ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Vision
and Geometry Lab

Microsoft

MVPDesc: Multi-View-Point Descriptor Learning for Registration of 3D Objects

Semester Thesis

Zijian Dong

Advisor: Dr. Johannes L. Schnberger and Dr. Christoph Vogel
Supervisor: Prof. Dr. Marc Pollefeys

September 21, 2019

Abstract

Significant to the matching of 3D objects is how to establish correspondences between 3D keypoints based on their 3D local descriptors extracted from various 3D representations. Among them, point clouds and multi-view images are easy to obtain and show better performance comparing to others. However, little effort focuses on exploring the complementary relationship between them. In this thesis, we investigate the spherical feature map for fusing 2D and 3D data and propose two approaches to deal with this problem. The first approach associates a sphere to a 3D point and projects it onto the multi-view images that observe the 3D point. These spheres are combined by weights given the angle between the viewing ray and the normal on the sphere. However, it fails in experiments due to not using the true geometry of point clouds for projection. Later MVPDesc is proposed, which can jointly fuse features from point cloud data and multi-view data. In this approach, the dominant vector of point clouds is first extracted by PCA to guide the stereographic projection. This dominant direction defines a projected plane that replaces the sphere as a proxy to combine the information of different views. For each Delaunay triangular region on projected planes, the color information is obtained via projection of original 3D points. After synthesizing each projected view and projecting it back to the sphere, the constructed sphere can be further processed by Spherical CNN to extract descriptors. To learn descriptors effectively, a triplet network with batch hard mining and soft margin loss is built to ensure fast convergence of the algorithm. Experiments show that our MVPDesc achieves robust matching performance and rotation-invariant property, which can be further applied to registration of rotated 3D structures.

Acknowledgements

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation of this Work

Matching of local 3D structure combines 3D information from various sources into a joint coordinate, which serves as an important element in applications such as Structure from Motion[17], SLAM[18], Re-localization[16] and robotic perception[5]. Significant to this task is how to determine correspondences between two 3D structures. To deal with this problem, plenty of 3D descriptors[14, 15, 22] are proposed to represent features of 3D keypoints. Specifically, generating descriptors from either the multi-view images[29] or the sparse point clouds[28] is proven to be flexible and achieves good performance, due to wide range of data source. Therefore, either multi-view based descriptors or point cloud based descriptors has received tremendous interest in recent years.

For multi-view based method, the input can be represented by a bunch of views captured from cameras in different directions. Thanks to the development of deep learning, descriptors from image patches can be efficiently extracted by convolutional neural networks(CNN), such as VGG[19] and ResNet[7]. For instance, MVDesc[29] is designed to represent the feature from multi-view data by aggregating features learned from each view via view-pooling. Although the image data is dense, ordered and can easily be processed by neural networks, these multi-view descriptors inevitably lose information of geometry and 3D structure. Therefore, they are easily influenced by various viewpoints, seasonal change and lighting condition.

For point cloud based method, the data is obtained from 3D cameras, LIDAR or 3D reconstruction methods, which preserve 3D geometry structure of the objects. Point clouds are ordered and CNN is not suitable for direcly processing them. Recently, various works are proposed to extract features from point clouds. PointNet[12] first designs a symmetric function suitable for learning spatial features from point clouds. PPFNet[3] extends PointNet for descriptor learning and robust matching of point clouds. However, since point clouds are always sparse in space, descriptors extracted from them have limited ability to represent 3D shapes.

After analyzing the drawbacks of existing methods, it is necessary to design a 3D descriptor

representing both appearance features from dense-structured images and local geometry features from sparse point clouds. This basic 3D information can be easily obtained from mapping pipelines such as SfM or visual SLAM.

Another large problem exists when considering registration of rotated shape. Current frameworks such as PointNet[12], PointNet++[13] and DGCNN[23] only concentrate on objects with canonical orientation and thus require immense data augmentation. Spherical CNN[1] proposes the spherical cross-correlation operation aiming to extract features from dense meshes, which is not suitable for point clouds. The difficulty lies in how to embed the information of the sparse point clouds onto the sphere.

Different from other approaches, which just concatenate features from different representations, we explored two novel approaches to construct a spherical feature map based on projection, exploiting relationship between 2D images and 3D points. In the first approach, a sphere is associated to one 3D point and projected onto multi-view images that observe the 3D point. These spheres are combined by weights given the angle between the viewing ray and normal vector on the sphere. However, experiments show this approach fails due to not using true geometry of the point clouds for projection. To address these fusion issues, we proposed a Multi-View-Point descriptor named MVPDesc to solve all mentioned problems. We first extract dominant direction of point clouds by PCA as the direction of stereographic projection. For each triangular region on projected views, the color information is collected via projection of original 3D points. After combining information of each projected view and project them back to the spherical feature map, this sphere can be further processed by Spherical CNN to extract rotation-invariant descriptors of the keypoint.

We also show how to learn such a framework in an efficient manner. To deal with the lack of training labels, triplet network is applied for training using the corresponding point triplets extracted from Structure-from-Motion(SfM) pipeline. Batch hard mining and soft margin loss are also used for ensuring the fast convergence of the algorithm.

1.2 Contributions

The main contributions of this thesis are the following:

1. We are the first to explicitly leverage the complementary relation between sparse point clouds and multi-view images for the description of keypoints to solve the task of point cloud registration. Specifically, we design a novel spherical feature map to fuse features from 2D/3D data.

2. A triplet network is designed and is trained using the dataset of 3D matching points extracted from SfM pipeline.

3. MVPDesc is explicitly designed for rotation invariance and can be obtained from any arbitrary views.

1.3 Thesis Organization

This thesis is structured in the following manner: First, we introduce various related work about descriptor learning of 3D keypoints and analyze their potential limitations in chapter2. Next, we will explain why and how we adopt these models to learn descriptors and also illustrate potential drawbacks of each approach in chapter 3. Following this part, experiments are designed to test the matching performance and rotation invariant property of our descriptors in chapter 4. In chapter 5, current achievements, failure cases, potential application and future work are discussed to help us improve the results. Finally, a conclusion is listed in chapter 6.

Chapter 2

Related Work

A large variety of 3D local feature descriptors have been proposed recently. In this section, we briefly review some related methods, which can be divided into two directions: hand-crafted models and learning-based models.

