Neighborhood Normalization for Robust Geometric Feature Learning

Xingtong Liu *[1], Benjamin D. Killeen *[1], Ayushi Sinha[1], Masaru Ishii[2], Gregory D. Hager[1], Russell H. Taylor[1], and Mathias Unberath[1]

[1]Johns Hopkins University
[2]Johns Hopkins Medical Institutions
{xingtongliu, unberath}@jhu.edu

Abstract

Extracting geometric features from 3D models is a common first step in applications such as 3D registration, tracking, and scene flow estimation. Many hand-crafted and learning-based methods aim to produce consistent and distinguishable geometric features for 3D models with partial overlap. These methods work well in cases where the point density and scale of the overlapping 3D objects are similar, but struggle in applications where 3D data are obtained independently with unknown global scale and scene overlap. Unfortunately, instances of this resolution mismatch are common in practice, e.g.*, when aligning data from multiple sensors. In this work, we introduce a new normalization technique, Batch-Neighborhood Normalization, aiming to improve robustness to mean-std variation of local feature distributions that presumably can happen in samples with varying point density. We empirically demonstrate that the presented normalization method's performance compares favorably to comparison methods in indoor and outdoor environments, and on a clinical dataset, on common point registration benchmarks in both standard and, particularly, resolution-mismatch settings. The source code and clinical dataset are available at https://github.com/lppllppl920/NeighborhoodNormalization-Pytorch.*

1. Introduction

Estimating point correspondences between 3D models is a key step in many computer vision tasks, *e.g.* 3D registration, tracking, and scene flow estimation pipelines. 3D descriptors facilitate this process by extracting discriminative and consistent geometric features for estimating correspondences, including hand-crafted features [41, 22, 42, 52], and learning-based descriptors [54, 25, 13, 12, 19]. Learning-based descriptors have gained prominence in recent years, achieving superior performance on common point cloud registration benchmarks [54, 18], but in general, these methods assume that the point density or resolution of different point clouds is the same. This is a reasonable assumption when using voxel grid downsampling to pre-process data since the resolution depends on a fixed voxel size.

However, resolution mismatch can often occur when data are not measured in physical 3D Euclidean space. Reconstructions from monocular videos, for example, are inherently scale-ambiguous. Without additional prior knowledge such as scene overlap ratio between samples, a consistent voxel size for grid downsampling is difficult to obtain, inevitably leading to variable point density across samples. Similarly, stereo camera setups are vulnerable to calibration errors that affect the accuracy of the depth estimation, so that the choice of the camera affects point cloud resolution. Even when physical distances can be estimated or measured from 3D sensors without errors, resolution mismatch can still arise due to hardware discretization, which results in the same scene producing point clouds with different resolutions when captured from different distances. High-resolution samples can be downsampled in this case, but this leads to computation inefficiency and information loss. Consequently, it is desirable to develop 3D geometric feature learning techniques that are robust to resolution mismatch.

We hypothesize that the lack of robustness to resolution mismatch is due in part to the global nature of common normalization methods in convolutional neural networks (CNNs), which are the basis of learning-based descriptors. When two point clouds exhibit resolution mismatch, corresponding regions contain different numbers and distributions of points, making a common feature representation more difficult to learn. This is especially challenging when global scales of samples are unknown, which, without ad-

*These authors contributed equally to this work

ditional prior knowledge, is inevitable in the case of reconstruction from monocular videos and calibration errors described above. By incorporating local point cloud statistics into internal normalization layers, we aim to improve the expressivity of the network across resolutions. We draw further inspiration from the contrast normalization approach present in the human vision system [34], which lends credence to increasing local normalization.

Contributions: Our main contributions are as follows: 1) We present a novel normalization technique, Batch-Neighborhood Normalization (B-NHN), that aims to increase the network's robustness to task-irrelevant mean-std variation of local feature distribution, where resolution mismatch that we try to deal with is a specific data variation potentially causing that. This technique is general and can be applied to any neural network architecture having the concept of convolution over local neighbors. Specifically, on 3DMatch [54] and KITTI odometry [18] datasets for geometric descriptor benchmarking, we show that our method performs favorably against the state-of-the-art on the standard benchmarks, and outperforms previous methods by a large margin on the created resolution-mismatch ones. 2) Additionally, we contribute a dataset of nasal cavities built from CT scans to benchmark the performance of geometric feature extraction methods on a medical video-CT registration task, where resolution mismatch is common.

