$, a bijective map. Then following [122] and assuming rigidity, the two sets are related by a correspondence and pose (motion), represented by a permutation

matrix $\mathbf{P} \in \mathbb{P}^n$ and a rigid transformation $\mathbf{T} = \{\mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3\}$, respectively. Then the L_2 error of point set registration reads:

$$d(\mathbf{X}, \mathbf{Y}|\mathbf{R}, \mathbf{t}, \mathbf{P}) = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{R}\mathbf{y}_{i(\mathbf{P})} - \mathbf{t}\|^2 \tag{3.1}$$

where $\mathbf{x}_i$ and $\mathbf{y}_{i(\mathbf{P})}$ are matched under $\mathbf{P}$. We assume $\mathbf{X}$ and $\mathbf{Y}$ are of equal cardinality ($|\mathbf{X}| = |\mathbf{Y}| = n$). In form of homogenized matrices, the following is equivalent:

$$d(\mathbf{X}, \mathbf{Y}|\mathbf{T}, \mathbf{P}) = \frac{1}{n} \|\mathbf{X} - \mathbf{P}\mathbf{Y}\mathbf{T}^T\|^2 \tag{3.2}$$

Two sets are ideally matching if $d(\mathbf{X}, \mathbf{Y}|\mathbf{T}, \mathbf{P}) \approx 0$. This view suggests that, to learn an effective representation, one should preserve a similar distance in the embedded space:

$$d_f(\mathbf{X}, \mathbf{Y}|\mathbf{T}, \mathbf{P}) = \frac{1}{n} \|f(\mathbf{X}) - f(\mathbf{P}\mathbf{Y}\mathbf{T}^T)\|^2 \tag{3.3}$$

and $d_f(\mathbf{X}, \mathbf{Y}|\mathbf{T}, \mathbf{P}) \approx 0$ also holds for matching points sets under any action of $(\mathbf{T}, \mathbf{P})$. Thus, for invariance, it is desirable to have: $f(\mathbf{Y}) \approx f(\mathbf{P}\mathbf{Y}\mathbf{T}^T)$. Ideally we would like to learn f being invariant to permutations $\mathbf{P}$ and as intolerant as possible to rigid transformations $\mathbf{T}$. Therefore, we choose to use a minimally handcrafted point set to deeply learn the representation.

This formulation motivates us to exploit PointNet architecture [152] which intrinsically accounts for unordered sets and consumes sparse input in our proposals. To boost the tolerance to transformations, we will benefit from point-pair-features, the true invariants under Euclidean isometry.

4 Learning Global Context Aware Local Features

In this chapter, we present our first method, *PPFNet*, which aims to extract distinctive and efficient 3D local features from point clouds. Unlike traditional handcrafted features, our method is data-driven thanks to the deep learning-backed framework. Compared to counterparts such as 3DMatch [220], which adopts a 3D convolutional network, our network consumes point clouds directly and enjoys the sparsity of point clouds to the full extent. With an efficient network design and loss function, our method manages to optimize features for all the local patches sampled from pairs of point clouds simultaneously with a comprehensive supervision signal. Extensive evaluations verified the improved feature qualities as well as matching performance.

4.1 Introduction

Local features serve as one of the most fundamental building blocks for a wide range of applications, so researchers are enthusiastic about getting more robust features. The success of deep learning drastically changes the research paradigm from creating explicit rules to describe a local patch to a data-driven manner. For 2D images, the convolutional network is the real game-changer, where the learned convolutional kernels can well capture the local patterns [82, 214]. To reproduce the success in learning 3D local features, two major factors need to be satisfied – sufficient 3D data and a proper network.

3DMatch [220] acquired point clouds fused from range scans and presented a benchmark with sufficient training data, and adopted a 3D convolutional network to process the data represented as *truncated signed distance function(TSDF)* grids and followed a siamese network architecture to train it. However, both the data preparation and the inference stage are quite heavy due to the volumetric representation. It would be favored if we can make better use of the sparsity of point clouds while learning robust 3D local features. This idea is made possible with the proposal of PointNet [151], which allows direct processing of a set of unordered points by a network. Different from the paradigm used in 3DMatch [220], where local patches are sampled and individually processed by a 3D convolutional network, our network can simultaneously consume all the local patches sampled from a partial scan. The computation and memory efficiency brought by point cloud sparsity and PointNet [151] limits the overload within GPU capacity.

In summary, *PPFNet* is introduced for the sake of learning discriminative and fast local features which describe 3D local patches from point clouds with increased tolerance to rotations. To fulfill the expectations, we first represent each local patch with an augmented set of simple

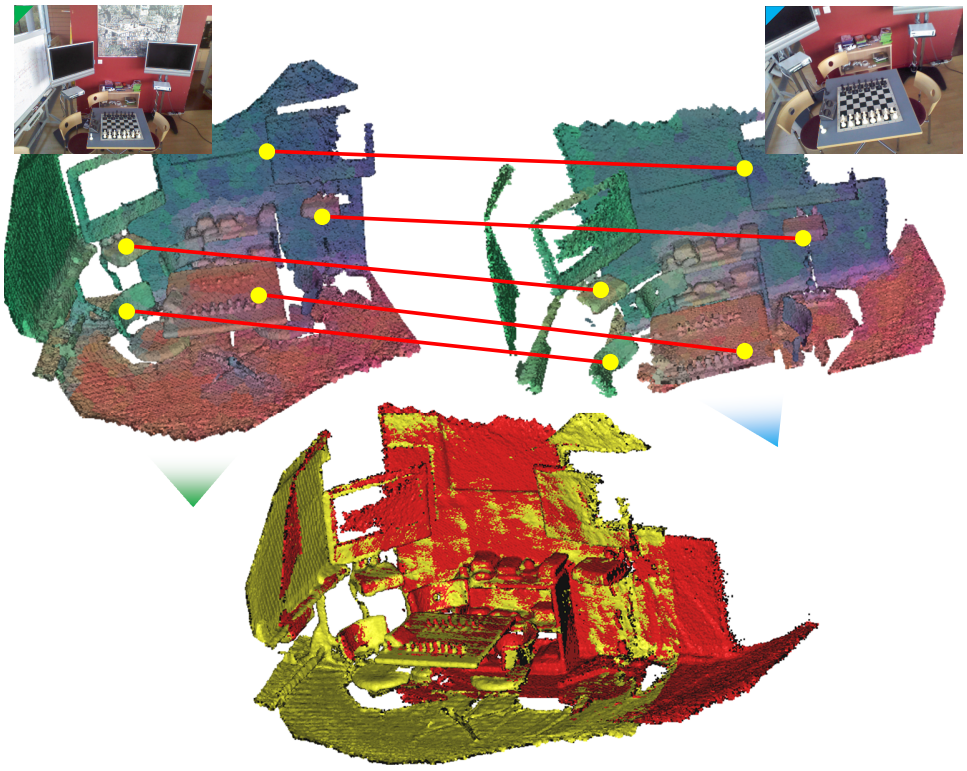


Fig. 4.1. PPFNet generates repeatable, discriminative descriptors and can discover the correspondences simultaneously given a pair of fragments, and they can be registered by estimating the relative pose using correspondences. Point sets are colored by a low dimensional embedding of the local feature for visualization. 3D data and the illustrative image are taken from 7-scenes dataset [175].

geometric relationships, including points, normals and point pair features(PPF) [56, 166]. A novel loss function, termed as *N-tuple loss*, is later proposed to simultaneously embed multiple pairs, including matching and non-matching, into a proper Euclidean domain. Our loss resembles the contrastive loss [80], but instead of pairs, we consider an N-combination of the input points within two scene fragments to boost the separability and reduce confusion in the feature space. Thanks to this many-to-many loss function, global context can be smoothly injected into the learning, i.e. PPFNet is aware of the other local features when establishing correspondence for a single one. In the meantime, thanks to the parallel processing, PPFNet is quite fast in the inference stage. All of those designs are combined as a new pipeline, which trains our network from correspondences in 3D fragment pairs. PPFNet extends PointNet [152] and thereby is natural for point clouds and neutral to permutations. Extensive evaluations show that PPFNet achieves state-of-the-art performance in accuracy, speed, robustness to point density, and tolerance to changes in 3D pose. Fig. 4.1 visualizes our features and illustrates the robust matching we can perform.

4.2 PPFNet

In this section, we provide details for all the core components of our *PPFNet*. We first explain how we prepare and represent the local patches, which describe the local geometries, from the sampled keypoints anchored on a 3D point cloud. Then we elaborate on our novel network

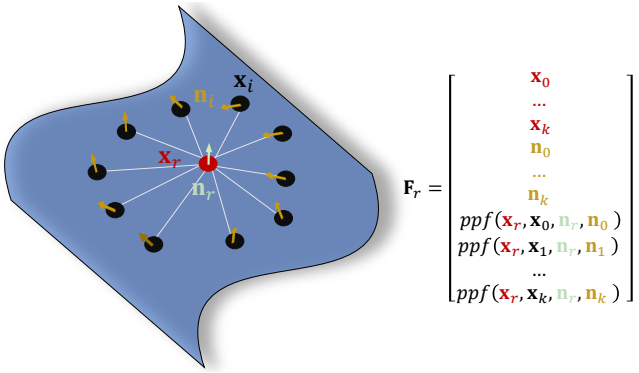


Fig. 4.2. Illustartion of the simplistic encoding for a local patch. It carries point coordinates, normals as well as PPFs.

architecture, specially designed with merit to consume data stored as unstructured point sets and infer the local features from those local patches. The output of our network is the set of local features associated with the original keypoints, as shown in Fig. 4.3. Last but not least, we explain our training method and, more importantly, a novel loss function which tries to solve the combinational correspondence problem in a global manner.

Local Patch Preparation and Representation

A local patch refers to the neighborhood surrounding a selected point. Given a reference point lying on a point cloud $\mathbf{x}_r \in \mathbf{X}$, we define a local region $\mathbf{\Omega} \subset \mathbf{X}$ and collect a set of points $\{\boldsymbol{m}_i\} \in \mathbf{\Omega}$ within this local vicinity. To enrich the geometric information, We also compute the normals for the point set [89]. Altogether, the oriented $\{\boldsymbol{x}_r \cup \{\boldsymbol{m}_i\}\}$ represent the geometry of the local patch.

Then we compute the point pair features for the local patch. Note in the original setting, for N points, N^2 PPFs can be computed. To avoid quadric pairing, our solution is to pair each neighboring point $\boldsymbol{m}_i$ only with the reference $\boldsymbol{x}_r$ and compute the PPFs. Complexity-wise, this is quite similar as using the points themselves, as we omit the quadratic pairing thanks to fixation of central reference point $\mathbf{x}_r$.

Our local patch is finally represented as a combined set of points, normals, and PPFs, as shown in Fig. 4.2. Despite the extra PPF components, our encoding strictly follows the same data organization as the raw point clouds. So it can be handled the same way as the point cloud. Also, in contrast to TSDF coversion [220], the computation of PPFs is quite fast, which does not bring a lot of burdens to overall efficiency. We will show that later in our experiments.

Network Architecture

The overall architecture of PPFNet is shown in Fig. 4.3. Our input consists of N local patches uniformly sampled from a fragment, which is a partial scan in the form of a point cloud. Due to the sparsity of point-style data representation and efficient GPU utilization of PointNet, PPFNet can absorb those N patches concurrently.

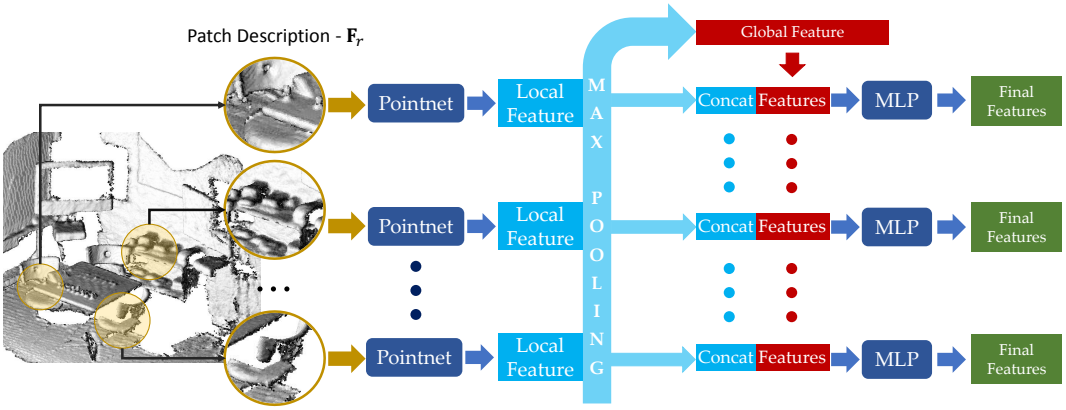


Fig. 4.3. PPFNet, our inference network, consists of multiple PointNets, each responsible for a local patch. To capture the global context across all local patches, we use a max-pooling aggregation and fusing the output back into the local description. This way we are able to produce stronger and more discriminative local representations.

The first module of PPFNet is a group of mini-PointNets that are responsible for extracting features from local patches. Weights and gradients are shared across all the PointNets during training. A max-pooling layer then aggregates all the local features into a global one, summarizing the distinct local information to the global context of the whole fragment. This global feature is then concatenated to every local feature. A group of MLPs is used to further fuse the global and local features into the final global context aware local descriptors.

N-tuple Loss

Our goal is to use PPFNet to extract features for local patches, a process of mapping from a high dimensional data space into a low dimensional feature space which is more separable. The distinctiveness of the resulting features is closely related to the separability in the embedded space. Ideally, the proximity of neighboring patches in the data space should be preserved in the feature space.

To this end, the state of the art seems to adopt two loss functions: contrastive loss [220] and triplet loss [109], which try to consider pairs and triplets respectively. Yet, a fragment consists of more than 3 patches, and in that case the widely followed practice trains networks by randomly retrieving 2/3-tuples of patches from the dataset. However, networks trained in such manner only learn to differentiate maximum 3 patches, preventing them from uncovering the true matching, which is combinatorial in the patch count. Especially for our PPFNet, which is able to process N patches from each fragment at a time, it would be both suboptimal and wasteful to only sample some pairs/triplets for the training.

Generalizing these losses to N-patches, we propose N-tuple loss, an N-to-N contrastive loss, to correctly learn to solve this combinatorial problem by catering for the many-to-many relations as depicted in Fig. 4.4. Given the ground truth transformation $\mathbf{T}$, N-tuple loss operates by constructing a correspondence matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ on the points of the aligned fragments. $\mathbf{M} = (m_{ij})$ where:

$$m_{ij} = \mathbb{1}(\|\mathbf{x}_i - \mathbf{T}\mathbf{y}_j\|_2 < \tau) \tag{4.1}$$

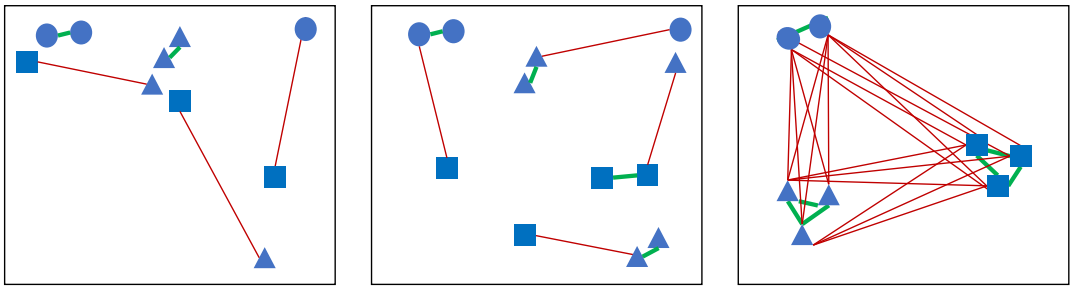


Fig. 4.4. Illustration of N-tuple sampling in feature space. Green lines link similar pairs, which are coerced to keep close. Red lines connect non-similar pairs, pushed further apart. Without N-tuple loss, there remains to be some non-similar patches that are close in the feature space and some distant similar patches. Our novel N-tuple method pairs each patch with all the others guaranteeing that all the similar patches remain close and non-similar ones, distant.

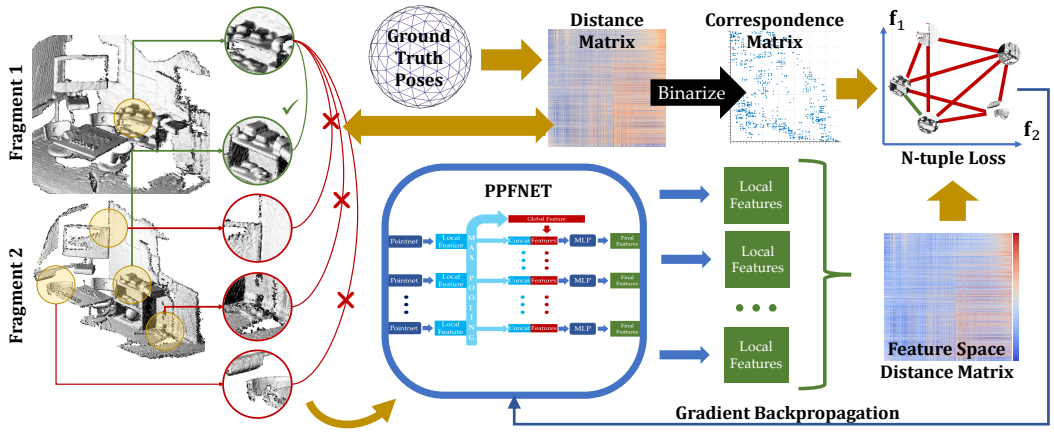


Fig. 4.5. Overall training pipeline of PPFNet. Local patches are sampled from a pair of fragments respectively, and feed into PPFNet to get local features. Based on these features a feature distance matrix is computed for all the patch pairs. Meanwhile, a distance matrix of local patches is formed based on the ground-truth rigid pose between the fragments. By binarizing the distance matrix, we get a correspondence matrix to indicate all the matching and non-matching relationships between patches. N-tuple loss is then calculated by coupling the feature distance matrix and correspondence matrix to guide the PPFNet to find an optimal feature space.

$\mathbb{1}(\cdot)$ is an indicator function. Likewise, we compute a feature-space distance matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ and $\mathbf{D} = (d_{ij})$ where

$$d_{ij} = \|f(\mathbf{x}_i) - f(\mathbf{y}_j)\|_2 \tag{4.2}$$

The N-tuple loss takes the two distance matrices as input to solve the correspondence problem. For simplicity of expression, we define an operation $\sum^*(\cdot)$ to sum up all the elements in a matrix. N-tuple loss can be written as:

$$L = \sum^* \left(\frac{\mathbf{M} \circ \mathbf{D}}{\|\mathbf{M}\|_2^2} + \alpha \frac{max(\theta - (1 - \mathbf{M}) \circ \mathbf{D}, 0)}{N^2 - \|\mathbf{M}\|_2^2} \right) \tag{4.3}$$

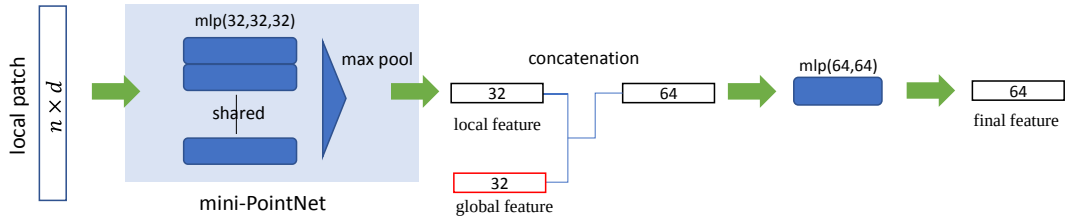


Fig. 4.6. Detailed pipeline with network specifications for processing a single local patch to obtain the final corresponding local feature in our PPFNet.

Here $\circ$ stands for Hadamard Product - element-wise multiplication. α is a hyper-parameter balancing the weight between matching and non-matching pairs and θ is the lower-bound on the expected distance between non-correspondent pairs.

We train PPFNet with N-tuple loss, as shown in Fig. 4.5, by drawing random pairs of fragments instead of patches. This also eases the preparation of training data. we use $\alpha = 1.0$ and $\theta = 1.0$ in our experiments.

Implementation

Next, we provide more details on our implementation, including the preparation of local patches, the algorithm for sampling keypoints and network specifics.

Local patches

Our input encoding uses a 17-point neighborhood to compute the normals for the entire scene, using the well-accepted plane fitting [89]. For each fragment, we anchor 2048 sample points distributed with spatial uniformity. These sample points act as keypoints and within their $30cm$ vicinity, they form the patch, from which we compute the local PPF encoding. Similarly, we down-sample the points within each patch to 1024 to facilitate the training as well as to increase the robustness of features to various point density and missing parts. For occasional patches with insufficient points in the defined neighborhood, we randomly repeat points to ensure identical patch size. PPFNet extracts compact descriptors of dimension 64.

Sampling algorithm

How we sample the point cloud down to 2048 keypoints (samples) plays an important role in learning. We try to be spatially as uniformly distributed as possible so that the samples are further apart and less dependent on one another. For this, we use the greedy algorithm given in Algorithm 1 inspired by [15]. Note that the algorithm involves a search over the so-far-sampled cloud, which we speed up using a voxel-grid.

Network

Due to the constraint of GPU memory, we adopt a minimized version of vanilla PointNet in our implementation. Fig. 4.6 demonstrates a pipeline for processing one single patch and more details of our network. The size of a local patch is $n \times d$, where $n = 1024$ is the number

Algorithm 1 Distance Constrained Sampling

Require: Source point cloud $\mathbf{X}$, Relative threshold τ
Ensure: Sampled point cloud $\mathbf{S}$ and its normals $\mathbf{N}$
  Compute normals for $\mathbf{X}$
  $\mathbf{S} \leftarrow []$
  $\mathbf{N} \leftarrow []$
  $R_d \leftarrow diameter(\mathbf{X})$
  for $\mathbf{x} \in \mathbf{X}$ **do**
    $d_{min} = \min_{(\mathbf{t} \in \mathbf{S})} |\mathbf{x} - \mathbf{t}|$
    if $(d_{min} > \tau R_d)$ **then**
      $\mathbf{S} \leftarrow \mathbf{S} \bigcup \mathbf{x}$
      $\mathbf{N} \leftarrow \mathbf{N} \bigcup \mathbf{n}(\mathbf{x})$ ▷ sample the normal as well
    end if
  end for

of points in the patch and d depends on the specific representation of local patch. For only point coordinates and normals, $d = 6$; for the one with extra PPF, $d = 10$. A patch is first sent into a mini-PointNet with three layers, each has 32 nodes, and then a max-pooling function aggregates all the information into a 32-dimensional local feature. After combining with the 32-dimensional global feature, it is further processed by a two-layer MLP, in which each layer has 64 nodes. The dimension of the final feature for the local patch is 64.

PPFNet is implemented in the popular Tensorflow [1]. The initialization uses random weights and ADAM [111] optimizer minimizes the loss. Our network operates simultaneously on all 2048 patches. The learning rate is set at 0.001 and exponentially decayed after every 10 epochs until 0.00001. Due to the hardware constraints, we use a batch size of 2 fragment pairs per iteration, containing 8192 local patches from 4 fragments already. This generates 2×2048^2 combinations for the network per batch.

4.3 Evaluation

In this section, we present comprehensive evaluations of our PPFNet. We compare the performance of our PPFNet against other handcrafted 3D local features as well as deeply learned competitors. Moreover, in-depth ablation studies for the key components of PPFNet are conducted.

4.3.1 Experiment Setup

Here we consider the baselines that we are going to compare against, the appropriate dataset to use and introduce the metric that we use to measure the matching performance and feature quality.

Baselines

We evaluate our method against hand-crafted baselines of Spin Images [99], SHOT [170], FPFH [166], USC [188], as well as 3DMatch [220], the state of the art deep learning based

3D local feature descriptor, the vanilla PointNet [152] and CGF [109], a hybrid hand-crafted and deep descriptor designed for compactness.

Dataset

We concentrate on real sets rather than synthetic ones and therefore our evaluations are the diverse 3DMatch RGBD benchmark [220], in which 62 different real-world scenes retrieved from the pool of datasets Analysis-by-Synthesis [192], 7-Scenes [175], SUN3D [208], RGB-D Scenes v.2 [120] and Halber et [81]. This collection is split into 2 subsets, 54 for training and validation, 8 for testing. The dataset typically includes indoor scenes like living rooms, offices, bedrooms, tabletops, and restrooms. More details about this dataset can be found in [220]. As our input consists of only point geometry, we solely use the fragment reconstructions captured by the Kinect sensor and not the color.

Our evaluation data consists of fragments from 7-scenes [175] and SUN3D [208] datasets.

Metric

Let us assume that a pair of fragments $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}$ and $\mathbf{Q} = \{\mathbf{q}_i \in \mathbb{R}^3\}$ are aligned by an associated rigid transformation $\mathbf{T} \in SE(3)$, resulting in a certain overlap. We then define a non-linear feature function $g(\cdot)$ for mapping from input points to feature space, and in our case, it summarizes the PPF computation and encoding to a codeword. The feature for point $\mathbf{p}_i$ is $g(\mathbf{p}_i)$, and $g(\mathbf{P})$ is the pool of features extracted for the points in $\mathbf{P}$. To estimate the rigid transformation between $\mathbf{P}$ and $\mathbf{Q}$, the typical approach finds a set of matching pairs in each fragment and associates the correspondences. The inter point pair set $\mathbf{M}$ is formed by the pairs $(\mathbf{p}, \mathbf{q})$ that lie mutually closely in the feature space by applying nearest neighbor search NN:

$$\mathbf{M} = \{\{\mathbf{p}_i, \mathbf{q}_i\}, g(\mathbf{p}_i) = NN(g(\mathbf{q}_i), g(\mathbf{P})),\ g(\mathbf{q}_i) = NN(g(\mathbf{p}_i), g(\mathbf{Q}))\ \} \tag{4.4}$$

True matches set $\mathbf{M}_{gnd}$ is the set of point pairs with a Euclidean distance below a threshold τ_1 under ground-truth transformation $\mathbf{T}$.

$$\mathbf{M}_{gnd} = \{\{\mathbf{p}_i, \mathbf{q}_i\} : (\mathbf{p}_i, \mathbf{q}_i) \in \mathbf{M}, ||\mathbf{p}_i - \mathbf{T}\mathbf{q}_i||_2 < \tau_1\} \tag{4.5}$$

We now define an inlier ratio for $\mathbf{M}$ as the percentage of true matches in $\mathbf{M}$ as $r_{in} = |\mathbf{M}_{gnd}|/|\mathbf{M}|$. To successfully estimate the rigid transformation based on $\mathbf{M}$ via registration algorithms, r_{in} needs to be higher than τ_2. For example, in a common RANSAC pipeline, in order to get 99.9% confidence in finding a subset with at least three correct matches $\mathbf{M}$, with an inlier ratio $\tau_2 = 5\%$, one would require at least 55258 iterations. Theoretically given $r_{in} > \tau_2$, it is highly probable a reliable local registration algorithm would work, regardless of the robustifier. So instead of using the local registration results to judge the quality of features, which would be both slow and not so straightforward, we define $\mathbf{M}$ with $r_{in} > \tau_2$ votes for a correct match of two fragments.

Each scene in the benchmark contains a set of fragments. Fragment pairs $\mathbf{P}$ and $\mathbf{Q}$ having an overlap above 30% under the ground-truth alignment are considered to match. They together

Tab. 4.1. Results of matching performance on the 3DMatch benchmark. *Kitchen* is from 7-scenes [175] and the rest from SUN3D [208].

	Spin Images [99]	SHOT [170]	FPFH [166]	USC [188]	PointNet [152]	CGF [109]	3DMatch [220]	3DMatch-2K [220]	PPFNet (ours)
Kitchen	0.1937	0.1779	0.3063	0.5573	0.7115	0.4605	0.5751	0.5296	**0.8972**
Home 1	0.3974	0.3718	0.5833	0.3205	0.5513	0.6154	**0.7372**	0.6923	0.5577
Home 2	0.3654	0.3365	0.4663	0.3077	0.5385	0.5625	**0.7067**	0.6202	0.5913
Hotel 1	0.1814	0.2080	0.2611	0.5354	0.4071	0.4469	0.5708	0.4779	**0.5796**
Hotel 2	0.2019	0.2212	0.3269	0.1923	0.2885	0.3846	0.4423	0.4231	**0.5769**
Hotel 3	0.3148	0.3889	0.5000	0.3148	0.3333	0.5926	**0.6296**	0.5185	0.6111
Study	0.0548	0.0719	0.1541	0.5068	0.4315	0.4075	**0.5616**	0.3904	0.5342
MIT Lab	0.1039	0.1299	0.2727	0.4675	0.5065	0.3506	0.5455	0.4156	**0.6364**
Average	0.2267	0.2382	0.3589	0.4003	0.4710	0.4776	0.5961	0.5085	**0.6231**

form the set of fragment pairs $\mathbf{S} = \{(\mathbf{P}, \mathbf{Q})\}$ that are used in evaluations. The quality of features is measured by the recall R of fragment pairs matched on $\mathbf{S}$:

$$R = \frac{1}{|\mathbf{S}|} \sum_{i=1}^{|\mathbf{S}|} \mathbb{1}\left(r_{in}\big(\mathbf{S}_i = (\mathbf{P}_i, \mathbf{Q}_i)\big) > \tau_2 \right) \tag{4.6}$$

We believe that this can show the true quality of the correspondence sets, which reflects the ability of the applied local features. Inspired by [109], we accredit recall as a more effective measure in our experiments, as the precision can always be improved by better corresponding pruning [33, 35].

4.3.2 Matching Performance

We start with the evaluation to demonstrate that our learned features by PPFNet could achieve better matching performance. In the evaluation, we run the same experiments for the aforementioned baselines. To set the experiments fairer, we also show a version of 3DMatch, where we use 2048 local patches per fragment instead of 5K, the same as in our method, denoted as 3DMatch-2K. For CGF, we use the provided pre-trained weights [109]. For the evaluation metric, We use $\tau_1 = 10cm$ and $\tau_2 = 0.05$.

As seen from Tab. 4.1, PPFNet outperforms all the handcrafted counterparts in the average recall. It also shows a consistent advantage over 3DMatch-2K, using an equal amount of patches. Finally and remarkably, we are able to show $\sim 2.7\%$ improvement on mean recall over the original 3DMatch, using only $\sim 40\%$ of the keypoints for matching. The performance boost from 3DMatch-2K to 3DMatch also indicates that having more keypoints is advantageous for matching. Our method expectedly outperforms both vanilla PointNet and CGF by 15%.

We continue to show in Tab. 4.2 that adding more samples brings benefit, but only up to a certain level (< 5K). For PPFNet, adding more samples also increases the global context and thus, following the advent in hardware, we have the potential to further widen the performance gap over 3DMatch, by simply using more local patches.

Tab. 4.2. Recall of 3DMatch for different number of sample patches used for matching.

Samples	128	256	512	1K	2K	5K	10K	20K	40K
Recall	0.24	0.32	0.40	0.47	0.51	0.59	0.59	0.56	0.60

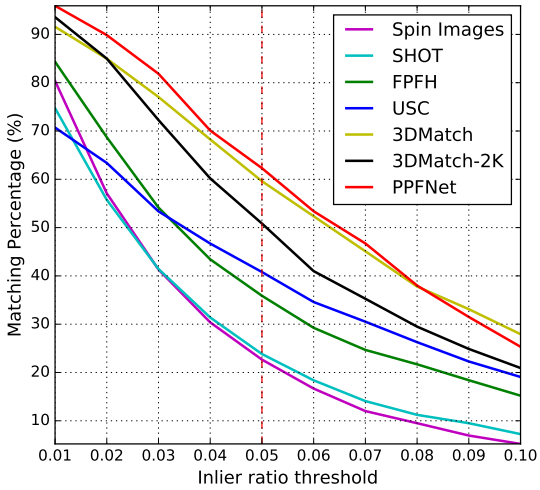


Fig. 4.7. Recall on 3DMatch benchmark. Our method consistently outperforms the state-of-the-art on matching task (no RANSAC is used) in terms of recall.

To show that we do not cherry-pick τ_2 but get consistent gains, we also plot the recall computed with the same metric for different inlier ratios in Fig. 4.7. There, for the practical choices of τ_2, PPFNet persistently remains above all others.

4.3.3 Runtime

Another advantageous property of PPFNet is efficient design and data processing. We found PPFNet to be lightning-fast in inference to extract local features and very quick in data preparation since we consume a very raw representation of the 3D data. The majority of our runtime is spent in the normal computation and this is done only once for the whole fragment. The PPF extraction is carried out within the neighborhoods of only 2048 sample points. Tab. 4.3 shows the average running times of different methods and ours on an NVIDIA TitanX Pascal GPU supported by an Intel Core i7 3.2GhZ 8 core CPU. Such dramatic speed-up in inference is enabled by the parallel-PointNet backend and our simultaneous correspondence estimation during inference for all patches. Currently, to prepare the input for the network, we only use CPU, leaving GPU idle for more work. This part can be easily implemented on the GPU to gain even further speed boosts.

4.3.4 Geometric Registration

Now we use PPFNet in a broader context of transformation estimation, i.e. the application of geometric registration as in [220]. To do so, we send all descriptors into the well established RANSAC based matching pipeline, in which the transformation between fragments is estimated

Tab. 4.3. Average per-patch runtime of different methods.

	input preparation	inference / patch	total
3DMatch	$0.31ms$ on GPU	$2.9ms$ on GPU	$3.21ms$
PPFNet	$2.24ms$ on CPU	$55\mu s$ on GPU	$\mathbf{2.25ms}$

Tab. 4.4. Results of geometric registration on the 3DMatch benchmark after applying RANSAC. *Kitchen* is from 7-scenes [175] and the rest from SUN3D [208].

	Spin Images [99]		SHOT [170]		FPFH [166]		USC [188]		PointNet [152]		CGF [109]		3DMatch [220]		3DMatch-2K		PPFNet	
	recall	prec.	recall	prec.	recall	prec.	recall	prec.	recall	prec.	recall	prec.	recall	prec.	recall	prec.	recall	prec.
Red Kitchen	0.27	0.49	0.21	0.44	0.36	0.52	0.52	0.60	0.76	0.60	0.72	0.54	0.85	0.72	0.80	0.54	**0.90**	0.66
Home 1	0.56	0.14	0.37	0.13	0.56	0.16	0.35	0.16	0.53	0.16	0.69	0.18	**0.78**	0.35	0.79	0.21	0.58	0.15
Home 2	0.35	0.10	0.30	0.11	0.43	0.13	0.47	0.24	0.42	0.13	0.46	0.12	**0.61**	0.29	0.52	0.14	0.57	0.16
Hotel 1	0.37	0.29	0.28	0.29	0.29	0.36	0.53	0.46	0.45	0.38	0.55	0.38	**0.79**	0.72	0.74	0.45	0.75	0.42
Hotel 2	0.33	0.12	0.24	0.11	0.36	0.14	0.20	0.17	0.31	0.18	0.49	0.15	0.59	0.41	0.60	0.22	**0.68**	0.22
Hotel 3	0.32	0.16	0.42	0.12	0.61	0.21	0.38	0.14	0.43	0.11	0.65	0.16	0.58	0.25	0.58	0.14	**0.88**	0.20
Study Room	0.21	0.07	0.14	0.07	0.31	0.11	0.46	0.17	0.48	0.16	0.48	0.16	0.63	0.27	0.57	0.17	**0.68**	0.16
MIT Lab	0.29	0.06	0.22	0.09	0.31	0.09	0.49	0.19	0.43	0.14	0.42	0.10	0.51	0.20	0.42	0.09	**0.62**	0.13
Average	0.34	0.18	0.27	0.17	0.40	0.21	0.43	0.27	0.48	0.23	0.56	0.23	0.67	0.40	0.63	0.24	**0.71**	0.26

by running a maximum of 50,000 RANSAC iterations on the resulted correspondence set. We evaluate the performance by transforming the source cloud to the target by estimated 3D pose and compute the point-to-point error, using the same metric as in [220].

Tab. 4.4 tabulates the results on the real datasets. Overall, PPFNet is again the top performer, while showing higher recall on a majority of the scenes and on the average. It is noteworthy that we always use 2048 patches, while allowing 3DMatch to use its original setting, 5K. Even so, we could get a better recall on more than half of the scenes. When we feed 3DMatch 2048 patches, to be on par with our sampling level, PPFNet dominates performance-wise on most scenes with higher average accuracy.

Fig. 4.8 demonstrates some qualitative fragment registration results, showing that learned features by PPFNet are able to cope with challenging point cloud matching problems under different situations.

4.3.5 Robustness to Point Density

Changes in point density, a.k.a. sparsity, is an important concern for point clouds, as it can change with sensor resolution or distance for 3D scanners. This motivates us to evaluate our algorithm against others in varying sparsity levels. We gradually decrease point density on the evaluation data and record the accuracy. Fig. 4.9 shows the significant advantage of PPFNet, especially under a severe loss of density. Such robustness is achieved due to the PointNet backend and the robust point pair features.