2.1 Hand-crafted Models

Early researches focus on establishing shape descriptors based on hand-crafted features. For the task of image matching, SIFT[11], is still probably the most well-known local feature descriptor in some tasks such as Structure from Motion(SfM)[17] and Visual Odometry(VO)[6]. Similar to their 2D counterpart, manual designed 3D descriptors are well studied during these years, including Fast Point Feature Histograms(FPFH)[14], Signature of Histogram Orientations(SHOT)[15], and Unique Shape Contexts(USC)[22]. These 2D/3D descriptors are mainly based on low-level geometric features and then not robust to difficult scenarios such as lighting and seasonal changes, cluster and occlusion, and viewpoint variations.

2.2 Learning-based Models

With the development of deep learning, great breakthroughs have been achieved in descriptor learning. 3D data has multiple popular representations such as multi-view images and point clouds, which leads to multiple approaches for learning.

2.2.1 View Based Model

For view-based approaches, the input data is a collection of views captured by cameras with different viewpoints. To recognize 3D shapes from multi-view images, Hang Su[20] proposes a multi-view convolutional neural network, which firstly learns the features of each view and aggregates views based on a view pooling layer. This method has achieved great success in the task of 3D

shape classification and retrieval. MVDesc[29] is the first multi-view descriptors for the description of 3D keypoints. However, these two approaches highly depend on view pooling for feature aggregation, which has the risk of smoothing local patterns[9]. Bidirectional long short-term memory (BiLSTM)[2] is also used to fuse information from various views. These three methods only learn features based on networks, but ignore the projection relationship between multiple views.

2.2.2 Point Cloud Based Model

Point clouds are another straightforward representation for describing 3D objects. With the development of 3D cameras and LIDAR, it has received lots of attention recently. PointNet[12] is the pioneer to achieve a network pipeline to deal with the permutation invariance of point clouds. PointNet++[13] further develops a hierarchical neural network, which is able to learn local features with increasing contextual scales. To further incorporate neighborhood information, DGCNN[23] proposes EdgeConv to obtain edge features between keypoints and their neighbors. For the task of point cloud registration, 3DFeat-Net[24] uses attention mechanisms to learn 3D feature detectors and descriptors based on weak supervision. However, point clouds are always sparse in space, which restrict the performance of descriptor learning in these approaches.

2.2.3 Multimodal Fusion Model

If geometry and image information are available, it appears beneficial to fuse both models to take advantage of their distinct properties. PVNet[25] extract local features from point clouds based on the global view features. To further explore the relationship between point clouds and multi-view data, PVRNet[26] achieves point-single-view fusion and point-multi-view fusion based on the learned relation module. Our work differs greatly from them by fusing the features from point clouds and multi-view images in an explicit way based on the known projection relationship between 2D images and 3D point clouds.

2.2.4 Spherical CNN

The above approaches have their distinct characteristics for learning features of 3D shapes, but they don't take the orientation of 3D shape into consideration. Spherical CNN[1] defines the spherical cross-correlation, which is rotation-equivariant. Based on this operation, this special network could extract features on spheres or even continuous meshes. However, it can't be directly applied for learning features from sparse point clouds and multi-view images. Based on Spherical CNN, PRIN[27] proposes a framework to generate rotation-invariant features for each point cloud. Nevertheless, this method just sample from point clouds without considering features from dense images.

2.2.5 Metric Learning

For metric learning, lots of work focuses on designing loss and training strategies for learning descriptors. Theoretically, an ideal loss formulation encourages matching patches to be close and different patches to be distant. Most of the work adopts two loss functions for learning descriptors: contrastive loss[29] and triplet[8] loss, which try to learn descriptors with pairs and triplets. To make triplet network converges fast, Alexander[8] proposes to apply hard-mining strategies for finding the hardest triplets for training networks. Log-likelihood loss formulation is choosen by L2-Net[21] and it is further extended by HardNet[10] with a structured loss. N-tuple loss is proven effective for incorporating global information into local descriptors in PPFNet[3]. In our task, we aim to find a proper loss and suitable training strategy, without the need of complicated tuning of parameters.

Despite some promising results in hand-crafted descriptors and learning-crafted descriptors, none of them can't develop a framework to combine features from multi-view images and sparse point clouds in an explicit and reasonable approach. So in this paper, we consider the multi-modal fusion problem and hope to explicitly generate a local descriptor from these two representations, which is also rotation-invariant and view-invariant.

Chapter 3

Methods

In this section, we propose to learn rotation-invariant and view-invariant descriptors for 3D key points, which combine multi-view fusion techniques and the geometric structure of sparse point clouds. Our inputs are two main representations for 3D shape: sparse point clouds and multi-view calibrated images. These representations can be easily obtained from structure from motion pipelines[17]. Our key idea is to combine local appearance information of multiple image local patches into a sphere around the point of interest to represent all the features of this 3D point. The motivation of this idea is intuitive and easy to understand. Firstly, because the sphere looks the same in different viewpoints, it is possible to combine images from arbitrary views in this sphere to form the omnidirectional vision. Secondly, inspired by the development of Spherical CNN[1], a 3D descriptor can be learned with a spherical feature map, the rotation-invariant property of which is quite meaningful for registration of rotated 3D objects[27]. Given those two kinds of input, we first explore two approaches to extract the spherical feature maps of the key point in an explicit way and discuss which one is more appropriate for our pipeline. Then we design our descriptor learning network based on spherical CNN and triplet metric learning. The final local features can be directly used for matching of 3D point clouds.

3.1 Constructing Spherical Feature Map

Sparse point clouds and multi-view images are common representations for 3D shape, but previous works often learn features from them separately such as PointNet[12, 13] and MVCNN[20]. Recently, some researchers have proposed a method based on Embedding Attention Fusion[25]. However, it is based on a weighted combination of the features learned from these two representations in an implicit way, without considering the known projection relationship between those 3D and 2D data. Given the projection matrix between 2D and 3D data, we developed two explicit approaches for constructing our spherical feature map. In the first method, only multi view images are used and they are projected onto the sphere based on projection matrix. To overcome the defects of the first method, another method based on stereographic projection and Delaunay Triangulation