2. Related Work

2.1. 3D Feature Descriptors

A great deal of research in 3D descriptors focuses on hand-crafted geometric descriptors. Local hand-crafted descriptors often process low-level geometric features, such as location, normal orientation, and curvature, with a hand-crafted algorithmic pipeline. In general, these types of descriptors are robust to partial correspondence but have relatively low distinguishability. Other hand-crafted methods use a global representation of shape, such as a functional map [26, 23, 1, 16, 9, 40], to generate dense correspondences between shapes that may undergo isometric or non-isometric deformation. These methods have better descriptiveness but are often considered unsuitable in presence of partial correspondence.

Early approaches to learning-based descriptors use data-driven parameterization to enhance the performance of hand-crafted descriptors [29, 50, 5, 7] Learning-based methods gained wider popularity with the advent of deep learning, which facilitated both local and global descriptors. Local descriptors [54, 25, 13, 12, 19, 3] focus on extracting feature descriptors from a local patch around the query point and usually have high generalizability across datasets. Global descriptors [10], on the other hand, aim to process the entire 3D data with a neural network in one forward pass, producing element-wise dense feature descriptions. In this branch, many works aim at shape correspondence. Research has explored global 3D architectures for learning deep functional maps which can estimate dense correspondences on shape pairs under various deformations [6, 28, 36, 8, 14, 21]. Some of these works [28, 14] combine deep learning with functional maps and are reasonably robust to partial correspondence, but the requirement of mesh connectivity information renders them inapplicable to point cloud registration. FCGF [10], which uses a global sparse voxelized architecture, aims at the task of sparse correspondence estimation in point clouds, the context of our work, and demonstrated superior performance on recent point cloud registration benchmarks [54, 18].

2.2. Normalization and Fusion Techniques

In recent years, many normalization techniques have been developed. Batch normalization [24], which is widely used, uses mini-batch statistics to approximate the larger distribution during training. It has been shown to reduce the internal covariate shift and smooth out the loss landscape [43], easing the optimization problem. Layer normalization [2] normalizes along all dimensions of a sample. Instance normalization [48] was originally developed for style transfer. It is similar to layer normalization but, instead of normalizing each sample, normalizes each channel in a sample independently. Group normalization [51] normalizes channels as different groups within a sample, tending to perform better than batch normalization in the case of small batch size. Local normalization techniques have also been proposed, such as local response normalization [27] and local context normalization [38], where the value of the center pixel or voxel is normalized using the statistics of its neighborhoods along either channel dimension, spatial dimension, or both. Fusion methods sidestep the issue of normalization selection by combining multiple techniques in a learnable proportion, potentially letting the network use the advantages of each. These fusion techniques include batch-instance normalization [37], switchable normalization [33], and sparse switchable normalization [45], and have shown better performance compared to using a single type of normalization in certain tasks.

3. Methods

In this work, we address the task of sparse point correspondence estimation for point cloud data. We use a modified FCGF [10] as our backbone, introduced in Sec. 3.2. The network takes a single point cloud sample as input and aims to produce point-wise feature descriptions that are consistent and distinctive across samples with overlap. To improve the robustness of the network to resolution mismatch across samples, we focus on developing a