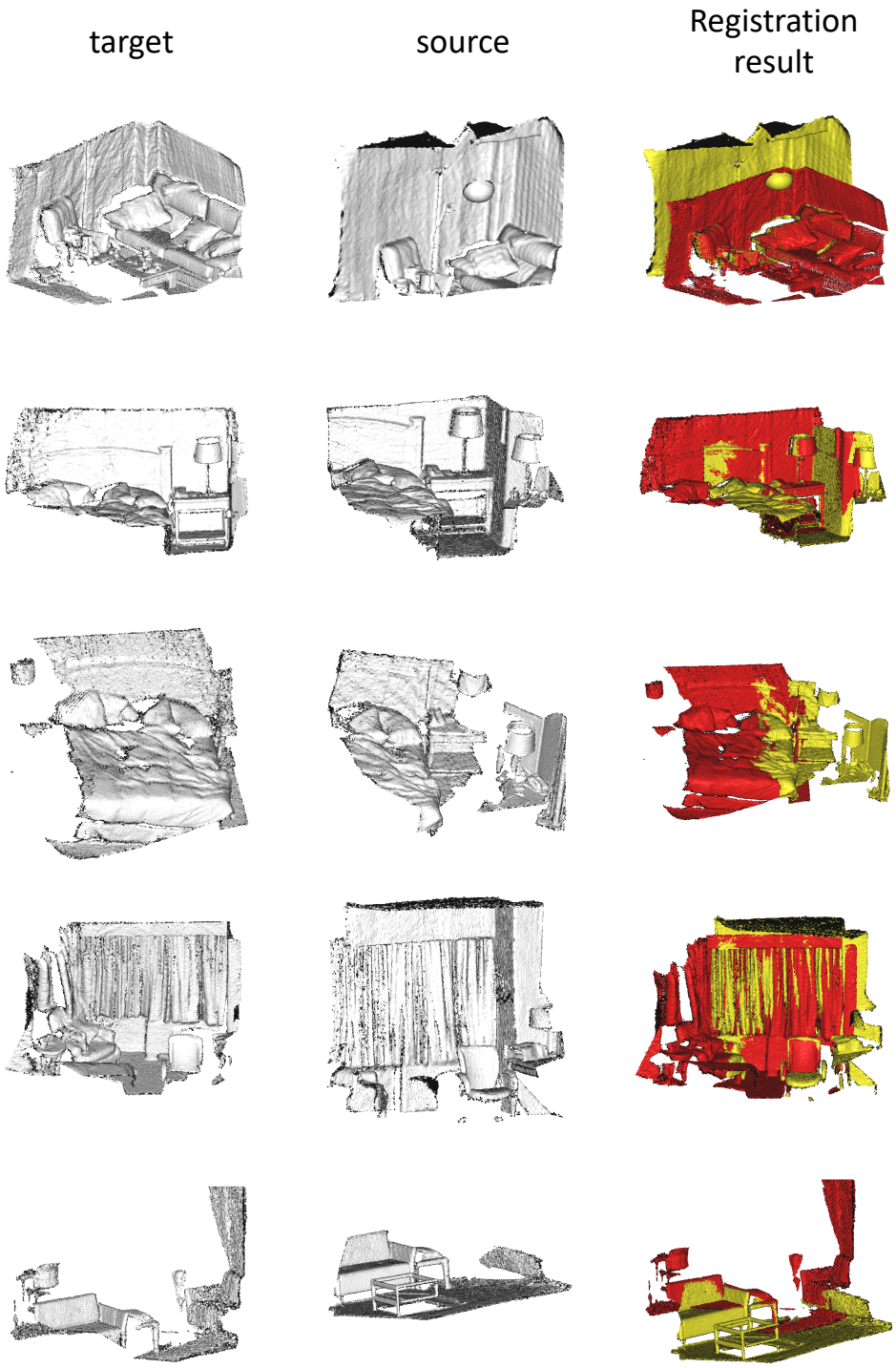


Fig. 4.8. Qualitative registration results of 5 fragment pairs using local features from PPFNet.

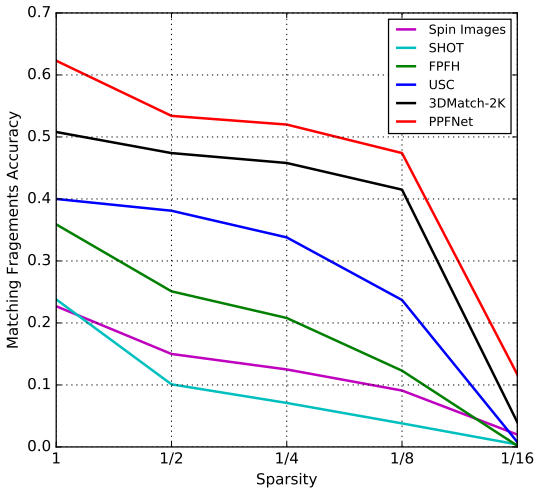


Fig. 4.9. Robustness to point density. Thanks to its careful design, PPFNet clearly yields the highest robustness to change in the sparsity of the input, even when only 6.25% of the input data is used.

4.3.6 Ablation Study

We continue to analyze the effect of those core designs of PPFNet, to see how they end up adding the learning of robust local features.

Impact of N-tuple loss

We train and test our network with 3 different losses: contrastive (pair) [80], triplet [88] and our N-tuple loss on the same dataset with identical network configuration. The pairs and triplets are sampled from the N^2 pairs used in the N-tuple loss. Inter-distance distribution of correspondent pairs and non-correspondent pairs are recorded for the train/validation data respectively.

Empirical results in Fig. 4.10 show that the theoretical advantage of our loss immediately transfers to practice: Features learned by N-tuple are better separable, i.e. non-pairs are more distant in the embedding space and pairs enjoy a lower standard deviation. N-tuples loss repels non-pairs further in comparison to contrastive and triplet losses because of its better knowledge of global correspondence relationships. Our N-tuple loss is general and thus we strongly encourage the application also to other domains such as pose estimation [203].

Impact of the global context

We argue that local features are dependent on the context. A corner belonging to a dining table should not share the similar local features of a picture frame hanging on the wall. A table is generally not supposed to be attached vertically on the wall. To assess the returns obtained from adding a global context, we simply remove the global feature concatenation, keep the rest of the settings unaltered, and re-train and test on the subsets of pairs of fragments. Our results are shown in Tab. 4.5, where injecting global information into local features improves the matching by 18% in training and 7% in the validation set as opposed to our baseline

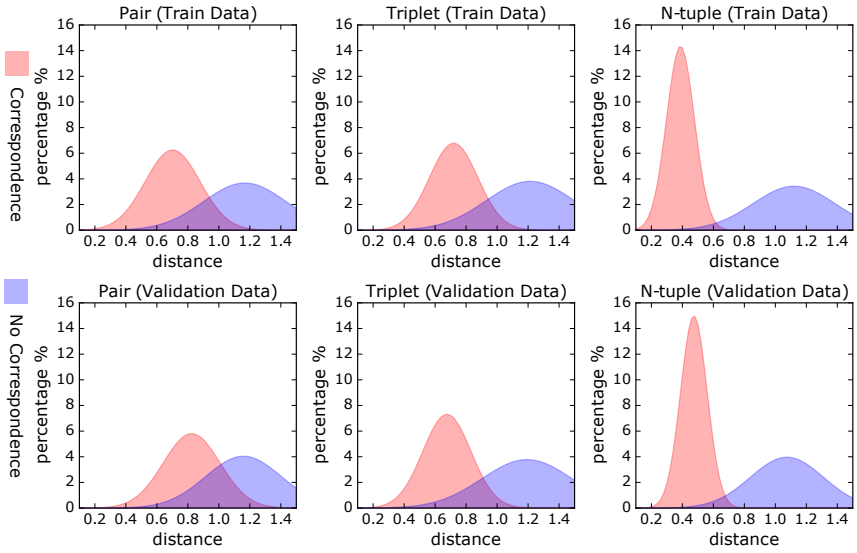


Fig. 4.10. N-tuple Loss (c) lets the PPFNet better separate the matching vs non-matching pairs w.r.t. the traditional contrastive (a) and triplet (b) losses.

Tab. 4.5. Effect of different components in performance: Values depict the number of correct matches found to be 5% inlier ratio.

Method	Train	Validation
Without points and normals	0%	6%
Vanilla PointNet [152]	47%	41%
Without global context	48%	46%
Without PPF	65%	48%
All combined	**67%**	**56%**

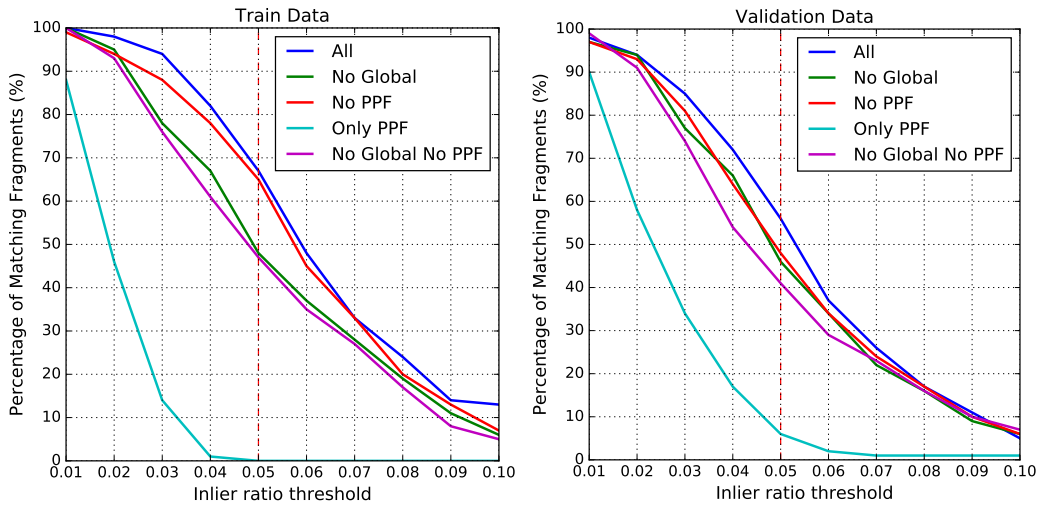
version of Vanilla PointNet [1], which is free of global context and PPFs. Such significance indicates that global features aid discrimination and are valid cues also for local descriptors.

Impact of PPF component

We now run a similar experiment and train two versions of our network, with/without incorporating PPF into the input. The contribution is tabulated in Tab. 4.5. There, a gain of 2% in training and 8% in validation is achieved, justifying that inclusion of PPF increases the discriminative power of the final features.

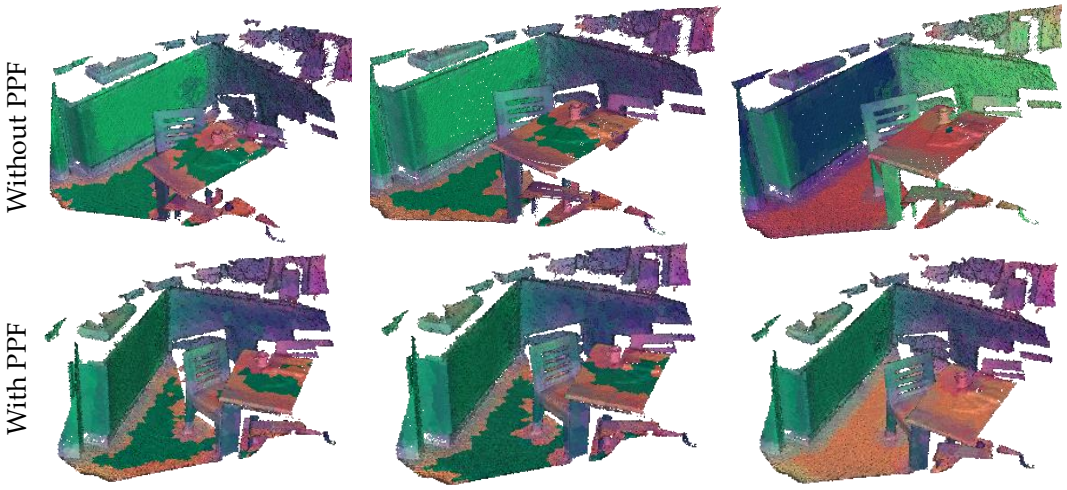
While being a significant jump, this is not the only benefit of adding PPF. Note that our input representation is composed of 33% rotation-invariant and 66% variant representations. This is already advantageous to the state of the art, where rotation handling is completely left to the network to learn from data. We hypothesize that the input guidance of PPF would aid the network to be more tolerant to rigid transformations. To test this, we gradually rotate fragments around the z-axis to 180° with a step size of 30° and then match the fragment to

[1] Note that this doesn't 100% correspond to the original version, as we modified PointNet with task-specific losses for our case.



(a) Training with different inputs (b) Test with different inputs

Fig. 4.11. Assessing different elements of the input on training and validation sets, respectively. Note that combining cues of global information and point pair features help the network to achieve the top results.



(a) Original Fragment **(b)** Rotate 30° around z axis **(c)** Rotate 60° around z axis

Fig. 4.12. Inclusion of PPF makes the network more robust to rotational changes as shown, where the appearance across each row is expected to stay identical, for a fully invariant feature.

the non-rotated one. As we can observe from Tab. 4.6, with PPFs, the feature is more robust to rotation and the ratio in the matching performance of the two networks opens as rotation increases. In accordance, we also show a visualization of the descriptors at Fig. 4.12 under small and large rotations. To assign each descriptor an RGB color, we use PCA projection from high dimensional feature space to 3D color space by learning a linear map [109]. It is qualitatively apparent that PPF can strengthen the robustness towards rotations. All in all, with PPFs we gain both accuracy and robustness to rigid transformation, the best of seemingly contradicting worlds. It is noteworthy that using only PPF introduces full invariance besides the invariance to permutations and renders the task very difficult to learn for our current network, which might be the fact that this representation is more sensitive to false pair

Tab. 4.6. Effect of point pair features in robustness to rotations.

z-rotation	0°	30°	60°	90°	120°	150°	180°
with PPF	100.0%	53.3%	35.0%	20.0%	8.3%	5.0%	0.0%
w/o PPF	100.0%	38.3%	23.3%	11.7%	1.7%	0.0%	0.0%

relationships. However, this problem is solved with our *PPF-FoldNet*, which will be introduced in the next chapter.

A major limitation of PPFNet is the quadratic memory footprint, limiting the number of used patches to 2K on our hardware. This is, for instance, why we cannot outperform 3DMatch on the fragments of *Home-2*. With upcoming GPUs, we expect to reach beyond 5K, the point of saturation.

4.4 Conclusion

We have presented a new 3d feature, *PPFNet*, which is specially tailored for the point cloud input. By generalizing the contrastive loss to N-tuple loss, we manage to fully take advantage of the available correspondence relationships. At the same time, we have shown how to learn a globally aware 3D descriptor by carefully design the training pipeline. Our features learned by PPFNet outperform the state of the arts not only in terms of recall but also speed, and are more capable of dealing with challenging scenarios,as shown in Fig. 4.13. Furthermore, we have shown that by incorporating set-input such as point pair features, our features are more robust to rotation changes, advantageous in developing invariance properties.

Yet, some limitations of PPFNet remain to be tackled. It is not fully rotation-invariant due to the existence of point coordinates and normals. The number of patches used is limited by the hardware resources. More importantly, it relies on the ground truth poses of scene fragment point clouds used as the training data. The availability and quality of those ground truth labels can affect the training of PPFNet a lot. Even when those ground-truth poses are accessible and accurate, the way of deciding pairs and non-pairs simply by distances in the Euclidean space can cause confusion for the network as well. In the next chapter, we will present *PPF-FoldNet*, an unsupervised feature learning framework, and completely free from those issues.

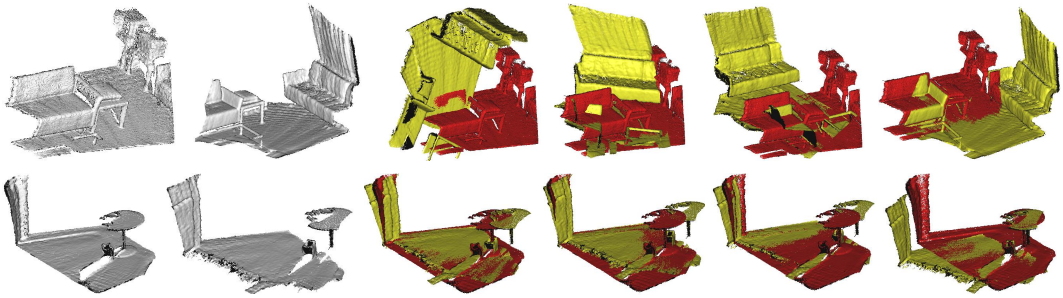


Fig. 4.13. Visualization of estimated transformations. Thanks to its robustness and understanding of global information, PPFNet can operate under challenging scenarios with confusing, repetitive structures as well as mostly planar scenes with less variation in geometry.

5 Learning Fully Rotation-invariant Local Features

With PPFNet, we explored the possibility of learning local features on point clouds, and we show that it makes feature extraction faster and the light-weighted operations make it feasible to process all the local patches at the same. With the richer global pairwise relationships, our learned features are quite powerful for the task of matching and pairwise registration. Yet it still requires the ground truth poses of point clouds to decide the pair labels, which can be erroneous. Also, while being more rotation-robust, it does not achieve full rotation-invariance.

In this chapter, we continue to present our second method on learning 3D local features from point clouds, termed as *PPF-FoldNet*. It is an unsupervised framework for learning local features, which means it does not need any ground truth pair/non-pair relationships between local patches to supervise the training. In the meanwhile, it benefits from the point pair features as the encoding, and achieves fully rotation-invariance. We show that it is an efficient yet competitive feature learning framework with extensive evaluations.

5.1 Introduction

By reviewing recent deep learning-based 3D local features, we could notice some common issues:

1. trained in a supervised way and requiring an abundant amount of labels in the form of pairs pairs [220], triplets [109] or N-tuple as introduced in the previous chapter;

2. sensitive to rotations [52, 220];

3. involving significant hand-crafted input preparation [109] or voxelization [220];

4. unsatisfactory performance [109, 152].

Deep learning is a data-driven technique. More training data generally lead to improved performance [53]. However, annotating the increased amount of data to get the ground truth labels requires excessive labors, posing another big challenge. As a result, we would like to argue for the fact that unsupervised learning is the solution for scalability. Motivated by this, an elegant architecture is mapped out in this chapter to tackle all the aforementioned problems, which we call *PPF-FoldNet*: an unsupervised, high-accuracy, 6DoF transformation invariant, sparse and fast 3D local feature learning network. PPF-FoldNet operates directly on

point sets, taking into account the point sparsity and permutation invariant set property, deals well with density variations, while significantly outperforming its rotation-variant counterparts even in the standard benchmarks.

Our network establishes theoretical rotation invariance inspired by the use a point pair feature (PPF) [13, 16, 52] encoding of the local 3D geometry into patches. Different from PPFNet [52], we do not incorporate the original points or normals into the encoding. The collection of these 4D PPFs are then sent to a FoldingNet [212]-like end to end auto-encoder (AE), trained to auto-reconstruct the PPFs, using a set distance. Our encoder is again PointNet-based and for decoding, we propose a similar folding scheme as in FoldingNet [212], where a low dimensional 2D grid lattice is folded onto a 4D PPF space and monitor the network evolution by a novel lossless visualization of the PPF space. Our overall architecture enjoys permutation invariance and fully utilizes the sparsity. Training our AE is far easier than training, say 3DMatch [220], as we do not need to sample pairs or triplets from a pre-annotated large dataset and we enjoy linear time complexity in the number of patches.

Extensive evaluations demonstrate that, PPF-FoldNet outperforms the state-of-the-art across the standard benchmarks, in which severe rotations are avoided. When arbitrary rotations are introduced in the input, our descriptors outperform related approaches by a large margin including even the best competitor, Khoury et. al.'s CGF [109], which also achieves rotation invariance. Moreover, we report better performance as the input sparsifies, as well as good generalization properties. Our qualitative evaluations will uncover how our network operates and give valuable interpretations. In a nutshell, our contributions can be summarized as:

- An auto-encoder, that unifies a PointNet encoding with a FoldingNet decoder;
- Use of well established 4D PPFs in this modified auto-encoder to learn rotation-invariant 3D local features without supervision;
- A novel look to the invariance of point pair features and derived from it, a new way of visualizing PPFs and monitoring the network progress.

5.2 PPF-FoldNet

PPF-FoldNet is based on the idea of auto-encoding a rotation-invariant but powerful representation of the point set (PPFs), such that the learned low dimensional embedding could be truly invariant. This is different than training the network with many possible rotations of the same input and forcing the output to be a canonical reconstruction. The latter would both be approximate and much harder to learn. Input to our network are local patches and unlike PPFNet, they are auto-encoded individually. The global context is lost in this case, however, processing and extracting features for each patch is becoming much easier. The latent low dimensional vector of the auto-encoder, *codeword*, is used as the local descriptor attributed to the point around which the patch is extracted.

Local Patch Representation

Same as in the previous chapter, our original input point cloud is a set of oriented points $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^6\}$, meaning that each point is decorated with a local normal (e.g. tangent space) $\mathbf{n} \in \mathbb{R}^3$: $\mathbf{x} = \{\mathbf{p}, \mathbf{n}\} \in \mathbb{R}^6$. A local patch is a subset of the input $\mathbf{\Omega}_{\mathbf{x}_r} \subset \mathbf{X}$ center around a reference point $\mathbf{x}_r$. We then encode this patch as a collection of pair features, computed between a central reference and all the other points:

$$\mathbf{F}_{\mathbf{\Omega}} = \{\, \mathbf{f}(\mathbf{x}_r, \mathbf{x}_1) \cdots \mathbf{f}(\mathbf{x}_r, \mathbf{x}_i) \cdots \mathbf{f}(\mathbf{x}_r, \mathbf{x}_N) \,\} \in \mathbb{R}^{4 \times N-1},\ i \neq r \tag{5.1}$$

The features between any pair (a.k.a. point pair features) are then defined to be a map $\mathbf{f} : \mathbb{R}^{12} \rightarrow \mathbb{R}^4$ sending two oriented points to three angles and the pair distance:

$$\mathbf{f} : (\mathbf{x}_r^T, \mathbf{x}_i^T)^T \rightarrow (\angle(\mathbf{n}_r, \mathbf{d}), \angle(\mathbf{n}_i, \mathbf{d}), \angle(\mathbf{n}_r, \mathbf{n}_i), \|\mathbf{d}\|_2)^T \tag{5.2}$$

$\mathbf{d} = \mathbf{p}_r - \mathbf{p}_i$. An angle computation for non-normalized vectors is given in [16]. Such encoding of local geometry resembles PPFNet's, but differs in the fact that we ignore the points and normals as they are orientation and local reference frame dependent. We instead use pure point pair features, thereby avoiding a canonical frame computation. Note that the dimensionality of this feature is still irreducible without data loss.

Proposition. *PPF representation* $\mathbf{f}$ *around* $\mathbf{x}_r$ *explains the original oriented point pair up to a rotation and reflection about the normal of the reference point.*

Proof. Let us consider two oriented points $\mathbf{x}_1$ and $\mathbf{x}_2$. We can always write the components of the associated point pair feature $\mathbf{f}(\mathbf{x}_1, \mathbf{x}_2)$ as follows:

$$\mathbf{n}_1^T \mathbf{n}_2 = f_1 \qquad \mathbf{n}_1^T \mathbf{d}_n = f_2 \qquad \mathbf{n}_2^T \mathbf{d}_n = f_3 \tag{5.3}$$

where $\mathbf{d}_n = \mathbf{d}/\|\mathbf{d}\|$. We now try to recover the original pair given its features. First, it is possible to write:

$$\begin{bmatrix} \mathbf{n}_1^T \\ \mathbf{n}_2^T \\ \mathbf{d}_n^T \end{bmatrix} \begin{bmatrix} \mathbf{n}_1 & \mathbf{n}_2 & \mathbf{d}_n \end{bmatrix} = \begin{bmatrix} 1 & f_1 & f_2 \\ f_1 & 1 & f_3 \\ f_2 & f_3 & 1 \end{bmatrix} \tag{5.4}$$

given that all vectors are of unit length. In matrix notation, Eq. 5.4 can be written as $\mathbf{A}^T\mathbf{A} = \mathbf{K}$. Then, by singular value decomposition, $\mathbf{K} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ and thus $\mathbf{A} = \mathbf{U}\mathbf{S}^{1/2}\mathbf{V}^T$. Note that, any orthogonal matrix (rotation and reflection) $\mathbf{R}$ can now be applied to $\mathbf{A}$ without changing the outcome: $(\mathbf{R}\mathbf{A})^T\mathbf{R}\mathbf{A} = \mathbf{A}^T\mathbf{R}^T\mathbf{R}\mathbf{A} = \mathbf{A}^T\mathbf{A} = \mathbf{K}$. Hence, such decomposition is up to finite-dimensional linear isometries: rotations and reflections. Since we know that the local patch is centered at the reference point $\mathbf{p}_r = \mathbf{0}$, we are free to choose an $\mathbf{R}$ such that the

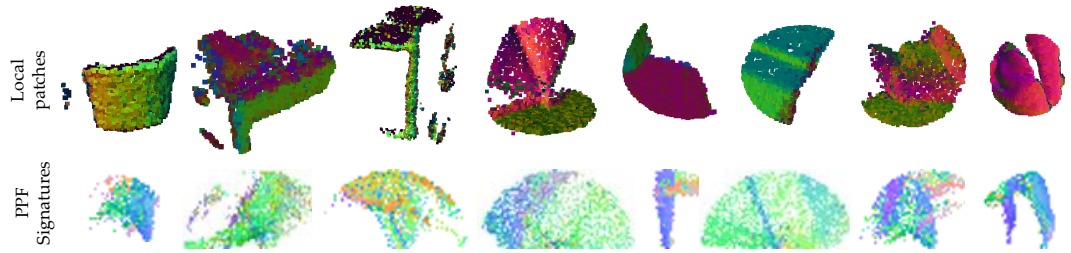


Fig. 5.1. Visualisation of some local patches and their correspondent PPF signatures.

normal vector of $\mathbf{p}_r$ ($\mathbf{n}_r$) is aligned onto one of the canonical axes, say $+\mathbf{z} = [0, 0, 1]^T$ (free to choose):

$$\mathbf{R} = \mathbf{I} + [\mathbf{v}]_x + [\mathbf{v}_x]^2 \frac{1 - n_r^z}{\|v\|} \tag{5.5}$$

where $\mathbf{v} = \mathbf{n}_r \times \mathbf{z}$, n_r^z is the z component of $\mathbf{n}_r$ and $\mathbf{I}$ is identity. $[\cdot]_x$ denotes skew symmetric cross product matrix. Because now $\mathbf{R}\mathbf{n}_r = \mathbf{z}$, any rotation θ and reflection ϕ about $\mathbf{z}$ would result in the same vector $\mathbf{z} = \mathbf{R}_z(\theta, \phi)\mathbf{z}$, $\forall \theta, \phi \in \mathbb{R}$. Any paired point can then be found in the canonical frame, up to two parameters as $\mathbf{p}_r \leftarrow \|\mathbf{d}\|\mathbf{R}_z(\theta.\phi)\mathbf{R}\mathbf{d}_n$, $\mathbf{n}_r \leftarrow \mathbf{R}_z(\theta, \phi)\mathbf{R}\mathbf{n}_r$. □ □

In the case where reflections are ignored (as they are unlikely to happen in 3D world), this leaves a single degree of freedom, rotation angle around the normal. Also note, once again that for the given local representation, the reference point $\mathbf{p}_r$ is common to all the point pairs.

Visualizing PPFs

PPFs live in a 4D space and thus it is not trivial to visualize them. While simple solutions such as PCA would work, we prefer a more geometrically meaningful and simpler solution. Proposition. 5.2 allows us to compute a signature of a set of point pairs by orienting the vectors $(\mathbf{n}_1, \mathbf{n}_2, \mathbf{d})$ individually for all points in order to align the difference vectors $\{\mathbf{d}_i\}$ with the $x - z$ plane by choosing an appropriate $\mathbf{R}_z(\theta.\phi)$. Such a transformation would not alter the features as shown. This way, the paired points can be transformed onto a common plane (image), where the location is determined by difference vector, in polar coordinates. The normal of the second point would not lie in this plane, but can be encoded as colors in that image. This way, it is possible to obtain a 2D visual, without any data loss, i.e. all components of the vector contribute to the visualization. In Fig. 5.1 we provide a variety of local patch and PPF visualizations from the datasets of concern.

Encoder and Decoder

PPF-FoldNet employs a PointNet-like encoder with skip-links and a FoldingNet-like decoding scheme. It is designed to operate on 4D-PPFs, as summarized in Fig. 5.2.

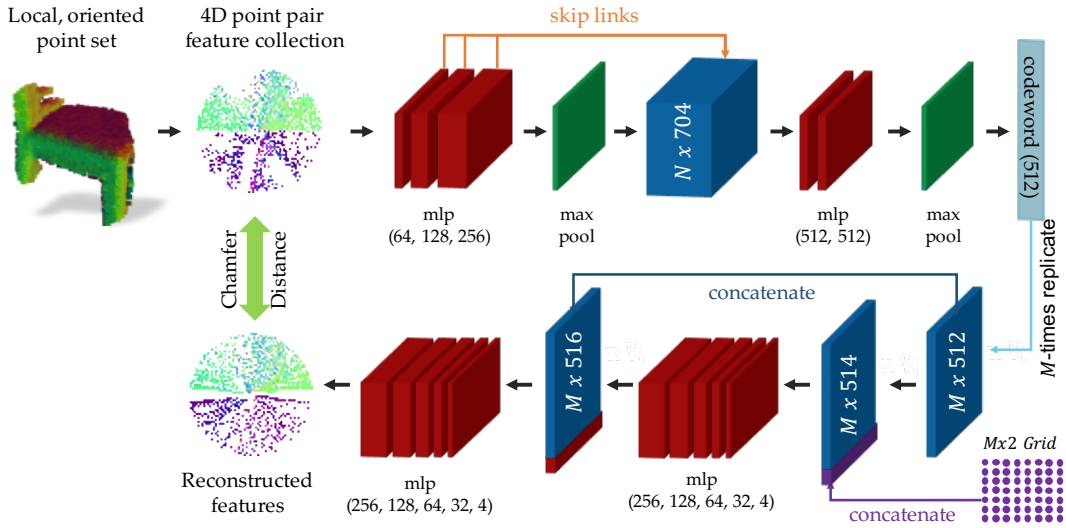


Fig. 5.2. PPF-FoldNet: The point pair feature folding network. The point cloud local patches are first converted into PPF representations, and then sent into the encoder to get compressed codewords. The decoder tries to reconstruct full PPFs from these codewords by folding. This forces the codewords to keep the most critical and discriminative information. The learned codewords are proven to be robust and effective as we will show across extensive evaluations.

Encoder

The input to our network, and thus to the encoder, is $\mathbf{F}_{\Omega}$, a local PPF representation, as described in the previous section. A three-layer, point-wise MLP (Multi Layer Perceptron) follows the input layer and subsequently, a max-pooling is performed to aggregate the individual features to a global one, similar to PointNet [152]. The low-level features are then concatenated with this global feature using skip-links. This results in a more powerful representation. Another two-layer MLP finally redirects these features to a final encoding, the codeword, whose dimension is 512.

Proposition. *The encoder structure of PPF-FoldNet is permutation invariant.*

Sketch of the proof. The encoder is composed of per-data-point functions (MLP), the RELU layers and max-pooling, all of which either do not affect the point order or are individually shown to be permutation invariant [152, 212]. Moreover, it is shown that the composition of functions is also invariant [212] and so is our encoder. We encourage the reader to the references for further details. □ □

In the end, altering the order of the PPF set will not affect the learned representation.

Decoder

Our decoder tries to reconstruct the whole set of point PPFs using a single codeword, which on the contrary, also forces the codeword to be informative and distill the most distinctive information from the high-dimensional input space. However, inspired by FoldingNet [212], instead of trying to upsample or interpolate point sets, the decoder will try to deform a

low-dimensional grid structure guided by the codeword. Each grid point gets concatenated to a replica of the codeword, resulting in an $M \times 514$ vector as input to what is referred as *folding operation* [212]. Folding can be a highly non-linear operation and thus is performed by two consecutive MLPs: the first folding results in a deformed grid, which is appended once again to the codewords and propagates through the second MLP, reconstructing the input PPFs. Moreover, compared to FoldingNet [212], we try to reconstruct a higher dimensional set, 4D vs 3D (2D manifold), we are better off using a deeper MLP - 5-layer as opposed to 3-layer of [212].

Other than simplifying and strengthening the decoding, the folding is also beneficial in making the network interpretable. For instance, it is possible to monitor the grid during subsequent iterations and grasp how the network evolves. To do so, in the following sections, we will trace the PPF sets by visualizing them as described in the previous section.

Chamfer Distance

Note that as the grid size M is not necessarily the same as the size of the input, N and the correspondences in 4D PPF space are lost when it comes to evaluating the loss. This requires a distance computation between two unequal cardinality point pair feature sets, which we measure via the well known Chamfer metric:

$$d(\mathbf{F}, \hat{\mathbf{F}}) = \max\left\{ \frac{1}{|\mathbf{F}|} \sum_{\mathbf{f} \in \mathbf{F}} \min_{\hat{\mathbf{f}} \in \hat{\mathbf{F}}} \|\mathbf{f} - \hat{\mathbf{f}}\|_2, \ \frac{1}{|\hat{\mathbf{F}}|} \sum_{\mathbf{f} \in \hat{\mathbf{F}}} \min_{\mathbf{f} \in \mathbf{F}} \|\mathbf{f} - \hat{\mathbf{f}}\|_2 \right\} \tag{5.6}$$

where $\hat{}$ operator refers to the reconstructed (estimated) set.

Implementation

Local patches

Our local patches are prepared the same way as in the experiments for PPFNet. To make a fair comparison with other methods reported in the previous chapter, we sample 2048 local patches from each fragment, but also provide an extended version that uses 5K in our evaluation since we are not memory bound like PPFNet. The data preparation stage ends with the PPFs calculated for the assembled local patches.

Network

PPF-FoldNet uses Tensorflow framework [1]. The initial values of all variables are initialized randomly by Xavier's algorithm. Global loss is minimized with an ADAM optimizer [111]. The learning rate starts at 0.001 and exponentially decays after every 10 epochs, truncated at 0.0001. We use batches of size 32.

5.3 Evaluation

In this section, we conduct numerous experiments to quantitatively and qualitatively evaluation on our PPF-FoldNet, showing its superior matching performance, even when severe rotations are presented.

5.3.1 Experiment Setup

Baselines

The methods that we evaluate against consist of three handcrafted features (Spin Images [99], SHOT [170], FPFH [166]) and four state-of-the-art deep learning based methods, including 3DMatch [220], CGF [109], FoldingNet [212], and our previous work PPFNet.

Dataset ane metric

We use the 3DMatch Benchmark Dataset [220] and matching performance metric for evaluation, the same as in the previous chapter.

5.3.2 Matching Performance

We first compare the matching performance of our features against the well-accepted works from the literature, on the 3DMatch benchmark with $\tau_1 = 10cm$ and $\tau_2 = 5\%$. Tab. 5.1 tabulates the findings. It is visible that, overall, our PPF-FoldNet could match far more fragment pairs in comparison to the other methods, except on scenes *Kitchen* and *Home*, where PPFNet and 3DMatch get a higher recall respectively. In all the other cases, PPF-FoldNet outperforms the state of the art by a large margin $> 9\%$ on the average. PPF-FoldNet has a recall of 68.04% when using 2K sample points (same as PPFNet), while PPFNet remains on 62.32%. Moreover, because PPF-FoldNet has no memory bottleneck, it can achieve an additional 3% over 2K version, when 5K points are used. Interestingly, FPFH is also constructed from a type of PPF features [166], but in a form of manual histogram summarization. Compared to FPFH, PPF-FoldNet has a 32.15% and 35.93% higher recall using 2K and 5K points respectively. It demonstrates the unprecedented strength of our advanced method to compress the PPFs. In order to reconstruct PPFs in the decoder at best, the network forces the bottleneck codeword to be compact as well as distilling the most critical and distinctive information in PPFs.

To illustrate that parameters in the evaluation metric are not tuned for our own good, we also repeat the experiments with different τ_1 and τ_2 values. To results are shown in Fig. 5.3a and Fig. 5.3b. In Fig. 5.3a, τ_1 is fixed at $10cm$, τ_2 increases gradually from 1% to 20%. When τ_2 is above 4%, PPF-FoldNet always has a higher recall than the other methods. Below 4%, some other methods could get a higher recall, but it is too strict for most of the registration algorithms anyways. It is further noteworthy that when τ_2 is set to 20%, the point where PPF-FoldNet still gets a recall above 20%, the performance of the other methods drop below 5%. This justifies that PPF-FoldNet is capable of generating much more sets of matching points with a high inlier ratio r_{in}. This brings a tremendous benefit for the registration algorithms. In

Tab. 5.1. Our results on the standard 3DMatch benchmark. *Red Kitchen* data is from 7-scenes [175] and the rest imported from SUN3D [208].

	Spin Image [99]	SHOT [170]	FPFH [166]	3DMatch [220]	CGF [109]	PPFNet [52]	FoldNet [212]	Ours	Ours-5K
Kitchen	0.1937	0.1779	0.3063	0.5751	0.4605	**0.8972**	0.5949	0.7352	0.7866
Home 1	0.3974	0.3718	0.5833	0.7372	0.6154	0.5577	0.7179	0.7564	**0.7628**
Home 2	0.3654	0.3365	0.4663	**0.7067**	0.5625	0.5913	0.6058	0.6250	0.6154
Hotel 1	0.1814	0.2080	0.2611	0.5708	0.4469	0.5796	0.6549	0.6593	**0.6814**
Hotel 2	0.2019	0.2212	0.3269	0.4423	0.3846	0.5769	0.4231	0.6058	**0.7115**
Hotel 3	0.3148	0.3889	0.5000	0.6296	0.5926	0.6111	0.6111	0.8889	**0.9444**
Study	0.0548	0.0719	0.1541	0.5616	0.4075	0.5342	**0.7123**	0.5753	0.6199
MIT Lab	0.1039	0.1299	0.2727	0.5455	0.3506	**0.6364**	0.5844	0.5974	0.6234
Average	0.2267	0.2382	0.3589	0.5961	0.4776	0.6231	0.6130	0.6804	**0.7182**

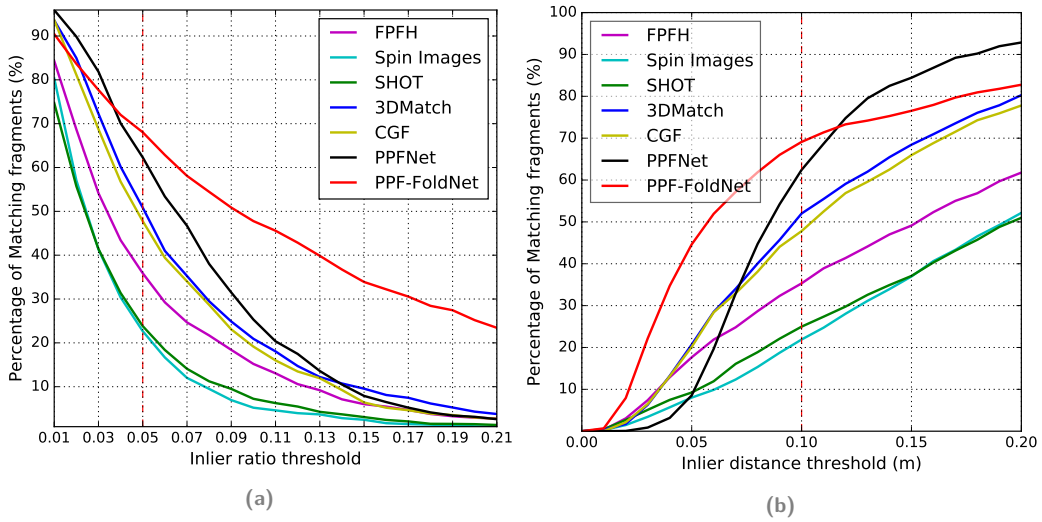


Fig. 5.3. Evaluations on 3DMatch benchmark: **(a)** Results of different methods under varying inlier ratio threshold **(b)** Results of different methods under varying point distance threshold.