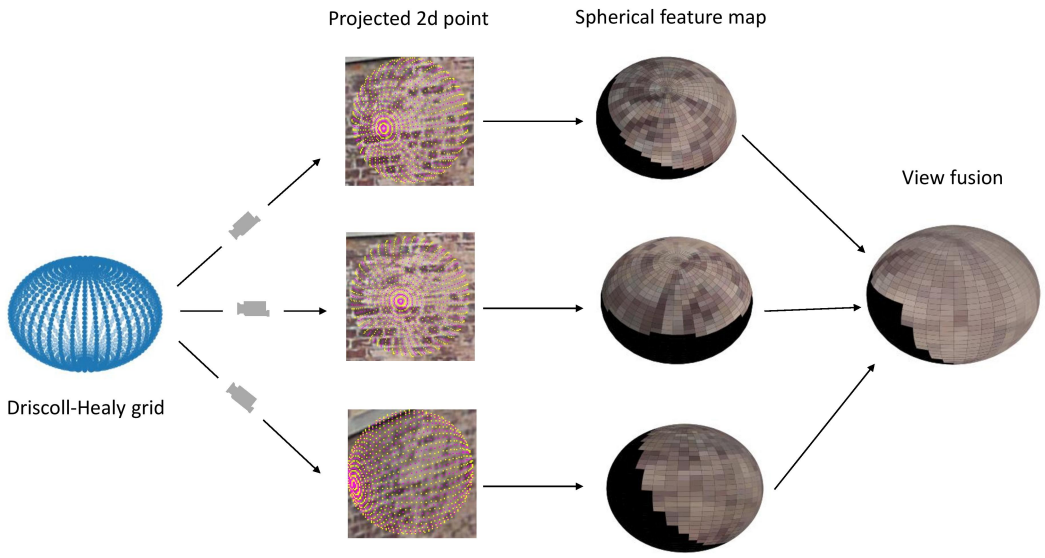


Figure 3.1: Detailed structure of generating a spherical feature map from multi-view images

is proposed.

3.1.1 Projecting multi-view images onto the sphere

For the first method, we make a weighted combination of appearance information of different views in two steps. First of all, we project 3D points of the sphere onto images and build the spherical feature map of each view. Next, these constructed spheres are combined by weights given the angle of viewing ray and the normal vectors on the sphere. The details are illustrated in the following parts.

Generate spherical feature map for single view

The detailed structure is illustrated in Fig 3.1. First, the enclosing sphere around the key point is generated and discretized using a Driscoll-Healy grid[4] with a designed bandwidth. Based on the projection matrices corresponding to different views, we project these 3D grids onto each view image. According to this projection method, the color information of each patch on the sphere can be represented by the average RGB value of its corresponding patch on the image. As a result, the spherical feature map corresponding to each single viewpoint can be established. For the back face of the sphere, the color information is assigned as 0.

Spherical map fusion

Next, the color information from different spherical feature maps should be condensed into one spherical feature map. Considering one small patch on the sphere, its normal vector can be approximated by the unit vector $\mathbf{n}$ pointing from the sphere centroid to the patch center. When this patch is observed by a camera set$\{c_1, c_2, \ldots, c_k\}$, the weight of patches is related to the unit camera vector $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}\}$ pointing from the sphere center to the camera center. The weight can be calculated by the following formula:

$$w_i = \frac{\mathbf{n} \cdot \mathbf{v_i}}{\sum_i \mathbf{n} \cdot \mathbf{v_i}}$$

Then the total spherical feature map can be represented as a weighted combination of spherical feature maps correlated to all observed images. The sample of generated spherical feature map is shown in Fig 3.1.

Potential drawbacks

Although this method looks intuitive and it is easy to construct a dense spherical feature map, there exists a large problem. According to Fig 3.2, we observe that the same 3D point on the sphere is projected to different 2d points on different images, which will cause wrong combination of spherical feature maps. It is because some 3D points on the sphere do not exist in the real world. If we project them into images, it will correspond to the projected 2D points corresponding to other 3D points. In the example on Fig 3.2, for the red point on the sphere, in the view of camera 1, it corresponds to the texture of sky. However, in the image of camera 2, it corresponds to the texture of the wall. When mixing features from the sky and the wall, the color of this point can be random and meaningless.

3.1.2 Stereographic projection map

To avoid the wrong combination of spherical feature maps mentioned in the first method, we limit the existing geometry to point clouds instead of points on spheres. Our key idea is to project local 3D points onto the sphere and extract the topology of unwrapped sphere map based on Delaunay Triangulation. Then we project original 3D points of these triangles into images to collect appearance information for unwrapped sphere map, and then project back to the sphere. The key issue is how to project the 3D points onto sphere without distortions.

Projection method

The first approach is to project 3D points towards the centroid of the sphere using a ray casting scheme as shown in Fig 3.3(a). For each point on the sphere, we send a ray towards the origin and collect the inclination angle θ from z axis and the azimuth angle ϕ from x axis. With these two angles, the points on the sphere can be mapped to points in a rectangular region with $\theta \in [0, 2\pi]$

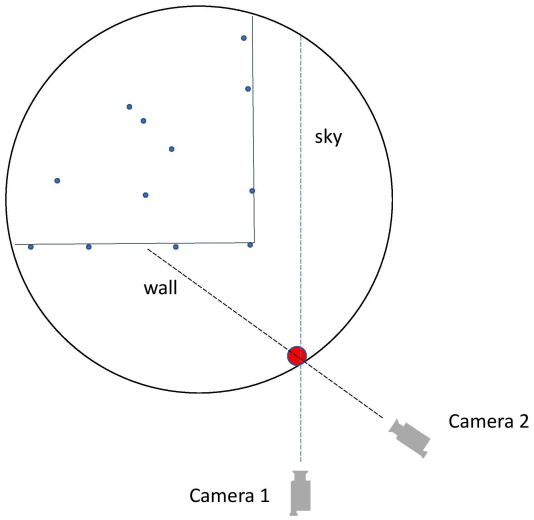
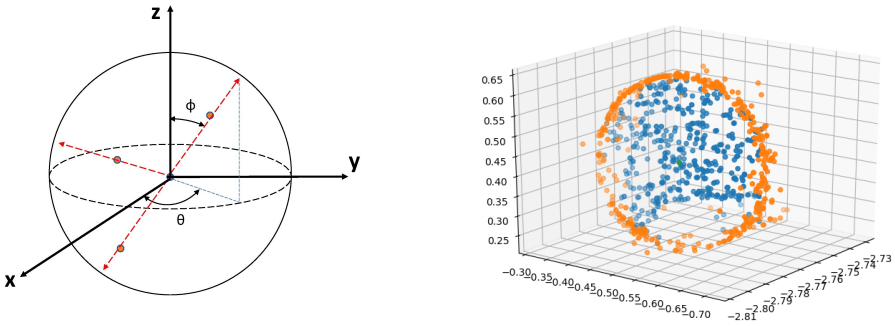


Figure 3.2: The potential drawback situation: the same point on the sphere could be projected to one point on the wall and one point on the sky in different images.

and $\phi \in [0, \pi]$. This region is called unwrapped sphere in this paper. As a result, it is easy to find triangles on the rectangle region and project the corresponding 3D points of this small triangle onto images. However, there is a severe problem for this method. Since our task specifically targets at learning local features and almost all local 3D region of key points obtained from SfM is a surface plane. When applying this method for projection, the projected 2d points on the sphere will form a ring with large distortion as shown in Fig 3.3(b). This is because the constructed surface plane is quite thin, and the key point is always at the center of the surface. It is shown in Fig 3.4, those generated triangles do not totally correspond to the reconstruction surface.