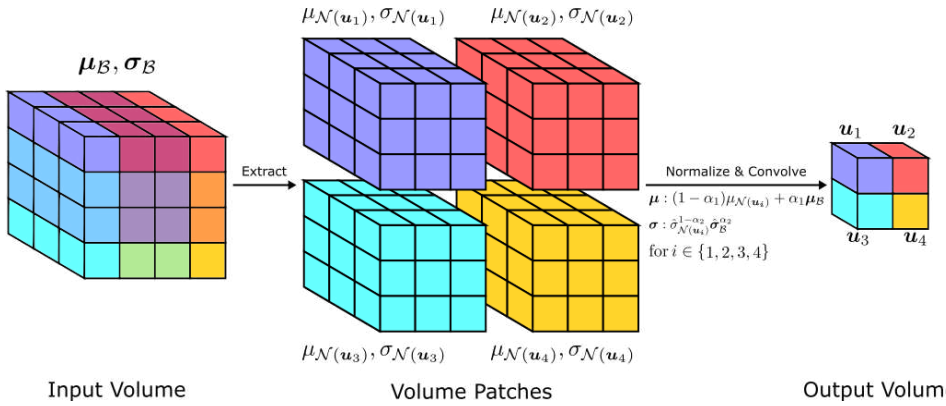

Figure 1. **Procedure of Batch-Neighborhood Normalization with convolution.** We use a dense 3D volume, with the size of 3x4x4, here to demonstrate the procedure handled naively, as in Eq. 4 and 6. The indicated 3D convolution here has a kernel size of 3x3x3 with no padding, and with both stride and dilation as 1. The input and output channel information is not shown here for simplicity. In this figure, the input volume first computes its channel-wise batch-norm statistics, $\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}_{\mathcal{B}} \in \mathbb{R}^{C_{\text{in}}}$. Four 3x3x3 volume patches are then extracted from the input volume with voxel overlap. The individual local statistics, $\mu_{\mathcal{N}(\boldsymbol{u}_i)}, \sigma_{\mathcal{N}(\boldsymbol{u}_i)} \in \mathbb{R}^1$, for each volume patch is then computed. After fusing the mentioned two types of statistics with learnable parameters $\alpha_1, \alpha_2 \in \mathbb{R}^1$, the volume patches are normalized with the corresponding fused statistics and then convolved with the 3D convolution filters to produce the output volume, which has a spatial size of 1x2x2. NHN is a special case of B-NHN where α_1, α_2 are all zeros. Please refer to Sec. 3.1 for the definitions of the symbols above.

new normalization technique, (Batch-)Neighborhood Normalization in Sec. 3.1, to replace the commonly used ones.

3.1. Reducing Resolution-based Variation Through Normalization

Resolution mismatch, *i.e*. point density mismatch in the point cloud, is a specific type of data variation where the number of points per unit volume varies. When the global scales of samples are known, a simple convolution alternate, *e.g*. averaging aggregation mode in a PointConv-based architecture, may already be good enough. However, in this work, we focus on the more challenging case where the scales are unknown and thus the resolution mismatch cannot be even recognized. In this case, the convolutional neural network may need to learn many sets of filters to produce consistent feature descriptions across samples. Because the same actual receptive field can only be likely covered in different network layers for samples with resolution mismatch, the intermediate feature representations in the same layer for these samples will vary. This leads to the potential variation of the mean and standard deviation of features. Therefore, we argue that removing or reducing such variation before any convolution operation may reduce the potential task-irrelevant information and thus improve robustness. It could also help the network distribute more resources on filters focusing on other variations that cannot be handled with normalization, such as spatial orientation. As a result, addressing variation introduced by resolution mismatch directly via normalization will potentially increase its capability of producing consistent features for samples with varying resolution and improve the expressivity of the network.

Based on the concepts above, we propose a new type of local normalization technique, (Batch-)Neighborhood Normalization ((B-)NHN). Unlike the previous local normalization techniques, *e.g*. LRN [27] and LCN [38], that treat the normalization as a standalone module, (B-)NHN is tightly coupled with the subsequent convolution operation, as shown in Fig. 1. Before a convolution kernel is applied to a volume patch, (B-)NHN is first applied to ensure the features in the patch are normalized. In the rest of this section, we describe NHN and B-NHN in the context of sparse voxelized 3D convolution. We refer to NHN and B-NHN as the basis for the NHN-Conv and B-NHN-Conv modules, respectively. In our experiments, we mainly demonstrate the benefit of the proposed normalization in the resolution-mismatch setting, but we emphasize that the technique is general and should also improve the robustness of a network to other types of data variation that affect the mean and standard deviation of local regions.

Neighborhood Normalization. Let $\boldsymbol{x}^{\text{in}}_{\boldsymbol{u}} \in \mathbb{R}^{C_{\text{in}}}$ be a C_{in}-dimensional feature vector in 3D space, at $\boldsymbol{u} \in \mathbb{R}^3$. Denote the convolution kernel weights as $\boldsymbol{W} \in \mathbb{R}^{M \times C_{\text{out}} \times C_{\text{in}}}$, where M is the size of the local neighborhood, and let $\boldsymbol{W_i} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ denote the kernel weights at spatial offset $\boldsymbol{i}$ from center. Thus, the output of a regular voxelized sparse 3D convolution at $\boldsymbol{u}$ is:

$$\boldsymbol{x}^{\text{out}}_{\boldsymbol{u}} = \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \boldsymbol{x}^{\text{in}}_{\boldsymbol{v}} \quad , \tag{1}$$

where $\mathcal{N}(\boldsymbol{u})$ is the local neighborhood of the voxel at $\boldsymbol{u}$. Let $\mu_{\mathcal{N}(\boldsymbol{u})} \in \mathbb{R}^1$ and $\sigma_{\mathcal{N}(\boldsymbol{u})} \in \mathbb{R}^1$ be the mean and standard deviation of the neighborhood, $\mathcal{N}(\boldsymbol{u})$, over both channel

and spatial dimensions.