Fig. 5.3b, τ_2 is fixed at 5%, τ_1 increases gradually from $0cm$ to $20cm$. When τ_1 is smaller than $12cm$, PPF-FoldNet constantly generates higher recall. This finding indicates that PPF-FoldNet matches more point pairs with a small distance error in the Euclidean space, which could efficiently decrease the rigid transformation estimation errors.

5.3.3 Matching Performance under Severe Rotations

To demonstrate the outstanding rotation-invariance property of PPF-FoldNet, we take random fragments from the evaluation set, gradually rotate them around z-Axis from $60°$ to $360°$ at a step size of $60°$. The matching results are shown in Fig. 5.4. As expected, both PPFNet and 3DMatch perform poorly as they operate on rotation-variant input representations. Handcrafted features or CGF also demonstrate robustness to rotations with a reasonable performance thanks to the reliance on the local reference frame. However, PPF-FoldNet stands out to be the best with much higher recalls and does not require computation of local reference frames.

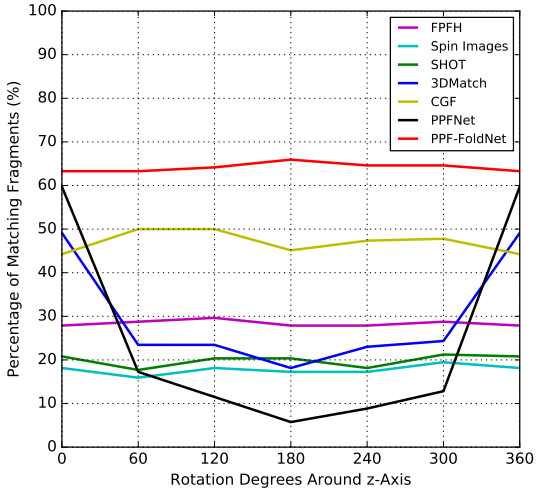


Fig. 5.4. Evaluations against rotations around the z-axis

Tab. 5.2. Our results on the rotated 3DMatch benchmark. *Red Kitchen* data is from 7-scenes [175] and the rest imported from SUN3D [208].

	Spin Image [99]	SHOT [170]	FPFH [166]	3DMatch [220]	CGF [109]	PPFNet [52]	FoldNet [212]	Ours	Ours-5K
Kitchen	0.1779	0.1779	0.2905	0.0040	0.4466	0.0020	0.0178	0.7352	**0.7885**
Home 1	0.4487	0.3526	0.5897	0.0128	0.6667	0.0000	0.0321	0.7692	**0.7821**
Home 2	0.3413	0.3365	0.4712	0.0337	0.5288	0.0144	0.0337	0.6202	**0.6442**
Hotel 1	0.1814	0.2168	0.3009	0.0044	0.4425	0.0044	0.0133	0.6637	**0.6770**
Hotel 2	0.1731	0.2404	0.2981	0.0000	0.4423	0.0000	0.0096	0.6058	**0.6923**
Hotel 3	0.3148	0.3333	0.5185	0.0096	0.6296	0.0000	0.0370	0.9259	**0.9630**
Study	0.0582	0.0822	0.1575	0.0000	0.4178	0.0000	0.0171	0.5616	**0.6267**
MIT Lab	0.1169	0.1299	0.2857	0.0260	0.4156	0.0000	0.0260	0.6104	**0.6753**
Average	0.2265	0.2337	0.3640	0.0113	0.4987	0.0026	0.0233	0.6865	**0.7311**

To further quantitatively evaluate how those methods perform under situations with severe rotations, we rotate all the fragments in the 3DMatch benchmark with randomly sampled axes and angles over the whole rotation space, and introduced a new benchmark – *Rotated 3DMatch Benchmark*. The same evaluation is once again conducted on this new benchmark. Keeping the accuracy evaluations identical, our results are shown in Table 5.2. 3DMatch and PPFNet completely failed on this new benchmark because of the variables brought in by large rotations. And again, PPF-FoldNet, surpasses all the other methods, achieving the best results in all the scenes, predominates the runner-up CGF by a large margin of 18.78% and 23.24% when using 2K and 5K points respectively.

5.3.4 Runtime

We run our algorithm on a machine loaded with NVIDIA TitanX Pascal GPU and an Intel Core i7 3.2GHz 8 core CPU. As shown in Tab. 5.3, computing features of an entire fragment takes about 4 seconds, whereas FPFH [166] is almost an order of magnitude slower.

Tab. 5.3. Runtime comparison (reported in seconds) for 2048 local patches.

FPFH [12]	PPF-FoldNet Preparation	PPF-FoldNet Inference	**PPF-FoldNet Total**
31.678	2.616	1.353	**3.969**

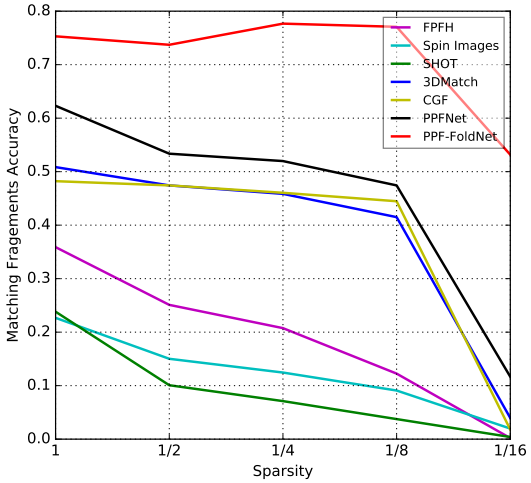


Fig. 5.5. Evaluating robustness again point density.

5.3.5 Robustness to Point Density

Thanks to the sparse representation of our input, PPF-FoldNet is also robust to the changes of point cloud density and noise. Fig.5.5 shows the performance of different methods when we gradually drop the points in the fragment from 100% to only 6.25%. We can see that PPF-FoldNet is least affected by the decrease of point cloud density. And especially when only 6.25% points are left in the fragments, the recall for PPF-FoldNet is still above 50% while PPFNet remains around 12% and the other methods almost fail. The result of PPFNet and PPF-FoldNet together demonstrate that PPF representation brings more robustness to point densities, which is a common problem existing in many point cloud representations.

5.3.6 PPF Construction Variants

We now study 3 identical networks, trained for 3 different invariant PPF formulations: Ours, PPFH (the PPF used in FPFH [166]) and Bobkov1 et. al. [20]. The latter has an added component of *occupancy ratio* based on grid space. We use a subset of the 3DMatch benchmark to train all networks for a given number of iterations to make a fair comparison and test them on the rotated fragments. Tab. 5.4 presents our findings, where all features perform similarly. Thus, we do not claim the superiority of the employed PPF representation, but stress that it is simple, easy to compute, intuitive, and easy to visualize. Due to the voxelization, *Bobkov1* is significantly slower than the others, and thanks to the lack of the local reference frame, our PPF is faster than *PPFH*'s. Using stronger pair primitives would favor PPF-FoldNet because our network design is agnostic to the PPF construction.

Tab. 5.4. Comparison of different PPF representations. Because of the close recall, we conclude that the PPF extraction methods can be selected depending on application and preference.

	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Average
PPFH	0.534	0.622	0.486	0.341	0.346	0.574	0.233	0.351	0.436
Bobkov1	0.514	0.635	0.510	0.403	0.433	0.611	0.281	0.481	0.483
Our-PPF	0.506	0.635	0.495	0.350	0.385	0.667	0.267	0.403	0.463

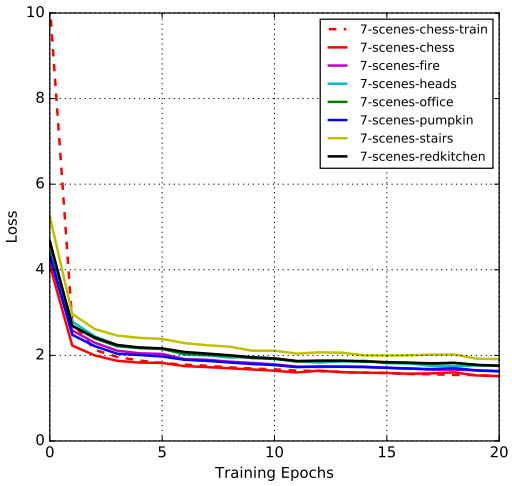


Fig. 5.6. Generalizability Test of PPF-FoldNet. Here it is only trained from the *Chess* scene, and tested on the rest of the 7 scenes.

5.3.7 Generalization Ability

State-of-the-art methods for learning 3D features rely heavily on the availability of extensively annotated data - such as ground truth matches between the pairs. In 3DMatch, CGF and PPFNet, contrastive, triplet and N-tuple losses are used respectively. Such supervision prevents these methods from immediately extending to different datasets without fine-tuning. It might occasionally be prohibitive to even obtain labeled data for new datasets.

This is different for PPF-FoldNet, where we learn a completely unsupervised representation. Thanks to the novel auto-encoder, we can operate on any available dataset without requiring auxiliary label information. Therefore, we are motivated to believe that PPF-FoldNet would generalize better to unseen data, even with a small subset of unlabeled data being available.

To test the aforementioned hypothesis, we propose an experiment, where PPF-FoldNet is trained on a small portion of scenes and tested on multiple different ones. We divide the *Chess* scene from 7-scenes dataset into train and test splits, train PPF-FoldNet from scratch, and measure the loss on the test data including *Chess* and other six scenes as well. Note that the remaining six scenes do not contribute to training. For all datasets, we plot the loss curves among iterations in Fig. 5.6. The dashed line stands for the average loss of training data for each epoch, and the other curves depict the average loss of test data from each scene

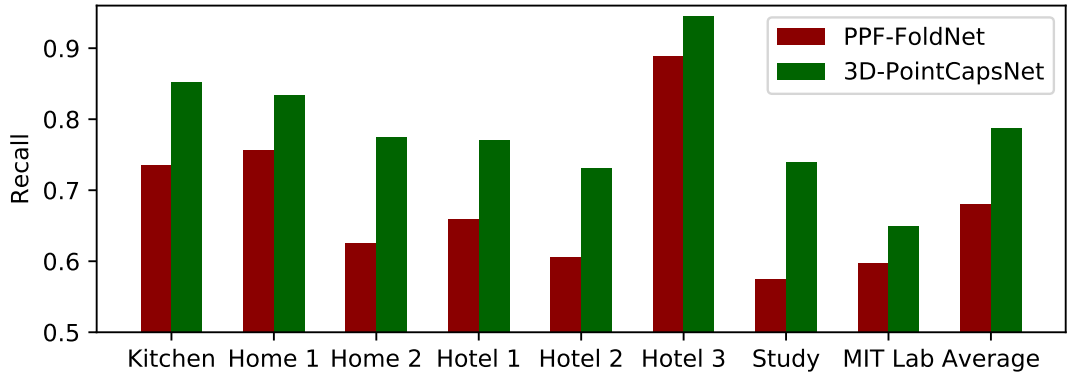


Fig. 5.7. Matching performance comparison between 3D-PointCapsNet and our original PPF-FoldNet. $2K$ points are used. It shows that with a more sophisticated encoder such as 3D-PointCapsNet, the learned features can be further improved. Our general idea of learning 3D local features with autoencders can be further applied with the progress of deep networks.

respectively. As the training proceeds, the losses for all 7 datasets decrease following a similar trend. Achieving such residual values validates that PPF-FoldNet can generalize to unseen input.

5.3.8 3D Point Capsule Network

PPF-FoldNet builds upon a general idea that an autoencoder could be utilized to extract compact yet distinctive features from PPF-encoded point clouds. To show that, we replace our network with *3D Point Capsule Network(3D-PointCapsNet)* [221] for the same task. The main differences the two networks lie on the encoder. While our PPF-FoldNet adopts a simple PointNet-like network, 3D-PointCapsNet uses *capsules* to group the input and different part parts of the intermediate feature is captured by different *latent capsules*. Fig. 5.7 shows that with this sophisticated design, the features learned by 3D-PointCapsNet can further improve the matching performance. More details about this network can be found in Appendix A.

5.3.9 Qualitative Results

Here we present more qualitative results to visually prove the superior performance of our PPF-FoldNet.

Monitoring network evolution

As our network is interpretable, it is tempting to qualitatively analyze the progress of the network. To do that we record the PPF reconstruction output at discrete time steps and visualize the PPFs as explained in § 5.2. Fig 5.8 shows such a visualization for different local patches. Formerly, thanks to the representation power, our network achieves high fidelity recovery of PPFs. Note that even though the network starts from a random initialization, it can quickly recover a desired point pair feature set, even with a small number of iterations. Next, for similar local patches (top and bottom rows), the reconstructions are similar, while for different ones, different.

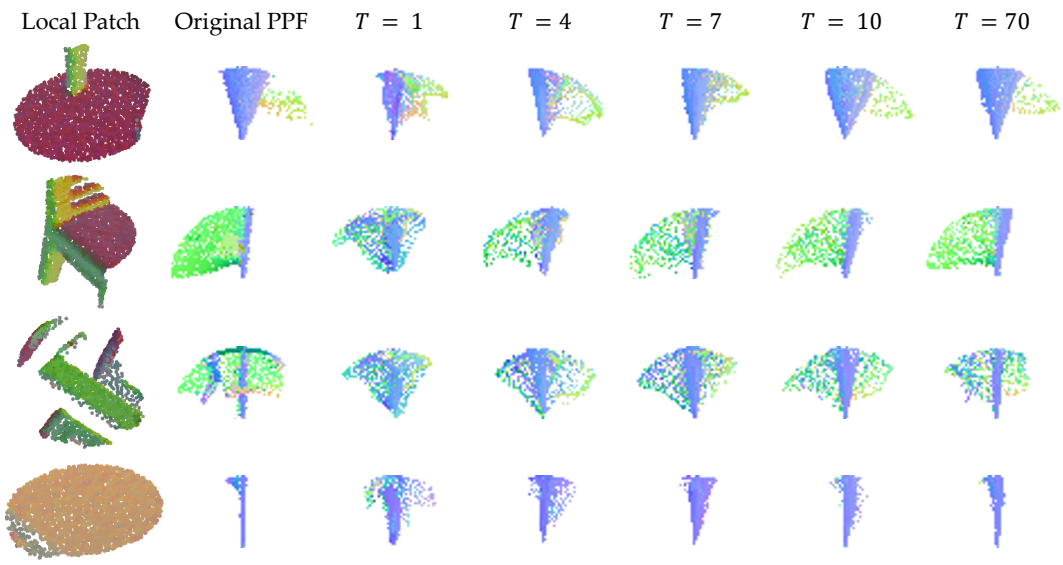


Fig. 5.8. Visualizing signatures of reconstructed PPFs. As the training converges, the reconstructed PPF signatures become closer to the original signatures. Our network reveals the underlying structure of the PPF space.

Visualizing the latent space

We now attempt to visualize the learned latent space and assess whether the embedding is semantically meaningful. To do that we compute a set of codewords and the associated PPF signatures. We then run the Barnes Hut T-SNE algorithm [131, 194] on the extracted codewords and form a two-dimensional embedding space, as shown in Fig. 5.9. At each 2D location we paint the PPF signature and thereby illustrate the distribution of PPFs along the manifold. We also plot the original patches which generated the codewords and their corresponding signatures as cutouts. Presented in Fig. 5.9, whenever the patches are geometrically and semantically close, the computed descriptors are close, and whenever the patches have less physical similarity, they get embedded into different parts of the space. This provides insight on the good performance and meaningful-ness in the relationships our network could learn.

In a further experiment, we extract a feature at each location of the point cloud. Then, we reduce the dimension of the latent space to three via TSNE [131], and colorize each point by the reduced feature vector. Qualitatively justifying the repeatability of our descriptors, the outcome is shown in Fig. 5.10. Notice that, descriptors extracted by the proposed approach lead to similar colors in matching regions among the different fragments.

Visualizing the matching result

From the quantitative results, PPF-FoldNet is expected to have better and more correct feature matches, especially when arbitrary rigid transformations are applied. To show this visually, we run different methods and ours across several fragments undergoing varying rotations. In Fig. 5.11 and 5.12 we show the matching regions, over uniformly sampled [15] keypoints on these fragments. It is clear that our algorithm performs the best among all in terms of discovering the most correct correspondences.

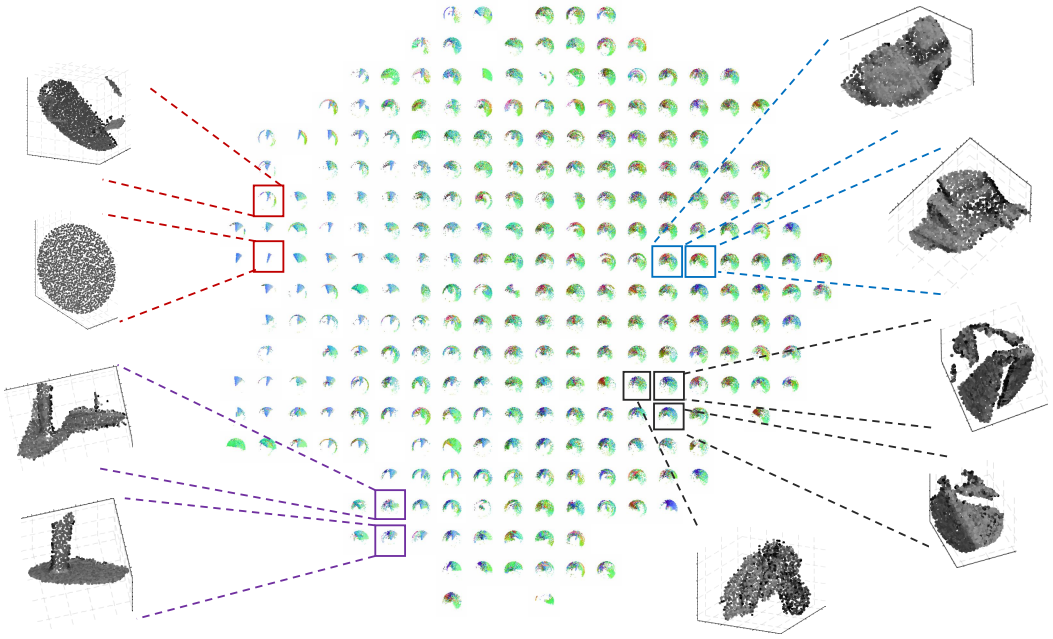


Fig. 5.9. Visualization of the latent space of codewords, associated PPFs and samples of clustered local 3D patches using TSNE [131, 194].

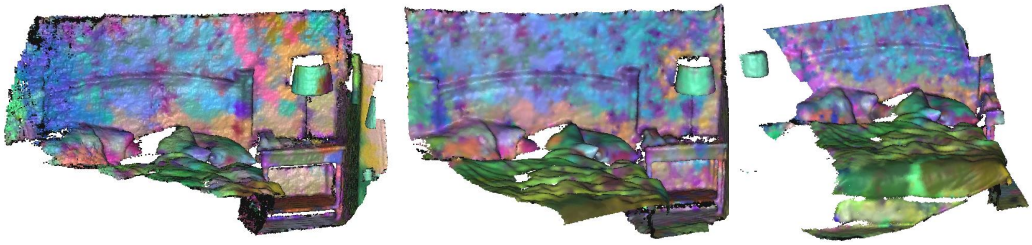


Fig. 5.10. Visualization of the latent feature space on fragments fused from different views. To map each feature to a color on the fragment, we use the TSNE embedding [131]. We reduce the dimension to three and associate each low dimensional vector to an RGB color.

5.4 Conclusion

In this section, we presented PPF-FoldNet. It is an unsupervised, rotation-invariant, uncomplex, intuitive and interpretable network tailored to learn 3D local features solely from point geometry information. PPF-FoldNet combines all the best attributes of PointNet, FoldingNet as well as our previous PPFNet. While being fully rotation invariant, PPF-FoldNet also manages to outperform all the state-of-the-art local features, including its deep learning-based peers trained with supervision, on both the standard benchmark and the much more challenging one with drastic pose variances. We are faithful that it also brings a promising new direction of learning 3D local features and see it as an important step towards the unsupervised revolution in 3D vision.

Until now, we are concentrating on the task of learning local features alone, and leave the pose estimation task to a RANSAC algorithm when necessary. However, we would also like to

investigate the possibility of adopting a network to empower the pose estimation stage, and designing an architecture which could optimize the two tasks concurrently. We will expand our focus to pose estimation in the following chapters.

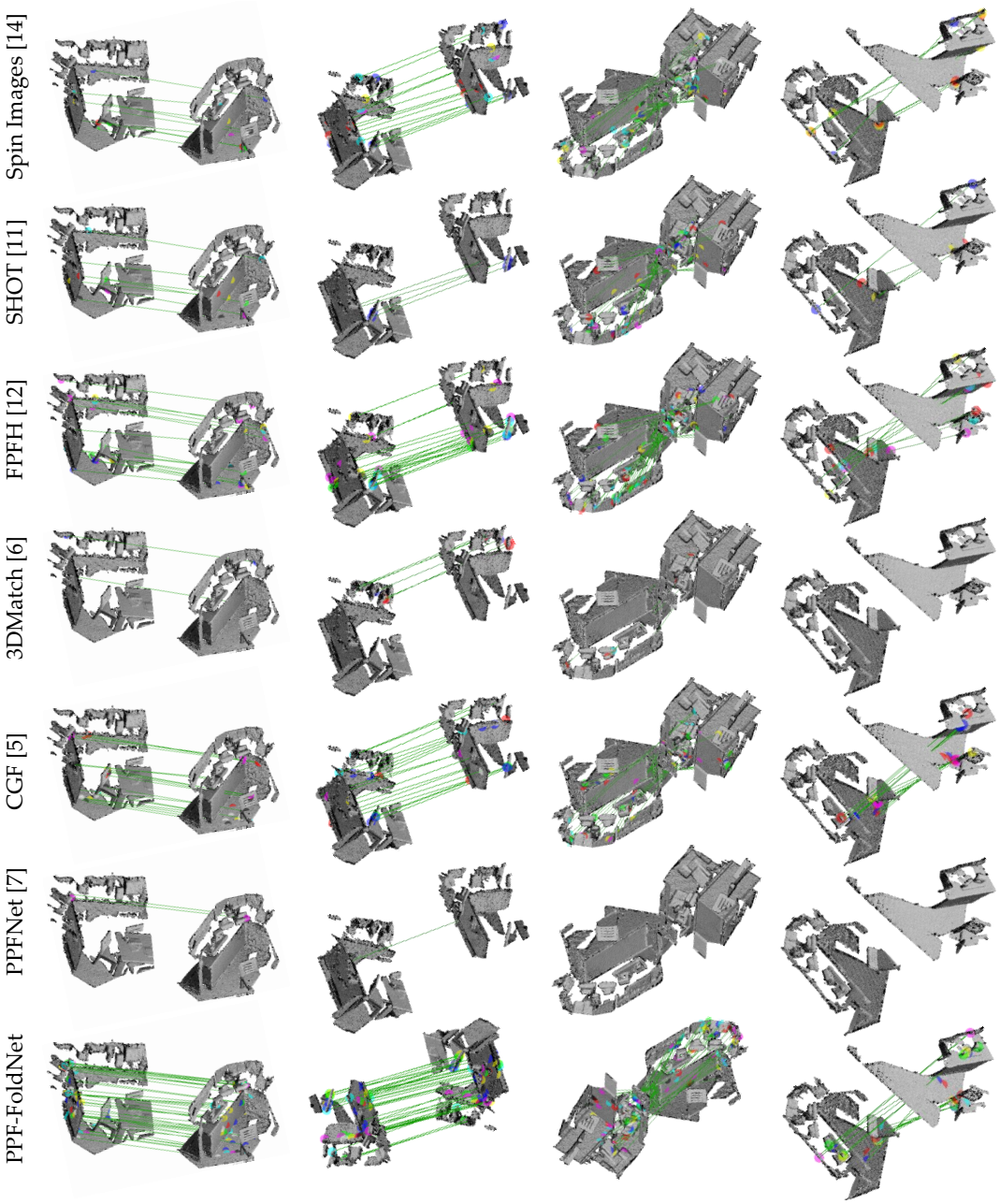


Fig. 5.11. Qualitative results of matching across different fragments and for different methods. When severe transformations involving rotations are present, only hand-crafted algorithms, CGF and our method achieve satisfactory matches. However, for PPF-FoldNet, the number of matches is significantly larger

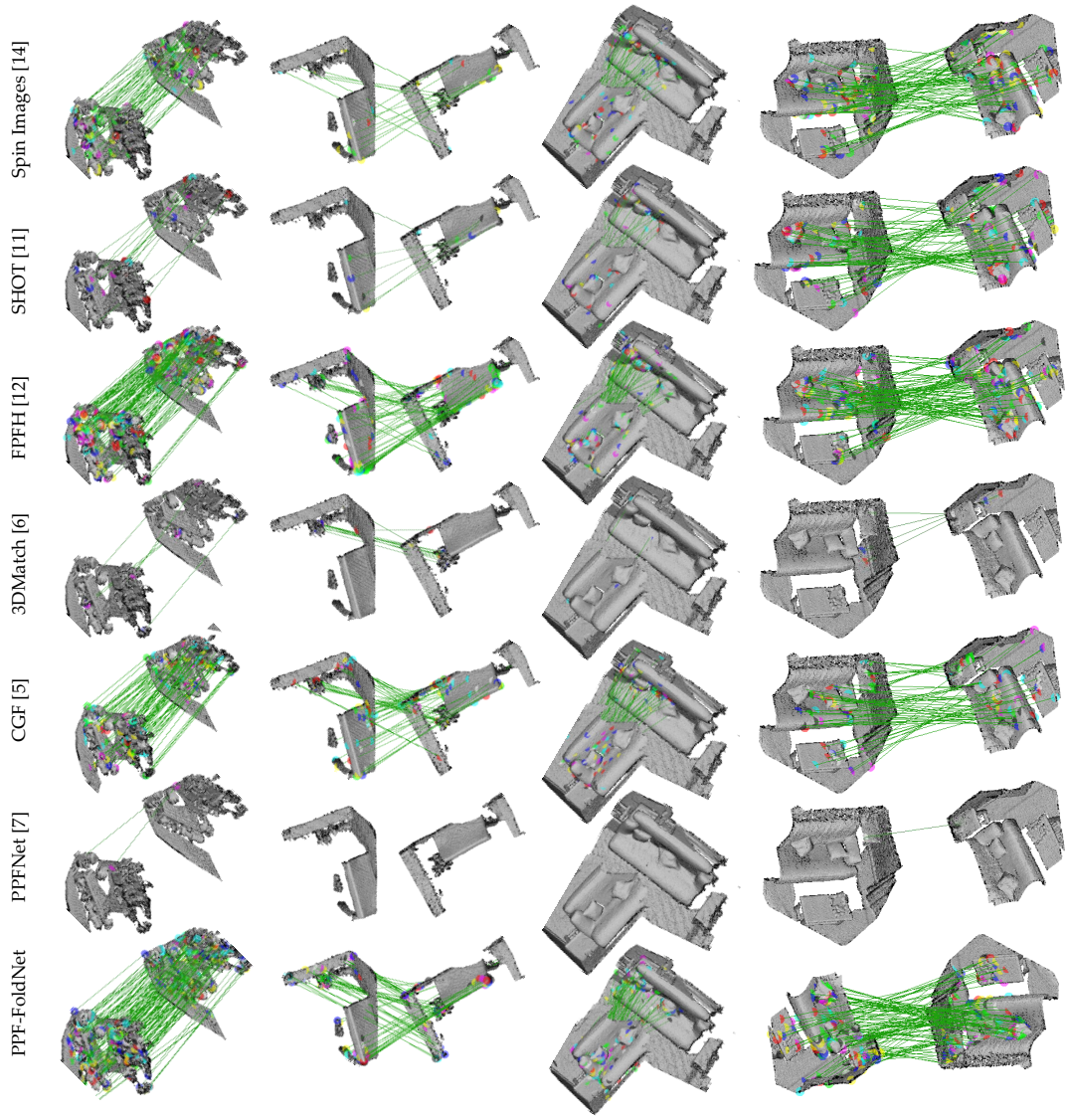


Fig. 5.12. Further qualitative results of matching across different fragments and for different methods. When severe transformations involving rotations are present, only hand-crafted algorithms, CGF and our method achieve satisfactory matches. However, for our PPF-FoldNet, the number of matches is significantly larger.

Part III

From Features to Poses

6 Introduction

This chapter precedes the next two chapters, providing motivation and related work on the topic of pose-centric researches, including pose estimation, pairwise registration, dealing with uncertainty and ambiguity in the applications of pose estimation.

6.1 Motivation

In the previous part, we are dedicated to learning more robust and discriminative 3D local features from point clouds, which can be used to obtain better correspondences. It is natural to plug our learned features into the existing feature-based pose estimation algorithms to improve the subsequent performance. We demonstrated it with the geometric registration application which adopts a RANSAC scheme to iteratively guess the set of inliers and compute the pose from the inliers using Kabsch-like algorithms [101].

However, within a RANSAC loop, only the coordinates of the matched keypoints are considered, the surrounding local neighborhood is left out. Moreover, for generating each pose hypothesis, at least three matching pairs would be required. It means for N pairs of matched keypoints, up to $\binom{N}{3}$ pose hypotheses would be generated and evaluated against. It would be desired if we could generate a pose hypothesis for each pair utilizing the information in the corresponding local patches. With the development of deep learning, directly regressing the pose has been observed in more and more applications. PoseNet [108] is such an example. It takes in a 2D image, and predict the camera location and orientation in the world coordinate to capture such an image. Inspired by this, we would like to undermine the pose information in the local patches with a network.

In the meanwhile, we also notice that in many pose estimation related applications, uncertainty and ambiguity are two major problems commonly faced by researchers. Uncertainty can be a good metric to measure the quality of predictions. When no uncertainty information is given, we have to treat all the results equally, which might lead to unexpected results. Ambiguity is another issue mostly caused by symmetries, occlusions, etc. , where identical observations can be acquired under different poses. Existing work either fails to handle them or requires sophisticated expertise. In our case, for estimating the pose of local geometries, these issues can become more obvious and prominent. Our goal is to come up with a generic framework, which can not only serve the purpose of pose estimation, but also provides a effective solution for these problems.

6.2 Related Work

In this section, we first introduce related work on the general topic of pose estimation. Then we move on to the pairwise registration, which is more related to the application targeted in this thesis. Finally, we present literature on the research of dealing with the problem of uncertainty and ambiguity.

6.2.1 Pose Estimation

Pose estimation is a basic yet critical task in many vision applications. Typical scenarios, where pose estimation is needed such as predicting the pose of CAD models in the images [102, 104, 217] or in the point clouds [150], obtaining poses of the camera in the space for given images [21, 23, 27, 28, 62, 105, 106, 108, 136], and predicting relative pose for the purpose of alignment [49, 51, 201], are prevalent.

Template matching was quite popular in the field of 6D object pose estimation, thanks to the availability of the CAD models [85, 87]. Correspondence matching demonstrates to be more robust to occlusions and clutter. Buch et al. [26] uses local shape features to obtain correspondence and further derive the poses. Zakharov et al. [218] learns to predict the dense correspondences between the image and the given CAD model. Apart from the correspondence based methods used in those applications [50, 218, 220], direct regression from the input is also quite popular due to its simplicity [49, 108, 207]. Shotton et al. [175] adopt a forest to regress the camera location in the scene coordinate. With the development of deep networks, it is getting much easier to extract powerful features from either image [53, 83] or 3D data [151, 154], which also paves the way for a more accurate direct regression. Kendall et. al. [108] adopted a convolutional network to regress the 6D camera location and orientation. PoseCNN [207] learns to regress the semantic labels and poses of CAD models from the images simultaneously. Despite the differences due to specific application fields, a common goal which aims to predict the 6D pose for the target, including rotation and translation, is shared. The results of those pose estimation applications would further serve the purpose of robotic manipulation, navigation, 3D reconstruction, AR/VR and so on.

6.2.2 Pairwise Registration

Pairwise registration typically involves a pair of point clouds with a similar scale and a certain amount of overlap with each other. It serves as an essential block in the 3D reconstruction where a set of partial scans are required to be aligned under a common coordinate, and the usual first step is to find the pairwise relative pose. The approaches to pairwise registration can be divided into two main research directions.

The first group branches off from Random Sample Consensus (RANSAC) [64]. These works rely on a set of putative matches of keypoints using feature matching and attempt to filter out the erroneous ones in an iterative way to get the optimal inlier set. A Kabsch-like [101] algorithm can compute a transformation that minimizes the distance error on the set of inliers. While being robust and able to deal with occlusions and clutter, a notable drawback of

RANSAC is the huge amount of trials required, especially when the inlier ratio is low and the expected confidence of finding a correct subset of inliers is high [38]. This encourages the researchers to propose accelerations to the original framework. At this time, the literature is filled with an abundance of RANSAC-derived methods [41, 42, 43, 114], unified under the USAC framework [157].

Another direction tries to align two point clouds globally. Iterative closest point (ICP) [11] and its descendants [11, 122, 187, 210] are representative algorithms which alternatively hypothesize a correspondence set based on the Euclidean distances and minimize the 3D registration error by progressively optimizing for the rigid pose. Despite being more efficient, it is quite difficult for ICP to handle outliers, which remains an open issue [31, 58, 96, 195]. Practical applications of ICP also incorporate geometric, photometric or temporal consistency cues [143] or odometry constraints [223], whenever available. ICP is prone to the initialization and is known to tolerate only up to a $15 - 30^{\circ}$ misalignment [13, 14]. As a result, ICP is often used where there is no big change in the movement is expected or only after the RANSAC stage to further refine the poses.

6.2.3 Dealing with Uncertainty

Typical CNNs [83, 176] are over-confident in their predictions [73, 225], as they do not qualify the predictions, i.e. all the predictions are assumed to be equally correct [49], and no side information of uncertainty is available to indicate how good or bad the predictions are. Moreover, these networks tend to approximate the conditional averages of the target data [19]. These undesired properties render the immediate outputs of those networks unsuitable for the quantification of calibrated uncertainty. Initial attempts to capture the uncertainty of camera relocalization involved Random Forests (RF) [25]. Valentin et. al. [193] stored GMM components at the leaves of a scene coordinate regression forest [175]. The modes are obtained via a mean shift procedure, and the covariance is explained by a 3D Gaussian. A similar approach later considered the uncertainty in object coordinate labels [22]. It is a shortcoming of RF that both of these approaches require hand crafted depth features. Moreover, their uncertainty is on the correspondences and not on the final camera pose. Thus a costly RANSAC [63] is required to propagate the uncertainty in the leaves to the camera pose.

Probability is the right way to capture uncertainty [10]. Kendall and Cipolla [106] improved PoseNet [108] with uncertainty by sampling posterior to approximate probabilistic inference. Mixture Density Network [19] learns to predict parameters of a Gaussian mixture distribution by using a neural network. Yet it suffers from problems like mode collapse and numeric instability, and Gaussian distributions are not ideal for modeling directional data. VidLoc [44] adapted MDNs [19] to model and predict uncertainty for the 6D relocalization problem. Besides the reported issues of MDNs, VidLoc incorrectly modeled the rotation parameters using Gaussian and lacked the demonstrations of uncertainty on rotations. Prokudin et. al. [149] replaced Gaussian distribution with von Mises distribution to enable estimation of a continuous probability space for head orientations. However, only poses aligned with a certain axis can be modeled by 2D von Mises distribution. Bingham distribution [12], on the other hand, is found to be a good way of analyzing quaternion distributions in a full

rotation space. Glover et. al.[67, 68] estimated parameters of Bingham distribution via *Sample Consensus*. Manhardt et. al. [135] used Bingham distribution as a tool of analyzing multiple hypotheses [135] in a post-processing stage. In no case was Bingham distribution incorporated into the network to enable end-to-end learning of parameters like in MDN [19], which is the main goal of this work.

6.2.4 Dealing with Ambiguity

Ambiguity is another important issue, which needs to be taken good care of in the context of pose estimation. In general, ambiguities arise where multiple legit solutions exist. In our scenario, it can be caused by the object's rotational symmetries or identical views acquired by cameras under different poses. It can to a certain extent be alleviated by using Dropout [65] as a Bayesian approximation, but even for moderate dimensions these methods still face difficulties in capturing multiple modes. In theory, this method can generate discrete samples from the multi-modal distribution, in practice, as we will demonstrate, the Monte Carlo scheme tends to draw samples around a single mode. This method also suffers from the large errors associated to PoseNet [108] itself. The pose estimation network of Pitteri et. al. [148] explicitly considered axis-symmetric objects whose pose cannot be uniquely determined. Likewise, Corona et. al. [46] addressed general rotational symmetries. All of these works require extensive knowledge about the object and cannot be extended to the scenario of localizing against a scene without having a 3D model. Note that they also cannot handle the case of self-symmetry and [46] additionally requires a dataset of symmetry-labeled objects, an assumption unlikely to be fulfilled in real applications.

Most prior work targeting ambiguities derives from the field of future prediction [125, 130]. [55, 78] proposed to generate multiple outputs as possible choices, and a *winner takes all (WTA)* strategy was proposed [78] and later widely adopted in other applications such as semantic segmentation [130]. Rupprecht et. al. [164] provided a better way to understand the benefits of this branch of methods with a mathematical formulation, and a relaxation term was introduced to WTA to facilitate convergence. In these works, only discrete outputs are considered instead of continuous space. To close the gap, Makansi et. al. [133] learned to fit parameters of a Gaussian mixture model to the generated point hypotheses in a two-stage training scheme with a variant of WTA loss. Manhardt et. al. [135] generated multiple quaternions as hypotheses for 6D pose estimation to deal with rotational symmetries and self-occlusion symmetries from visual data.