To solve this projection problem, we move the projection center along the direction of the normal vector of the plane. Given local 3D point clouds, the normal vector is easily obtained via PCA. Then the rest projection procedure is the same as the first method. Fig 3.5 shows a detailed illustration of this method and Fig 3.6 shows the Delaunay triangulation of the corresponding unwrapped sphere map. This method in Fig 3.6 performs better than the first approach displayed in Fig 3.4, but there still exist distortions around edges of the surface. It is because that the 3D points close to the center form much looser regions on the sphere, compared to the points far away from the center. That makes our model lose much information for learning the descriptors.

After these discussions, an approach based on stereographic projection is chosen for constructing the spherical feature map. The detail of this method is shown in Fig 3.7. Given sparse point clouds and multi-view images, the dominant direction of point clouds corresponding to each view is first extracted by PCA. After that, we project original 3D points on the plane tangent to sphere and collect appearance information from each view for triangles on projected 2d plane. Later color



(a) (b)

Figure 3.3: (a): Details of ray casting scheme. The ray is cast from the surface of the sphere towards the origin. (b): Examples of ray casting scheme: the 3D points are blue and the projected points are orange.

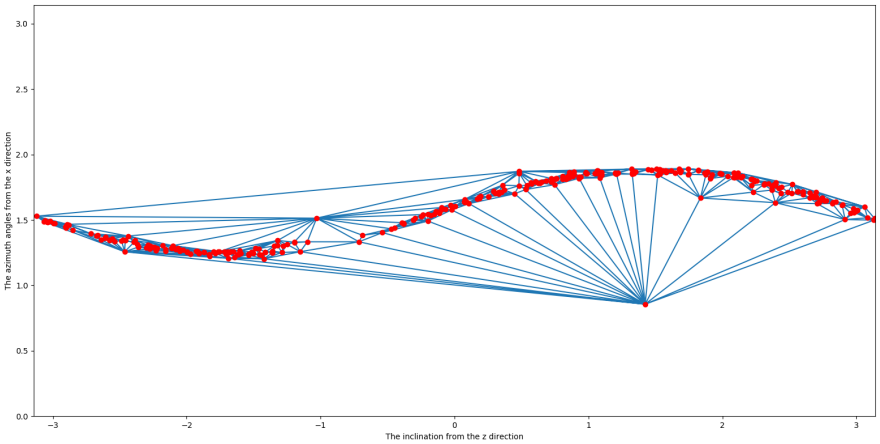
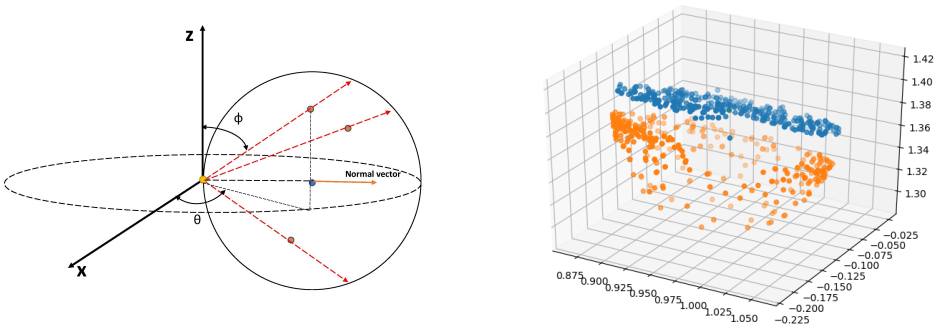


Figure 3.4: Delaunay triangulation of projected 3D points on the unwrapped sphere



(a) (b)

Figure 3.5: (a): First, we obtain the normal vector of 3d point clouds based on PCA and move the projection center along the direction of normal vector. The ray is cast from the surface of the sphere towards the new projection center. (b): Examples of the ray casting scheme towards the new projection center.

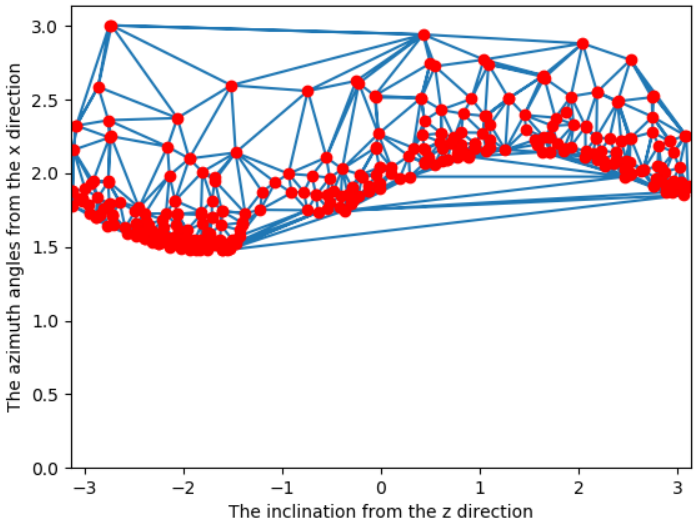


Figure 3.6: Delaunay triangulation of projected 3D points on the unwrapped sphere based on ray casting scheme towards the new projection center.