$$\mu_{\mathcal{N}(\boldsymbol{u})} = \frac{1}{|\mathcal{N}(\boldsymbol{u})| \cdot C_{\text{in}}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \mathbf{1}^T \boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} \tag{2}$$

$$\sigma^2_{\mathcal{N}(\boldsymbol{u})} = \frac{1}{|\mathcal{N}(\boldsymbol{u})| \cdot C_{\text{in}}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \mathbf{1}^T \left(\boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} - \mu_{\mathcal{N}(\boldsymbol{u})}\mathbf{1}\right)^2 \tag{3}$$

where $\mathbf{1} \in \mathbb{R}^{C_{\text{in}}}$ is the all-ones vector. The NHN-Conv operation, described below, consists of neighborhood normalization followed by a convolution.

$$\boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \left(\boldsymbol{\gamma} \circ \left(\frac{\boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} - \mu_{\mathcal{N}(\boldsymbol{u})}}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}} \right) + \boldsymbol{\beta} \right) \quad , \tag{4}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^{C_{\text{in}}}$ and $\boldsymbol{\beta} \in \mathbb{R}^{C_{\text{in}}}$ are per-channel scaling and bias weights, $\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})} = \sqrt{\sigma^2_{\mathcal{N}(\boldsymbol{u})} + \epsilon}$, and ϵ prevents zero division. $\circ$ denotes the Hadamard product, or element-wise multiplication. Although Eq. 4 is intuitively simple, it requires inefficient duplication of memory, since the same input voxel may need to be normalized with different statistics for each overlapping neighborhood window. Fortunately, a reformulation of the equation solves this issue.

$$\begin{aligned} \boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = & \frac{1}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} W_{\boldsymbol{v}-\boldsymbol{u}} \left(\boldsymbol{\gamma} \circ \boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} \right) \\ & - \frac{\mu_{\mathcal{N}(\boldsymbol{u})}}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} W_{\boldsymbol{v}-\boldsymbol{u}} \boldsymbol{\gamma} + \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} W_{\boldsymbol{v}-\boldsymbol{u}} \boldsymbol{\beta} \quad . \end{aligned} \tag{5}$$

The equivalent formulation in Eq. 5 is desirable because it decouples the computation of local statistics, $\mu_{\mathcal{N}(\boldsymbol{u})}$ and $\sigma_{\mathcal{N}(\boldsymbol{u})}$, from the 3D convolution operation, removing unnecessary duplication. Further implementation details are provided in the supplementary material.

Batch-Neighborhood Normalization. Though NHN is well-suited to developing resolution-robust features, it has the inherent property of removing local mean and variance information before applying convolution. In some applications, it may be beneficial to preserve a portion of that, which leads to the introduction of Batch-Neighborhood Normalization (B-NHN). As one might expect, B-NHN fuses the channel-wise batch-norm statistics [24] and the sample-wise statistics of the local neighborhood in a learnable manner. Like NHN, B-NHN is not a standalone operation but is performed along with the subsequent convolution, resulting in the B-NHN-Conv layer:

$$\boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} W_{\boldsymbol{v}-\boldsymbol{u}} \hat{\boldsymbol{x}}_{\boldsymbol{v}}^{\text{in}} \quad \text{, where} \tag{6}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{v}}^{\text{in}} = \boldsymbol{\gamma} \circ \left(\frac{\boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} - (1-\alpha_1)\mu_{\mathcal{N}(\boldsymbol{u})} - \alpha_1 \boldsymbol{\mu}_{\mathcal{B}}}{\hat{\sigma}^{1-\alpha_2}_{\mathcal{N}(\boldsymbol{u})} \cdot \hat{\boldsymbol{\sigma}}^{\alpha_2}_{\mathcal{B}}} \right) + \boldsymbol{\beta} \quad , \tag{7}$$

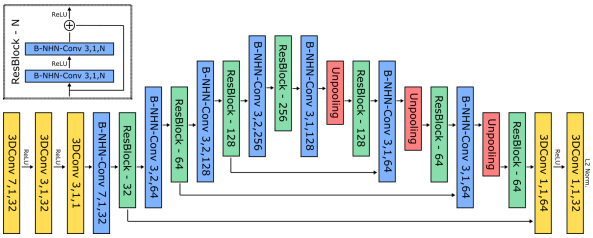

Figure 2. **Overall network architecture.** 3DConv stands for sparse 3D convolution, where the three numbers are kernel size along one spatial dimension, stride size, and output channel size. B-NHN-Conv is defined in Eq. 6 and the numbers have the same meaning as those in 3DConv. ResBlock is a residual block of B-NHN-Conv layers, shown on the upper-left corner. The skipping arrows mean skip connection with a concatenation operation. Unpooling is a nearest neighbor upsampling operation that copies the value of an input voxel to all occupied ones in a 2x2x2 region of the output volume. We replaced the combination of B-NHN-Conv and Unpooling with the transposed version of B-NHN-Conv, in the experiments of the standard 3DMatch benchmark. In KITTI benchmarks, the kernel sizes of the 1st and 4th layers are 5 instead of 7. In other experiments, we used the one in the figure. When NHN is used instead, all B-NHN-Convs are replaced with NHN-Conv layers.