7 3D Local Features for Direct Pairwise Registration

In the previous half of this thesis, we are dedicated to obtaining better local features to improve the pool of correspondences that we could get. A typical practice to estimate the rigid transformation based on the set of matches is *RANSAC*, which iteratively samples at least three pairs randomly to generate one pose hypothesis with a Kabsch-like algorithm and evaluate it. The best one by the evaluation criterion is kept as the final pose estimation. The specific implementation of the RANSAC algorithm can impact the final pose estimation results a lot, even assuming the set of given correspondences is fixed. There has been a wide range of works focusing on improving the algorithm [42, 43, 114, 157]. However, they all focus solely on the set of matched keypoints, and ignore the local geometric patterns in their neighborhoods.

In this chapter, we present a deep learning-based framework to simultaneously learn local features, as well as regress the poses from the features directly. To achieve this, each local patch will be encoded into a pose-invariant and pose-variant feature with networks. The invariant feature will be used to find the match while another network will try to recover the pose from the features. In summary, our method learns to regress the relative pose between two point clouds via the features of the set of matched local patches. This way, only one matched pair is required to generate one pose hypothesis, which largely decreases the validation process. Comprehensive experiments demonstrate that our method not only achieves better pose estimation performance, but also requires less computation and evaluation time.

7.1 Introduction

Local features extraction and matching have fueled computer vision for many years, and having 3d local features at hand is usually considered as an intermediate step towards solving more complicated and challenging 3D vision problems. One of the most representative problems of such is to estimate the six-degree-of-freedom (6DoF) rigid transformation between pairs of 3D data, which is also termed *3D pairwise registration*. It is no doubt that the quality of the used 3d features has a huge impact on the final estimation results [75], the subsequent ways to solve the pose problem on the set of established correspondences are also quite important, so directly solving the pose problem is alluring. However, recent deeply learned 3D descriptors [109, 220] as well as our previous work, *PPFNet* and *PPF-FoldNet*, are not tailored for this task we consider. Especially to facilitate the matching purpose, it is desired in the design philosophy to get rid of the pose information in the resulted local features. Hence, a typical pose estimation paradigm is composed of nearest neighbor search and subsequent exhaustive RANSAC iterations, which is quite exhaustive and computationally inefficient.

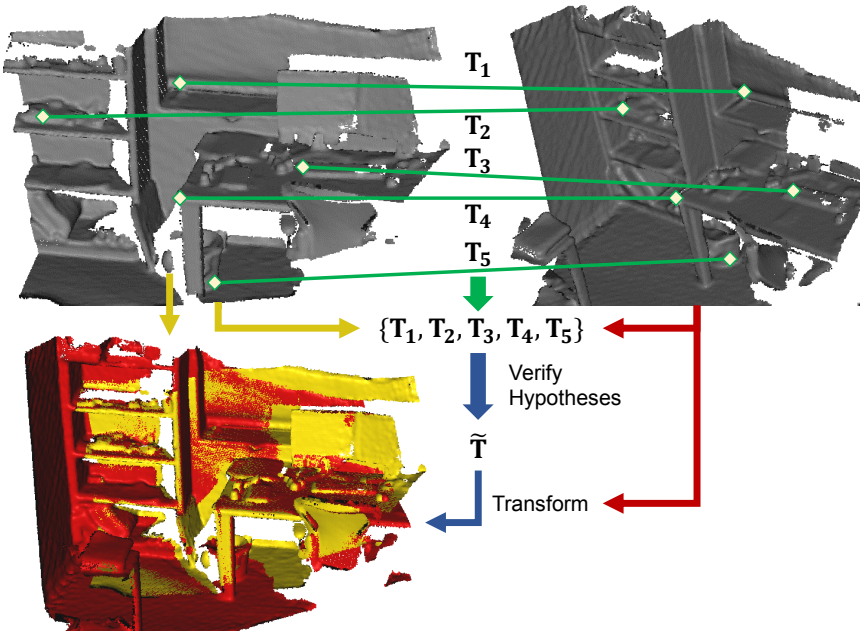


Fig. 7.1. Our method provides not only powerful features for establishing correspondences, but also directly predicts a rigid transformation attached to each correspondence. Final estimation of the rigid pose between fragment pairs can then be made efficiently by operating on the pool of pose predictions.

Here we would like to argue that descriptors can not only be used for finding correspondence, but also amenable to provide cues for direct computation of local rotations. Motivated by this idea, we propose a novel and robust end-to-end framework for local feature-based pairwise registration of pairs point clouds, as shown in Fig. 7.1. We start by coupling our previous PPF-FoldNet with a pose-variant peer which preserves the orientation information in the resulted features. Then we decouple the common 3D structure information from the pose information with subtraction in the latent space. This can result in features solely explaining the pose variability up to a reasonable approximation. With the observation that poses make well registration for local patches lead to good global alignment and vice versa, we propose a simple yet effective hypothesize-and-verify scheme to find the optimal pose from the set of pose hypotheses that are generated by prediction of our network from the set of matched features.

To make the aforementioned idea to work as expected, we should make sure the orientations associated with the keypoints, i.e. the centers of the local patches, are reliably assigned. Unfortunately, finding such repeatable orientations of local patches immediately calls for local reference frames (LRF), which are by themselves a large source of ambiguity and error [146]. Therefore, we instead choose to learn to estimate the *relative* transformations between pairs of local patches instead of canonical poses of each, which is much easier to obtain as they should be identical with the global relative pose between the holistic point clouds where those local patches are sampled from. We find that relative motion to be more robust and also easier to train, which eliminates the definition of unique and robust canonical poses as there is indeed no inherent ground-truth for the canonical pose of a patch. To this end, we introduce *RelativeNet*, a specialized architecture for relative pose estimation.

We train all of our networks within the framework end-to-end by combining three loss functions: 1) Chamfer reconstruction loss for the unsupervised PPF-FoldNet [50], 2) Weakly-supervised relative pose cues for the transformation-variant local features, 3) A feature-consistency loss which enforces the nearby points to give rise to nearby features in the embedding space. Fig. 7.2 depicts the complete architecture of our framework. We evaluate our method extensively against multiple widely accepted benchmark datasets of 3DMatch-benchmark [220] and Redwood [37], on the important tasks of feature matching and geometric registration. On our assessments, we improve the state of the art by 6.83% in pairwise registration while reducing the runtime by 20 folds. This dramatic improvement in both aspects stems from the weak supervision making the local features capable of spilling rotation estimates and thereby easing the job of the final transformation estimator. The interaction of three multi-task losses in return enhances all predictions.

Overall, our contributions are:

1. Invariant + pose-variant network for local feature learning designed to generate pose-related descriptors that are insensitive to geometrical variations.

2. A multi-task training scheme which could assign orientations to matching pairs and simultaneously strengthen the learned descriptors for finding better correspondences.

3. Improvement of geometric registration performance on given correspondence set using direct network predictions both in terms of speed and accuracy.

7.2 Method

Ideally the information carried in a purely geometric local patches can be divided into two parts. One is the *structure*, summarized by the sample points themselves $\mathbf{P} = \{\mathbf{p}_i \,|\, \mathbf{p}_i \in \mathbb{R}^{N\times 3}\}$ where $\mathbf{p} = [x, y, z]^\top$. The other is *motion*, which in our context corresponds to the 3D transformation or the *pose* $\mathbf{T}_i \in SE(3)$ holistically orienting and spatially positioning the point set $\mathbf{P}$:

$$SE(3) = \left\{ \mathbf{T} \in \mathbb{R}^{4\times 4} \colon \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \right\}. \tag{7.1}$$

where $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$. A point set $\mathbf{P}_i$, representing a local patch can be viewed as a transformed replica of its canonical version $\mathbf{P}_i^c$: $\mathbf{P}_i = \mathbf{T}_i \otimes \mathbf{P}_i^c$. Most of the cases, it is non-trivial to find such an absolute pose $\mathbf{T}_i$ to transform the point set back to its canonical form. A popular way to tackle this problem involves computing *local reference frames (LRF)* [170]. However, it is also known to be less reliable when corruptions are presented in the data [146]. Instead, we base our idea on the premise that a good local pose estimation based on a pair of patches leads to a good global alignment of the two fragments. So when the relative pose between the fragments are known, it is easy to obtain the ground truth relative poses between local patches. In summary, the basic idea behind our method is to decouple the pose component from the structure information to facilitate a pose recovering task. We devise a data-driven predictor network capable of regressing the pose for arbitrary patches and showing

good generalization properties. Fig. 7.2 depicts our architectural design. In the following part, we tackle the problem of relative pose labeling without the need for a canonical frame computation.

Generalized Pose Prediction

A naive way to achieve tolerance to 3D-structure is to train the network for pose prediction conditioned on a database of input patches and leave the invariance up to the network [51, 220]. Unfortunately, networks trained in this manner either demand a very large collection of unique local patches or simply lack generalization. To alleviate this drawback, we opt to eliminate the structural components by training an invariant-equivariant network pair and using the intermediary latent space arithmetic. We characterize an equivariant function Ψ as [204]:

$$\Psi(\mathbf{P}) = \Psi(\mathbf{T} \otimes \mathbf{P}^c) = g(\mathbf{T})\Psi(\mathbf{P}^c) \tag{7.2}$$

where $g(\cdot)$ is a function dependent only upon the pose. When $g(\mathbf{T}) = \mathbf{I}$, Ψ is said to be $\mathbf{T}$-*invariant* and for the scope of our application, for any input $\mathbf{P}$ leads to the outcome of the canonical one $\Psi(\mathbf{P}) \leftarrow \Psi(\mathbf{P}^c)$. Note that Eq. (7.2) is more general than Cohen's definition [45] as the group element $\mathbf{T}$ is not restricted to act linearly. Within the content of this chapter, the term *equivariant* will loosely refer to such *quasi-equivariance* or *co-variance*. When $g(\mathbf{T}) \neq \mathbf{I}$, we further assume that the action of $\mathbf{T}$ can be approximated by some additive linear operation:

$$g(\mathbf{T})\Psi(\mathbf{P}^c) \approx h(\mathbf{T}) + \Psi(\mathbf{P}^c). \tag{7.3}$$

$h(\mathbf{T})$ being a probably highly non-linear function of $\mathbf{T}$. Hopefully, a network can be trained to achieve the approximation and keep the loss as small as possible. By plugging Eq. (7.3) into Eq. (7.2), we arrive at:

$$\Psi(\mathbf{P}) - \Psi(\mathbf{P}^c) \approx h(\mathbf{T}) \tag{7.4}$$

that is, the difference in the latent space can approximate the pose up to a non-linearity, h. We approximate the inverse of h by a four-layer MLP network $h^{-1}(\cdot) \triangleq \rho(\cdot)$ and propose to regress the motion (rotational) terms:

$$\rho(\mathbf{b}) \approx \mathbf{R} \,|\, \mathbf{t} \tag{7.5}$$

where $\mathbf{b} = \Psi(\mathbf{P}) - \Psi(\mathbf{P}^c)$. Note that $\mathbf{b}$ solely explains the motion and hence, can generalize to any local patch structure, leading to a powerful pose predictor under our mild assumptions.

The manifolds formed by deep networks are found sufficiently close to a Euclidean flatness. This rather flat nature has already motivated prominent works such as GANs [69] to use simple latent space arithmetic to modify faces, objects etc. Our assumption in Eq. (7.3) follows a similar premise. Semantically speaking, by *subtracting out* the structure specific information from point cloud features, we end up with descriptors that are pose/motion-focused.

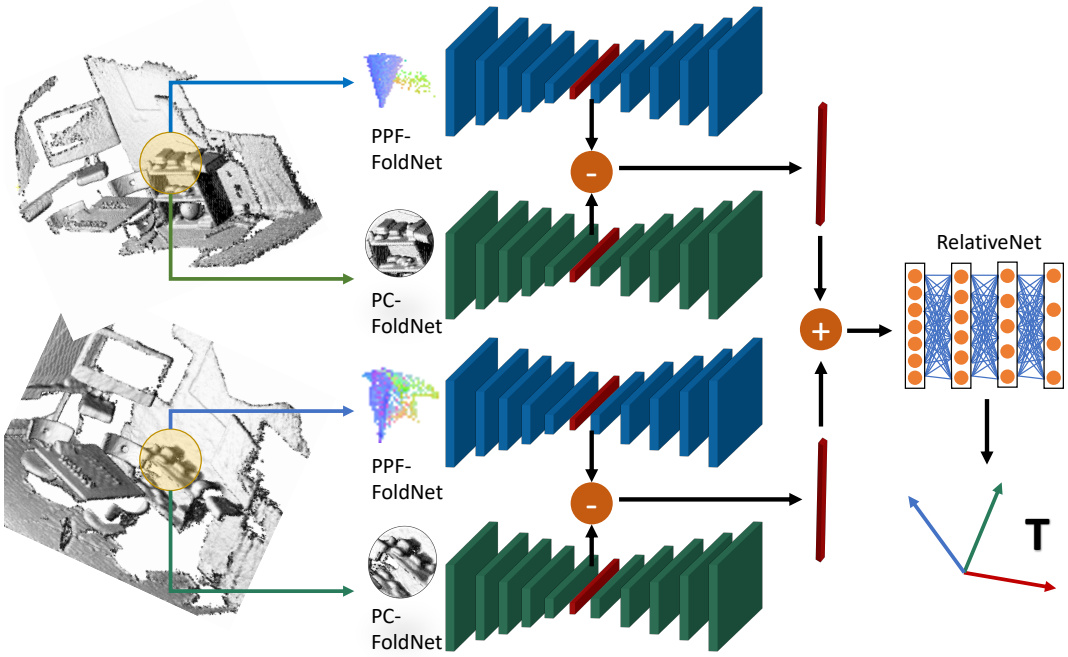


Fig. 7.2. Overview of proposed pipeline. Given two point clouds, we first feed all the patches into PPF-FoldNet and PC-FoldNet auto-encoders to extract invariant and pose-variant local descriptors, respectively. Patch pairs are then matched by their intermediate invariant features. The pairs that are found to match are further processed to compute the discrepancy between invariant PPF-based features and PC-based features. These ratio features belonging to pairs of matching keypoints are concatenated and sent into RelativeNet, generating relative pose predictions. Multiple signals are imposed on reconstruction, pose prediction and feature consistency during the training stage.

Relative pose estimation

Note that by its design, $\rho(\cdot)$ can be used to directly regress the absolute pose to a canonical frame. Yet, due to the aforementioned difficulties of defining a unique local reference frame, it is not advised [146]. Since our scenario considers a pair of scenes, and we observe that corresponding local structures of two scenes (i, j), that are well-registered under a rigid transformation $\mathbf{T}_{ij}$, also align well with $\mathbf{T}_{ij}$. As a result, the relative pose between local patches could be easily obtained by calculating the relative pose between the fragments and vice versa. We can safely estimate a *relative pose* rather than the absolute, ousting the prerequisite for a nicely estimated LRF. This also helps us to easily forge the labels needed for training. Thus, we model $\rho(\cdot)$ by a relative pose predictor, *RelativeNet*, as shown in Fig. 7.2.

We will use these ideas in the following subsection to design our networks, and explain how to train them.

Network Design

To realize our generalized relative pose prediction, we need to implement three key components: the invariant network $\Psi(\mathbf{P}^c)$ where $g(\mathbf{T}) = \mathbf{I}$, the network $\Psi(\mathbf{P})$ that varies as a function of the input and the MLP $\rho(\cdot)$. The recent PPF-FoldNet [50] auto-encoder is luckily very suitable to model $\Psi(\mathbf{P}^c)$, as it is unsupervised, works on point patches and achieves true invariance thanks to the point pair features (PPF) fully marginalizing the motion terms.

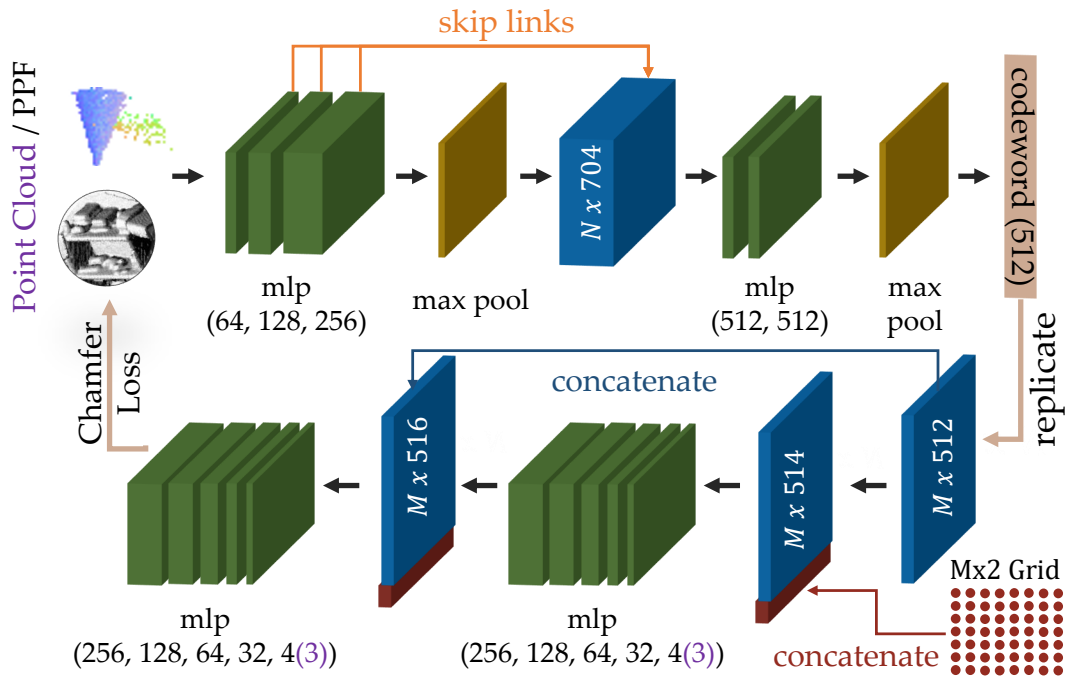


Fig. 7.3. The architecture of PC/PPF-FoldNet. Depending on the input source, the number of last layers of unfolding module is 3 for point clouds and 4 for point pair features, respectively.

Interestingly, keeping the network architecture identical as PPF-FoldNet, if we were to substitute the PPF part with the 3D points themselves ($\mathbf{P}$), the intermediate feature would be dependent upon both structure and pose information. We coin this version as *PC-FoldNet* and use it as our equivariant network $\Psi(\mathbf{P}) = g(\mathbf{T})\Psi(\mathbf{P}^c)$. We rely on using PPF-FoldNet and PC-FoldNet to learn rotation-invariant and -variant features respectively. They share the same architecture while take in a different encoding of local patches, as shown in Fig. 7.3. Taking the difference of the encoder outputs of the two networks, i.e. the latent features of PPF- and PC-FoldNet respectively, results in new features which specialize almost exclusively on the pose (motion) information. Since the encoded features from PPF-FoldNet for both patches are almost identical, excluding the common information from the features from PC-FoldNet has the potential to keep only the distinct information (mostly introduced by different poses) and facilitate the training as well. Those features are subsequently fed into the generalized pose predictor RelativeNet to recover the rigid relative transformation. The overall architecture of our complete relative pose prediction is illustrated in Fig. 7.2.

Multi-Task Training Scheme

We train our networks with multiple cues, including supervised and unsupervised. In particular, our loss function L is composed of three parts:

$$L = L_{rec} + \lambda_1 L_{pose} + \lambda_2 L_{feat} \tag{7.6}$$

L_{rec}, L_{pose} and L_{feat} are the reconstruction, pose prediction and feature consistency losses, respectively. For the sake of clarity, we omit the function arguments.

Reconstruction loss

L_{rec} reflects the reconstruction fidelity of PC/PPF-FoldNet. To enable the encoders of PPF/PC-FoldNet to generate good features for pose regression, as well as for finding robust local correspondences, similar to the steps in PPF-FoldNet[50], use the *Chamfer Distance* as the metric to train the both of the auto-encoders in an unsupervised manner:

$$L_{rec} = \frac{1}{2}\Big(d_{cham}(\mathbf{P}, \hat{\mathbf{P}}) + d_{cham}(\mathcal{F}_{ppf}, \hat{\mathcal{F}}_{ppf})\Big) \tag{7.7}$$

$$d_{cham}(\mathbf{X}, \hat{\mathbf{X}}) = \max\left\{ \frac{1}{|\mathbf{X}|} \sum_{\mathbf{x}\in\mathbf{X}} \min_{\hat{\mathbf{x}}\in\hat{\mathbf{x}}} \|\mathbf{x} - \hat{\mathbf{x}}\|_2,\ \frac{1}{|\hat{\mathbf{X}}|} \sum_{\hat{\mathbf{x}}\in\hat{\mathbf{X}}} \min_{\mathbf{x}\in\mathbf{X}} \|\mathbf{x} - \hat{\mathbf{x}}\|_2 \right\}. \tag{7.8}$$

$\hat{}$ operator denotes the reconstructed (estimated) set and $\mathcal{F}_{ppf}$ the PPFs of the points computed identically as [50].

Pose prediction loss

A correspondence of two local patches are centralized and normalized before being sent into PC/PPF-FoldNets. This cancels the translational part $\mathbf{t} \in \mathbb{R}^3$. The main task of our pose prediction loss is then to enable our RelativeNet to predict the relative rotation $\mathbf{R}_{12} \in SO(3)$ between given patches $(1, 2)$. Hence, a natural choice for L_{pose} describes the discrepancy between the predicted and the ground truth rotations:

$$L_{pose} = \|\mathbf{q} - \mathbf{q}^*\|_2 \tag{7.9}$$

Note that we choose to parameterize the spatial rotations by quaternions $\mathbf{q} \in \mathbb{H}_1$, the Hamiltonian 4-tuples [18, 30] due to: 1) decreased the number of parameters to regress, 2) lightweight projection operator - vector-normalization.

Translation $\mathbf{t}^*$, conditioned on the hypothesized pair $(\mathbf{p}_1, \mathbf{p}_2)$ and the predicted rotation $\mathbf{q}^*$ can be computed by:

$$\mathbf{t}^* = \mathbf{p}_1 - \mathbf{R}^*\mathbf{p}_2 \tag{7.10}$$

where $\mathbf{R}^*$ corresponds to the matrix representation of $\mathbf{q}^*$. Such an L2 error is easier to train with negligible loss compared to the geodesic metric.

Feature consistency loss

Unlike [50], our RelativeNet requires pairs of local patches for training. Thus, we can additionally make use of pair information as an extra *weak supervision* signal to further facilitate the training of our PPF-FoldNet. We hypothesize that such guidance would improve the quality of intermediate latent features that were previously trained in a fully unsupervised fashion. In specific, correspondent features subject to noise, missing data or clutter would generate a high reconstruction loss causing the local features to be different even for the same local patches. Also, only matched pairs are used here can avoid noises in the wrong non-matching pair relationships as shown in PPFNet. This new information helps us to guarantee that the features extracted from identical patches live as close as possible in the embedded

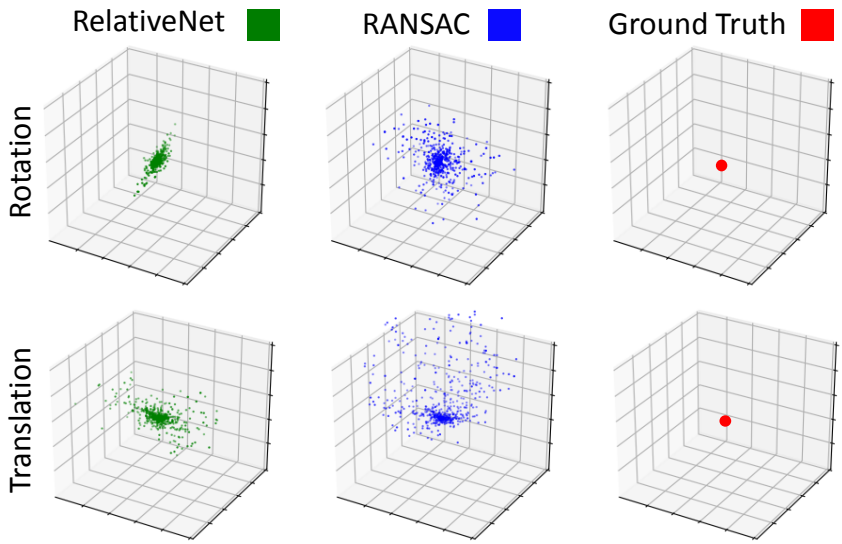


Fig. 7.4. Comparison between the hypotheses generated by our Direct Prediction and RANSAC pipeline. The first row shows the rotational component as 3D Rodrigues vectors, and the second row shows the translational component. Hypotheses generated by our RelativeNet are more centralized around the ground truth.

space, which is extremely beneficial since we establish local correspondences by searching their nearest neighbor in the feature space. The *feature consistency loss* L_{feat} reads:

$$L_{feat} = \sum_{(\mathbf{p}_i, \mathbf{q}_i) \in \mathbf{\Gamma}} \|\mathbf{b}_{\mathbf{p}_i} - \mathbf{b}_{\mathbf{q}_i}\|_2 \tag{7.11}$$

$\mathbf{\Gamma}$ represents the set of correspondent local patches and $\mathbf{b}_{\mathbf{p}}$ is the feature extracted at $\mathbf{p}$ by the PPF-FoldNet, $\mathbf{b}_{\mathbf{p}} \in \mathcal{F}_{ppf}$.

Hypotheses Selection

The final stage of our algorithm involves selecting the best hypotheses among many, produced per each sample point. The full 6DoF pose is parameterized by the predicted 3DoF orientation (Eq. (7.9)) and the translation (Eq. (7.10)) conditioned on matching points (3DoF). For our approach, having a set of correspondences is equivalent to having a pre-generated set of transformation hypotheses since each keypoint is associated a 3D rotation. Note that this is contrary to the standard RANSAC approaches where $m = 3$-correspondences parameterize the pose, and establishing N correspondences can lead to $\binom{N}{m}$ hypotheses to be verified. Our small number of hypotheses, already linear in the number of correspondences, makes it possible to exhaustively evaluate the putative matching pairs for verification. We further refine the estimate by recomputing the transformation using all the surviving inliers. The hypothesis with the highest score would be kept as the final decision.

Fig. 7.4 shows that both translational and rotational components of our hypothesis set are tighter and have smaller deviation from the true pose as opposed to the standard RANSAC hypotheses.

Tab. 7.1. Results on 3DMatch benchmark for fragment matching recall [50, 220].

	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Average
3DMatch [220]	0.5751	0.7372	0.7067	0.5708	0.4423	0.6296	0.5616	0.5455	0.5961
CGF [109]	0.4605	0.6154	0.5625	0.4469	0.3846	0.5926	0.4075	0.3506	0.4776
PPFNet [51]	**0.8972**	0.5577	0.5913	0.5796	0.5769	0.6111	0.5342	0.6364	0.6231
FoldingNet [212]	0.5949	0.7179	0.6058	0.6549	0.4231	0.6111	0.7123	0.5844	0.613
PPF-FoldNet [50]	0.7352	0.7564	0.625	0.6593	0.6058	0.8889	0.5753	0.5974	0.6804
Ours	0.7964	**0.8077**	**0.6971**	**0.7257**	**0.6731**	**0.9444**	**0.6986**	**0.6234**	**0.7458**

Implementation details

We represent a local patch by randomly collecting 2K points around a reference one within 30cm vicinity. To provide relative pose supervision, we associate each patch a pose fetched from the ground truth relative transformations. Local correspondences are established by finding the mutually closest neighbors in the feature space. Our implementation is based on PyTorch [144], a widely used deep learning framework.

7.3 Evaluation

7.3.1 Experiment Setup

The same as before, we train our networks using the training split of the 3DMatch benchmark dataset [220]. Real local patches with both different structures and different poses can be sampled from the training fragments. We assess our performance on the test set against the other data-driven feature algorithms as well as the prosperous variants derived from the RANSAC-family on the tasks of feature matching and geometric registration.

7.3.2 Matching Performance

To show that our novel training scheme can help improve the learned local features, we begin by putting our local features at test for the fragment matching evaluation, which reflects the feature quality by measuring how many good correspondence sets could be found using the features. A fragment pair is said to be matched if a true correspondence ratio of 5% and above is achieved, the same as in the previous matching performance evaluations. In Table 7.1 we report the recall of various data driven descriptors, including 3DMatch [220], CGF [109], PPFNet [51], FoldingNet [212], PPF-FoldNet [50], as well as ours. It is remarkable to see that our network outperforms the supervised PPFNet [51] by $\sim 12\%$ and the unsupervised PPF-FoldNet [50] by $\sim 6\%$. Note that, we are architecturally identical to PPF-FoldNet and hence the improvement is enabled primarily by the multi-task training signals, interacting towards a better minimum and decoupling of the shape and pose within the architecture. Thanks to the siamese structure of our network, we can provide both rotation-invariant features like [50], or upright ones, similar to [51].

Tab. 7.2. Geometric registration performance comparison. The first part lists the performances of some state-of-the-art deeply learned local features combined with RANSAC. The second part shows the performances of our features combined with RANSAC and its variants. The third part shows the results of our features combined with our pose prediction module directly. Not only our learned features are more powerful, but also our pose prediction module demonstrates superiority over RANSAC family.

			Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Average
Different Feautures + RANSAC	3DMatch [220]	Rec.	0.8530	0.7830	0.6101	0.7857	0.5897	0.5769	0.6325	0.5111	0.6678
		Prec.	0.7213	0.3517	0.2861	0.7186	0.4144	0.2459	0.2691	0.2000	0.4009
	CGF [109]	Rec.	0.7171	0.6887	0.4591	0.5495	0.4872	0.6538	0.4786	0.4222	0.5570
		Prec.	0.5430	0.1830	0.1241	0.3759	0.1538	0.1574	0.1605	0.1033	0.2251
	PPFNet [51]	Rec.	**0.9020**	0.5849	0.5723	0.7473	0.6795	0.8846	0.6752	0.6222	0.7085
		Prec.	0.6553	0.1546	0.1572	0.4159	0.2181	0.2018	0.1627	0.1267	0.2615
Our Features + RANSAC variants	USAC [157]	Rec.	0.8820	0.7642	0.6101	0.7527	0.6538	0.8077	0.6709	0.5778	0.7149
		Prec.	0.5083	0.1397	0.1362	0.2972	0.1536	0.1329	0.1530	0.1053	0.2033
	SPRT [42]	Rec.	0.8797	0.7453	0.6101	0.7253	0.6538	0.8462	0.6624	0.4444	0.6959
		Prec.	0.5170	0.1341	0.1374	0.3158	0.1599	0.1384	0.1593	0.0881	0.2062
	LR [114]	Rec.	0.8753	0.7925	0.6038	0.7198	**0.7051**	0.7692	0.6667	0.5556	0.7110
		Prec.	0.5019	0.1348	0.1294	0.2854	0.1549	0.1190	0.1465	0.1012	0.1967
	RAN SAC	Rec.	0.8530	0.7642	0.6038	0.7033	0.6667	0.7692	0.6496	0.5111	0.6901
		Prec.	0.5527	0.1614	0.1479	0.3647	0.1825	0.1587	0.1658	0.1139	0.2309
Our Features + Pose Prediction		Rec.	0.8998	**0.8302**	**0.6352**	**0.8242**	0.6923	**0.9231**	**0.7650**	**0.6444**	**0.7768**
		Prec.	0.5437	0.1778	0.1807	0.4011	0.2061	0.2087	0.1843	0.1465	0.2561

7.3.3 Geometric Registration with RANSAC

To further demonstrate the superiority of our learned local features, we evaluate them for the task of geometric registration. Local features are first extracted and then a set of local correspondences are established by nearest neighbor search in the latent space. Out of these putative matches, a subsequent RANSAC iteratively selects a subset of minimally 3 correspondences in order to estimate a rigid pose. The best relative rigid transformation between the fragment pair is then the one with the highest inlier score. For the sake of fairness among all the methods and to have a controlled setting where the result depends only on the differences in descriptors, we use the simple RANSAC framework [157] across all methods to find the best matches.

The first part of Table 7.2 shows how well different local features could aid RANSAC to register fragments on the 3DMatch Benchmark. Recall and precision are computed the same way as in 3DMatch [220]. For this evaluation, recall is a more important measure, because the precision can be improved by employing better hypothesis pruning schemes filtering out the bad matches without harming recall [109, 114]. The registration result shows that our method is on par with or better than the best performer PPFNet [51] on average recall while using a much more light-weighted training pipeline. Interestingly, our recall on this task drops when compared to the one of the fragment matching. This means that for certain fragment pairs, even though the inlier ratio is above 5%, RANSAC fails to do the work. Thus, one is motivated to seek better ways to recover the rigid transformation from 3D correspondences.

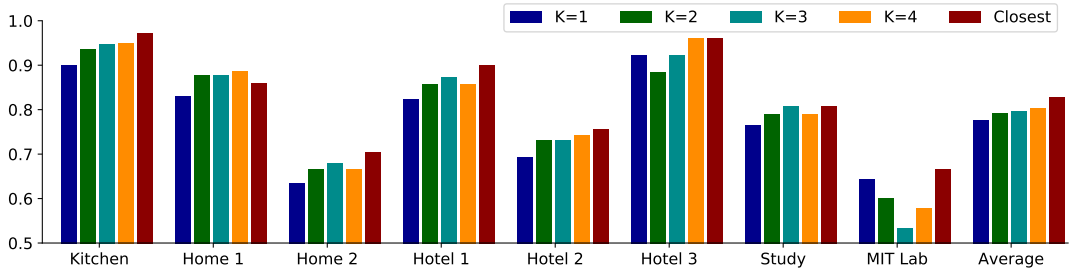


Fig. 7.5. The impact of using different methods to find correspondences. As the number of mutual correspondences kept, K, increases, more hypotheses are verified leading to a trade-off between recall and computation time.

7.3.4 Geometric Registration with RelativeNet

We now evaluate the contributions of RelativeNet in fixing the aforementioned breaking cases of RANSAC. Thanks to our architecture, we are able to endow each correspondence with pose information. Normally, each of these correspondences are expected to be good. However, in practice this is not the case. Hence, we devise a linear search to find the best of those, as explained in Section 7.2. In Table 7.2 (bottom), we report our results as an outcome of this verification, on the same 3DMatch Benchmark. As we can see, with the same set of correspondences, our method could yield a much higher recall, reaching up to 77.68%, around 8% higher than what is achievable by RANSAC. This is 7% higher than PPFNet. Also, this number is around 3% higher than the recall in fragment matching, which means that not only pairs with good correspondences are registered, but also some challenging pairs with even less than 5% inlier ratio are successfully registered, pushing the potential of matched correspondences to the limit.

It is noteworthy to point out that the iterative scheme of RANSAC requires finding at least 3 correct correspondences to estimate $\mathbf{T}$, whereas it is sufficient for us to rely on a single correct match. Moreover due to downsampling [15], poses computed directly from 3-points are crude, whereas patch-wise pose predictions of our network are less prone to the accuracy of exact keypoint location.

7.3.5 Comparison against the RANSAC-family

To further demonstrate the power of RelativeNet, we compare it with some of the state-of-the-art variants of RANSAC, namely USAC [157], SPRT [42] and Latent RANSAC (LR) [114]. Those methods are proved to be both faster and more powerful than the vanilla version [114, 157].

All the methods are given the same set of putative matching points found by our rotation-invariant features. The results depicted in Table 7.2 shows that even a simple hypothesis pruning combined with our data-driven RelativeNet can surpass an entire set of hand-crafted methods, achieving approximately 6.19% higher recall than the best obtained by USAC and 2.61% better than the highest precision obtained by standard RANSAC. In this regard, our method takes a dominant advantage on 3D pairwise geometric registration.

Tab. 7.3. The average runtime for registering one fragment pair and the number of hypotheses generated and verified.

	USAC [157]	SPRT [42]	LR [114]	Ours
Time(s)	0.886	2.661	0.591	0.013 + 0.016
# Hypos	30220	672223	2568 (46198)	335

Tab. 7.4. Average number (#) of correspondences obtained by different methods of assignments. $K = k$ refers to retaining k-mutual neighbors.

	$K = 1$	$K = 2$	$K = 3$	$K = 4$	Closest
# Matches	335	1099	1834	2609	3664

7.3.6 Runtime

Speed is another important factor regarding any pairwise registration algorithm and it is of interest to see how our work compares to the state of the art in this aspect. We implement our hypotheses verification part based on USAC to make the comparison fair with other USAC-based implementations.

The average time needed for registering a fragment pair is recorded in Table 7.3, feature extraction time excluded. All timings are done on an Intel(R) Core(TM) i7-4820K CPU @ 3.70GHz with a single thread. Note that, our method is much faster than the fastest RANSAC-variant *Latent-RANSAC* [114]. The average time for generating all hypotheses for a fragment pair by RelativeNet is about 0.013s, and the subsequent verification costs 0.016s, making up around 0.03s in total. An important reason why we can terminate so quickly is that the number of hypotheses generated and verified is much smaller compared to the RANSAC methods. While LR is capable of reducing this amount significantly, the number of surviving hypotheses to be verified is still much more than ours.