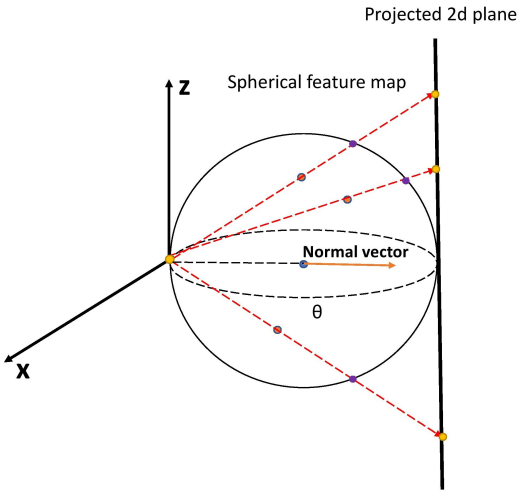


Figure 3.7: Schematic diagram of stereographic projection: the orange points are 3D points; the violet points are on the spherical feature map; the yellow points are on the projected 2d plane.

information of multi views can be combined on this plane and then it is projected back onto the sphere. Since this process can further be regarded as a part of stereographic projection, each point on the sphere will correspond to a unique point on this plane.

Sphere fusion

One advantage of this fusion process is that since each point on the sphere correspond to the unique point on the projected plane and the projected planes are the same for all different views. Therefore, it is easy to directly average the common projected regions and complement the unique regions belonging to specific views. Fig 3.8 shows the projected plane of three different views and the combination of these three views. Comparing with original image patch, there exist less distortions and the combination of image is reasonable and explicit. The reconstructed spherical feature map is shown in Fig 3.8.

3.2 Descriptor Learning Network

3.2.1 Architecture of spherical CNN

Our goal is to use spherical CNN to extract features for spherical feature map, which is a process of mapping from a high dimensional data space to low dimensional feature space[1]. The overall architecture of our descriptor learning network is shown in Fig 3.9(a). It mainly consists of an initial S^2conv-BN-ReLU block followed by two $SO(3)$conv-BN-ReLU blocks. The S^2conv block

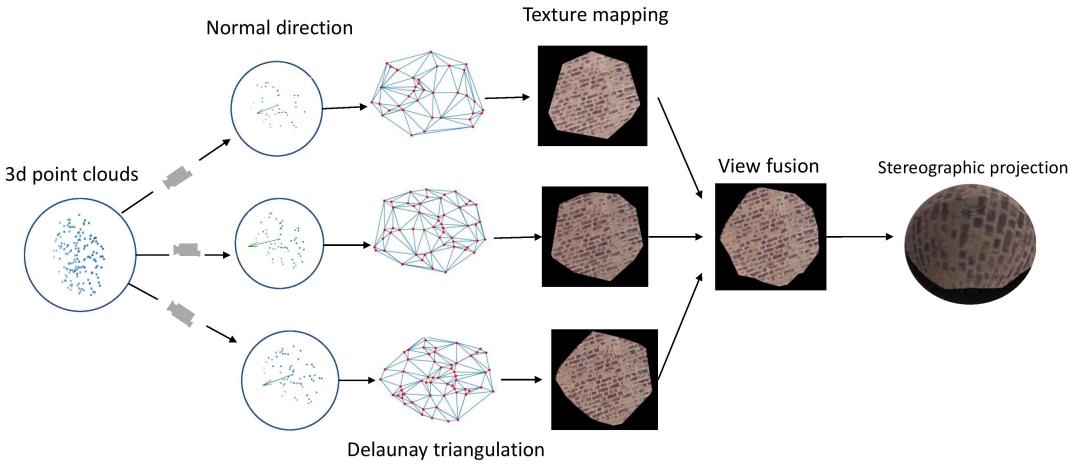


Figure 3.8: Detailed illustration of spherical feature construction based on stereographic projection

is the convolution operations for S^2 and its output is $SO(3)$ signals, which can be processed by $SO(3)$conv layers to extract features. The resulting features are pooled using a max pooling layer and then fed into a linear layer, which could be set flexibly to revise the dimension of descriptors. The hyperparameters of the network are shown in the experimental part.

3.2.2 Soft margin loss

The state of the art adopts two loss functions for learning descriptors: contrastive[29] and triplet[8] loss, which try to learn descriptors with pairs and triplets. We train a triplet network f to learn descriptors, which takes an anchor x_a, a positive x_p (of the same class as an anchor) and negative x_n (of different class than an anchor) example. The objective is to learn embeddings such that the anchor is (by some margin) closer to the positive example than it is to the negative example. The triplet loss can be defined as follows:

$$L(x_a, x_p, x_n) = \max\{d(f_\theta(x_a), f_\theta(x_p)) - d(f_\theta(x_a), f_\theta(x_n)) + \text{margin}, 0\}$$

$$d(x_i, y_i) = ||x_i - y_i||_2$$

According to [8], it is beneficial to replace the hinge function by a smooth approximation using the softplus function. Although behaving similarly as the hinge function, this function decays exponentially without a hard cut-off. We finally find this soft-margin loss behave better for our task

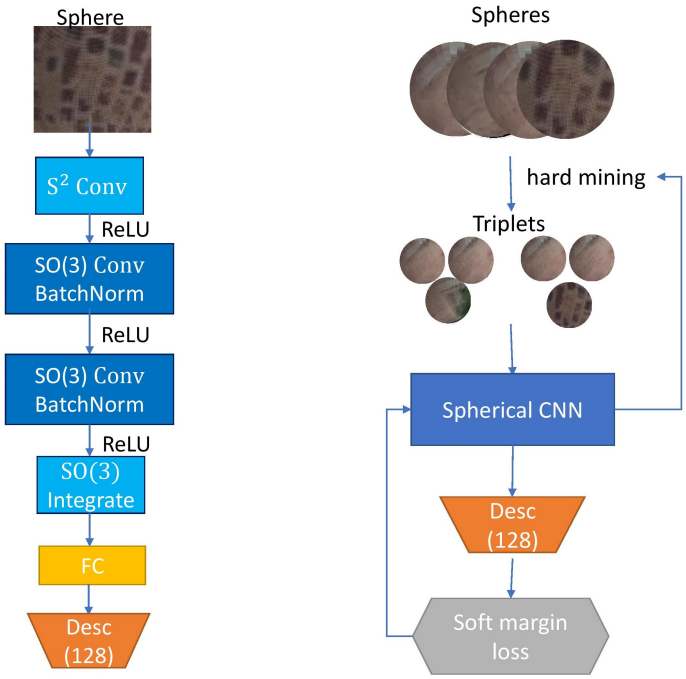


Figure 3.9: (a): Spherical CNN for learning descriptors for spherical feature maps (b): Triplet training process based on hard mining and soft margin loss

in the experiment, since it could pull together samples from the same class as much as possible. The soft-margin loss can be defined as follows:

$$L(x_a, x_p, x_n) = \log(1 + \exp(d(f_\theta(x_a), f_\theta(x_p)) - d(f_\theta(x_a), f_\theta(x_n)) + \text{margin}))$$

3.2.3 Batch hard mining

There exist many problems in triplet networks. First, the number of triplets grows cubically with the number of examples and it is impossible to process them all and then the training converges slowly[8]. Second, the triplets are generated randomly. When the training continues, the loss of more triplets becomes small or 0, which can prevent the network from training. Third, each sphere is used for the computation of only one triplet and can't be reused for other corresponding triplets. The computation is wasted. On the other hand, if only the hardest triplets are chosen, the outliers can be selected with a wrong proportion and make the network unable to learn normal associations.