where $\boldsymbol{\mu}_{\mathcal{B}} \in \mathbb{R}^{C_{\text{in}}}$ and $\boldsymbol{\sigma}_{\mathcal{B}} \in \mathbb{R}^{C_{\text{in}}}$ are the batch-norm statistics, and $\hat{\boldsymbol{\sigma}}_{\mathcal{B}} = \sqrt{\boldsymbol{\sigma}^2_{\mathcal{B}} + \epsilon}$. $\alpha_1 \in \mathbb{R}^1$ and $\alpha_2 \in \mathbb{R}^1$ are the learnable fusion parameters that control the portion of batch and neighborhood information in the mean and standard deviation, respectively. Note the vector division is simple element-wise or Hadamard division. Fig. 1 visualizes the B-NHN-Conv operation for multiple neighborhoods in a single-channel input volume. α_1 controls an arithmetic weighting of batch and neighborhood means, and α_2 controls a geometric mean of the standard deviations. The geometric mean here is to enable an efficient reformulation, similar to Eq. 5:

$$\begin{aligned} \boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = & \frac{1}{\hat{\sigma}^{1-\alpha_2}_{\mathcal{N}(\boldsymbol{u})}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} W_{\boldsymbol{v}-\boldsymbol{u}} \left(\frac{\boldsymbol{\gamma}}{\hat{\boldsymbol{\sigma}}^{\alpha_2}_{\mathcal{B}}} \left(\boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} - \alpha_1 \boldsymbol{\mu}_{\mathcal{B}} \right) + \boldsymbol{\beta} \right) \\ & - (1-\alpha_1) \frac{\mu_{\mathcal{N}(\boldsymbol{u})}}{\hat{\sigma}^{1-\alpha_2}_{\mathcal{N}(\boldsymbol{u})}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} W_{\boldsymbol{v}-\boldsymbol{u}} \frac{\boldsymbol{\gamma}}{\hat{\boldsymbol{\sigma}}^{\alpha_2}_{\mathcal{B}}} \quad . \end{aligned} \tag{8}$$

3.2. Network Architecture

Because we specifically confront the resolution mismatch setting, we use a global 3D architecture for descriptor learning. This is because local 3D descriptors usually crop a region with a fixed radius around the point of interest and extract feature representations based on that region. In the case of resolution mismatch with unknown global scales, regions with a fixed radius may contain dramatically different amounts of the actual, real-world volume. A global 3D architecture is better suited to this resolution mismatch

because it has a receptive field that theoretically encompasses the entire volume, regardless of resolution. Intermediate layers in the network can therefore encode both local geometric and global context information of the entire 3D scene. In principle, this allows for encoding similar representations of an object with different resolutions.

FCGF [10] is a voxelized sparse CNN with encoder-decoder architecture and skipping connections, one of the first methods to apply a global architecture to 3D descriptor learning for sparse correspondence estimation. It achieves superior performance on the 3DMatch and KITTI registration benchmark [54, 18], and so we use the FCGF architecture with modification as the primary backbone for our proposed method. Fig. 2 shows the overall network architecture and we refer to this architecture as Mink., short for MinkowskiNet, in the following sections. **Loss function.** Whereas the original FCGF uses hardest contrastive (HC) loss [10], we use the relative response (RR) loss in [30]. We observe that the performance of HC loss is much worse in the resolution-mismatch benchmark than in the standard one. This may be due to the need to dynamically adjust the hyperparameters based on the relative resolution for a given sample pair. For each sampled point correspondence in the ground truth, RR loss compares the source point feature embedding with the features of all points in the target sample, maximizing the similarity between features in the ground truth and the estimated point correspondence while minimizing the similarity of all non-corresponding points.