7.3.7 Impact of Correspondence Estimation

We put 5 different ways to constructing putative matching pair sets under an ablation study. Strategies include: (1) keeping different numbers of mutual closest neighboring patches $k = 1 \dots 4$, each dubbed as $K = k$ and (2) keeping the nearest neighbor for all the local patches from both fragments as a match pair, dubbed *Closest*. These strategies are applied on the same set of local features to estimate initial correspondences for further registration. The results of each method on different scenes and their average are plotted in Fig. 7.5. As k increases and the criteria for accepting a neighbor to be a pair relaxes, we observe an overall trend of increasing registration recall on different sequences. Not surprisingly, this trend is most obvious in the *Average* column. This is of course not sufficient to conclude that

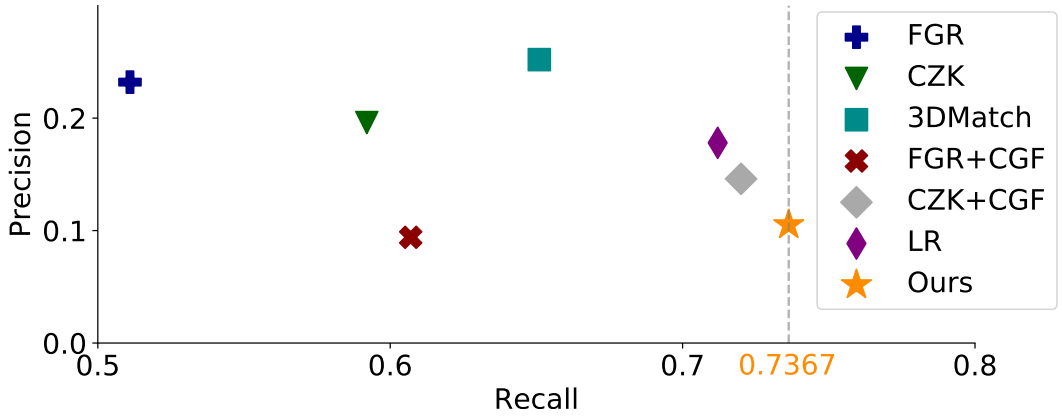


Fig. 7.6. Geometric registration performance of various methods on Redwood Benchmark [37].

relaxation helps correspondences. The second important observation is that the number of established correspondences also increases as this condition relaxes. Table 7.4 tabulates the average number of putative matches found by different methods. As we can see, the size of the correspondence set increases rapidly as we relax the standard and keep more neighbors. The average amount of putative matches found by *Closest* is around 3664, much larger than $K = 1$'s 334, approximately 10 times more, meaning that a subsequent verification would need more time to process them. Hence, we arrive at the conclusion that if recall/accuracy is the main concern, more putative matches should be kept. If, conversely, speed is an issue, *Mutual-1* could achieve a rather satisfying result quicker.

7.3.8 Generalization to Unseen Data

To show that our algorithm could generalize well to other datasets, we evaluate its performance on the well-known and challenging global registration benchmark provided by Choi et. al., the Redwood Benchmark [37]. This dataset contains four different synthetic scenes with a sequence of fragments. Our network is not finetuned with any synthetic data, instead, the weights trained with real data from the 3DMatch dataset is used directly. We follow the evaluation settings as Choi et. al. for easy and fair comparison, and report the registration results in Fig. 7.6. This precision and recall plot also depicts results achieved by some recent methods including FGR [223], CZK [37], 3DMatch [220], CGF+FGR [109], CGF+CZK [109], and Latent-Ransac [114]. Among them, 3DMatch and CGF are data-driven. 3DMatch was trained with real data on the same data source as ours, while CGF trained with synthetic data. Note that our method shows $\sim 8.5\%$ higher recall against 3DMatch. Although we are not using any synthetic data for finetuning, we still achieve a better recall of 2.4% w.r.t. CGF and its combination with CZK. In general, our method outperforms all the other state-of-the-art methods on the aspect of recall on Redwood Benchmark [37], which validates the generalizability and good performance of our method simultaneously. Note that while in general, the maximal precision is low across all the methods, it is not hard to improve it when the recall is high. To show that recall is the primary measure, we ran a global optimization [37] on our initial results, bringing precision up to 91% without a big loss of recall - still at 73%.

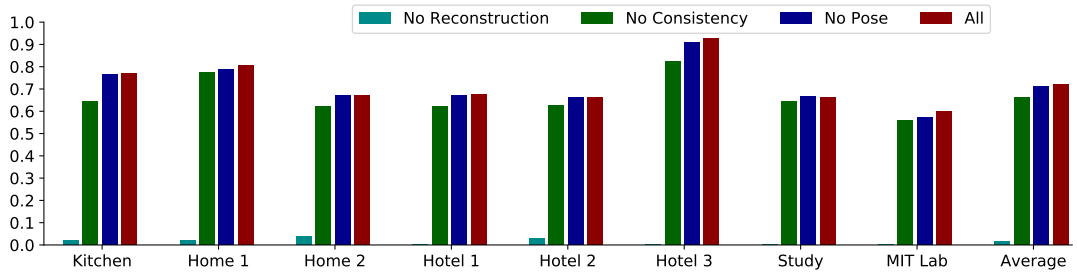


Fig. 7.7. Influences of different supervision signals. Reconstruction is the most essential loss for our network to generate local features for matching tasks. Without it the descriptive-ness is lost. When all losses are combined, the network learns to extract the most powerful features and achieves the best performance.

7.3.9 Ablation Study

Does multi-task training scheme help to boost the feature quality?

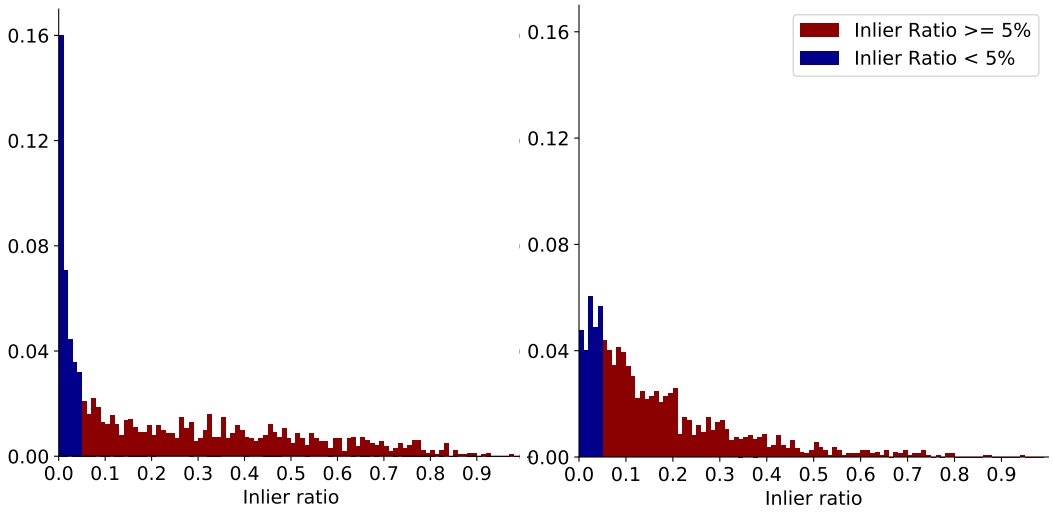
In order to find out how multi-task training affects the quality of the learned intermediate features, we trained several networks with combinations of different supervision signals. For the sake of controlled experimentation, all networks are made to have identical architecture. They are trained with the same data for 10 epochs. Hence, the only variable remains to be the objective function used for each group.

In total, there are four networks to be compared. The first one is trained with all the available supervision signals, i.e. reconstruction loss, feature consistency loss and pose prediction loss. Regarding the other three groups, each of the networks is trained with one of the three signals excluded. For simplicity, those groups are tagged as *All*, *No Reconstruction*, *No Consistency* and *No Pose* respectively. The fragment matching results using features from different networks are shown in Fig. 7.7.

As shown in Fig. 7.7, with all the training signals on, the learned features are the most robust and outperform all the others which lack at least one piece of information and thus suffer a performance drop. When no reconstruction loss is applied, the learned features almost always fail at matching. It is therefore the most critical loss to minimize. The absence of pose prediction loss has the least negative influence. Yet, RelativeNet must learn to predict the relative pose for given patch pairs. Without this, the later stages of the pipeline such as hypotheses generation and verification cannot continue. Feature consistency loss contributes most to the improved quality of learned features. These results validate that our multi-task training scheme takes full advantage of all the available information to drive the performance of learned local features to a higher level.

Matching of invariant vs pose-variant features

Our method extracts two kinds of local features using two different network components. The ones extracted by PPF-FoldNet are fully rotation-invariant, while local features of PC-FoldNet change as the pose of local patches vary. Experimentation contained in the paper used local features from PPF-FoldNet only to establish correspondences thanks to its superior property of invariance. Here, we use invariant and equivariant features to match fragment pairs separately,



(a) Equivariant Feature Matching (b) Invariant Feature Matching

Fig. 7.8. Inlier ratio distribution of fragment pair matching result using different local features from our framework. **(a)** Matching results using equivariant features extracted by PC-FoldNet. **(b)** Matching results using invariant features extracted by PPF-FoldNet. Blue part stands for the portion of fragment pairs with correspondence inlier ratio smaller than 5%. Matching results by invariant features demonstrate a better quality for further registration procedure.

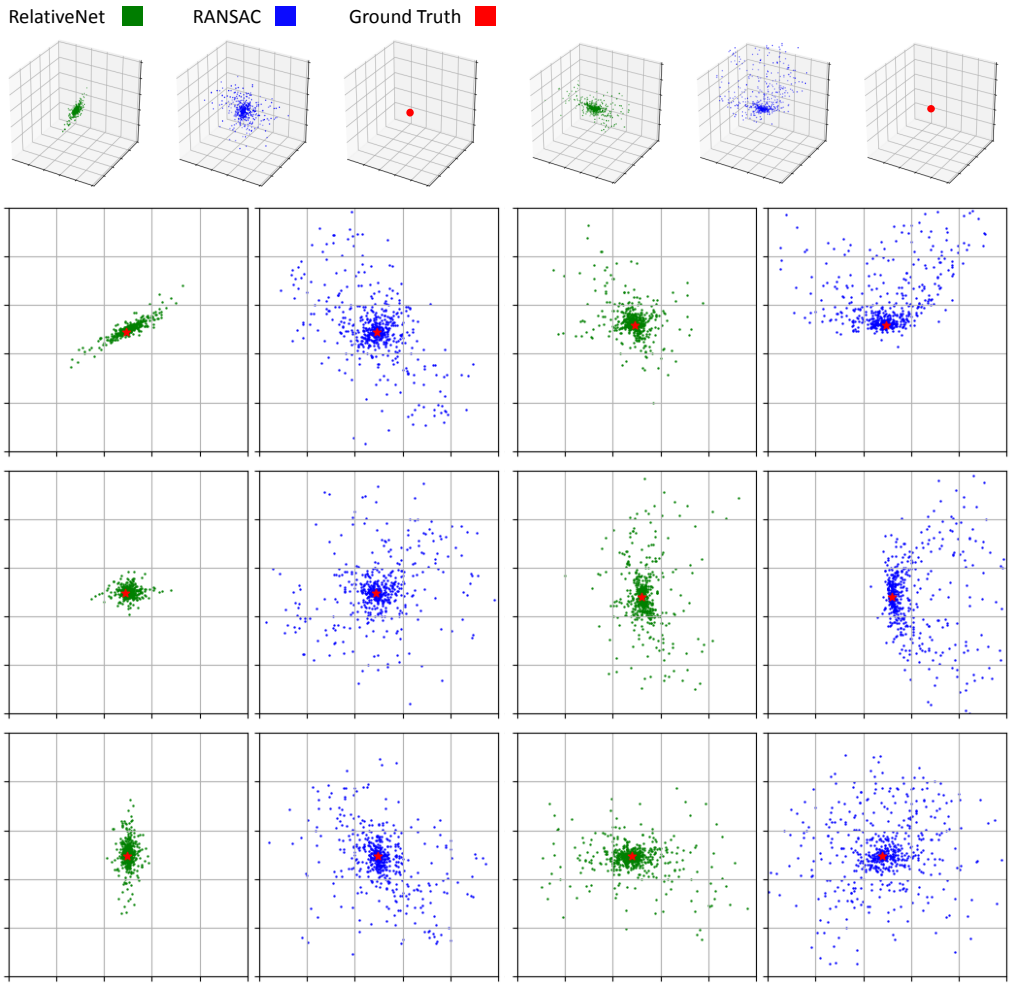
and compare their matching performance. This is important in validating our choice that invariant features are more suitable for nearest neighbor queries.

Fig. 7.8 exhibits the distribution of correspondence inlier ratio for the matched fragment pairs by using different local features. Matching results of equivariant features show a huge amount of fragment pairs having correspondences with only a small fraction of inliers (less than 5%). Invariant features though, manage to provide many fragment pairs with a set of correspondences with over 10% true matches. It proves that invariant features are better at finding good correspondence set for further registration stage. All in all, rotation-invariant features extracted by PPF-FoldNet is more suitable for finding putative local matches. Note that this was also verified by [50].

7.3.10 Quantitative Results

Distribution of hypotheses

Fig. 7.9 shows the distribution of poses predicted by RelativeNet and poses determined by running RANSAC on the randomly selected subsets of corresponding points. Each hypothesis is composed of a rotational and translational part. The former is represented as a Rodrigues vector to keep it in $\mathbb{R}^3$. It is obvious that hypotheses predicted by RelativeNet are centered more around the ground truth pose, both in rotation and translation. It also reveals the reason why the hypotheses of our network could facilitate an easier and faster registration procedure.



(a) Rotation Part (b) Translation Part

Fig. 7.9. Hypotheses distribution comparison between ones generated by RANSAC using randomly selected subset of correspondences and ones predicted by our RelativeNet. Rotation and translation parts are shown separately. The first row plots the distributions in 3D space and the following three rows are correspondent 2D projections from three different orthogonal view directions.

Qualitative comparison against RANSAC

Fig. 7.10 shows some challenging cases where only a small number of correct correspondences are established. In these examples, RANSAC fails to recover the pose information from the small set of inliers hidden in a big set of mismatches. However, a registration procedure with the aid of RelativeNet could succeed with the correct result. The qualitative comparison demonstrates that our method is robust at registering fragment pairs even in extreme cases where insufficient inliers are presented.

3D reconstruction of indoor scene

Finally, we apply our method in registering multiple scans to a common reference frame,i.e. 3D reconstruction of the whole scene. To do that, we first align pairwise scans and obtain the most likely relative pose per pair. These poses are then fed into a global registration pipeline [37]. Note that while this method can use a global iterative closest point alignment [11] in the final

stage, we deliberately omit this step to emphasize the quality of our pairwise estimates. Hence, the outcome is a rough, but nevertheless an acceptable alignment on which we can optionally apply the global-ICP refining the points and scans. The results are shown in Fig. 7.11 on the *Red Kitchen* sequence of the 7-scenes [175] as well as in Fig. 7.12 on the Sun3D Hotel sequence [208], a part of 3DMatch benchmark [220].

7.4 Conclusion

In this chapter, a unified end-to-end framework for extracting local features and estimating rigid relative pose between pairs of point clouds is proposed. We show first that it is feasible to adopt a network to regress the relative pose between pairs of local patches with specially designed features. In the meanwhile, the multi-task training scheme also leads to improved local features which could find better correspondences to benefit the subsequent pose estimation task. Comprehensive experiments on the 3DMatch benchmark again validate the superiority of our method. When compared with the state-of-the-art RANSAC methods, we show that the pose predictions by our RelativeNet from the given putative matched pairs are also shown to be both more clustered and the process is much faster. Furthermore, we analyzed how different methods of establishing local correspondences would impact registration performance. The outstanding performance of our method on the challenging synthetic Redwood benchmark strongly support our claim that our method is not only effective, but also generalizes well to new datasets.

However, in the current pipeline, all the hypotheses are treated equally and exhaustively evaluated to obtain the final pose estimation. It is desirable to assign each prediction with a score to indicate the confidence, so we could further filter out bad predictions to decrease computation and speed up the process. Also, for local patches, their geometries are simplified compared to the whole point clouds, which relieves the learning burden for the network. However, it also means more symmetric patches can be observed, which leads to a problem called *ambiguity*. The existence of ambiguity in the data could potentially harm the training of pose network a lot. In the next chapter, we will look into those problems in a more generic manner, as they exist commonly for many other pose-related vision applications as well.

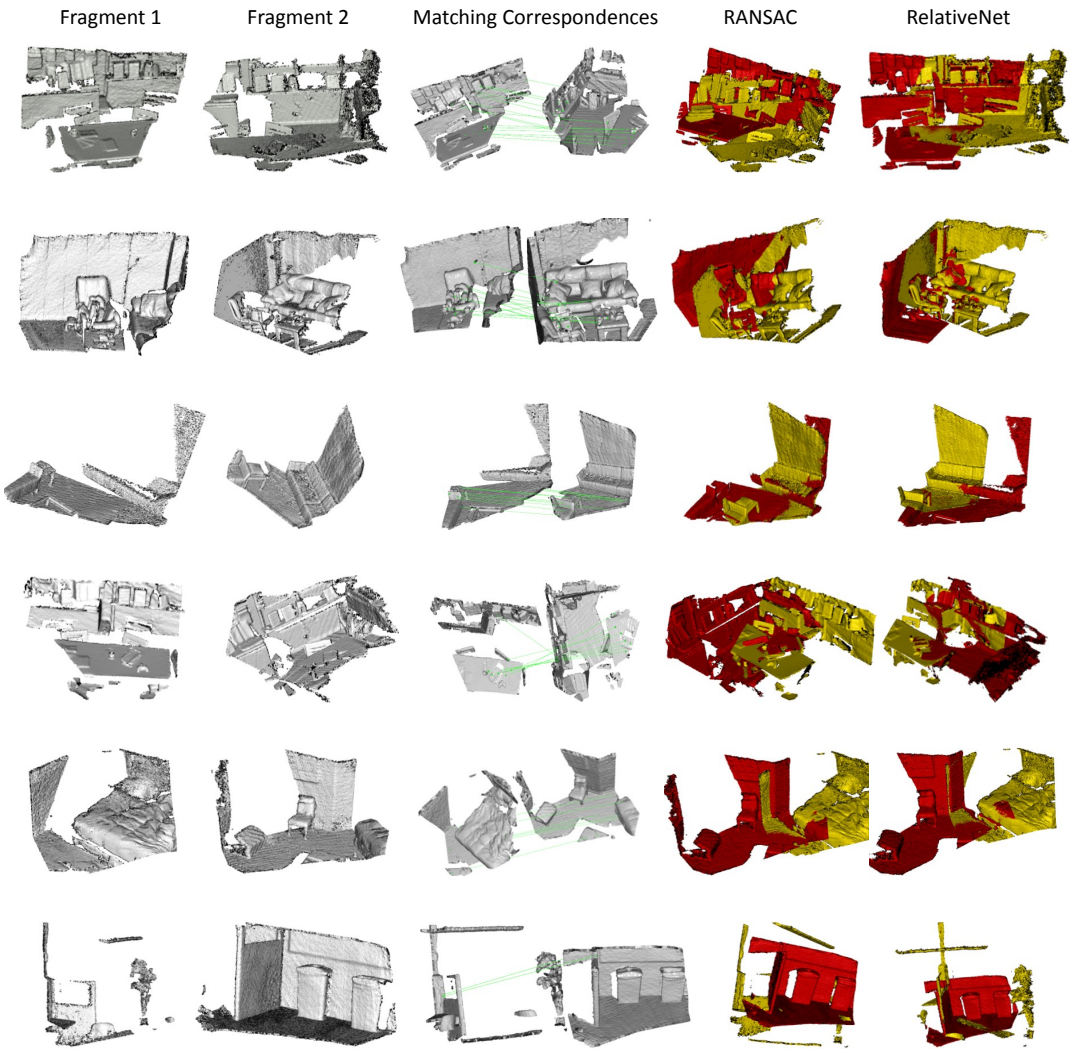
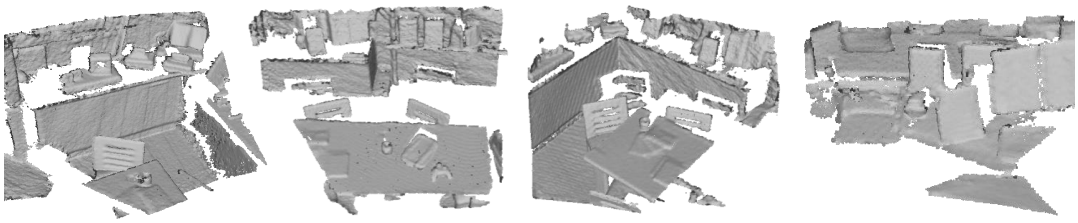


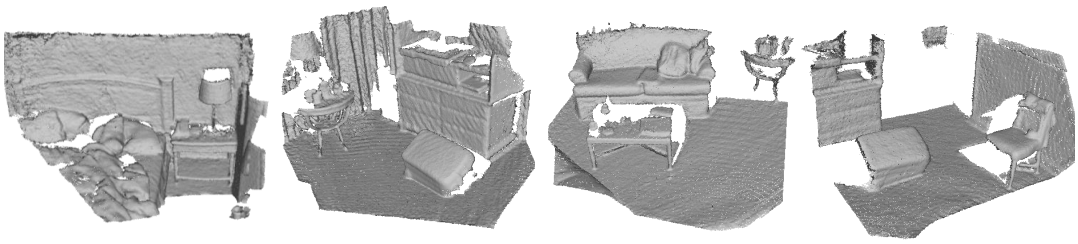
Fig. 7.10. Some challenging fragment pairs with only a small number of correct correspondences. RANSAC fails to estimate the correct relative poses between them while our network is able to produce successful registration results. Especially, for the fragment pair in the last row, only two correct local correspondences are found, which doesn't satisfy the minimum number of inliers required by RANSAC, but still correctly handled by our method.



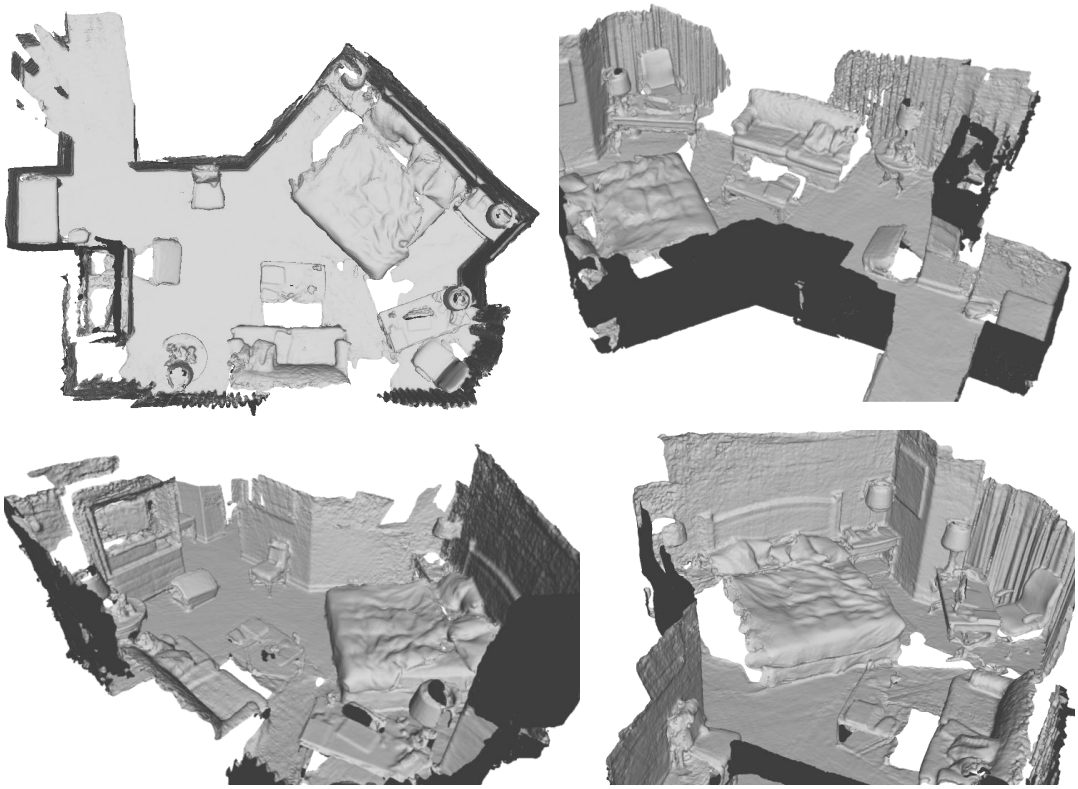
(a) Snapshots of individual scans of the Red Kitchen sequence.

(b) Views of the reconstruction obtained by running our method on multiple pairwise scans (No ICP)

Fig. 7.11. Reconstruction by 3D alignment on the entire Red Kitchen sequence of the 7scenes dataset [175]. We first compute the pairwise estimates by our method and feed them into the pipeline of [37] for obtaining the poses in a globally coherent frame. Note that this dataset is a real one, acquired by a Kinect scanner. We make no assumptions on the order of acquisition.



(a) Snapshots of individual scans of the Sun3D Hotel sequence.



(b) Views of the reconstruction obtained by running our method on multiple pairwise scans (No ICP)

Fig. 7.12. Reconstruction by 3D alignment on the entire Sun3D Hotel sequence. The reconstruction procedure is identical to the one of Fig. 7.11.

8 Uncertainty and Ambiguity in Pose Estimation

In the previous chapter, we show the feasibility of adopting a network to predict the relative pose between pairs of local patches. Similarly, network-based methods can be found targeting the other vision tasks such as camera relocalization [108] and 6D object pose estimation from images [207]. The powerful learning capacity of deep networks easies the pose prediction task by a large margin. Yet, most of them do not give any uncertainty information about the predictions, so we have to choose to trust all of them and treat them equally in further applications. In the meanwhile, ambiguity is another common issue that might exist in many situations, but are mostly ignored by those methods.

Now we explicitly target the problems of *uncertainty* and *ambiguity* in pose estimation. We propose a generic framework, termed *Deep Bingham network (DBN)*. It provides a solid solution to those problems, and can be used as an easy replacement for the direct regression model used in many pose-related applications. Our DBN learns to inference the essential parameters of a Bingham distribution for a given input, which can be used to model the distribution of the predicted pose. Based on the type of Bingham distributions modeled, our DBN can be further classified into Unimodal Bingham Network (UBN) and Multimodal Bingham Network (MBN). We show that the uncertainty information can be indicated by the entropy of the distributions, while ambiguity can be tackled by capturing the diverse modes using MBN. Experiments demonstrate that our DBN can well tackle those problems and further improve the pose estimation results.

8.1 Introduction

With the advancements in autonomous driving and 3D data capture, the need for efficient and accurate processing of 3D data has become a critical issue. At the backbone of numerous 3D systems lies point cloud object processing and pose estimation, an essential tool for the understanding of the shapes surrounding us. The capability to make sense of the objects enables multitudes of applications such as retrieval or alignment [13, 34]. In many of the cases, objects exhibit symmetries and are observed under varying orientations. In these situations, the machinery of perception has to be made robust to the variations in the input that do not alter the object geometry. For decades, vision scholars have worked on finding the unique solution [85, 90, 150, 171, 218, 219]. However, this trend is now witnessing a fundamental challenge. A recent school of thought has begun to point out that for highly complex and ambiguous real environments, obtaining a single solution for the *correct pose* is simply not sufficient. A scene with repeated structures can lead to similar views under different locations and orientations. And for objects with rotational symmetries, similar point

clouds could be captured with multiple poses around the symmetric axis. These findings have led to a paradigm shift towards estimating a range of solutions, in the form of full probability distributions [5, 17, 18] or at least solutions that estimate the uncertainty in the solutions [106, 135].

In this paper, we propose a data-driven pose estimation algorithm that can explain the multiple plausible solutions of an ambiguous input in the form of continuous multimodal predictions on the Riemannian manifold of rotations, as well as capture uncertainties by the entropy of underlying distributions. In specifics, we model rotations by a mixture of anisotropic Bingham distributions [12] that are well suited to the nature of quaternion parameterization. We started with a unimodal Bingham distribution based network, termed *UBN*. We then architect a multi-hypotheses prediction network similar to the one proposed by [135, 164], termed *MBN*. Our multi-headed network yields particle predictions that spread across the posterior in order to capture different modes. Unlike [164], we also predict the mixture weights and variances anchored on each mode resulting in a full continuous Bingham mixture distribution. Note that our scheme largely alleviates the mode collapse issues attributed to *mixture density networks* (MDN) without resorting to a full particle scheme like [164]. We propose to train our networks by a multi-task loss that drives the network to a good optimum, as we validate by the experiments.

We extensively evaluated our methods on the task of *point cloud pose estimation*, and obtained superior results against the stat-of-the-arts. Our method is also flexible in the sense that it can be used with a wide variety of backbone architectures. Moreover, we modify our *RelativeNet* from the previous chapter with our new framework and an improvement on the pairwise registration is also achieved. In Appendix B, we include extra experiments by applying our method to the application of camera relocalization, to further demonstrate that our generic framework can be easily adopted to improve the performance of applications related to pose estimation.

Having a continuous distribution of plausible solutions at hand is useful in multiple fronts: 1) estimation of the uncertainty [18] and a reliable confidence, 2) direct use of the sampled solution space to characterize the 3d object symmetries or configuration of data acquisition, 3) determining the best solution not through a naive conditional averaging but a scheme that is aware of multiple weighted modes. Note that while many types of symmetries do exist, we avoid making a distinction and rather try to capture this nuance in the multimodal predictions without explicit supervision unlike [46, 148].

In a nutshell, our contributions are:

1. We provide a general framework for continuously modeling conditional density functions on quaternions using Bingham distributions. Both unimodal and multimodal models are proposed in our work and extensively evaluated.

2. We devise a novel way of tailoring networks to fit in our framework and effective multi-task training schemes well suited to the complex and non-convex posteriors that we have, to enable efficient optimization of the necessary parameters while getting rid of

problems like mode collapse, numeric instability existing in original Mixture Density Network.

3. We exhaustively evaluate our methods on the pose-related applications, demonstrating the validity of our approach, showing that uncertainties captured by our network align well with actual rotation errors with regard to ground-truth poses, and ambiguities could be well handled by our deeply learned Bingham mixture model, both in the quality of the single best prediction as well as the ability to capture multiple ambiguous modes.

8.2 The Bingham Distribution

Derived from a zero-mean Gaussian, the Bingham distribution [12] (BD) is an antipodally symmetric probability distribution conditioned to lie on $\mathbb{S}^{d-1}$ with probability density function (PDF) $\mathcal{B} : \mathbb{S}^{d-1} \rightarrow R$:

$$\mathcal{B}(\mathbf{x}; \mathbf{\Lambda}, \mathbf{V}) = (1/F) \exp(\mathbf{x}^T \mathbf{V} \mathbf{\Lambda} \mathbf{V}^T \mathbf{x}) \tag{8.1}$$

$$= (1/F) \exp\left(\sum_{i=1}^{d} \lambda_i (\mathbf{v}_i^T \mathbf{x})^2 \right) \tag{8.2}$$

where $\mathbf{V} \in \mathbb{R}^{d \times d}$ is an orthogonal matrix ($\mathbf{V}\mathbf{V}^T = \mathbf{V}^T\mathbf{V} = \mathbf{I}_{d \times d}$) describing the orientation, $\mathbf{\Lambda} \in \mathbb{R}^{d \times d}$ is called the *concentration matrix* with $0 \geq \lambda_1 \geq \cdots \geq \lambda_{d-1}$:

$$\mathbf{\Lambda} = \text{diag}\left([\, 0, \lambda_1, \lambda_2, \ldots, \lambda_{d-1} \,] \right) \tag{8.3}$$

It is easy to show that adding a multiple of the identity matrix $\mathbf{I}_{d \times d}$ to $\mathbf{V}$ does not change the distribution [12]. Thus, we conveniently force the first entry of $\mathbf{\Lambda}$ to be zero. Moreover, since it is possible to swap columns of $\mathbf{\Lambda}$, we can build $\mathbf{V}$ in a sorted fashion. This allows us to obtain *the mode* very easily by taking the first column of $\mathbf{V}$. Due to its antipodally symmetric nature, the mean of the distribution is always zero. F in Eq. (8.1) denotes the *the normalization constant* dependent only on $\mathbf{\Lambda}$ and is of the form:

$$F \triangleq |S_{d-1}| \cdot {}_1F_1\Big(1/2,\, d/2,\, \mathbf{\Lambda}\Big), \tag{8.4}$$

where $|S_{d-1}|$ is the surface area of the d-sphere and ${}_1F_1$ is a confluent hypergeometric function of matrix argument [84, 117]. In practice, this quantity is approximated by a look-up table interpolation, lending itself to differentiation [116, 118].

Relationship to Quaternions

The antipodal symmetry of the PDF makes it amenable to explain the topology of quaternions, i. e., $\mathcal{B}(\mathbf{x}; \cdot) = \mathcal{B}(-\mathbf{x}; \cdot)$ holds for all $\mathbf{x} \in \mathbb{S}^{d-1}$. In 4D when $\lambda_1 = \lambda_2 = \lambda_3$, one can write $\mathbf{\Lambda} = \text{diag}([1, 0, 0, 0])$. In this case, Bingham density relates to the dot product of two quaternions $\mathbf{q}_1 \triangleq \mathbf{x}$ and the mode of the distribution, say $\bar{\mathbf{q}}_2$. This induces a metric of the form: $d_{\text{bingham}} = d(\mathbf{q}_1, \mathbf{q}_2) = (\mathbf{q}_1 \cdot \bar{\mathbf{q}}_2)^2 = \cos(\theta/2)^2$. Bingham distributions have been extensively

used to represent distributions on quaternions [67, 68, 117]; however, to the best of our knowledge, never for the problem we consider here.

Relationship to Other Representations

Note that geometric [8] or measure theoretic [61], there are multitudes of ways of defining probability distributions on the Lie group of 6D rigid transformations [79]. A naive choice would be to define Gaussian distribution on the Rodrigues vector (or exponential coordinates) [141] where the geodesics are straight lines [139]. However, as our purpose is not tracking but direct regression, in this work we favor quaternions as continuous and minimally redundant parameterizations without singularities [30, 71] and use the Bingham distribution that is well suited to their topology. We handle the redundancy $\mathbf{q} \equiv -\mathbf{q}$ by mapping all the rotations to the northern hemisphere.

Constructing Orientation Matrices V

Ensuring the orientation matrix $\mathbf{V}$ to be orthonormal is tricky as adding a regularization term such as $||\mathbf{V}^\top\mathbf{V} - \mathbf{I}||_F$ during optimization cannot guarantee a valid orthonormal matrix. In this work, different to [29], we investigate three different ways to construct $\mathbf{V}$:

1. **Gram-Schmidt process** A straightforward way is to first estimate an unconstrained Euclidean matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$ and then ortho-normalize it into $\mathbf{V}$ via Gram-Schmidt (GS) process. In this case, the column vectors $\mathbf{v_i}$ of $\mathbf{V}$ are computed from the column vectors $\mathbf{m_i}$ as follows

$$\hat{\mathbf{v}}_\mathbf{i} = \mathbf{m}_i - \sum_{k=1}^{i-1} \langle \mathbf{v}_k, \mathbf{m}_i \rangle \cdot \mathbf{v}_k \text{ , where } \mathbf{v}_i = \frac{\hat{\mathbf{v}}_i}{\|\hat{\mathbf{v}}_i\|}. \tag{8.5}$$

   This GS procedure requires prediction of 16 values, an over-parametrization of the degrees of freedom in $\mathbf{V}$. In the following, we refer to this process as *Gram-Schmidt (GS) strategy*.

2. **Matrix representation** To use the minimal degrees of freedom, an elegant way proposed by Birdal [18] is to estimate the mode $\mathbf{q} \in \mathbb{H}_1$ and subsequently find a set of vectors orthonormal to $\mathbf{q}$. Fortunately, the quaternions can be linearly represented by matrices. In other words, there exists an injective homomorphisms from $\mathbb{H}_1$ to the matrix ring $\mathrm{M}(4, \mathbb{R})$. The result is a frame bundle $\mathbb{H}_1 \rightarrow \mathcal{F}\mathbb{H}_1$ composed of four unit basis vectors: the mode and its orthonormals:

$$\mathbf{V}(\mathbf{q}) \triangleq \begin{bmatrix} q_1 & -q_2 & -q_3 & q_4 \\ q_2 & q_1 & q_4 & q_3 \\ q_3 & -q_4 & q_1 & -q_2 \\ q_4 & q_3 & -q_2 & -q_1 \end{bmatrix}. \tag{8.6}$$

   It is easy to verify that the matrix valued function $\mathbf{V}(\mathbf{q})$ is orthonormal for every $\mathbf{q} \in \mathbb{H}_1$. $\mathbf{V}(\mathbf{q})$ further gives a convenient way to represent quaternions as matrices paving the way

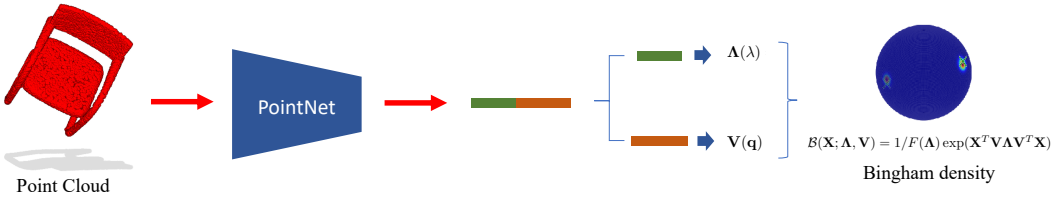


Fig. 8.1. The pipeline for Unimodal Bingham Network. The input point cloud is processed by an adequate backbone network (here we use PointNet) to output a 7-d vector from the last layer, which is later used to form $\mathbf{\Lambda}$ and $\mathbf{V}$ for a Bingham distribution.

to linear operations, such as quaternion multiplication or orthonormalization without the Gram-Schmidt. We refer this one as *Birdal Strategy*.

3. **Cayley transformation** Utilizing the Cayley transform, which describes a mapping from skew-symmetric matrices to special orthogonal matrices, we propose a third way to construct $\mathbf{V}$: Given the mode $\mathbf{q}$ (not necessarily with unit norm), we compute $\mathbf{V}$ as:

$$\mathbf{V} = (\mathbf{I}_{d\times d} - \mathbf{S})^{-1}(\mathbf{I}_{d\times d} + \mathbf{S}), \tag{8.7}$$

where $\mathbf{I}_{d\times d}$ is the identity matrix and

$$\mathbf{S}(\mathbf{q}) \triangleq \begin{bmatrix} 0 & -q_1 & q_4 & -q_3 \\ q_1 & 0 & q_3 & q_2 \\ -q_4 & -q_3 & 0 & -q_1 \\ q_3 & -q_2 & q_1 & 0 \end{bmatrix} \tag{8.8}$$

is a skew-symmetric matrix parameterized by $\mathbf{q}$. Similar to *Birdal* this allows us to only estimate four values and even removes the need of normalization to obtain a valid quaternion during optimization. We term this construction as *Cayley Strategy*.

Note that *Birdal* and *Cayley* strategies require a reduced number of predictions to yield $\mathbf{V}$ compared to *Gram-Schmidt*. We will show later that they also demonstrate better performance in Table 8.5, especially the *Birdal* strategy, which is used as the default one in our experiments.