Inspired by the paper[8], batch semi-hard mining is used to deal with all the mentioned problems. The core idea is to form batches by randomly sampling P classes (one class including all features extracted from each 3D keypoint), and then randomly sampling K spherical feature of each class (each feature formed by a group of different views), therefore resulting in a batch of $P \cdot K$ spheres. Then for the anchor in each sample, we choose the hardest positive and hardest negative samples in the batch when constructing triplets for computing the loss. The batch hard loss could be calculated as follows:

$$L(\theta; X) = \sum_{i=1}^{P} \sum_{a=1}^{K} \log(1 + \exp(\overbrace{\max_{p=1,\ldots,K} d(f_\theta(x_a^i), f_\theta(x_p^i))}^{\text{hardest positive}} - \underbrace{\min_{\substack{j=1,\ldots,P \\ n=1,\ldots,K \\ j \neq i}} d(f_\theta(x_a^i), f_\theta(x_n^j))}_{\text{hardest negative}} + \text{margin}))$$

,where X is a mini-batch and a data point and the data point corresponds to the j-th spherical feature map of i-th 3D point in the batch. The total structure of triplet training process is shown in Fig 3.9(b)

Chapter 4

Experiments and Results

4.1 Data Preparation

Our aim is to learn correspondences between 3D key points, which demands millions of ground truth annotations labeled manually. Current available data sets for 3D matching generally lack enough multi-view patches and they are solely focus on one representation for 3D shape, such as point clouds or multi-view images. For example, the RGB-D reconstruction dataset of 3DMatch[28] only contains 2D RGB-D patches and 3D patches without corresponding multi-view images. Therefore, we prepare the training data based on Structure-from-Motion (SfM) database generated by COLMAP[17]. The scene named South-Building contains more than 60k 3D points and over 128 view images and each 3D point can be projected to at least 8 images. The corresponding 2d points are obtained from the projection of each 3D point according to the method in paper[17], which is mainly based on SIFT matching. For the view number to construct a spherical feature map, we only adopt 4 views in our network, which is a good trade-off between accuracy and computational efficiency. A positive training pairs is constructed by two independent set of multi-view image pairs corresponding to the same 3D point cloud, while a negative pair is from different 3D points. A total number of 6k triplet pairs is formed based on our strategy.

4.2 Network Setup

When constructing the spherical feature map, we finally select the bandwidth as 20. Our network structure is shown in Fig 3.9. We use 64, 128, 256 features for the S^2,and the two $SO(3)$ layers respectively. In each layer, we reduce the resolution of the network from 20,10 to 6 in the final layer. The final network takes 4GB of memory at batch size of 20. We train the network using the Adam optimizer with a weight decay of 0.0001. The training is generally converged within 200 epochs, which only takes 1 hour to train.

4.3 Performance Evaluation

Our experimental evaluation is split up into three main parts. In the first sequence of experiments, the matching performance and rotation-invariant property of our first proposed method is shown. We further discuss the drawbacks of this method and why the fusion of multiple views is not reasonable. The second section shows the performance of our selected triplet loss and training method. Finally, the third section shows the performance of our second method based on stereographic projection and discusses the advantage of this model for fusing different views

4.3.1 Matching performance of the first method

Multi-view sphere

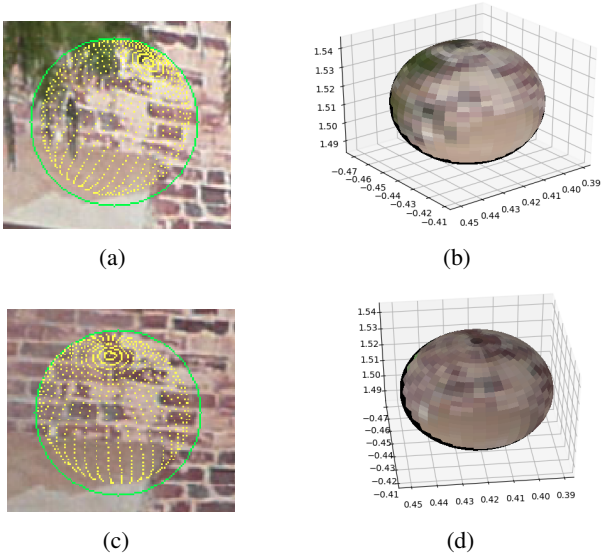
Setup: We first choose the dataset which we establish following 4.1. Its training dataset consists of 2000 3D key points extracted from SIFT and we randomly sample 4 independent views from the camera set for each point and repeat this process 3 times for selecting 3 subsets. The 4-view patches are used to construct spherical feature map based on the first method introduced in 3.1.1. The training dataset finally consists of 6000 spherical feature maps from 2000 classes. We also select 2000 totally different 3D key points and follow the same procedure to generate the validation dataset. Then we generate 6000 triplet pairs by randomly choosing two spherical feature maps from one class and the other from a different class. These triplets are taken as input to generate descriptors. For the point matching task, we apply putative matching across the validation dataset and judge whether the correspondences with smallest distance between their descriptors are from the same class. We use this 1-top accuracy as measurement. We also make comparisons with the baseline SIFT. For this baseline model, we only choose the Triplet loss without hard batch mining.

Result: From statistics in Table 4.1, the first method only achieves an accuracy of 19.75 percent for the matching task, which is much worse than the matching result of the SIFT baseline. We try to visualize the result of the first method and the result is shown in Fig 4.1. From Fig 4.1, the constructed spheres from the same 3D point is totally different from different views. This underlines our investigation from Sec 3.1.1. As shown in Fig 4.1, the same point of one sphere can be projected into green pixel in Fig 4.1(a) which corresponds to trees, but on Fig 4.1(b), it projected onto the brown pixels on the wall. This will further cause the wrong combination of pixel values in the process of view fusion.