Let $\mathcal{U}_\mathrm{t}$ be the set of all point locations in the target sample, $\mathcal{G}$ be a random subset of ground truth point correspondences between source and target sample. Let $\boldsymbol{f}^\mathrm{s}_{\boldsymbol{u}_\mathrm{s}}$ and $\boldsymbol{f}^\mathrm{t}_{\boldsymbol{u}_\mathrm{t}}$ be the geometric features of source and target location $\boldsymbol{u}_\mathrm{s}$ and $\boldsymbol{u}_\mathrm{t}$, respectively. The RR loss is expressed as

$$\mathcal{L}_\mathrm{rr} = -\frac{1}{|\mathcal{G}|} \sum_{(\boldsymbol{u}_\mathrm{s}, \boldsymbol{u}_\mathrm{t}) \in \mathcal{G}} \log \left(\frac{\mathrm{e}^{\sigma \boldsymbol{f}^\mathrm{s}_{\boldsymbol{u}_\mathrm{s}} \cdot \boldsymbol{f}^\mathrm{t}_{\boldsymbol{u}_\mathrm{t}}}}{\sum_{\boldsymbol{u} \in \mathcal{U}_\mathrm{t}} \mathrm{e}^{\sigma \boldsymbol{f}^\mathrm{s}_{\boldsymbol{u}_\mathrm{s}} \cdot \boldsymbol{f}^\mathrm{t}_{\boldsymbol{u}}}} \right) - \log \left(\frac{|\mathcal{U}_\mathrm{t}|}{N} \right) \quad (9)$$

where σ is a scale factor to enlarge the value range of feature correlation and N is a constant factor. To account for the fact that the size of the input 3D point cloud could vary, we add a second term, as shown above, to make sure the loss value for samples of various sizes is consistent. This term does not affect the gradient but is important for monitoring progress during training.

4. Experiments

We evaluate our approach on standard benchmark datasets 3DMatch [54] and KITTI [18], as well as on a clinical dataset of nasal cavities, in both the standard resolution setting and the resolution mismatch setting, which we generate. Section 4.3, 4.4, and 4.5 provide greater details about the data processing for each benchmark, respectively; here, we provide a brief overview of the experiments and the resolution mismatch setting generated for each. Fig. 3 shows sample pairs from each dataset in the case of resolution mismatch, with colors representing the learned feature embedding from our descriptor.

First, the 3DMatch [54] dataset contains indoor scenes processed from RGB-D images into point clouds. To generate the resolution mismatch benchmark, we use the same hyperparameters as [54], but instead of sampling points from a TSDF volume, we use the Marching Cubes [32] algorithm to extract a triangular mesh surface, from which we take the vertices as a point cloud. Compared with applying operations such as voxel grid downsampling on point clouds, applying a remeshing operation, *e.g.* ACVD [49], on meshes more accurately simulates the resolution variation encountered in practice. This is because it allows for sampling along the surface while preserving mesh geometry. Second, the KITTI odometry dataset [18] depicts outdoor environments, captured using a lidar sensor. Because no mesh surfaces can be extracted from lidar data, we simply used voxel grid downsampling to mimic the resolution variation in the resolution-mismatch benchmark. We split both KITTI and 3DMatch according to [10].

Finally, we present a dataset of nasal cavity volumes to evaluate our NHN-based descriptor for the application of video-CT registration [4]. To produce this data, we have built a statistical shape model of the entire nasal cavity using 52 CT scans collected from The Cancer Imaging Archive [11]. The shape model was generated using the PCA-based method in [46]. As with 3DMatch, we generate the resolution mismatch variant of this data by applying a grid downsampling after remeshing. The data split was determined using a different range of mode weights of the shape model in the training, validation, and testing phase.

4.1. Training

For each benchmark, we train the FCGF network using the SGD optimizer with momentum 0.9 and cyclic learning rate $\in \left[10^{-4}, 7 \times 10^{-4}\right]$. For the RR loss, we set $\sigma = 20$ and, for each sample, use 10 random positive pairs for loss calculation per iteration. For models with B-NHN, all pairs of α_1 and α_2 are initialized with 0.5 in the standard benchmarks. In the resolution-mismatch ones, two-stage training is adopted. All parameters except α_1 and α_2, which are fixed to 0.0, are trained to convergence; the whole network is then jointly fine-tuned. For all experiments, the network is trained until the validation performance plateaus and the batch size is 4. Data augmentation is different for each dataset and is described below for each result.

4.2. Evaluation Metrics

We use two common evaluation metrics reported in previous works [53, 20, 10]. To evaluate descriptor performance, we report the feature-match recall for each dataset. Additionally, for the KITTI standard benchmark, we evaluate relative translation and rotation errors.