8.3 Deep Bingham Networks

Targeting uncertainty and ambiguity issues in pose estimation, we adopt deep networks and predict the underlying posterior distribution of the target pose in an end-to-end style. We consider the situation where we observe an input point cloud $\mathbf{X} \in \mathbb{R}^{N\times 3}$ and assume the availability of a predictor function $\mathbf{V} \triangleq V_{\mathbf{\Gamma}}(\mathbf{X})$ parameterized by $\mathbf{\Gamma} = \{\mathbf{\Gamma}_i\}$. $V(\cdot)$ is an orthogonal orientation matrix computed using any of the strategies introduced in the previous section. Note that predicting entities that are non-Euclidean easily generalizes to the prediction of Euclidean quantities such as translations e.g. $\mathbf{t} \in \mathbb{R}^3$ when the constraints are removed.

To this end, we investigate two models: **Unimodal Bingham Network (UBN)** models the pose by a single Bingham distribution. In addition to computing a single best prediction, the entropy of the resulting distribution can be used as a measure of the uncertainty. **Multimodal Bingham Network (MBN)** predicts a multimodal Bingham distribution. It extends the advantage of UBN with an extra capability of capturing different modes lying in the data, thus dissolving ambiguities.

Note the definitions of our Bingham networks do not include any specific network architectures, i.e. they are agnostic of the backbone networks and can be combined with any existing networks other than the reported ones in our paper.

Unimodal Bingham Network (UBN)

UBN takes an observation in the form of a point cloud $\mathbf{X} \in \mathbb{R}^{N\times 3}$, and predicts the essential parameters of a unimodal Bingham distribution of the target pose, which describes the orientation of the object of interest.

We assume the availability of a predictor function $\mathbf{V}_{\mathbf{\Gamma}}(\mathbf{X})$ parameterized by $\mathbf{\Gamma} = \{\mathbf{\Gamma}_i\}$ and that $\mathbf{V}(\cdot)$ can output a proper orthogonal matrix following either one of the previously introduced strategies of construction. The first column of $\mathbf{V}$ represents the correct values of the rotation $\mathbf{q}_i \in \mathbb{H}_1$, admitting a non-ambiguous prediction, hence a posterior of single mode. We use the predicted rotation to set the most likely value (mode) of a Bingham distribution:

$$p_{\mathbf{\Gamma}}(\mathbf{q} \,|\, \mathbf{X}; \mathbf{\Lambda}) = (1/F)\, \exp\left(\mathbf{q}^\top \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top \mathbf{q}\right), \tag{8.9}$$

and let $\mathbf{q}_i$ differ from this value up to the extent determined by $\mathbf{\Lambda} = \{\lambda_i\}$. For the sake of brevity we use $\mathbf{V} \equiv \mathbf{V}_{\mathbf{\Gamma}}(\mathbf{X})$. In this work, we model $\mathbf{\Gamma}$ by a deep neural network.

While for certain applications, fixing $\mathbf{\Lambda}$ can work, in order to capture the variation in the input, it is recommended to adapt $\mathbf{\Lambda}$ [149]. Thus, we introduce it among the unknowns. To this end we define the function $\mathbf{\Lambda}_{\mathbf{\Gamma}}(\mathbf{X})$ or in short $\mathbf{\Lambda}$ for computing the concentration values depending on the current input $\mathbf{X}$ and the parameters $\mathbf{\Gamma}$. Our final model for the unimodal case reads:

$$p_{\mathbf{\Gamma}}(\mathbf{q} \,|\, \mathbf{X}) = \frac{\exp\left(\mathbf{q}^\top \mathbf{V}_{\mathbf{\Gamma}}(\mathbf{X})\mathbf{\Lambda}_{\mathbf{\Gamma}}(\mathbf{X})\mathbf{V}_{\mathbf{\Gamma}}(\mathbf{X})^\top \mathbf{q}\right)}{F(\mathbf{\Lambda}_{\mathbf{\Gamma}}(\mathbf{X}))} \tag{8.10}$$

$$= \frac{\exp\left(\mathbf{q}^\top \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top \mathbf{q}\right)}{F(\mathbf{\Lambda})} \tag{8.11}$$

The second row follows from the short-hand notations and is included for clarity. For the rest of this section, we stick with this simplified notations.

Inferring Bingham parameters

To produce a Bingham probability model with a network, we need to propose an end-to-end inference paradigm for $\mathbf{\Lambda}$ and $\mathbf{V}$, conditioned only on the input $\mathbf{X}$. We need to ensure all values in $\mathbf{\Lambda}$ to be non-positive and in descending order and $\mathbf{V}$ to be an orthogonal matrix.

Assume a backbone feature network is available, the original output is unconstrained and does not satisfy the properties of $\mathbf{V}$ and $\mathbf{\Lambda}$. Further processing of those raw values is necessary. In order to keep the efforts on modifications to the lowest level, we propose only to change the last layer without adapting the remaining part of the network.

Three values are needed in $\mathbf{\Lambda}$. In order to keep the diagonal values $(\lambda_1, \lambda_2, \lambda_3)$ sorted in **descending order** and **non-positive**, we make use of softplus activation and accumulative sum, with an extra negating operation.

$$\lambda_1 = -\phi(o_1) \tag{8.12}$$

$$\lambda_2 = -\phi(o_1) - \phi(o_2) \tag{8.13}$$

$$\lambda_3 = -\phi(o_1) - \phi(o_2) - \phi(o_3) \tag{8.14}$$

$\phi(\cdot)$ is the *softplus* activation function and $o_i (i = 1, 2, 3)$ are three values taken from the original output of the network.

For constructing $\mathbf{V}$, based on the specific strategies adopted, 4 or 16 values are needed from the original network output. Also, for *Birdal Strategy*, we need to normalize the four values first to make it a legal quaternion to feed into Eq. (8.6)

The hard-to-compute entity F is shown in Eq. (8.4) to depend solely on $\mathbf{\Lambda}$. To enable fast inference of F and gradient flow from $\mathbf{\Lambda}$ through F, we make use of a lookup table that is pre-computed based on a set of predefined range of values in $\mathbf{\Lambda}$.

Entropy as uncertainty

After obtaining the probability function $\mathcal{B}$ of a Bingham distribution, its entropy can be computed.

$$E(\mathcal{B}) = \log(F) - \mathbf{\Lambda} \frac{\nabla F(\mathbf{\Lambda})}{F} \tag{8.15}$$

Information theory [97] proves that higher entropy is an indicator of higher uncertainty. Similar to [133], we treat the entropy of the predicted Bingham distribution as a practical measure of uncertainty [199]. To make it a more straightforward to understand the score, we pass the entropy values through a sigmoid function which constrains the uncertainty scores within the range $(0, 1)$.

$$U = \text{sigmoid}(E(\mathcal{B})) = \frac{1}{1 + e^{-E(\mathcal{B})}} \tag{8.16}$$

Bingham loss

Given a collection of observations $\mathcal{X} = \{\mathbf{X}_i\}$ and associated rotations $\mathcal{Q} = \{\mathbf{q}_i\}$, we learn the parameters $\mathbf{\Gamma}$ of our unimodal Bingham network by minimizing the negative log-likelihood:

$$\mathcal{L}(\mathbf{q}\,,\, \mathcal{B}(\mathbf{\Lambda}, \mathbf{V})) = \log F\big(\Lambda\big) - \mathbf{q}^\top \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top \mathbf{q} \tag{8.17}$$

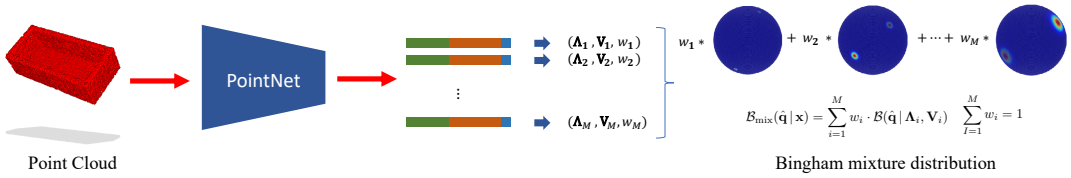


Fig. 8.2. The pipeline for Multimodal Bingham Network. The same network is used as in UBN, only the last layer is modified to output $M \times (7+1)$ units to form M groups of parameters and weights for different Bingham components. Different modes counting for ambiguity are captured by different Bingham distribution.

As shown in Fig. 8.1, we compose $\mathbf{V} : \mathbf{V}_{\mathbf{\Gamma}}(\mathbf{X})$ and $\mathbf{\Lambda} : \mathbf{\Lambda}_{\mathbf{\Gamma}}(\mathbf{X})$ during training so as to evaluate the density. This maximizes the probability of ground truth $\mathbf{q}_i$ evaluated on the associated Bingham distribution $\mathcal{B}(\mathbf{\Lambda}_i, \mathbf{V}_i)$. In short we obtain the optimal parameters $\mathbf{\Gamma}^\star$ by

$$\mathbf{\Gamma}^\star = \arg\min_{\mathbf{\Gamma}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{q}_i, \mathcal{B}(\mathbf{\Lambda}_i, \mathbf{V}_i) \,|\, \mathbf{\Gamma}) \tag{8.18}$$

Note once again that $\mathbf{\Lambda}$ and $\mathbf{V}$ are dependant upon $\mathbf{X}$, thus $\mathbf{\Lambda}_i \equiv \mathbf{\Lambda}_{\mathbf{\Gamma}}(\mathbf{X}_i)$ and $\mathbf{V}_i \equiv \mathbf{V}_{\mathbf{\Gamma}}(\mu(\mathbf{X}_i))$.

We implemented all the necessary Bingham operators as PyTorch [144] extensions to allow an efficient end-to-end training of our Bingham networks.

Multimodal Bingham Network (MBN)

While UBN is able to grant the predictions with uncertainty information, it cannot handle ambiguities such as objects with rotational symmetries, or scenes with identical views at different locations. Inspired by MDN [19], we resort to multimodality and propose a Multimodal Bingham Network for predicting multiple Bingham distributions to explain a single observation $\mathbf{X}$, and eventually yielding a Bingham mixture model, as depicted in Fig. 8.2.

Bingham mixture model (BMM)

A Bingham mixture model can be easily extended from a set of unimodal Bingham models by assigning each one a weight factor and combining them linearly to form a continuous distribution space with multimodality. Each unimodal *component* captures a *peak* presence of a valid solution. MBN then aims to predict a Bingham mixture model (BMM) for any given input $\mathbf{X}$, storing individual component predictions in different *branches*.

We build MBN on top of UBNs. Apart from the parameters $(\mathbf{\Lambda}_i, \mathbf{V}_i)$ for each unimodal component, it also predicts a set of weights $\{w_i\}_{i=1}^{M}$. The probability density function $\mathcal{B}_{\text{mix}}$ can be written as:

$$\mathcal{B}_{\text{mix}}(\mathbf{q} \,|\, \mathbf{X}) = \sum_{i=1}^{M} w_i \cdot \mathcal{B}(\mathbf{q} \,|\, \mathbf{\Lambda}_i, \mathbf{V}_i) \;\; : \;\; \sum_{i=1}^{M} w_i = 1 \tag{8.19}$$

M is the number of components. This way, different poses associated with the same input $\mathbf{X}$ can acquire high probabilities in different components.

Mixture Bingham loss

Similar to Eq. (8.18), we can define *Mixture Bingham Loss* as the negative log-likelihood of the mixture distribution model, which can be viewed as a weighted sum of *Bingham losses* per each component.

$$\mathcal{L}_{\text{MB}}(\mathbf{q}\,,\,\mathcal{B}_{\text{mix}}) = -\log(\mathcal{B}_{\text{mix}}(\mathbf{q}\,|\,\mathbf{X})) \tag{8.20}$$

Previous work on Mixture Density Network [19, 133] has shown that directly optimizing all the parameters at the same time can lead to numerical instabilities and problems such as mode collapse might arise. Thus a proper solution to tackle these issues is critical.

RWTA loss

A "Winner Takes All" training scheme has been proven to be effective for coping with ambiguities [135, 164]. In WTA, each iteration updates only the branch that generates the closest prediction to the ground truth. This provably leads to the Voronoi tessellation of the output space.

Let $\mathcal{B}_i$ be the i-th component of our Bingham Mixture Model. At each training iteration, we select the component giving the best prediction concerning the current ground truth. We could select the best component by checking the l_1 distances of the predicted quaternion $\hat{\mathbf{q}}$ about the given ground truth $\mathbf{q}$ [135].

$$i^* = \arg\min_i |\mathbf{q} - \hat{\mathbf{q}}_i| \tag{8.21}$$

An alternative way is to check the probabilities of the ground truth $\mathbf{q}$ on the set of predicted Bingham distributions, and keep the one where it gets the highest value.

$$i^* = \arg\max_i \mathcal{B}_i(\mathbf{q}\,|\,\mathbf{\Lambda}_i, \mathbf{V}_i) \tag{8.22}$$

Then we optimize $(\mathbf{\Lambda}_i, \mathbf{V}_i)$ of $\mathcal{B}_i$ by minimizing its corresponding Bingham Loss. Rupprecht et. al. [164] find that allowing a small portion of gradients from the sum of losses would help avoid "dead" components that never get updated because of their bad random initializations, which can be considered as a *relaxed* version of WTA.

$$\mathcal{L}_{\text{RWTA}}(\mathbf{q}\,,\,\mathcal{B}_{\text{mix}}) = \sum_{i=1}^{M} \pi_i \mathcal{L}(\mathbf{q}\,,\,\mathcal{B}_i(\mathbf{\Lambda}_i, \mathbf{V}_i)) \tag{8.23}$$

$$\pi_i = \begin{cases} 1-\epsilon & \text{if } i\text{-th component is selected} \\ \frac{\epsilon}{M-1} & \text{else} \end{cases} \tag{8.24}$$

$\mathcal{L}_{\text{RWTA}}$ is able to guide the multiple unimodal components to cover different modes of the ground truth distribution. Yet, it cannot not represent a continuous distribution due to the lack of $\{w_i\}$.

Cross Entropy loss

We provide an alternative way to explicitly train the weights for the components in MBN by forming it as a classification problem.

$$\mathcal{L}_{\text{CE}}(\mathbf{q}, \mathcal{B}_{\text{mix}}) = \sum_{i=1}^{M} -(y_i \cdot \log(w_i) + (1 - y_i) \cdot \log(1 - w_i)), \tag{8.25}$$

w_i is the predicted weight for the $i-th$ component and y_i is the associated label given by the selection results by either Eq. (8.21) or Eq. (8.22).

$$y_i = \begin{cases} 1, & \text{if } i = i^* \\ 0, & \text{otherwise} \end{cases}. \tag{8.26}$$

Based on those loss functions, we propose two independent training schemes.

- **Mixture Bingham + RWTA**. It follows the conventional MDN training [19] scheme, but imports extra RWTA loss to help guide the training to overcome the existing problems.

$$\mathcal{L}_{\text{MBN}} = \mathcal{L}_{\text{MB}} + \mathcal{L}_{\text{RWTA}} \tag{8.27}$$

- **Cross Entropy + RWTA**. It relies on RWTA loss to train each individual components for capturing different modes and Cross Entropy loss to assign proper weights for each one.

$$\mathcal{L}_{\text{MBN-CE}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{RWTA}} \tag{8.28}$$

We will show later in our experiments that both schemes could facilitate the training process as well as further improve the performance. To differentiate, we name the MBNs trained with the two schemes as *MBN* and *MBN-CE* respectively.

8.4 Application to Point Cloud Pose Estimation

We first evaluate the performance of our Deep Bingham Networks on the task of class-level point cloud pose estimation. We assume the category of the test object is known and an individual network is trained for each category. The pose of a point cloud can be expressed by a combination of rotation $\mathbf{R} \in SO(3)$ and translation $\mathbf{t} \in \mathbb{R}^3$. For 3D objects, translation can be canceled out using the centroid or a common anchor, whereas the rotation has to be estimated, which is also the main source of ambiguities in this application. Similarly, the latter quantity is parameterized by a unit quaternion $\hat{\mathbf{q}} \in \mathbb{H}_1$.

8.4.1 Implementation and Training Details

Our Bingham Networks and Losses are implemented using Pytorch [144] and we use PointNet [151] as the backend to process point clouds. We train for each class 500 epochs and in each epoch 100 quaternions are randomly sampled to rotate training objects. We set the learning rate at 0.001 and use Adam solver [112] to optimize the network parameters.

Birdal Strategy is used as the default way of constructing $\mathbf{V}$. The default training loss for MBN is a combination of *Mixture Bingham Loss* and *RWTA Loss* and the best component for computing *RWTA Loss* is chosen by probability.

8.4.2 Experiment Setup

Baselines

Due to the differences in the specific targeted applications and network selections, it is very hard to have a fair direct comparison. We extensively studied the state-of-the-art methods on pose/rotation estimation and evaluated the loss functions commonly employed by them:

- **L1 loss**. The most popular one is $L_p\,(p = 1, 2)$ loss, and it has been most widely used in recent work for rotation regression [3, 105, 123, 201, 202]. Previous work has found that L_1 outperforms L_2 in general [105], so we mainly take L_1 loss for comparison: $\mathcal{L}_1 = ||\mathbf{q} - \hat{\mathbf{q}}||_1$.

- **Cosine loss**. A metric often used for measuring distance on the spherical manifold, cosine loss respects the vectorial nature of quaternions [132]: $\mathcal{L}_{\cos} = 1 - |\mathbf{q} \cdot \hat{\mathbf{q}}|$.

- **PLoss**. Defined on the points rather than the rotations themselves, PLoss [207, 216] yields point-wise Euclidean distances: $\mathcal{L}_{\text{PLoss}} = ||\mathbf{q} \circ \mathbf{x} - \hat{\mathbf{q}} \circ \mathbf{x}||_2$.

We unify the above loss functions under a common framework for a fair comparison with our Bingham losses. The same network, dataset and training schemes and tasks are used for different losses. Note when training with those losses, only the quaternions parts of our Bingham Networks are trained. It is also possible to incorporate our Bingham losses in their tasks to train their networks, simply by adding more outputs to the last layer without touching the rest of the architecture.

To further showcase the power of the proposed algorithm, we include several independent state-of-the-art baselines as well. In particular, PointNetLK [3] and IT-Net [216] are two state-of-the-art deep learning-based algorithms for iteratively estimating the relative poses between pairs of point clouds. For a fair comparison, we pair the canonical and rotated point clouds to compose the input and try to predict the relative rotation. This notion is identical to what our MBN tries to predict. The remaining configurations are kept the same as in our evaluation. Moreover, we include MC-Dropout from BayesianPoseNet [107], which samples from posterior to infer the pose as well as uncertainty.

Tab. 8.1. Point cloud pose estimation results on ModelNet10. The values are scaled by 10^2.

	L1	Cosine	Ploss	PointNetLK	IT-Net	MC-Dropout	UBN	MBN-5	MBN-10	MBN-25	MBN-50	MBN-CE	MBN-MB
Bathtub	4.126	7.141	2.262	10.110	11.015	5.994	1.064	0.728	0.557	**0.490**	0.504	0.790	0.805
Bed	1.815	3.049	1.610	12.330	7.106	1.907	0.918	0.331	**0.235**	0.267	0.332	0.790	0.656
Chair	0.653	1.108	0.859	8.280	3.272	0.866	0.999	0.734	**0.600**	0.727	0.663	0.790	0.970
Desk	6.101	8.190	4.529	10.730	11.299	6.068	3.190	2.129	**1.953**	2.615	2.656	2.620	2.510
Dresser	4.524	6.753	2.888	7.260	8.500	5.124	2.285	1.423	**1.372**	1.669	1.771	2.620	2.166
Monitor	3.005	5.547	2.172	13.230	11.184	2.980	2.038	1.009	0.988	**0.984**	1.169	1.150	1.243
Night Stand	3.661	4.144	2.987	5.700	6.348	3.535	1.943	1.602	1.282	**1.248**	1.281	1.600	2.066
Sofa	0.727	1.368	0.786	12.460	3.763	0.820	0.620	**0.314**	0.327	0.352	0.408	0.300	0.444
Table	10.825	16.820	1.253	16.550	15.537	7.063	0.871	**0.428**	0.519	0.506	0.566	0.780	0.740
Toilet	0.609	1.389	0.582	7.430	3.746	0.730	0.846	0.576	0.544	0.401	**0.377**	0.570	0.769
Average	3.605	5.551	1.993	10.410	8.180	3.509	1.477	0.927	**0.838**	0.926	0.973	1.201	1.237

Dataset

We choose ModelNet10 [206] as the benchmark to conduct our evaluations. It contains 10 classes, and objects from each class have unique geometries as well as different levels of symmetries, which is ideal for validating our method in terms of uncertainty and ambiguity. We conduct class-level object pose estimation on this dataset, and follow the original train/test split.

Metrics

Due to the potential ambiguities in the point clouds, it is inappropriate to use angular error to measure the qualities of the predictions with regard to the ground truths as different poses might align the point clouds equally well. Instead, we measure the *Chamfer distance* (CD) between the point clouds rotated by the ground truth and predicted pose. In order to show how multiple modes are captured by our mixture Bingham networks, we also measure *Self-EMD* (SEMD) [133], the earth movers distance [163] of turning a multi-modal distribution into a unimodal one. With this measure we can evaluate the diversity of predictions, where the unimodal distribution is chosen as the predicted mode of the corresponding method. Note that this measure by itself does not give any indication about the accuracy of the prediction.

8.4.3 Quantitative Evaluation

UBN and MBN output continuous distributions instead of discrete poses. To evaluate their performance on the task of pose estimation, a single pose prediction for UBN is decided by taking the mode of the predicted continuous Bingham distribution, which equals the pose with the highest chance in the according probability space. Similarly, for MBN, the component with the largest weight is selected and its mode serves as the single prediction in this evaluation.

Performance comparison

Table 8.1 showcases the average *Chamfer Distance* results of all the baselines as well as our Deep Bingham Networks, including both UBN and MBN.

With comparison to all the baselines, our UBN not only achieves best performances across all the classes, but also provides extra uncertainty information. We attribute the improvement to the fact that Bingham distribution well reflects the real distance between the predictions and

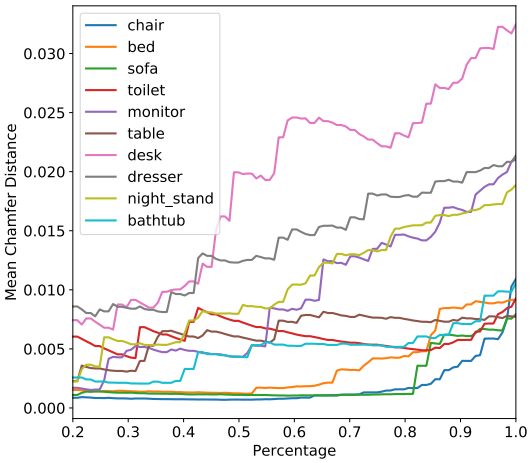


Fig. 8.3. As uncertainty threshold increases, the average CD of predictions whose uncertainties are below threshold increases accordingly.

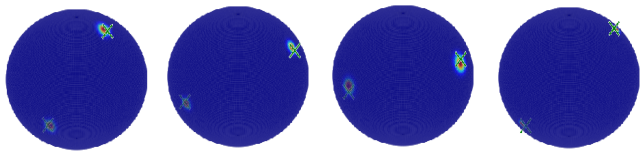
ground truth, thus lead to better convergence. Moreover, the remarkable improvement against MC-Dropout further validates the advantage of explicitly modeling a meaningful distribution. However, like other baselines, UBN is incapable of dealing with ambiguities from point clouds with rotational symmetries. This is strongly demonstrated by the further improvements obtained by our MBN, where different modes triggering ambiguities are captured by different components of the underlying multimodal Bingham model.

PointNetLK and IT-Net work similarly as ICP and they are proven to be more robust [3, 216]. Yet as we can see from Table 8.1, they could not cope with the drastic changes in the poses that well and this leads to inferior performances on this evaluation. Different to all the baseline competitors, our MBN does not require a canonical point set to estimate the orientation of the input. Instead, it relies on a high-level abstracted/implicit canonical notion for the entire class. Moreover, this canonical form is learned from the training data, making the algorithm more robust and efficient.

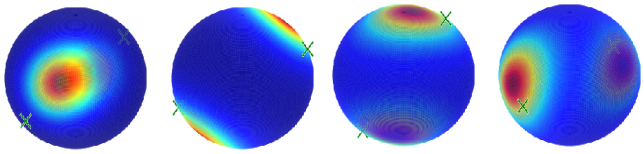
To see how the number of components would impact the performance of MBNs, we provide results for several versions of MBNs with 5, 10, 25, and 50 components respectively. As the number of components increases from 5 to 10, a further improvement can be clearly observed. However, as the number of components continues to grow up to 25 and 50, the performances are not getting better or even slightly drops, which indicates the saturation of the mode diversity existing in the data and the increased learning complexity. But overall, all our MBNs outperform the other baselines which lack the ability to handle ambiguity.

Uncertainty analyses

To better understand how our uncertainty gauge works, we plotted Fig. 8.3 by computing the average CDs of predictions with uncertainties less than a varying number of thresholds. As the uncertainty threshold decreases, the average CDs also decrease accordingly, which indicates that predictions with less uncertainty tend to align the point clouds better with the ground truth. We visualize the predicted Bingham distributions along with the ground truths

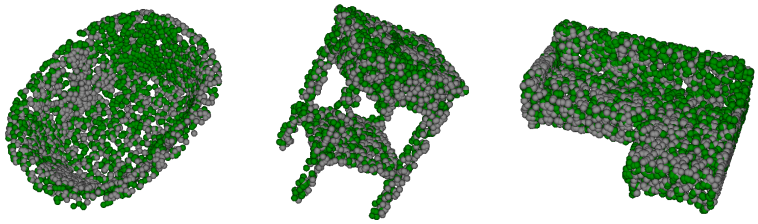


(a) Good predictions tend to have small uncertainty



(b) Bad predictions tend to have big uncertainty

Fig. 8.4. Bingham distributions generated by Unimodal Bingham Network. The ground truth poses are marked with "×".



Chamfer Distance ($\times 10^{-2}$)	0.246	0.380	0.118
Angular Error (radian)	3.14	1.55	0.05

Fig. 8.5. Commonly used angular error would fail as a good metric for the predictions which are different from the ground truth pose but still make good alignment for objects with ambiguities. In this case, chamfer distance could better reflect the quality of predictions. Gray points are from the ground truth point cloud and green ones from predicted point cloud.

in Fig. 8.4. Good predictions in general clustered and peaked distributions, while bad ones tend to spread and result in higher uncertainties.

8.4.4 Qualitative Evaluation

To illustrate that angular error might not be an optimal metric here, Fig. 8.5 shows the overlaps of point clouds rotated by ground-truth poses and predicted poses. In all the displayed cases, the predictions achieve good alignment with the ground truths, which can be well indicted by the small chamfer distances. However, if we use angular error as the metric to qualify the results, even though it could still obtain small values for unambiguous objects (e.g. sofa), but it might disregard the first two predictions with big errors due to the ambiguities in the objects.

Fig. 8.6 demonstrates how MBN is able to capture different modes in the data when ambiguities are presented. For objects with rotational symmetries, different poses which could equivalently align them are captured by different components. It is possible to further derive the symmetric axis of those objects based on the set of various predictions. In cases where an object does

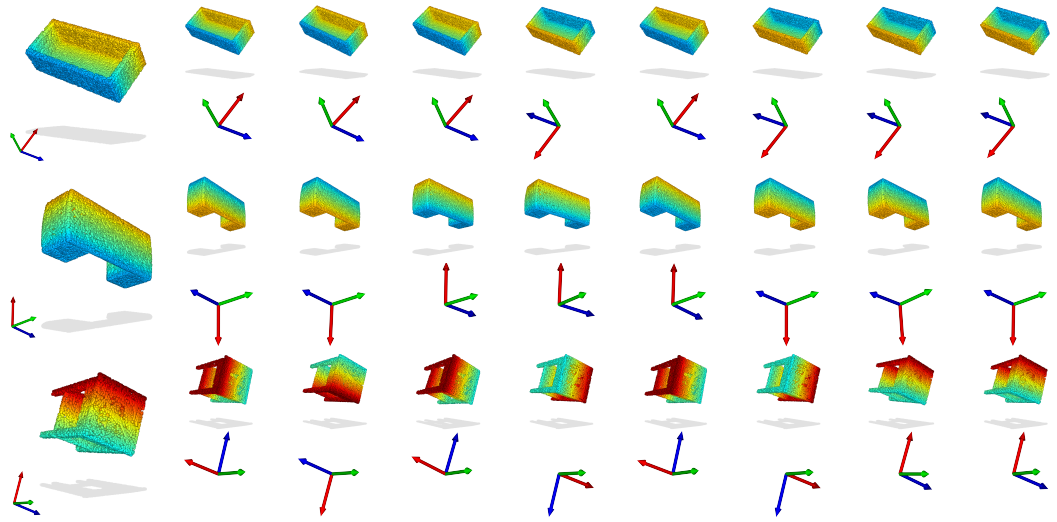


Fig. 8.6. Ambiguous poses could be well captured by different components of our Multimodal Bingham Network. The first column shows the object under ground-truth poses. The rest of the columns show predicted poses (represented as a local reference frame) and the correspondent rotated object. Point clouds are colored by the coordinates in the non-rotated version.

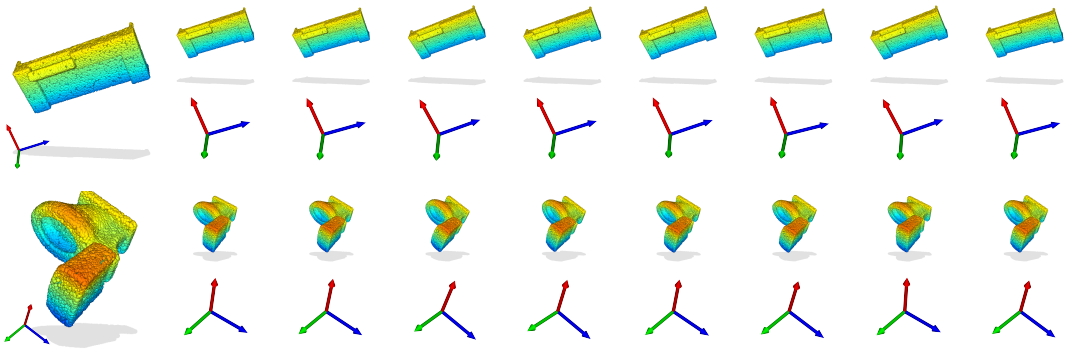


Fig. 8.7. For non-ambiguous objects, different components would generate similar predictions, where all modes correctly collapse. This can also be used to check the existence of ambiguities.

not carry ambiguity, we can see from Fig. 8.7 that all the components tend to agree on the predictions. It shows that extra components are not burdensome for non-ambiguous objects. This kind of predictions could be also further utilized as a sign to indicate whether the given point cloud is rotational symmetric or not.

8.4.5 Choice of Best Branch during Training

In our evaluation, the probability is used as the default metric in RWTA to select the best branch during training. Here we also investigate the choice using l1 and compare the two strategies as described in Eq. (8.21) and Eq. (8.22). As we can see from Table 8.2, performance-wise the two criteria are close to each other. However, as the probabilities are anyway needed for the final RWTA loss, it reduces the total amount of computation by using them for branch selection as well.

Tab. 8.2. Comparison between best branch selection criteria for MBN.

Selection	L1	Prob
CD	1.057	0.973

Tab. 8.3. Comparison between different MHP variants, including WTA, EWTA[133] with RWTA[164]. Average SEMD and chamfer distances ($\times 10^{-2}$) on ModelNet10 across all the classes.

	MB	WTA	EWTA(25)	EWTA(50)	RWTA
SEMD	0.425	**0.777**	0.639	0.513	0.729
CD	1.237	1.105	1.090	1.160	**0.973**

8.4.6 Other Multiple Hypotheses Prediction Strategies

In our benchmark evaluation, we stick with the relaxed version of WTA, termed as RWTA. Recently, [133] proposed EWTA, an evolving version of WTA, to further alleviate the collapse problems of the RWTA training schemes proposed in [164]. Updating the top k hypothesis instead of only the best one, EWTA increases the number of hypotheses that are actually used during training, resulting in fewer wrong mode predictions that do not match the actual distribution.

We compared different versions of MHP training schemes for our applications, including WTA, RWTA and EWTA. The results can be found in Table 8.3. As it is not straightforward how k should be chosen in EWTA, we 1) start with $k = K$, where K is the number of hypotheses and gradually decrease k until $k = 1$ (as proposed in [133]) and 2) start with the best half hypotheses, i.e. $k = 0.5 \cdot K$. We set $K = 50$ in our experiments. We have found k to strongly influence the accuracy of our model. In our applications, however, we have found the wrong predictions to have high uncertainty so that, if desired, they can easily be removed. Overall, RWTA results in the highest performance than WTA and EWTA in both application scenarios. Therefore, we chose to remain with RWTA to train our models. This implicitly admits $k = 1$.

8.4.7 Impact of RWTA Loss

To show how RWTA Loss helps overcome problems of mode collapse as well as facilitate the training process for MBN, we introduce a training instance following the MDN [19] using only Mixture Bingham Loss, dubbed as *MBN-MB*. From Table 8.1, we can see that MBN-50 demonstrates constantly improved performances over MBN-MB with the same configuration. Table 8.3 lists a complete comparison with MBN trained with only MB loss and the other versions combined with one of the WTA-based loss. We can see WTA-based losses not only improve the performance, but also increases the diversity in the multiple predictions which is illustrated by larger SEMD values.

Tab. 8.4. Results using continous 6D representation [224] to model rotations instead of a Bingham distribution on the quaternion. Point cloud pose estimation results on ModelNet10 across all the classes.

	Geo	UBN	MDN	MC-Dropout	MBN-CE	MBN
Bathtub	6.246	7.228	0.513	4.048	0.445	**0.427**
Bed	1.533	1.730	0.517	1.407	**0.277**	0.346
Chair	0.559	0.702	**0.515**	0.774	0.543	0.537
Desk	4.404	4.650	3.144	4.050	**3.023**	3.193
Dresser	4.792	3.799	2.216	2.590	**2.121**	2.333
Monitor	1.694	2.124	**1.076**	2.136	1.178	1.161
Night Stand	3.756	3.656	1.559	2.502	1.445	**1.434**
Sofa	0.313	0.374	0.319	0.503	0.324	**0.312**
Table	9.913	8.685	**0.473**	3.161	0.654	0.621
Toilet	0.427	0.873	1.065	0.750	**0.266**	0.423
Average	3.364	3.382	1.140	2.192	**1.028**	1.079

Tab. 8.5. Average chamfer distances ($\times 10^{-2}$) on point cloud pose estimation, averaged over ModelNet10 dataset using different strategies of constructing $\mathbf{V}$, including Gram-Schmidt (*G*), Cayley Transform (*C*) and Birdal et. al. [18] (*B*).

UBN	MBN-MB	MBN
G / C / B	G / C / B	G / C / B
4.654 / 2.821 / **1.477**	3.233 / 2.458 / **1.237**	2.867 / 1.597 / **0.973**

8.4.8 Generalization to Other Rotation Parameterization

The best choice of rotation parameterization for training deep learning models is an open question. PoseNet [108] proposed to use quaternions due to the ease of normalization. The ambiguities can be resolved by mapping the predictions to one hemisphere. MapNet [24] further showed improvements in using the axis angle representation. Recently it has been shown that any representation with four or fewer degrees of freedom suffers from discontinuities in mapping to $SO(3)$. This might harm the performance of deep learning models. Instead, [224] proposed a continuous 6D or 5D representation. We ablate in this context by mapping all predictions to the proposed 6D representation and model them using a GMM, similar to an MDN. Table 8.4 lists results for point cloud pose estimation. In Table 8.4, 'Geo' refers to a direct regression using the geodesic loss proposed in [224]. We can see there is a clear improvement in the results when the model is lifted from a single prediction to multiple predictions, which further validates ambiguities could be well handled with multimodality. Also, in both MBN variants, when RWTA loss is incorporated, the performance could be further boosted from pure MDN, which qualifies our proposed training schemes as well.

Tab. 8.6. Filtering hypotheses for registration using uncertainty information. The first row is the threshold value below which the hypotheses survive. The second row is the registration recall, which reaches 77.7% with all the hypotheses. The third row is the percentage of filtered hypotheses. The lower the uncertainty threshold, the more hypotheses are dropped. With the aid of our uncertainty, more than 20% of the hypotheses could be neglected without harming the performance. Even when 54.1% hypotheses are dropped, the performance decreases only by 4.7%.

Threshold	0.30	0.25	0.20	0.15	0.10
Ave. Recall	0.777	0.773	0.769	0.742	0.730
Drop Rate	0.184	0.221	0.280	0.388	**0.541**

Tab. 8.7. Registration results comparison between the original direct regression ("Direct") and the version modified with MBN.

	Registration Recall		Rotation Error		Translation Error	
	Direct	MBN	Direct	MBN	Direct	MBN
Kitchen	0.8998	**0.9198**	3.2501	**1.8287**	0.0700	**0.0398**
Home 1	0.8302	0.8302	2.9146	**1.8213**	0.0804	**0.0538**
Home 2	0.6352	**0.6604**	3.8968	**1.8299**	0.1184	**0.0610**
Hotel 1	0.8242	**0.8352**	3.1396	**1.6517**	0.0979	**0.0542**
Hotel 2	0.6923	**0.7179**	5.2479	**3.4842**	0.1326	**0.0797**
Hotel 3	0.9231	0.9231	5.0966	**1.8840**	0.0524	**0.0301**
Study	0.7650	0.7650	2.5968	**1.3525**	0.0913	**0.0489**
MIT Lab	**0.6444**	0.6000	4.9337	**3.8331**	0.1285	**0.0993**
Average	0.7768	**0.7814**	3.8845	**2.2107**	0.0964	**0.0583**

8.5 Application to Geometric Registration

To further demonstrate that our UBN and MBN are generic frameworks which could be useful in practical 3D applications, we apply them on the pairwise point cloud registration task.

8.5.1 Uncertainty Information with UBN

We first modify the *RelativeNet* from the previous chapter by adding extra output units to predict *concentration parameters* for $\mathbf{\Lambda}$ and then train it with our Bingham loss. Once trained, we could obtain an extra uncertainty score for each hypothesis predicted by the network. Note in the last stage of the registration pipeline, a pool of pose hypotheses would be generated and exhaustively evaluated. With our extra uncertainty information, it is possible to filter out some hypotheses with high uncertainties to accelerate the registration, without harming the performance much, as shown in Table 8.6.