Single-view sphere

Setup: To verify whether the bad performance of the first method is caused by the wrong combination of spheres, we construct the spherical feature map solely based on one single view. The rest of setup is the same as the setup for the multi-view approach.

Result: From Table 4.1, it shows that the method without the fusion of different views achieves an accuracy of 23.5 percent, which is higher than that of the multi-view method. This proves that our discussion above is correct. However, the accuracy is still quite low. There mainly exist two



(a) (b)

(c) (d)

Figure 4.1: (b),(d) are two spherical feature maps of the same 3D point (a),(c) are their corresponding 2d images and projected 2d points.We could find that the same point on the sphere can correspond to different colors on different images.

Table 4.1: Top-1 matching accuracy of different methods

Methods	Top-1 matching ac
Multi-view sphere + Triplet loss	
Single-view sphere + Triplet loss	
Single-view sphere + Hardest negative mining	
Single-view sphere + Hard mining(Rand)	
Single-view sphere + Soft margin loss + Single-view sphere + Soft margin loss + Hard mining(Rand) Hard mining(Rand)	
Stereographic projection map + Soft margin loss + Hard mining(Rand)	
SIFT(method in SfM)	

reasons. First, according to Sec 3.2.2 and Sec 3.2.3, it is difficult to train a triplet network and make it converge. Some training strategies such as batch hard mining and soft margin loss could be used to make it converge quickly. Second, utilizing only a single view image loses much information for the 3D point, which we seek to overcome with our developed methods for view fusion. We will discuss these two aspects in the following parts of our experiment.

Evaluation of training strategies

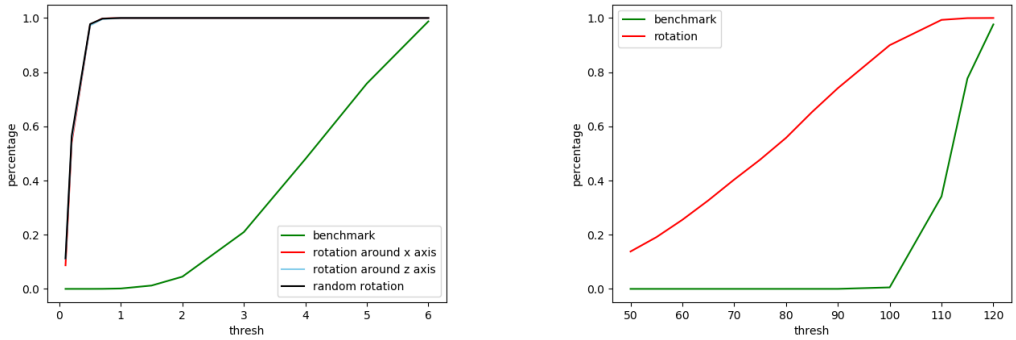
To evaluate the performance of the training strategies introduced in 3.2.2 and 3.2.3, we first test the hard-mining strategy. When we feed a network with mini batches, there exist two strategies to choose triplet selection. First, the hardest negatives are chosen for each positive pair. Second, we choose hard negative (triplets with positive triplet loss value) randomly for each positive pair. From Table 4.1, we find both random hard mining and hardest mining have better performance than the normal training strategy.

Specifically, the random hard mining achieves an accuracy of 37.5 percent, which is the best among all the methods. To evaluate the effect of soft margin loss, we change the triplet loss into soft margin loss. Finally, after training 30 epochs, it achieves an accuracy of 50.53 percent.

Evaluation of rotation-invariant property

Since the property of rotation invariance is quite significant for the task of 3D matching, we finally evaluate the performance of our descriptors with respect to rotations of the input. We create one test dataset including 2000 original spherical feature maps in which the original spherical feature map rotates around the x axis (y axis), z axis and any arbitrary axis respectively. We trained our model on the non-rotated training set introduced in 4.1 and evaluated it on the rotated testing dataset. To verify our rotation-invariant property, SIFT descriptors are used to compare with our methods. Since SIFT descriptor is only for 2D point, we just extract SIFT descriptors for the same 3D point in different views and regard them as different descriptors with respect to rotations.

To verify the rotation-invariant property, we calculate the Euclidean distance between descriptors corresponding to two different matching points and draw the curve of the percentage of the distances below thresholds as the benchmark for comparison as shown in Fig 4.2(a). The other lines in Fig 4.2(a) corresponds to the difference between the rotated sphere and the original sphere from the same 3D keypoint. Comparing our method and SIFT descriptors with their corresponding benchmark, we could obtain the result in Fig 4.2. From Fig 4.2(a), we obtain that under the thresholds of 1, when 95 percent of rotated descriptors satisfy the requirement, only 0.3 percent of descriptors in benchmark satisfy it. In contrast, for SIFT in Fig 4.2(b), when 95 percent of rotated descriptor satisfy the threshold, 50 percent of descriptors also satisfy this threshold. This means it is much safer to our descriptor for matching two rotation 3D objects comparing to SIFT. Furthermore, under any arbitrary rotation, our descriptor performs the same performance. Therefore, the descriptors trained by our method are rotation-invariant, which are useful in the task of 3D matching regardless of rotation.



(a) (b)

Figure 4.2: (a) Rotation invariant property of our descriptor: the green line correspond to benchmark which calculates the distance between descriptors from different points. The other lines correspond to distance between original descriptors and rotated descriptors. (b) Rotation invariant property of SIFT: the green line corresponds to difference between descriptors from two different points; the red line corresponds to difference between two descriptors extracted from two views of the same point. The horizontal axis corresponds to different thresholds. The vertical axis represents the percentage of descriptors of which the euclidean distance from original descriptors is below the threshold.

4.3.2 Matching performance of the second method

Setup: To test the performance of our second method and compare with the first method, we generate the training and test dataset following the procedure introduced in 3.1.2. We use the same 3D points as the first experiment and obtain their corresponding neighboring 3D points with a radius equal to 0.1. Similarly, 4 views are used for generating spherical feature maps and these maps have the same dimension as those in first method. We choose the structure of spherical CNN in 4.2 with soft-margin loss and batch hard mining.

Result: As shown in Table 4.1, our network achieves an top-1 accuracy of 62.7 percent, which is much higher than the top-1 accuracy of the first method. The top-5 accuracy for matching is 85.5 percent and the top-10 accuracy can reach over 90 percent. However, the accuracy is still lower than the method of SIFT. The reason is due to the fact that when generating SfM dataset, SIFT is used to generate corresponding pairs of 2d points. Therefore, all the 2D matching points can be matched by SIFT and that is why SIFT descriptors achieve such a high accuracy.