Feature-match recall. Intuitively, feature-match recall measures the percentage of sample pairs where the random sample consensus (RANSAC) method [17] can recover the ground truth pose with high confidence. It is defined as

$$\mathcal{R} = \frac{1}{M} \sum_{s=1}^{M} \mathbb{1}\left(I_s > \tau_2\right) \quad \text{, where} \tag{10}$$

$$I_s = \frac{1}{N_s} \sum_{(i,j)\in\Omega_s} \mathbb{1}\left(\left\|\boldsymbol{T}^* \boldsymbol{x}_i - \boldsymbol{y}_j\right\|_2 < \tau_1\right) \quad . \tag{11}$$

M is the number of sample pairs for evaluation. (i, j) denotes an element of the found correspondence set, Ω_s, of the sample pair s. $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively come from the first and second samples under matching. $\boldsymbol{T}^* \in \mathrm{SE}(3)$ is the ground truth pose. τ_1 is the inlier distance threshold and τ_2 is the inlier recall threshold. Same as [10], for each point of the smaller sample in a pair, the one in the other sample that has the most similar feature description is treated as the found correspondence. No further pruning is applied to the correspondence set. Also, following the convention of [10], we use N_s, the number of points of the smaller sample in s. Because $N_s \geq |\Omega_s|$, this results in a feature-match recall no larger than if Eq. 11 were to have $|\Omega_s|$ in place of N_s.

Relative translation and rotation error. The Relative Translation Error (RTE) and Relative Rotation Error (RRE) measure the registration errors after RANSAC initialization, using the extracted features. This is an indirect measurement of feature quality, which we report for the KITTI standard benchmark, following common convention. RTE is defined as $\left\|\hat{\boldsymbol{t}} - \boldsymbol{t}^*\right\|_2$, where $\hat{\boldsymbol{t}}$ and $\boldsymbol{t}^*$ are the estimated and ground truth translation, respectively. RRE is defined as $\arccos\left((\mathrm{Tr}(\hat{\boldsymbol{R}}^{\mathrm{T}}\boldsymbol{R}^*) - 1)/2\right)$, where $\hat{\boldsymbol{R}}$ and $\boldsymbol{R}^*$ are the estimated and ground truth rotation matrices, respectively.

4.3. 3DMatch

Standard benchmark. First, we evaluate our descriptor on the standard 3DMatch benchmark, as in [54], which assumes all samples have the same resolution. To preprocess 3DMatch, point cloud data are first downsampled with a grid size of $5\mathrm{cm}$. We then apply training augmentations, including full-range random rotation and random scaling $\in [0.8, 1.2]$. In this experiment, we mainly evaluate normalization techniques when used in the state-of-the-art descriptor learning architecture FCGF [10]. These include the commonly used BatchNorm (BN) [24], InstanceNorm (IN) [48], and Batch-InstanceNorm (BIN) [37]; as well as the proposed Neighborhood Normalization (NHN) and Batch-Neighborhood Normalization (B-NHN). We do not include Local Context Normalization (LCN) [38], as described in Sec. 2.2, in this analysis because adapting LCN would require the integral image calculation, an intrinsically serial operation [15], of every internal 3D feature map over the sparse volume data, making it impractical in typical applications for point cloud correspondence estimation. In addition to FCGF, we evaluate the performance of other representative 3D architectures [47, 13, 39, 44] which excel at dense prediction tasks like semantic segmentation. Adapting such architectures to make them suitable as 3D descriptors and thereby enable a fair comparison requires some slight modifications, the details of which are provided in the supplementary material. Results are shown in Table 1, which lists the feature-match recall for all methods at $\tau_1 = 10\mathrm{cm}$ and $\tau_2 = 0.05$. When using other normalization techniques with Mink., such as BN, IN, and BIN, other than NHN and B-NHN, we substitute the B-NHN-Conv modules with the normalization module following with a normal 3D convolution for a fair comparison.

Method	$\tau_2 = .05$	$\tau_2 = .10$	$\tau_2 = .20$	$\tau_2 = .30$
KPConv [47]	0.798	0.517	0.163	0.050
PPNet [31]	0.478	0.250	0.057	0.015
PointNet++ [39]	0.471	0.201	0.026	0.002
DCM-Net [44]	0.001	0.00	0.00	0.00
FCGF [10] (5cm)	**0.935**	0.852	0.613	0.401
Mink.+BN	0.924	0.832	0.588	0.387
Mink.+IN	0.607	0.359	0.136	0.054
Mink.+BIN	0.692	0.422	0.157	0.049
Mink.+NHN	0.866	0.670	0.357	0.166
Mink.+B-NHN	0.933	**0.852**	**0.634**	**0.428**

Table 1. **Evaluation of feature-match recall in the 3DMatch standard benchmark.** All models in this table were trained and evaluated using the point cloud data downsampled with the grid size of 5 cm. The results of FCGF were estimated using the pretrained model provided by [10]. Note that we did not include the results of FCGF with 2.5cm grid size here, which is the state-of-the-art result reported in [10], for a fair comparison. For B-NHN, α_1 and α_2 are 0.63 ± 0.48 and 0.64 ± 0.31, respectively.