8.5.2 Further Improvement with MBN

Next we follow the architecture of MBN to modify our RelativeNet by adding more output units to the last layer to generate the essential parameters for multiple Bingham distributions and their weights. Hypotheses with low confidences are filtered. General improvement on the registration recall is observed and recorded in Table 8.7.

To further demonstrate the improved pose estimation results, we compute the *Oracle Error* of the pool of predictions about the ground truth pose. It measures the accuracy of the best prediction in the set of generated hypotheses. We can see that in Table 8.7, both the rotation and the translation errors are decreased across all the test scenes. We attribute these improvements to the well-handled ambiguities in the data.

8.6 Conclusion

We have proposed elegant solutions in this chapter to enable end-to-end modeling of Bingham distributions [12] for 3D rotation estimation via deep networks, for both unimodal (UBN) and multimodal (MBN) cases. We illustrate it is feasible to train a network to regress parameters of a Bingham distribution for obtaining pose predictions as well as uncertainty information. An MDN-like Multimodal Bingham network is designed targeting ambiguity issues that cannot be handled by Unimodal Bingham Network. Novel training schemes which resort to WTA strategy to facilitate the success of training a Bingham Mixture model, avoiding mode collapse. We exhaustively evaluated our methods on two fundamental pose-related vision tasks and demonstrated its superiority over the state-of-the-art, obtaining consistently better mode predictions. We further demonstrated that our solutions can be easily incorporated into other network-based pose estimation applications to improve their performances without heavy modifications. In Appendix B, we validated this claim again by applying our method to the application of camera relocalization.

Part IV

Conslusion

9 Conclusion

In this chapter, we conclude the methods and findings of this thesis, further analyze the limitations and potential future directions.

9.1 Summary

We thoroughly studied deep learning on point clouds for features learning and pose estimation in this thesis.

The first two methods are focusing on learning more robust and distinctive local features from point clouds. *PPFNet* is the first work that empirically validates our proposal. Thanks to the efficient feature processing on point clouds, we manage to process all the local patches sampled from point clouds simultaneously. It also inspires us to generalize the contrastive loss to N-tuple loss, which fully utilizes all the available correspondence relationships, and redesign the training pipeline by injecting global information into local features. Experiments demonstrate that PPFNet manages to produce local features with better qualities than both handcrafted and other deep learning-based counterparts. *PPF-FoldNet* extends the idea with pure PPF, achieving fully rotation invariance while enjoying the sparsity of point clouds with a novel autoencoder structure. Further improvement on the matching performance justifies the superiority of this design. It cancels the necessity of pairwise labels yet manages to obtain remarkable results. We believe it opens a new paradigm of learning 3D local features unsupervised, as the same idea could be applied with more sophisticated networks to further improve the feature quality, e.g. *3D-PointCapsNet* [221].

The next work unifies the task of feature learning as well as pose estimation. It extends the unsupervised feature learning idea of *PPF-FoldNet* with an additional relative pose regression network, thus achieving an end-to-end pose prediction, eliminating the tedious RANSAC process. In the meanwhile, the multi-task training scheme improves the matching and pose prediction performances simultaneously. Encoding each local patch with both rotation-variant and rotation-invariant features suit the nature of different tasks. Our method is shown to be not only more robust, but also faster than various state-of-the-art RANSAC methods. Extensive evaluation on the Redwood benchmark validates that our method could also generalize well to unseen datasets.

In the last part, we explicitly target the uncertainty and ambiguity problems in pose estimation. An elegant solution that enables an end-to-end Bingham distribution modeling is proposed. Based on the underlying distribution, a unimodal Bingham network (UBN) and multimodal Bingham network (MBN) are designed to handle *pure uncertainty* and *uncertainty + ambiguity* scenarios respectively. We illustrate that it is feasible to train a network to regress parameters

of a Bingham distribution, for both unimodal and multimodal cases. Our novel training schemes help avoid mode collapse and numerical instability in the training procedure. We exhaustively evaluated and ablated our generic Bingham networks on the task of category-level object point cloud pose estimation, showing their ability to capture uncertainty and ambiguity information. By incorporating the framework of UBN and MBN into our previous work on the application of geometric registration, a reasonable improvement on the performance is observed as well.

Altogether, these works provide a solid insight into the field of deep learning on 3D local features as well as pose estimation on rigid point clouds.

9.2 Limitations

Point Pair Features (PPFs) are heavily used in this thesis as the basic way of encoding raw point clouds. It brings rotation invariance into the extracted features, and by fixing the center point of a local patch as the anchor point, a quadric growth of PPFs is avoided. However, this procedure comes with certain information loss. Also, as its computation involves point normals. If this information is missing in the given point cloud and the normal estimation results are bad, the performance can be severely harmed. In the meantime, our features aim to encode pure geometric information, and texture is completely ignored. So local regions without rich structure information such as planes cannot be well handled. Also, the existence of these structure-less local patches can even harm the learning of local features. For PPFNet, another limitation is that it assumes the point clouds to be matched are of similar scale, which makes it nontrivial to extend it to cases like matching objects to scenes, that differ a lot in sizes. For Multimodal Bingham Network (MBN), the number of branches has to be manually set. However, the optimal number might vary with different applications. It would be desired to leave it to the network to make the decision for specific applications.

9.3 Future Work

Looking back at the robust performance of PPFNet and PPF-FoldNet, it is noteworthy the keypoints are uniformly sampled. A natural way to further performance improvement could be to adopt some more sophisticated 3D keypoint detectors for this purpose [177]. However, a more alluring direction would be to train a network for keypoint detection and feature extraction simultaneously, or assigning each keypoint with a weight to emphasize attention [54, 98, 142]. Also, we believe the performance can be further boosted by incorporating visual clues such as color and semantic labels [93]. Despite the trivial effort in computing PPFs from a point cloud, it would be desirable that special operations, which could achieve the same results and be embedded into the network, could be designed.

Up until now, this thesis has shown that working on point cloud directly is quite memory-friendly and fast in computation due to the sparse representation. Yet the irregular structure means random memory access, which potentially harms the efficiency. Efforts trying to combine sparsity with regularity have been witnessed in recent years. Some researchers

propose to build models on point clouds with grid partitions [126, 209]. Whereas another promising direction leads to *Sparse 3D convolution* [39, 40, 70], which has been proven to be powerful on learning 3D features [40] as well. It would be interesting to further explore those possibilities with feature learning and pose estimation on 3D data.

For pose estimation from latent features, rotation equivariance is another interesting topic to continue with. Zhao et al. [222] present an architecture to process point clouds in an equivariant way with respect to the $SO(3)$ group, enabling robust pose estimation. Srivastava et al. [180] come up with a similar idea to disentangle pose with representation using capsules. They are proven to be working well with complete 3D models. How to extend those fancy ideas to partial noisy point clouds remains to be studied.

Finally, we leave it as future work to learn local features and poses from deformable objects and dynamic scenes. Unlike the data we dealt with in this thesis, the local geometries could change due to non-rigidity and dynamics, which brings about another level of challenges for the network as well as data processing.

Part V

Appendix

A 3D Point Capsule Network

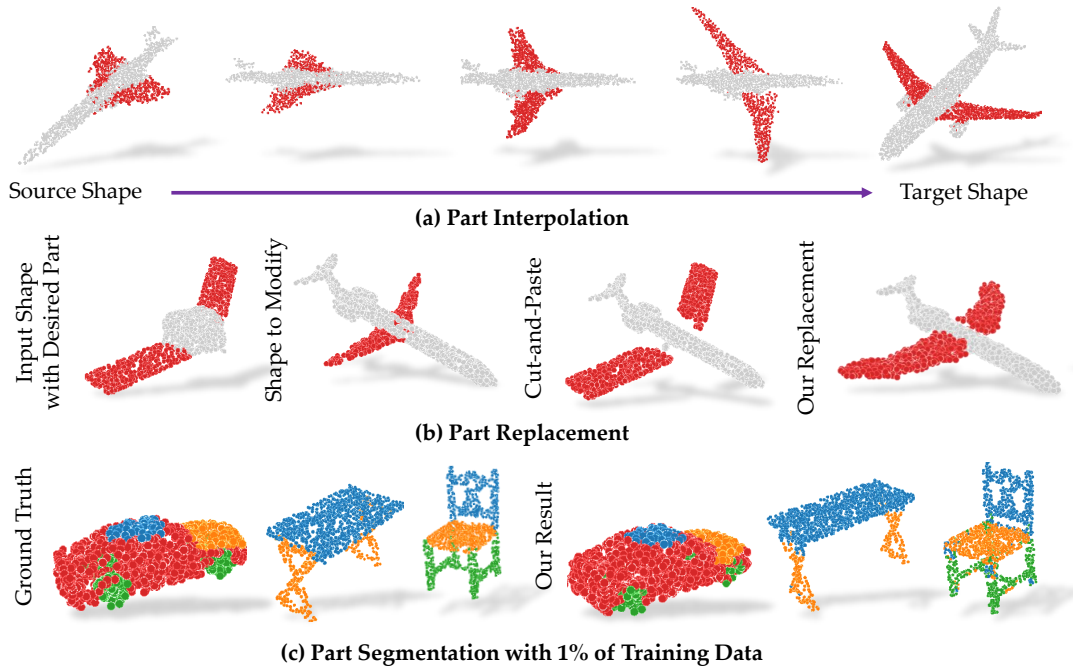


Fig. A.1. Our *3D-PointCapsNet* improves numerous 3D tasks while enabling interesting applications such as latent space part interpolation or complete part modification, an application where a simple cut-and-paste results in inconsistent outputs.

Inspired by the renowned capsule network [168], we propose the unsupervised *3D Point Capsule Network (3D-PointCapsNet)* in this project. It is an auto-encoder specially designed for generic representation learning on unstructured 3D data. With the embedding of the powerful routing-by-agreement algorithm developed for the capsule network [168], geometric relationships between local parts are fully respected by our network. It demonstrates robust learning and generalization ability. Same as PPFNet and PPFFoldNet, our 3D-PointCapsNet also uses PointNet-like layers [151] to assemble the encoder, thus it is able to process on sparse point clouds directly. The main difference is that, multiple max-pooled features will be generated and further organized by the unsupervised dynamic routing algorithm into a latent representation. It is composed of a group of *latent capsules* - stacked latent activation vectors specifying the features of the shapes and their likelihood. With those latent capsules, the restriction of representing the latent space using a single compact vector is broken. Now different local regions discovered by the routing algorithm are explicitly controlled by their corresponding latent capsule. Together they make it possible to reconstruct the complete 3D shapes by concentrating on different local parts.

To achieve that, we also come up with a novel 3D point-set decoder that takes those latent capsules as the input, with which an improved reconstruction, as well as more operational

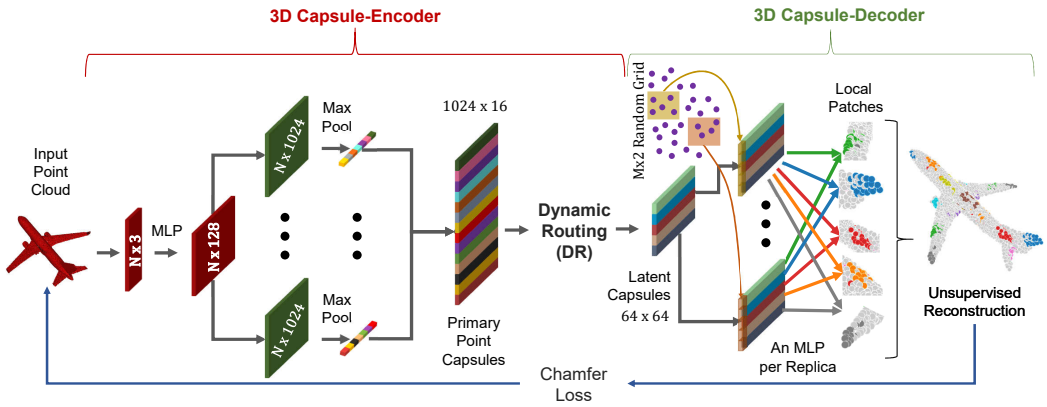


Fig. A.2. 3D Point Capsule Network. Our capsule-encoder accepts an $N \times 3$ point cloud as input and uses an MLP to extract $N \times 128$ features from it. These features are then sent into multiple independent convolutional-layers with different weights, each of which is max-pooled to a size of 1024. The pooled features are then concatenated to form the *primary point capsules* (PPC) (1024×16). A subsequent dynamic routing clusters the PPC into the final *latent capsules*. Our decoder, responsible for reconstructing point sets given the latent features, endows the latent capsules with random 2D grids and applies MLPs $(64 - 64 - 32 - 16 - 3)$ to generate multiple point patches. These point patches target different regions of the shape thanks to the DR [168]. Finally, we collect all the patches into a final point cloud and measure the Chamfer distance to the input to guide the network to find the optimal reconstruction. In this figure, part-colors encode capsules.

capabilities, are enabled, as illustrated in Fig. A.1. With the observation that the latent capsules focusing on semantically instead of spatially across the point cloud of interest, they can be regarded as various shape parameters learned in an unsupervised manner and permit further operational possibilities. We show that it is achievable to provide the network with only limited task-specific supervision to excel at solving individual sub-tasks. For example, the capsules will learn to specialize on different parts of the given shapes in the part segmentation task.

Fig. A.2 depicts the overall architecture of the proposed *3D-PointCapsNet* as a deep 3D point cloud autoencoder. More details would be revealed in the following *encoder* and *decoder* part respectively.

Encoder

We assume the input of our network is an $N \times d$ point cloud. Here N is fixed as 2048 in our experiments, and for a typical point sets, $d = 3$. Similar to PointNet [151], a point-wise multi-layer perceptron (MLP) (3-64-128-1024) is used to extract individual local feature maps. As suggested by capsule networks [168] to diversify the learning, multiple convolutional layers, which do not share the weights, will take the local feature maps as input to generate distinct summaries of the input shape with varied attention score. Those responses are further max-pooled to obtain a global latent representation. These descriptors are then concatenated into a set of vectors named *primary point capsules*, $\mathcal{F}$. The size of $\mathcal{F}$ depends upon the size $S_c := 1024$ and the number $K := 16$ of independent kernels at the last layer of MLP. dynamic routing [168] is then used to embed the primary point capsules into higher-level *latent capsules*. Each capsule is independent and can be considered as a *cluster centroid* (codeword) of the

primary point capsules. The total size of the latent capsules is fixed to 64×64 (i.e. , 64 vectors each sized 64).

Decoder

Our decoder treats the latent capsules as a feature map and uses MLP(64-64-32-16-3) to reconstruct a patch of points $\hat{\mathbf{X}}_i$, where $|\hat{\mathbf{X}}_i| = 64$. At this point, instead of replicating only a single vector as performed in [72, 212], the entire capsule is duplicated m times. To each replica, we append a unique randomly synthesized grid $\mathbf{P}_i$ specializing it to a local area. This operation further stimulates diversity. We arrive at the final shape $\hat{\mathbf{X}}_i$ by propagating those replicas through a final MLP for each patch and stacking the output patches together. Here we choose $m = 32$ to reconstruct $|\hat{\mathbf{X}}| = 32 \times 64 = 2048$ points to keep the same quantity as the input.

B

Deep Bingham Networks for Camera Relocalization

Camera relocalization is the term for determining the 6-DoF rotation and translation parameters of a camera concerning a known 3D world. It is a key technology in enabling a multitude of applications such as augmented reality, autonomous driving, human-computer interaction (HCI) and robot guidance, thanks to its extensive integration in simultaneous localization and mapping (SLAM) [32, 57, 169], structure from motion (SfM) [173, 190], metrology [14] and visual localization [147, 175]. Lying in its core is also a pose estimation problem, so it is again haunted by the problems of uncertainty and ambiguity . To support our claim that our method is generic and can be easily extended, we further evaluate Deep Bingham Networks on the application of relocalizing cameras.

Modelling Translation

As a camera's pose is defined by its orientation as well as its position, we predict the rotation by our Deep Bingham Network. Further, as they reside in the Euclidean space, we use mixture density networks [19] to incorporate translations. In more detail, for a sample input image $\mathbf{X} \in \mathbb{R}^{W \times H \times 3}$, we obtain a predicted translation $\hat{\mathbf{t}} \in \mathbb{R}^{c=3}$ from a neural network with parameters $\mathbf{\Gamma}$. This prediction is set to the most likely value of a multivariate Gaussian distribution with covariance matrix

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_d^2 \end{bmatrix}_{c \times c}, \tag{B.1}$$

where $\mathbf{\Sigma}$ is predicted by our model. As a result, our model for a unimodal Gaussian is defined as:

$$p_{\mathbf{\Gamma}}(\mathbf{t} \mid \mathbf{X}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{t} - \hat{\mathbf{t}})^\top \mathbf{\Sigma}^{-1}(\mathbf{t} - \hat{\mathbf{t}})\right)}{(2\pi)^{c/2}|\mathbf{\Sigma}|^{1/2}}, \tag{B.2}$$

where $c = 3$ and both $\hat{\mathbf{t}}$ as well as $\mathbf{\Sigma}$ are trained by maximizing its log-likelihood.

Similar to forming a Bingham Mixture Model, we can equally compute a Gaussian Mixture Model with K components and corresponding weights $\boldsymbol{\pi}$, such that $\sum_{j=1}^{K} \pi_j = 1$, to obtain a multi-modal solution. Again both $\hat{\mathbf{t}}$ and $\mathbf{\Sigma}$ as well as $\boldsymbol{\pi}$ are learned by the network and trained by maximizing the log-likelihood of the mixture model. Note that, in this case, the components of $\hat{\mathbf{t}}$ are assumed to be statistically independent within each distribution component. However,

it has been shown that any density function can be approximated up to a certain error by a Gaussian mixture model with underlying kernel function as defined in Eq. (B.2) [19, 138].

Experimental Setup

When evaluating our method we consider two cases: (1) camera relocalization in non-ambiguous scenes, where we aim to not only predict the camera pose, but the posterior of both rotation and translation that can be used to associate each pose with a measure of uncertainty; (2) we create a highly ambiguous environment, where similar-looking images are captured from very different viewpoints. We show the problems current regression methods suffer from in handling such scenarios and in contrast show the merit of our proposed method.

Baselines

We compare our approach to current state-of-the-art direct camera pose regression methods, PoseNet [105] and MapNet [24], that output a single pose prediction. More importantly, we assess our performance against two state-of-the-art approaches, namely BayesianPoseNet [106] and VidLoc [44], that are most related to our work and predict a distribution over the pose space by using dropout and mixture density networks, respectively.

Datasets

In addition to the standard datasets of 7-Scenes [175] and Cambridge Landmarks [108], we created synthetic as well as real datasets, that are specifically designed to contain repetitive structures and allow us to assess the real benefits of our approach. For synthetic data we render table models from 3DWarehouse[1] and create camera trajectories, e.g. a circular movement around the object, such that ambiguous views are ensured to be included in our dataset. Specifically we use a *dining table* and a *round table* model with discrete modes of ambiguities. In addition, we create highly ambiguous real scenes using Google Tango and the graph-based SLAM approach RTAB-Map [119]. We acquire RGB and depth images as well as distinct ground truth camera trajectories for training and testing. We also reconstruct those scenes. However, note that only the RGB images and corresponding camera poses are required to train our model and the reconstructions are used for visualization only. In particular, our training and test sets consist of 2414 and 1326 frames, respectively. Note that during training, our network sees only a single pose label per image.

Metrics

Note that, under ambiguity, the best mode is unlikely to exist. In those cases, as long as we can generate a hypothesis that is close to the Ground Truth (GT), our network is considered successful. For this reason, in addition to the standard metrics and the weighted mode, we will speak of the so-called *Oracle* error, assuming an oracle that can choose the best of all

[1] https://3dwarehouse.sketchup.com/

Tab. B.1. Evaluation in non-ambiguous scenes, displayed is the median rotation and translation error. (Numbers for MapNet on the Cambridge Landmarks dataset are taken from [172]). BPN depicts BayesianPoseNet [24].

Dataset	7-Scenes							Cambridge Landmarks				
[° / m]	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Kings	Hospital	Shop	St. Marys	Street
PoseNet	4.48/0.13	11.3/0.27	13.0/0.17	5.55/0.19	4.75/0.26	5.35/0.23	12.4/0.35	1.04/0.88	3.29/3.2	3.78/0.88	3.32/1.57	25.5/20.3
MapNet	3.25/0.08	11.69/0.27	13.2/0.18	5.15/0.17	4.02/0.22	4.93/0.23	12.08/0.3	1.89/1.07	3.91/1.94	4.22/1.49	4.53/2.0	-
BPN	7.24/0.37	13.7/0.43	12.0/0.31	8.04/0.48	7.08/0.61	7.54/0.58	13.1/ 0.48	4.06/1.74	5.12/2.57	7.54/1.25	8.38/2.11	-
VidLoc	-/0.18	-/0.26	-/0.14	-/0.26	-/0.36	-/0.31	-/0.26	-	-	-	-	-
UBN	4.97/0.1	12.87/0.27	14.05/0.12	7.52/0.2	7.11/0.23	8.25/0.19	13.1/0.28	1.77/0.88	3.71/1.93	4.74/0.8	6.19/1.84	24.08/16.8

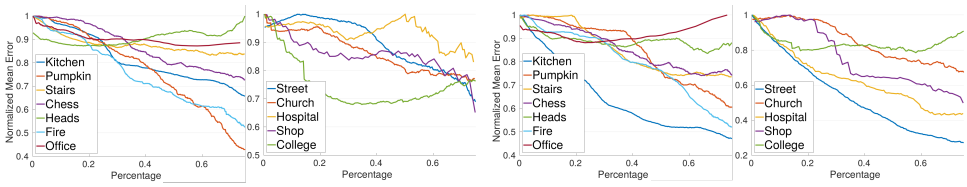
predictions: the one closest to the GT. Also, we report the *Self-EMD* (SEMD) [133] to measure the diversity of the predictions, the same as in the application of point cloud pose estimation.

Evaluation in Non-ambiguous Scenes

We first evaluate our method on the publicly available 7-Scenes [175] and Cambridge Landmarks [108] datasets. As most of the scenes contained in these datasets do not show highly ambiguous environments, we consider them to be non-ambiguous. Though, we can not guarantee that some ambiguous views might arise in these datasets as well, such as in the *Stairs* scene of the 7-Scenes dataset. Both datasets have extensively been used to evaluate camera pose estimation methods. Following the state-of-the-art, we report the median rotation and translation errors, the results of which can be found in Table B.1. In comparison to methods that output a single pose prediction (e.g. PoseNet [105] and MapNet [24]), our methods achieves similar results. Yet, our network provides an additional piece of information that is the uncertainty. On the other hand, especially in translation, our method outperforms uncertainty methods, namely BayesianPoseNet [106] and VidLoc [44], on most scenes.

Uncertainty evaluation

One benefit of our method is that we can use the resulting variance of the predicted distribution as a measure of uncertainty in our predictions. The resulting correlation between pose error



(a) Rot. Uncertainty (b) Trans. Uncertainty

Fig. B.1. Uncertainty evaluation on the 7-Scenes and Cambridge Landmarks datasets, showing the correlation between predicted uncertainty and pose error. Based on the entropy of our predicted distribution uncertain samples are gradually removed. We observe that as we remove the uncertain samples the overall error drops indicating a strong correlation between our predictions and the actual erroneous estimations.

Tab. B.2. % correct poses on our ambiguous scenes for several thresholds. We report the results of our MBN as MBN-M, where M is the number of hypothesis used.

	Threshold	PoseNet [108]	MC-Dropout [106]	UBN	MBN-MB	MBN-CE	MBN-5	MBN-10	MBN-25	MBN-50
Blue Chairs (A)	10° / 0.1m	0.19	0.39	0.29	0.24	0.35	0.40	**0.48**	0.35	0.39
	15° / 0.2m	0.69	0.78	0.73	0.75	0.81	0.85	**0.92**	0.80	0.79
	20° / 0.3m	0.90	0.88	0.86	0.80	0.82	0.89	**0.96**	0.87	0.85
Meeting Table (B)	10° / 0.1m	0.0	0.04	0.02	0.01	0.05	0.03	0.07	**0.08**	0.03
	15° / 0.2m	0.05	0.13	0.12	0.07	0.28	0.26	**0.33**	0.31	0.32
	20° / 0.3m	0.10	0.22	0.19	0.10	0.39	0.34	**0.42**	0.38	0.41
Staircase (C)	10° / 0.1m	0.14	0.13	0.11	0.04	0.18	**0.20**	0.18	0.18	0.17
	15° / 0.2m	0.45	0.32	0.48	0.15	**0.50**	0.47	0.47	0.47	0.49
	20° / 0.3m	0.60	0.49	0.62	0.25	**0.68**	0.64	0.66	0.64	0.64
Staircase Extended (D)	10° / 0.1m	0.07	0.02	0.06	0.06	0.09	0.10	0.06	0.09	**0.11**
	15° / 0.2m	0.31	0.14	0.26	0.21	0.39	0.43	0.38	0.44	**0.46**
	20° / 0.3m	0.49	0.31	0.41	0.32	0.58	0.59	0.60	0.62	**0.64**
Seminar Room (E)	10° / 0.1m	0.37	0.18	0.11	0.06	0.35	**0.39**	0.38	0.35	0.37
	15° / 0.2m	0.81	0.58	0.36	0.23	0.83	0.77	0.78	**0.84**	0.80
	20° / 0.3m	0.90	0.78	0.57	0.40	**0.95**	0.88	0.93	0.94	0.92
Average	10° / 0.1m	0.15	0.15	0.12	0.08	0.20	0.23	**0.24**	0.21	0.22
	15° / 0.2m	0.46	0.39	0.39	0.28	0.56	0.56	**0.58**	0.57	0.57
	20° / 0.3m	0.60	0.54	0.53	0.37	0.68	0.67	**0.71**	0.69	0.69

Tab. B.3. Average % correct oracle poses on our ambiguous scenes for several thresholds.

Threshold	MC-Dropout Oracle	MBN-Oracle-5	MBN-Oracle-10	MBN-Oracle-25	MBN-Oracle-50
10° / 0.1m	**0.28**	0.26	0.27	0.27	0.26
15° / 0.2m	0.60	0.56	0.60	**0.67**	**0.67**
20° / 0.3m	0.70	0.69	0.75	0.80	**0.84**

and uncertainty can be seen in Fig. B.1, where we gradually remove the most uncertain predictions and plot the mean error for the remaining samples. The strong inverse correlation between the actual errors vs our confidence shows that whenever our algorithm labels a prediction as uncertain it is also likely to be a bad estimate.

It has been shown that current direct camera pose regression methods still have difficulties in generalizing to views that differ significantly from the camera trajectories seen during training [172]. However, we chose to focus on another problem these methods have to face and analyze the performance of direct regression methods in a highly ambiguous environment. In this scenario, even similar trajectories can confuse the network and easily lead to wrong predictions, for which our method proposes a solution.

Evaluation in ambiguous scenes

We start with quantitative evaluations on our synthetic as well as real scenes before showing qualitative results of our and the baseline methods.

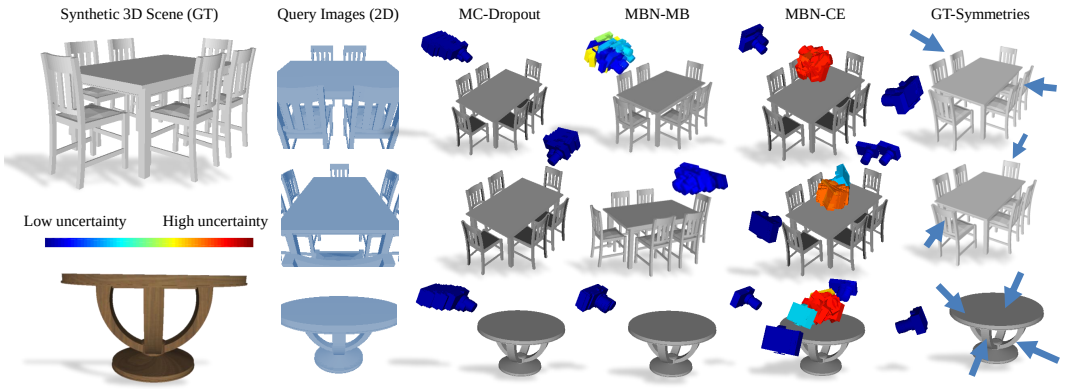


Fig. B.2. Qualitative results on our synthetic *dining* and *round table* datasets. Camera poses are colored by uncertainty. Viewpoints are adjusted for best perception.

Tab. B.4. SEMD of our method and MC-Dropout indicating highly diverse predictions by our method in comparison to the baseline.

Method/Scene	A	B	C	D	E
MC-Dropout	0.06	0.11	0.13	0.26	0.10
MBN-CE	1.19	2.13	2.04	3.81	1.70
MBN	**1.20**	**2.53**	**2.24**	**4.35**	**2.22**

Quantitative evaluations

Due to the design of our synthetic table scenes, we know that there are two and four possible modes for each image in *dining* and *round* table scenes respectively. Hence, we analyze the predictions of our model by computing the accuracy of correctly detected modes of the true posterior. A mode is considered as found if there exists one pose hypothesis that falls into a certain rotational (5°) and translational (10% of the diameter of GT camera trajectory) threshold of it. In the dining table, MC-Dropout obtains an accuracy of 50%, finding one mode for each image, whereas the accuracy of our method on average achieves 96%. On the round table, our model shows an average detection rate of 99.1%, in comparison to 24.8% of MC-Dropout.

On our real scenes, we report the recall, where a pose is considered to be correct if both the rotation and translation errors are below a pre-defined threshold. Table B.2 shows the accuracy of our baseline methods in comparison to ours for various thresholds. Especially on our *Meeting Table* scene, it can be seen that the performance of direct camera pose regression methods that suffer from mode collapse drops significantly due to the presence of ambiguities in the scene. Thanks to the diverse mode predictions of our MBNs, which is indicated by the high Oracle accuracy, see Table B.3, as well as the high SEMD shown in Table B.4, we are able to improve upon our baselines predictions.

Tab. B.5. % correctly detected modes for various translational thresholds (in meters).

Scene	Method	0.1	0.2	0.3	0.4
A	MC-Dropout	0.11	0.15	0.16	0.16
	MBN-CE	0.36	**0.79**	**0.80**	**0.80**
	MBN	**0.42**	0.76	0.77	0.77
B	MC-Dropout	0.04	0.07	0.09	0.11
	MBN-CE	0.10	**0.43**	**0.63**	**0.73**
	MBN	**0.12**	0.41	0.60	**0.73**

Further, by a semi-automatic labeling procedure, we can extract GT modes for the *Blue Chairs* and *Meeting Table* scenes. This way, we can evaluate the entire set of predictions against the GT. Table B.5 shows the percentage of correctly detected modes for our method in comparison to MC-Dropout when evaluating with these GT modes. The results support our qualitative observations, that MC-Dropout suffers from mode collapse such that even with increasing threshold the number of detected modes does not increase significantly.

Qualitative evaluations

Qualitative results of our proposed model on our synthetic table datasets are shown in Fig. B.2. MC-Dropout as well as our finite mixture model, *MBN-MB*, suffer from mode collapse. In comparison, the proposed MHP model is able to capture plausible, but diverse modes as well

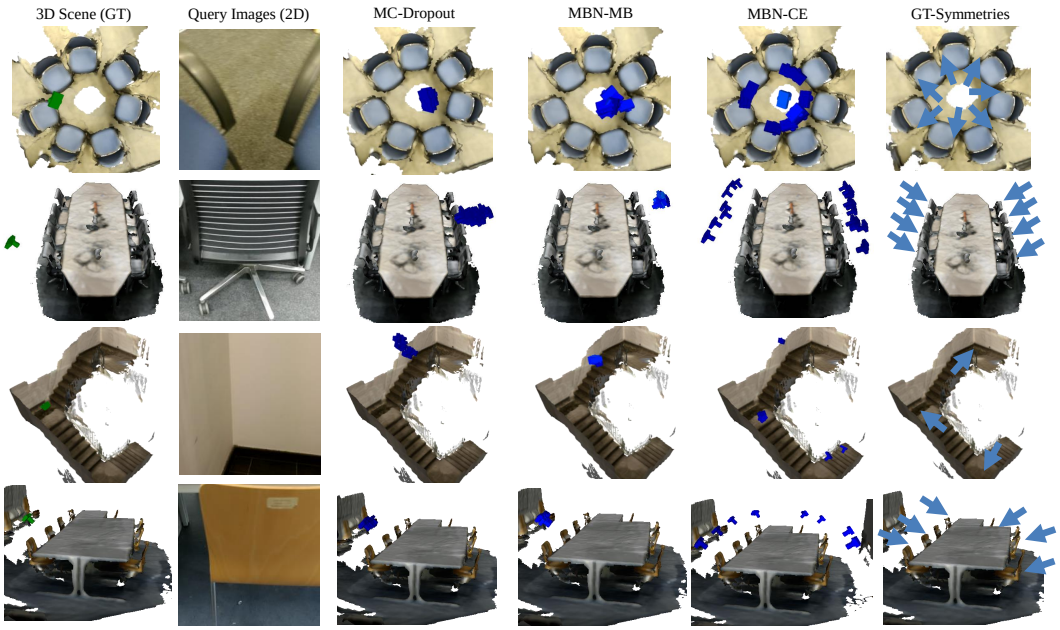


Fig. B.3. Qualitative results in our ambiguous dataset. For better visualization, camera poses have been pruned by their uncertainty values.

Tab. B.6. Averaged percentage of correct poses for different backbone networks over all scenes.

	Threshold	PoseNet	UBN	MBN-MB	MC-Dropout	MBN-CE	MBN
ResNet-34	10° / 0.1m	0.15	0.12	0.08	0.15	0.20	**0.22**
	15° / 0.2m	0.46	0.39	0.28	0.39	0.56	**0.57**
	20° / 0.3m	0.60	0.53	0.37	0.54	0.68	**0.69**
Inception-v3	10° / 0.1m	0.11	0.10	0.11	0.08	**0.18**	0.17
	15° / 0.2m	0.38	0.33	0.38	0.31	0.49	**0.50**
	20° / 0.3m	0.55	0.53	0.52	0.49	**0.63**	**0.63**

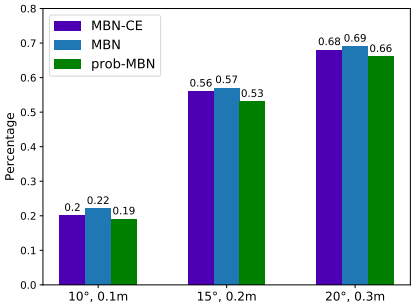


Fig. B.4. Influence on the choice of the best hypothesis our MHP training scheme. We compare between l_1 (MBN-CE and MBN) and choosing the best branch according to the probability (prob-MBN).

as associated uncertainties. In contrast to other methods that obtain an uncertainty value for one prediction, we obtain uncertainty values for each hypothesis. This way, we could easily remove non-meaningful predictions, that for example can arise in the RWTA training schemes.

Fig. B.3 shows qualitative results on our ambiguous real scenes. Again, MC-Dropout and MBN-MB suffer from mode collapse. Moreover, these methods are unable to make reasonable predictions given highly ambiguous query images. Most profoundly, in our *Meeting Table* scene, the predicted camera poses fall on the opposite side of the GT one due to the symmetric structure.

Backbone network

To evaluate the effect of different network architectures on our model, we change the backbone network of ours and the SoTA baseline methods. As most of the recent SoTA image-based localization methods [7, 24, 145] use a version of ResNet, we compare between ResNet variants: ResNet-18, ResNet-34 and ResNet-50 and Inception-v3 [184]. All networks are initialized from an ImageNet [53] pre-trained model. We report our findings in Table B.6. When comparing the performance of different ResNet variants all networks showed on average similar accuracy. Therefore, we only report the results of ResNet-34 in Table B.6. Naturally, all methods are slightly dependant on the features that serve as input to the final pose regression

layers. However, regardless of the backbone network used, our MBN shows, on average, superior performance over the baseline methods.

Choice of best branch

The choice of the best branch in multiple hypothesis predictions depends on the chosen distance function comparing the prediction to the ground truth label.

In this work, we compare between the l_1 norm (see Eq. (8.21)) and choosing the branch with highest probability density (see Eq. (8.22)), and report the results in Fig. B.4. For camera re-localization, we have found l_1 to slightly outperform probability based choices.

C

List of Authored and Co-authored Publications

Authored

1. Haowen Deng, Tolga Birdal, Slobodan Ilic, "*PPFNet: Global Context Aware Local Features for Robust 3D Point Matching*", In IEEE Computer Vision and Pattern Recognition (CVPR), Salt Lake City, United States, June 2018

2. Haowen Deng, Tolga Birdal, Slobodan Ilic, "*PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors*", In European Conference On Computer Vision (ECCV), Munich, Germany, September 2018

3. Haowen Deng, Tolga Birdal, Slobodan Ilic, "*3D Local Features for Direct Pairwise Registration*", In IEEE Computer Vision and Pattern Recognition (CVPR), Long Beach, United States, June 2019

4. Haowen Deng, Mai Bui, Yongheng Zhao, Leonidas Guibas, Slobodan Ilic, Tolga Birdal, "*Deep Bingham Networks: Dealing with Uncertainty and Ambiguity in Pose Estimation*", 2020, [under submission]

Co-authored

1. Yongheng Zhao, Tolga Birdal, Haowen Deng, Federico Tombari, "*3D Point Capsule Networks*", In IEEE Computer Vision and Pattern Recognition (CVPR), Long Beach, United States, June 2019

2. Mai Bui, Tolga Birdal, Haowen Deng, Shadi Albarqouni, Leonidas Guibas, Slobodan Ilic, Nassir Navab, "*6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference*", In European Conference On Computer Vision (ECCV), Glasgow, United Kingdom, August 2020

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, et al. “Tensorflow: Large-scale machine learning on heterogeneous distributed systems”. In: *arXiv preprint arXiv:1603.04467* (2016) (cit. on pp. 33, 50).

[2] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. “Learning Representations and Generative Models for 3D Point Clouds”. In: *International Conference on Machine Learning (ICML)*. 2018 (cit. on p. 24).

[3] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey. “PointNetLK: Robust & efficient point cloud registration using PointNet”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7163–7172 (cit. on pp. 99, 101).