Chapter 5

Discussion

5.1 Achievement

In this paper, We first explore the first approach to construct the spherical feature map solely dependent on multi-view images. According to experiments and analysis, we explain the bad performance as not using the true geometry of the point clouds for projection. After that, we revise our framework and the results in experiments show that our second framework, i.e. MVPNet, can take neighboring sparse point clouds and their corresponding local image patches as inputs to reconstruct signals on the spherical feature map in an explicit and reasonable approach. The descriptors learned by Spherical CNN from this spherical feature map are shown to be rotation invariant and can be further applied for the task of 3D registration or matching.

5.2 Failure Cases

Though our framework is suitable for most situations, we also discover some exceptions during experiments. Sometimes the model tends to fail when there exist two or more planes in the keypoint neighborhood. This happens for instance if the keypoint is a corner point of a building or for noisy points in space. Then our framework can't find a suitable normal vector for the 3D points. To deal with this issue, we hope to find two or more principle normal vectors for the local 3D points and form three spherical feature maps for one point. These three spherical feature maps can be processed together by Spherical CNN. Furthermore, due to computational considerations, our resolution of spherical feature map, which is defined by bandwidth, is restricted within 20. However, better results can be obtained with higher resolution.

5.3 Application

Similar to SIFT on 2D images, our MVPNet can produce rotation-invariant 3D point descriptors based on images and sparse point clouds. That is quite useful for incorporating local 3D maps

from SfM or SLAM into a globally consistent map, since the corresponding relationship between 2D images and 3D points are stored when building these maps. Besides, with the development of 3D cameras such as Kinect, it is easier to obtain sparse 3D point clouds together with their corresponding 2D images. Like we have done for 2D images, we could build a 3D map and establish one 3D descriptor library based on our method. Then given a 3D point and its corresponding local image patches, we could match it with our library and easily know where it locates in the 3D map.

5.4 Future Work

In the future, we need to solve the failure cases mentioned in 5.2 as well as improve the computational efficiency of Spherical CNN and design normal vectors of 3D points. Furthermore, instead of using raw pixel information, the features learned from RGB image by CNN can be used for constructing our spherical feature map later. And our pipeline of constructing spherical feature map can be incorporated into our descriptor learning network, which enables end-to-end training and back-propagation through all of the process. Finally, more experiments will be designed for different application scenarios and various dataset. Also, comparison with other 3D descriptors is necessary and will be conducted into these experiments.

Chapter 6

Conclusion

In contrast to current trends, we have addressed the importance of exploring complementary relation between sparse point clouds and multi-view images. We have studied two different methods. The first approach associates a sphere to a 3D point and projects it onto the images that observe the 3D points. By experiments, the failure of this approach is due to not using the correct geometry of the point clouds for projection. Later we propose the descriptor, i.e. MVPDesc, which can jointly combine point cloud data and multi-view data for learning 3D descriptors. In our framework, the normal vector of point clouds is extracted by PCA, which is regarded as the direction of stereographic projection. After this projection, texture mapping is conducted via copying corresponding texture from images to each patch of Delaunay triangle on the projected feature map and synthesizing them together. Then this projected feature map could be projected back onto a sphere based on stereographic projection. Finally spherical CNN follows to extract rotation-invariant features for each spherical feature map. Different from other approaches, our method provides an explicit explanation for combining features from 2D/3D data with less distortion. This framework can be applied to 3D point feature matching, 3D shape alignment and robust re-localization. Furthermore, the rotation-invariant property of our descriptor is also demonstrated by our experiments.

Bibliography

[1] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018.

[2] Guoxian Dai, Jin Xie, and Yi Fang. Siamese cnn-bilstm architecture for 3d shape representation learning. In *IJCAI*, pages 670–676, 2018.

[3] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 195–205, 2018.

[4] James R Driscoll and Dennis M Healy. Computing fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994.

[5] Peter R Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *arXiv preprint arXiv:1806.08756*, 2018.

[6] Friedrich Fraundorfer and Davide Scaramuzza. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robotics & Automation Magazine*, 19(2):78–90, 2012.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[9] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Advances in Neural Information Processing Systems*, pages 4826–4837, 2017.

[10] Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Repeatability is not enough: Learning affine regions via discriminability. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 284–300, 2018.

[11] Pauline C Ng and Steven Henikoff. Sift: Predicting amino acid changes that affect protein function. *Nucleic acids research*, 31(13):3812–3814, 2003.

[12] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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*, pages 652–660, 2017.

[13] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[14] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217. IEEE, 2009.

[15] Samuele Salti, Federico Tombari, and Luigi Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, 2014.

[16] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1744–1756, 2016.

[17] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] Rasha Sheikh, Stefan OBwald, and Maren Bennewitz. A combined rgb and depth descriptor for slam with humanoids. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1718–1724. IEEE, 2018.

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[20] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[21] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 661–669, 2017.

[22] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique shape context for 3d data description. In *Proceedings of the ACM workshop on 3D object retrieval*, pages 57–62. ACM, 2010.

[23] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

[24] Zi Jian Yew and Gim Hee Lee. 3dfeat-net: Weakly supervised local 3d features for point cloud registration. In *European Conference on Computer Vision*, pages 630–646. Springer, 2018.

[25] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 1310–1318. ACM, 2018.

[26] Haoxuan You, Yifan Feng, Xibin Zhao, Changqing Zou, Rongrong Ji, and Yue Gao. Pvrnet: Point-view relation neural network for 3d shape recognition. *arXiv preprint arXiv:1812.00333*, 2018.

[27] Yang You, Yujing Lou, Qi Liu, Lizhuang Ma, Weiming Wang, Yuwing Tai, and Cewu Lu. Prin: Pointwise rotation-invariant network. *arXiv preprint arXiv:1811.09361*, 2018.

[28] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and T Funkhouser. 3dmatch: Learning the matching of local 3d geometry in range scans. *arXiv preprint arXiv:1603.08182*, 3, 2016.

[29] Lei Zhou, Siyu Zhu, Zixin Luo, Tianwei Shen, Runze Zhang, Mingmin Zhen, Tian Fang, and Long Quan. Learning and matching multi-view descriptors for registration of point clouds. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 505–522, 2018.