[4] K. S. Arun, T. S. Huang, and S. D. Blostein. “Least-squares fitting of two 3-D point sets”. In: *IEEE Transactions on pattern analysis and machine intelligence* 5 (1987), pp. 698–700 (cit. on p. 17).

[5] R. Arun Srivatsan, M. Xu, N. Zevallos, and H. Choset. “Probabilistic pose estimation using a Bingham distribution-based linear filter”. In: *The International Journal of Robotics Research* 37.13-14 (2018), pp. 1610–1631 (cit. on p. 90).

[6] V. Badrinarayanan, A. Kendall, and R. Cipolla. “Segnet: A deep convolutional encoder-decoder architecture for image segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495 (cit. on p. 3).

[7] V. Balntas, S. Li, and V. Prisacariu. “Relocnet: Continuous metric learning relocalisation using neural nets”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 751–767 (cit. on p. 127).

[8] T. D. Barfoot and P. T. Furgale. “Associating uncertainty with three-dimensional poses for use in estimation problems”. In: *IEEE Transactions on Robotics* 30.3 (2014), pp. 679–693 (cit. on p. 92).

[9] H. Bay, T. Tuytelaars, and L. Van Gool. “Surf: Speeded up robust features”. In: *European conference on computer vision*. Springer. 2006, pp. 404–417 (cit. on pp. 4, 23).

[10] J. O. Berger. *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013 (cit. on p. 67).

[11] P. J. Besl and N. D. McKay. “Method for registration of 3-D shapes”. In: *Robotics-DL tentative*. International Society for Optics and Photonics. 1992, pp. 586–606 (cit. on pp. 5, 67, 84).

[12] C. Bingham. “An antipodally symmetric distribution on the sphere”. In: *The Annals of Statistics* (1974), pp. 1201–1225 (cit. on pp. 67, 90, 91, 107).

[13] T. Birdal and S. Ilic. “CAD Priors for Accurate and Flexible Instance Reconstruction”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 133–142 (cit. on pp. 5, 24, 46, 67, 89).

[14] T. Birdal, E. Bala, T. Eren, and S. Ilic. “Online inspection of 3d parts via a locally overlapping camera network”. In: *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2016, pp. 1–10 (cit. on pp. 67, 121).

[15] T. Birdal and S. Ilic. “A Point Sampling Algorithm for 3D Matching of Irregular Geometries”. In: *International Conference on Intelligent Robots and Systems (IROS 2017)*. IEEE. 2017, pp. 6871–6878 (cit. on pp. 32, 57, 79).

[16] T. Birdal and S. Ilic. “Point Pair Features Based Object Detection and Pose Estimation Revisited”. In: *3D Vision*. IEEE. 2015, pp. 527–535 (cit. on pp. 14, 24, 46, 47).

[17] T. Birdal and U. Simsekli. “Probabilistic Permutation Synchronization using the Riemannian Structure of the Birkhoff Polytope”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 11105–11116 (cit. on p. 90).

[18] T. Birdal, U. Simsekli, M. O. Eken, and S. Ilic. “Bayesian Pose Graph Optimization via Bingham Distributions and Tempered Geodesic MCMC”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 308–319 (cit. on pp. 75, 90, 92, 105).

[19] C. M. Bishop. “Mixture density networks”. In: (1994) (cit. on pp. 7, 67, 68, 96–98, 104, 121, 122).

[20] D. Bobkov, S. Chen, R. Jian, M. Z. Iqbal, and E. Steinbach. “Noise-Resistant Deep Learning for Object Classification in Three-Dimensional Point Clouds Using a Point Pair Descriptor”. In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 865–872 (cit. on p. 54).

[21] E. Brachmann, A. Krull, S. Nowozin, et al. “DSAC-differentiable RANSAC for camera localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6684–6692 (cit. on p. 66).

[22] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, et al. “Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3364–3372 (cit. on p. 67).

[23] E. Brachmann and C. Rother. “Learning less is more-6d camera localization via 3d surface regression”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4654–4662 (cit. on pp. 5, 66).

[24] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. “Geometry-aware learning of maps for camera localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2616–2625 (cit. on pp. 105, 122, 123, 127).

[25] L. Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32 (cit. on p. 67).

[26] A. G. Buch, H. G. Petersen, and N. Krüger. “Local shape feature fusion for improved matching, pose estimation and 3D object recognition”. In: *SpringerPlus* 5.1 (2016), p. 297 (cit. on p. 66).

[27] M. Bui, S. Albarqouni, S. Ilic, and N. Navab. “Scene Coordinate and Correspondence Learning for Image-Based Localization”. In: *British Machine Vision Conference (BMVC)*. 2018 (cit. on p. 66).

[28] M. Bui, C. Baur, N. Navab, S. Ilic, and S. Albarqouni. “Adversarial Networks for Camera Pose Regression and Refinement”. In: 2019 (cit. on pp. 5, 66).

[29] M. Bui, T. Birdal, H. Deng, et al. “6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference”. In: *arXiv preprint arXiv:2004.04807* (2020) (cit. on p. 92).

[30] B. Busam, T. Birdal, and N. Navab. “Camera Pose Filtering with Local Regression Geodesics on the Riemannian Manifold of Dual Quaternions”. In: *IEEE International Conference on Computer Vision Workshop (ICCVW)*. Oct. 2017, pp. 2436–2445 (cit. on pp. 75, 92).

[31] Á. P. Bustos and T.-J. Chin. “Guaranteed outlier removal for point cloud registration with correspondences”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.12 (2018), pp. 2868–2882 (cit. on pp. 5, 67).

[32] C. Cadena, L. Carlone, H. Carrillo, et al. “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age”. In: *IEEE Transactions on robotics* 32.6 (2016), pp. 1309–1332 (cit. on p. 121).

[33] D. Campbell, L. Petersson, L. Kneip, and H. Li. “Globally-Optimal Inlier Set Maximisation for Simultaneous Camera Pose and Feature Correspondence”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on p. 35).

[34] A. X. Chang, T. Funkhouser, L. Guibas, et al. “Shapenet: An information-rich 3d model repository”. In: *arXiv preprint arXiv:1512.03012* (2015) (cit. on p. 89).

[35] T.-J. Chin, P. Purkait, A. Eriksson, and D. Suter. “Efficient globally optimal consensus maximisation with tree search”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 2413–2421 (cit. on p. 35).

[36] C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam. “Voting-based pose estimation for robotic assembly using a 3D sensor”. In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE. 2012, pp. 1724–1731 (cit. on p. 23).

[37] S. Choi, Q.-Y. Zhou, and V. Koltun. “Robust Reconstruction of Indoor Scenes”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on pp. 71, 81, 84, 87).

[38] S. Choi, T. Kim, and W. Yu. “Performance evaluation of RANSAC family”. In: *Journal of Computer Vision* 24.3 (1997), pp. 271–300 (cit. on pp. 5, 67).

[39] C. Choy, J. Gwak, and S. Savarese. “4d spatio-temporal convnets: Minkowski convolutional neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3075–3084 (cit. on p. 113).

[40] C. Choy, J. Park, and V. Koltun. “Fully Convolutional Geometric Features”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 8958–8966 (cit. on p. 113).

[41] O. Chum and J. Matas. “Matching with PROSAC-progressive sample consensus”. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 220–226 (cit. on pp. 5, 67).

[42] O. Chum and J. Matas. “Optimal randomized RANSAC”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.8 (2008), pp. 1472–1482 (cit. on pp. 5, 67, 69, 78–80).

[43] O. Chum, J. Matas, and J. Kittler. “Locally optimized RANSAC”. In: *Joint Pattern Recognition Symposium*. Springer. 2003, pp. 236–243 (cit. on pp. 5, 67, 69).

[44] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. “Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6856–6864 (cit. on pp. 67, 122, 123).

[45] T. Cohen and M. Welling. “Group equivariant convolutional networks”. In: *International conference on machine learning*. 2016, pp. 2990–2999 (cit. on p. 72).

[46] E. Corona, K. Kundu, and S. Fidler. “Pose Estimation for Objects with Rotational Symmetry”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 7215–7222 (cit. on pp. 68, 90).

[47] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05)*. Vol. 1. IEEE. 2005, pp. 886–893 (cit. on p. 4).

[48] M. Defferrard, X. Bresson, and P. Vandergheynst. “Convolutional neural networks on graphs with fast localized spectral filtering”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 3844–3852 (cit. on p. 25).

[49] H. Deng, T. Birdal, and S. Ilic. “3D Local Features for Direct Pairwise Registration”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019 (cit. on pp. 5, 66, 67).

[50] H. Deng, T. Birdal, and S. Ilic. “PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors”. In: *The European Conference on Computer Vision (ECCV)*. Sept. 2018, pp. 602–618 (cit. on pp. 5, 66, 71, 73, 75, 77, 83).

[51] H. Deng, T. Birdal, and S. Ilic. “Ppfnet: Global context aware local features for robust 3d point matching”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 195–205 (cit. on pp. 5, 66, 72, 77, 78).

[52] H. Deng, T. Birdal, and S. Ilic. “Ppfnet: Global context aware local features for robust 3d point matching”. In: *Computer Vision and Pattern Recognition (CVPR). IEEE* 1 (2018) (cit. on pp. 45, 46, 52, 53).

[53] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on pp. 3, 5, 19, 45, 66, 127).

[54] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018) (cit. on p. 112).

[55] D. Dey, V. Ramakrishna, M. Hebert, and J. Andrew Bagnell. “Predicting multiple structured visual interpretations”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2947–2955 (cit. on p. 68).

[56] B. Drost, M. Ulrich, N. Navab, and S. Ilic. “Model globally, match locally: Efficient and robust 3D object recognition”. In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. Ieee. 2010, pp. 998–1005 (cit. on pp. 24, 28).

[57] H. Durrant-Whyte and T. Bailey. “Simultaneous localization and mapping: part I”. In: *IEEE robotics & automation magazine* 13.2 (2006), pp. 99–110 (cit. on pp. 4, 121).

[58] B. Eckart, K. Kim, and J. Kautz. “HGMR: Hierarchical Gaussian Mixtures for Adaptive 3D Registration”. In: *The European Conference on Computer Vision (ECCV)*. Sept. 2018 (cit. on pp. 5, 67).

[59] D. W. Eggert, A. Lorusso, and R. B. Fisher. “Estimating 3-D rigid body transformations: a comparison of four major algorithms”. In: *Machine vision and applications* 9.5-6 (1997), pp. 272–290 (cit. on p. 18).

[60] G. Elbaz, T. Avraham, and A. Fischer. “3D Point Cloud Registration for Localization Using a Deep Neural Network Auto-Encoder”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017 (cit. on p. 25).

[61] L. Falorsi, P. de Haan, T. R. Davidson, and P. Forré. “Reparameterizing Distributions on Lie Groups”. In: *arXiv preprint arXiv:1903.02958* (2019) (cit. on p. 92).

[62] W. Feng, F.-P. Tian, Q. Zhang, and J. Sun. “6d dynamic camera relocalization from single reference image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 4049–4057 (cit. on p. 66).

[63] M. A. Fischler and R. C. Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395 (cit. on pp. 18, 67).

[64] M. A. Fischler and R. C. Bolles. “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. In: *Commun. ACM* (1981) (cit. on pp. 5, 66).

[65] Y. Gal and Z. Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *international conference on machine learning*. 2016, pp. 1050–1059 (cit. on p. 68).

[66] R. Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448 (cit. on p. 3).

[67] J. Glover and L. P. Kaelbling. “Tracking the spin on a ping pong ball with the quaternion Bingham filter”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 4133–4140 (cit. on pp. 68, 92).

[68] J. Glover, G. Bradski, and R. B. Rusu. “Monte carlo pose estimation with quaternion kernels and the bingham distribution”. In: *Robotics: science and systems*. Vol. 7. 2012, p. 97 (cit. on pp. 68, 92).

[69] I. Goodfellow, J. Pouget-Abadie, M. Mirza, et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680 (cit. on p. 72).

[70] B. Graham. “Sparse 3D convolutional neural networks”. In: *arXiv preprint arXiv:1505.02890* (2015) (cit. on p. 113).

[71] F. S. Grassia. “Practical parameterization of rotations using the exponential map”. In: *Journal of graphics tools* 3.3 (1998), pp. 29–48 (cit. on p. 92).

[72] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry. “AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018 (cit. on p. 119).

[73] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. “On calibration of modern neural networks”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1321–1330 (cit. on p. 67).

[74] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok. “A comprehensive performance evaluation of 3D local feature descriptors”. In: *International Journal of Computer Vision* 116.1 (2016), pp. 66–89 (cit. on pp. 4, 24).

[75] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and J. Zhang. “Performance evaluation of 3D local feature descriptors”. In: *Asian Conference on Computer Vision*. Springer. 2014, pp. 178–194 (cit. on pp. 4, 23, 24, 69).

[76] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan. “Rotational projection statistics for 3D local surface description and object recognition”. In: *International journal of computer vision* 105.1 (2013), pp. 63–86 (cit. on pp. 4, 24).

[77] Y. Guo, F. A. Sohel, M. Bennamoun, J. Wan, and M. Lu. “RoPS: A local feature descriptor for 3D rigid objects based on rotational projection statistics”. In: *Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on*. IEEE. 2013, pp. 1–6 (cit. on pp. 4, 23).

[78] A. Guzman-Rivera, D. Batra, and P. Kohli. “Multiple choice learning: Learning to produce multiple structured outputs”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 1799–1807 (cit. on p. 68).

[79] A. Haarbach, T. Birdal, and S. Ilic. “Survey of Higher Order Rigid Body Motion Interpolation Methods for Keyframe Animation and Continuous-Time Trajectory Estimation”. In: *3D Vision (3DV), 2018 Sixth International Conference on*. IEEE. 2018, pp. 381–389 (cit. on p. 92).

[80] R. Hadsell, S. Chopra, and Y. LeCun. “Dimensionality reduction by learning an invariant mapping”. In: *Computer vision and pattern recognition, 2006 IEEE computer society conference on*. Vol. 2. IEEE. 2006, pp. 1735–1742 (cit. on pp. 28, 39).

[81] M. Halber and T. Funkhouser. “Fine-To-Coarse Global Registration of RGB-D Scans”. In: (2017) (cit. on p. 34).

[82] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. “Matchnet: Unifying feature and metric learning for patch-based matching”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3279–3286 (cit. on p. 27).

[83] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on pp. 3, 5, 66, 67).

[84] C. S. Herz. “Bessel Functions of Matrix Argument”. In: *Annals of Mathematics* 61.3 (1955), pp. 474–523 (cit. on p. 91).

[85] S. Hinterstoisser, V. Lepetit, S. Ilic, et al. “Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes”. In: *Asian conference on computer vision*. Springer. 2012, pp. 548–562 (cit. on pp. 66, 89).

[86] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige. “Going further with point pair features”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 834–848 (cit. on p. 24).

[87] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas. “Detection and fine 3D pose estimation of texture-less objects in RGB-D images”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 4421–4428 (cit. on p. 66).

[88] E. Hoffer and N. Ailon. “Deep metric learning using triplet network”. In: *International Workshop on Similarity-Based Pattern Recognition*. Springer. 2015, pp. 84–92 (cit. on p. 39).

[89] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. *Surface reconstruction from unorganized points*. Vol. 26.2. ACM, 1992 (cit. on pp. 13, 29, 32).

[90] R. Horaud, B. Conio, O. Leboulleux, and B. Lacolle. “An analytic solution for the perspective 4-point problem”. In: *Proceedings CVPR’89: IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 1989, pp. 500–507 (cit. on p. 89).

[91] B. K. Horn. “Closed-form solution of absolute orientation using unit quaternions”. In: *Josa a* 4.4 (1987), pp. 629–642 (cit. on p. 17).

[92] B. K. Horn, H. M. Hilden, and S. Negahdaripour. “Closed-form solution of absolute orientation using orthonormal matrices”. In: *JOSA A* 5.7 (1988), pp. 1127–1135 (cit. on p. 17).

[93] J. Hou, A. Dai, and M. Nießner. “3d-sis: 3d semantic instance segmentation of rgb-d scans”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4421–4430 (cit. on p. 112).

[94] H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V. G. Kim, and E. Yumer. “Learning local shape descriptors with view-based convolutional networks”. In: *CoRR* abs/1706.04496 (2017). arXiv: 1706.04496 (cit. on p. 25).

[95] J. Huang and S. You. “Point cloud labeling using 3d convolutional neural network”. In: *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE. 2016, pp. 2670–2675 (cit. on p. 25).

[96] F. Järemo Lawin, M. Danelljan, F. Shahbaz Khan, P.-E. Forssén, and M. Felsberg. “Density Adaptive Point Set Registration”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018 (cit. on pp. 5, 67).

[97] E. T. Jaynes. “Information theory and statistical mechanics”. In: *Physical review* 106.4 (1957), p. 620 (cit. on p. 95).

[98] Z. Jian Yew and G. Hee Lee. “3DFeat-Net: Weakly Supervised Local 3D Features for Point Cloud Registration”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 607–623 (cit. on p. 112).

[99] A. E. Johnson and M. Hebert. “Using spin images for efficient object recognition in cluttered 3D scenes”. In: *IEEE Transactions on pattern analysis and machine intelligence* 21.5 (1999), pp. 433–449 (cit. on pp. 4, 23, 33, 35, 37, 51–53).

[100] W. Kabsch. “A discussion of the solution for the best rotation to relate two sets of vectors”. In: *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 34.5 (1978), pp. 827–828 (cit. on p. 17).

[101] W. Kabsch. “A solution for the best rotation to relate two sets of vectors”. In: *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), pp. 922–923 (cit. on pp. 5, 17, 65, 66).

[102] A. Kanezaki, Y. Matsushita, and Y. Nishida. “Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5010–5019 (cit. on pp. 5, 66).

[103] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. “Large-scale video classification with convolutional neural networks”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2014, pp. 1725–1732 (cit. on p. 19).

[104] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. “SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1521–1529 (cit. on pp. 5, 66).

[105] A. Kendall and R. Cipolla. “Geometric loss functions for camera pose regression with deep learning”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5974–5983 (cit. on pp. 5, 66, 99, 122, 123).

[106] A. Kendall and R. Cipolla. “Modelling uncertainty in deep learning for camera relocalization”. In: *2016 IEEE international conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 4762–4769 (cit. on pp. 5, 66, 67, 90, 122–124).

[107] A. Kendall and Y. Gal. “What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?” In: *Advances in Neural Information Processing Systems (NIPS)*. 2017 (cit. on p. 99).

[108] A. Kendall, M. Grimes, and R. Cipolla. “PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization”. In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2015 (cit. on pp. 5, 65–68, 89, 105, 122–124).

[109] M. Khoury, Q.-Y. Zhou, and V. Koltun. “Learning Compact Geometric Features”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on pp. 23, 25, 30, 34, 35, 37, 41, 45, 46, 51–53, 69, 77, 78, 81).

[110] L. Kiforenko, B. Drost, F. Tombari, N. Krüger, and A. G. Buch. “A performance evaluation of point pair features”. In: *Computer Vision and Image Understanding* (2017) (cit. on pp. 4, 23, 24).

[111] D. Kinga and J. B. Adam. “A method for stochastic optimization”. In: *International Conference on Learning Representations (ICLR)*. 2015 (cit. on pp. 33, 50).

[112] D. P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on p. 99).

[113] T. N. Kipf and M. Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016) (cit. on p. 25).

[114] S. Korman and R. Litman. “Latent RANSAC”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6693–6702 (cit. on pp. 5, 67, 69, 78–81).

[115] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on p. 3).

[116] A. Kume and A. T. Wood. “Saddlepoint approximations for the Bingham and Fisher–Bingham normalising constants”. In: *Biometrika* 92.2 (2005), pp. 465–476 (cit. on p. 91).

[117] G. Kurz, I. Gilitschenski, S. Julier, and U. D. Hanebeck. “Recursive estimation of orientation based on the Bingham distribution”. In: *Information Fusion (FUSION), 2013 16th International Conference on*. IEEE. 2013, pp. 1487–1494 (cit. on pp. 91, 92).

[118] G. Kurz, I. Gilitschenski, F. Pfaff, et al. “Directional Statistics and Filtering Using libDirectional”. In: *arXiv preprint arXiv:1712.09718* (2017) (cit. on p. 91).

[119] M. Labbé and F. Michaud. “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation”. In: *Journal of Field Robotics* 36.2 (2019), pp. 416–446 (cit. on p. 122).

[120] K. Lai, L. Bo, and D. Fox. “Unsupervised feature learning for 3d scene labeling”. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 3050–3057 (cit. on p. 34).

[121] Y. LeCun, Y. Bengio, and G. Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444 (cit. on p. 3).

[122] H. Li and R. Hartley. “The 3D-3D registration problem revisited”. In: *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE. 2007, pp. 1–8 (cit. on pp. 5, 25, 67).

[123] S. Liao, E. Gavves, and C. G. Snoek. “Spherical Regression: Learning Viewpoints, Surface Normals and 3D Rotations on n-Spheres”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9759–9767 (cit. on p. 99).

[124] T.-Y. Lin, M. Maire, S. Belongie, et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755 (cit. on p. 3).

[125] W. Liu, W. Luo, D. Lian, and S. Gao. “Future frame prediction for anomaly detection–a new baseline”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6536–6545 (cit. on p. 68).

[126] Z. Liu, H. Tang, Y. Lin, and S. Han. “Point-Voxel CNN for efficient 3D deep learning”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 963–973 (cit. on p. 113).

[127] J. Long, E. Shelhamer, and T. Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440 (cit. on p. 3).

[128] D. G. Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60.2 (2004), pp. 91–110 (cit. on p. 4).

[129] D. G. Lowe. “Object recognition from local scale-invariant features”. In: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*. Vol. 2. Ieee. 1999, pp. 1150–1157 (cit. on pp. 4, 23).

[130] P. Luc, N. Neverova, C. Couprie, J. Verbeek, and Y. LeCun. “Predicting deeper into the future of semantic segmentation”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 648–657 (cit. on p. 68).

[131] L. v. d. Maaten and G. Hinton. “Visualizing data using t-SNE”. In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605 (cit. on pp. 57, 58).

[132] S. Mahendran, H. Ali, and R. Vidal. “3D pose regression using convolutional neural networks”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2174–2182 (cit. on p. 99).

[133] O. Makansi, E. Ilg, O. Cicek, and T. Brox. “Overcoming Limitations of Mixture Density Networks: A Sampling and Fitting Framework for Multimodal Future Prediction”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7144–7153 (cit. on pp. 68, 95, 97, 100, 104, 123).

[134] F. Manessi, A. Rozza, and M. Manzo. “Dynamic Graph Convolutional Networks”. In: *arXiv preprint arXiv:1704.06199* (2017) (cit. on p. 24).

[135] F. Manhardt, D. M. Arroyo, C. Rupprecht, et al. “Explaining the Ambiguity of Object Detection and 6D Pose from Visual Data”. In: IEEE/CVF. 2019 (cit. on pp. 68, 90, 97).

[136] D. Massiceti, A. Krull, E. Brachmann, C. Rother, and P. H. Torr. “Random forests versus Neural Networks—What’s best for camera localization?” In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5118–5125 (cit. on p. 66).

[137] D. Maturana and S. Scherer. “Voxnet: A 3d convolutional neural network for real-time object recognition”. In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 922–928 (cit. on pp. 11, 19, 24, 25).

[138] G. J. McLachlan and K. E. Basford. *Mixture models: Inference and applications to clustering*. Vol. 84. M. Dekker New York, 1988 (cit. on p. 122).

[139] A. Morawiec and D. Field. “Rodrigues parameterization for orientation and misorientation distributions”. In: *Philosophical Magazine A* 73.4 (1996), pp. 1113–1130 (cit. on p. 92).

[140] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. “ORB-SLAM: a versatile and accurate monocular SLAM system”. In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163 (cit. on p. 23).

[141] R. M. Murray. *A mathematical introduction to robotic manipulation*. CRC press, 1994 (cit. on p. 92).

[142] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. “Large-Scale Image Retrieval With Attentive Deep Local Features”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on pp. 23, 112).

[143] J. Park, Q.-Y. Zhou, and V. Koltun. “Colored point cloud registration revisited”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 143–152 (cit. on pp. 5, 67).

[144] A. Paszke, S. Gross, S. Chintala, et al. “Automatic Differentiation in PyTorch”. In: *NIPS Autodiff Workshop*. 2017 (cit. on pp. 77, 96, 99).

[145] V. Peretroukhin, B. Wagstaff, M. Giamou, and J. Kelly. “Probabilistic Regression of Rotations using Quaternion Averaging and a Deep Multi-Headed Network”. In: *arXiv preprint arXiv:1904.03182* (2019) (cit. on p. 127).

[146] A. Petrelli and L. Di Stefano. “On the repeatability of the local reference frame for partial shape matching”. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2244–2251 (cit. on pp. 70, 71, 73).

[147] N. Piasco, D. Sidibé, C. Demonceaux, and V. Gouet-Brunet. “A survey on visual-based localization: On the benefit of heterogeneous data”. In: *Pattern Recognition* 74 (2018), pp. 90–109 (cit. on p. 121).

[148] G. Pitteri, M. Ramamonjisoa, S. Ilic, and V. Lepetit. “On Object Symmetries and 6D Pose Estimation from Images”. In: IEEE. 2019 (cit. on pp. 68, 90).

[149] S. Prokudin, P. Gehler, and S. Nowozin. “Deep Directional Statistics: Pose Estimation with Uncertainty Quantification”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 534–551 (cit. on pp. 67, 94).

[150] C. R. Qi, O. Litany, K. He, and L. J. Guibas. “Deep Hough Voting for 3D Object Detection in Point Clouds”. In: *arXiv preprint arXiv:1904.09664* (2019) (cit. on pp. 5, 66, 89).

[151] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 652–660 (cit. on pp. 5, 24, 27, 66, 99, 117, 118).

[152] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* 1.2 (2017), p. 4 (cit. on pp. 6, 19, 26, 28, 34, 35, 37, 40, 45, 49).

[153] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. “Volumetric and multi-view cnns for object classification on 3d data”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 5648–5656 (cit. on pp. 13, 25).

[154] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *arXiv preprint arXiv:1706.02413* (2017) (cit. on pp. 5, 24, 66).

[155] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. “Frustum PointNets for 3D Object Detection From RGB-D Data”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018 (cit. on p. 24).

[156] X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun. “3D Graph Neural Networks for RGBD Semantic Segmentation”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on pp. 24, 25).

[157] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J.-M. Frahm. “USAC: a universal framework for random sample consensus.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8 (2013), pp. 2022–2038 (cit. on pp. 5, 67, 69, 78–80).

[158] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788 (cit. on p. 3).

[159] S. Ren, K. He, R. Girshick, and J. Sun. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems*. 2015, pp. 91–99 (cit. on p. 3).

[160] B. Richard. “Dynamic programming”. In: *Princeton University Press* 89 (1957), p. 92 (cit. on p. 3).

[161] G. Riegler, A. O. Ulusoy, H. Bischof, and A. Geiger. “OctNetFusion: Learning Depth Fusion from Data”. In: *arXiv preprint arXiv:1704.01047* (2017) (cit. on p. 12).

[162] G. Riegler, O. Ulusoy, and A. Geiger. “OctNet: Learning Deep 3D Representations at High Resolutions”. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. July 2017 (cit. on pp. 11, 12).

[163] Y. Rubner, C. Tomasi, and L. J. Guibas. “The earth mover’s distance as a metric for image retrieval”. In: *International journal of computer vision* 40.2 (2000), pp. 99–121 (cit. on p. 100).

[164] C. Rupprecht, I. Laina, R. DiPietro, et al. “Learning in an uncertain world: Representing ambiguity through multiple hypotheses”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 3591–3600 (cit. on pp. 7, 68, 90, 97, 104).

[165] O. Russakovsky, J. Deng, H. Su, et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252 (cit. on p. 19).

[166] R. B. Rusu, N. Blodow, and M. Beetz. “Fast point feature histograms (FPFH) for 3D registration”. In: *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on*. IEEE. 2009, pp. 3212–3217 (cit. on pp. 4, 23, 24, 28, 33, 35, 37, 51–54).

[167] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz. “Persistent Point Feature Histograms for 3D Point Clouds”. In: *Proceedings of the 10th International Conference on Intelligent Autonomous Systems (IAS-10), Baden-Baden, Germany*. 2008 (cit. on pp. 4, 23, 24).

[168] S. Sabour, N. Frosst, and G. E. Hinton. “Dynamic routing between capsules”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 3856–3866 (cit. on pp. 117, 118).

[169] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. “Slam++: Simultaneous localisation and mapping at the level of objects”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013, pp. 1352–1359 (cit. on pp. 23, 121).

[170] S. Salti, F. Tombari, and L. Di Stefano. “SHOT: Unique signatures of histograms for surface and texture description”. In: *Computer Vision and Image Understanding* 125 (2014), pp. 251–264 (cit. on pp. 4, 23, 33, 35, 37, 51–53, 71).

[171] T. Sattler, M. Havlena, F. Radenovic, K. Schindler, and M. Pollefeys. “Hyperpoints and fine vocabularies for large-scale location recognition”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2102–2110 (cit. on p. 89).

[172] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. “Understanding the Limitations of CNN-based Absolute Camera Pose Regression”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3302–3312 (cit. on pp. 123, 124).

[173] J. L. Schonberger and J.-M. Frahm. “Structure-from-motion revisited”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 4104–4113 (cit. on pp. 4, 121).

[174] Y. Shen, C. Feng, Y. Yang, and D. Tian. “Neighbors Do Help: Deeply Exploiting Local Structures of Point Clouds”. In: *ArXiv e-prints* (Dec. 2017). arXiv: 1712.06760 [cs.CV] (cit. on p. 24).

[175] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. “Scene coordinate regression forests for camera relocalization in RGB-D images”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2930–2937 (cit. on pp. 28, 34, 35, 37, 52, 53, 66, 67, 85, 87, 121–123).

[176] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *CoRR* abs/1409.1556 (2014) (cit. on pp. 3, 67).

[177] I. Sipiran and B. Bustos. “Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes”. In: *The Visual Computer* 27.11 (2011), p. 963 (cit. on p. 112).

[178] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng. “Convolutional-recursive deep learning for 3d object classification”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 656–664 (cit. on p. 25).

[179] S. Song and J. Xiao. “Deep sliding shapes for amodal 3D object detection in RGB-D images”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 808–816 (cit. on p. 25).

[180] N. Srivastava, H. Goh, and R. Salakhutdinov. “Geometric Capsule Autoencoders for 3D Point Clouds”. In: *arXiv preprint arXiv:1912.03310* (2019) (cit. on p. 113).

[181] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. “Multi-view convolutional neural networks for 3d shape recognition”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 945–953 (cit. on pp. 13, 19).

[182] Y. Sun, D. Liang, X. Wang, and X. Tang. “DeepID3: face recognition with very deep neural networks. CVPR”. In: (2015) (cit. on p. 3).

[183] C. Szegedy, W. Liu, Y. Jia, et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9 (cit. on p. 3).

[184] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826 (cit. on p. 127).

[185] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. “Deepface: Closing the gap to human-level performance in face verification”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1701–1708 (cit. on p. 3).

[186] M. Tatarchenko, A. Dosovitskiy, and T. Brox. “Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs”. In: *CoRR* abs/1703.09438 (2017) (cit. on p. 12).

[187] R. Toldo, A. Beinat, and F. Crosilla. “Global registration of multiple point clouds embedding the Generalized Procrustes Analysis into an ICP framework”. In: *3DPVT 2010 Conference*. 2010 (cit. on pp. 5, 67).

[188] F. Tombari, S. Salti, and L. Di Stefano. “Unique shape context for 3D data description”. In: *Proceedings of the ACM workshop on 3D object retrieval*. ACM. 2010, pp. 57–62 (cit. on pp. 4, 23, 33, 35, 37).

[189] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. “Learning spatiotemporal features with 3d convolutional networks”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 4489–4497 (cit. on p. 19).

[190] S. Ullman. “The interpretation of structure from motion”. In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), pp. 405–426 (cit. on p. 121).

[191] S. Umeyama. “Least-squares estimation of transformation parameters between two point patterns”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 4 (1991), pp. 376–380 (cit. on p. 17).

[192] J. Valentin, A. Dai, M. Nießner, et al. “Learning to navigate the energy landscape”. In: *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE. 2016, pp. 323–332 (cit. on p. 34).

[193] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. Torr. “Exploiting uncertainty in regression forests for accurate camera relocalization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 4400–4408 (cit. on p. 67).

[194] L. van der Maaten. “Barnes-Hut-SNE”. In: *ArXiv e-prints* (Jan. 2013). arXiv: 1301.3342 [cs.LG] (cit. on pp. 57, 58).

[195] J. Vongkulbhisal, B. I. Ugalde, F. De la Torre, and J. P. Costeira. “Inverse Composition Discriminative Optimization for Point Cloud Registration”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2993–3001 (cit. on pp. 5, 67).

[196] E. Wahl, U. Hillenbrand, and G. Hirzinger. “Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification”. In: *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*. IEEE. 2003, pp. 474–481 (cit. on pp. 4, 24).

[197] M. W. Walker, L. Shao, and R. A. Volz. “Estimating 3-D location parameters using dual number quaternions”. In: *CVGIP: image understanding* 54.3 (1991), pp. 358–367 (cit. on p. 17).

[198] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. “O-cnn: Octree-based convolutional neural networks for 3d shape analysis”. In: *ACM Transactions on Graphics (TOG)* 36.4 (2017), p. 72 (cit. on p. 12).

[199] Q. A. Wang. “Probability distribution and entropy as a measure of uncertainty”. In: *Journal of Physics A: Mathematical and Theoretical* 41.6 (2008), p. 065004 (cit. on p. 95).

[200] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. “Dynamic Graph CNN for Learning on Point Clouds”. In: *ArXiv e-prints* (Jan. 2018). arXiv: 1801.07829 [cs.CV] (cit. on p. 24).

[201] Y. Wang and J. M. Solomon. “Deep Closest Point: Learning Representations for Point Cloud Registration”. In: *arXiv preprint arXiv:1905.03304* (2019) (cit. on pp. 5, 66, 99).

[202] Y. Wang and J. M. Solomon. “PRNet: Self-Supervised Learning for Partial-to-Partial Registration”. In: *arXiv preprint arXiv:1910.12240* (2019) (cit. on p. 99).

[203] P. Wohlhart and V. Lepetit. “Learning descriptors for object recognition and 3d pose estimation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3109–3118 (cit. on p. 39).

[204] D. Worrall and G. Brostow. “CubeNet: Equivariance to 3D Rotation and Translation”. In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss. 2018 (cit. on p. 72).

[205] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 82–90 (cit. on p. 25).

[206] Z. Wu, S. Song, A. Khosla, et al. “3d shapenets: A deep representation for volumetric shapes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1912–1920 (cit. on pp. 11, 19, 25, 100).

[207] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. “PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes”. In: 2018 (cit. on pp. 5, 66, 89, 99).

[208] J. Xiao, A. Owens, and A. Torralba. “Sun3d: A database of big spaces reconstructed using sfm and object labels”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1625–1632 (cit. on pp. 34, 35, 37, 52, 53, 85).

[209] Q. Xu, X. Sun, C.-Y. Wu, P. Wang, and U. Neumann. “Grid-GCN for Fast and Scalable Point Cloud Learning”. In: *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020 (cit. on p. 113).

[210] J. Yang, H. Li, and Y. Jia. “Go-icp: Solving 3d registration efficiently and globally optimally”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1457–1464 (cit. on pp. 5, 67).

[211] M.-D. Yang, C.-F. Chao, K.-S. Huang, L.-Y. Lu, and Y.-P. Chen. “Image-based 3D scene reconstruction and exploration in augmented reality”. In: *Automation in Construction* 33 (2013), pp. 48–60 (cit. on p. 4).

[212] Y. Yang, C. Feng, Y. Shen, and D. Tian. “FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018 (cit. on pp. 24, 46, 49–53, 77, 119).

[213] D. Yarotsky. “Geometric features for voxel-based surface recognition”. In: *arXiv preprint arXiv:1701.04249* (2017) (cit. on p. 25).

[214] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. “Lift: Learned invariant feature transform”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 467–483 (cit. on pp. 4, 23, 27).

[215] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. “PU-Net: Point Cloud Upsampling Network”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018 (cit. on p. 24).

[216] W. Yuan, D. Held, C. Mertz, and M. Hebert. “Iterative Transformer Network for 3D Point Cloud”. In: *arXiv preprint arXiv:1811.11209* (2018) (cit. on pp. 99, 101).

[217] S. Zakharov, W. Kehl, B. Planche, A. Hutter, and S. Ilic. “3d object instance recognition and pose estimation using triplet loss with dynamic margin”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 552–559 (cit. on pp. 5, 66).

[218] S. Zakharov, I. Shugurov, and S. Ilic. “Dpod: Dense 6d pose object detector in rgb images”. In: *arXiv preprint arXiv:1902.11020* (2019) (cit. on pp. 5, 66, 89).

[219] B. Zeisl, T. Sattler, and M. Pollefeys. “Camera pose voting for large-scale image-based localization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2704–2712 (cit. on p. 89).

[220] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. “3dmatch: Learning local geometric descriptors from rgb-d reconstructions”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1802–1811 (cit. on pp. 5, 6, 11, 23–25, 27, 29, 30, 33–37, 45, 46, 51–53, 66, 69, 71, 72, 77, 78, 81, 85).

[221] Y. Zhao, T. Birdal, H. Deng, and F. Tombari. “3D Point-Capsule Networks”. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2019 (cit. on pp. 56, 111).

[222] Y. Zhao, T. Birdal, J. E. Lenssen, E. Menegatti, L. Guibas, and F. Tombari. “Quaternion Equivariant Capsule Networks for 3D Point Clouds”. In: *arXiv preprint arXiv:1912.12098* (2019) (cit. on p. 113).

[223] Q.-Y. Zhou, J. Park, and V. Koltun. “Fast global registration”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 766–782 (cit. on pp. 5, 67, 81).

[224] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. “On the continuity of rotation representations in neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5745–5753 (cit. on p. 105).

[225] M. Zolfaghari, Ö. Çiçek, S. M. Ali, F. Mahdisoltani, C. Zhang, and T. Brox. “Learning Representations for Predicting Future Activities”. In: *arXiv preprint arXiv:1905.03578* (2019) (cit. on p. 67).

List of Figures

List of Tables