Resolution-mismatch benchmark. In this benchmark, we evaluate the performance of 3D descriptors when a sample pair may contain models with different resolutions. During training, we simulate resolution variation by applying ACVD [49] to meshes with random target edge length $\in [3, 30]$ cm. In addition, full-range random rotation was used for training augmentation. Note that in both training and evaluation, the network input is still a point cloud, which consists of vertices of the remeshed sample. All the following experiments involving remeshing operations also converted remeshed samples to point clouds as network input. To generate a sparse 3D volume as input for voxelized sparse CNN architectures (FCGF and Mink.), we further

ϕ	$\{1, 1.5, 2\}$			$\{2.5, 3, 3.5\}$			$\{4, 4.5, 5\}$		
τ_2	.05	.10	.20	.05	.10	.20	.05	.10	.20
KPConv [47]	0.121	0.024	0.002	0.015	0.004	0.000	0.034	0.009	0.003
PPNet [31]	0.043	0.007	0.000	0.074	0.014	0.003	0.141	0.032	0.007
PointNet++ [39]	0.004	0.000	0.000	0.014	0.003	0.001	0.042	0.010	0.002
FCGF [10]	0.421	0.220	0.070	0.354	0.112	0.009	0.414	0.156	0.012
Mink.+BN	0.380	0.174	0.046	0.354	0.127	0.015	0.438	0.185	0.024
Mink.+IN	0.206	0.066	0.009	0.271	0.084	0.013	0.362	0.150	0.025
Mink.+BIN	0.106	0.012	0.001	0.168	0.022	0.001	0.257	0.056	0.003
Mink.+NHN	0.468	0.265	0.085	0.497	0.270	0.059	0.528	0.294	0.064
Mink.+B-NHN	**0.494**	**0.289**	**0.106**	**0.521**	**0.308**	**0.091**	**0.579**	**0.366**	**0.104**

Table 2. **Evaluation of feature-match recall in the 3DMatch resolution-mismatch benchmark.** The same operation is also applied to other tables that show results under the resolution-mismatch setting. Here and in the following benchmarks, unless stated otherwise, "FCGF" indicates that we trained the original architecture in [10] with the same setting as Mink.. For B-NHN, α_1 and α_2 are 0.07 ± 0.12 and 0.04 ± 0.15, respectively.

applied grid downsampling to the point cloud with a grid size equal to the sampled target edge length. This is to simulate the case of resolution mismatch without knowledge of global scales. For methods [39] that use point locations as input features, the samples are further scaled to equalize the average edge lengths to remove the global scale information. The same processing is applied to other resolution-mismatch benchmarks. During evaluation, the target edge lengths were 3 cm and 1.5ϕ cm for a sample pair. ϕ was set to 9 numbers $\in [1, 5]$; for each ϕ value, all testing sample pairs, same as the standard benchmark, were evaluated. The inlier distance threshold τ_1 is set to 3ϕ cm to allow for more error in coarser-resolution samples, thus the results for different sets of ϕ are not directly comparable. We did not evaluate local descriptors for reasons mentioned in Sec. 3.2 and the time-consuming data pre-processing procedures that make training infeasible. Table 2 lists the feature-match recall performance of all evaluated methods under various sample resolution ratio ϕ and inlier ratio threshold τ_2. Note that, in all the resolution-mismatch benchmarks, the results for each set of ϕ are averaged for display purposes. As can be seen, Mink.+B-NHN achieved top performance in all three sets of resolution ratio ϕ. It also shows that PointConv-based methods [47, 31] with averaging aggregation alone cannot handle this task well.

4.4. KITTI

Standard benchmark. As in [53, 10], we use lidar point cloud data and GPS information provided in the KITTI odometry dataset for 3D descriptor evaluation. In the standard benchmark, a point cloud was downsampled with a grid size of 0.3 m. Training augmentation consisted of a single random scaling $\in [0.8, 1.2]$ per sample pair. For evaluation, we report registration performance and feature-match recall, with $\tau_1 = 0.3$ m and $\tau_2 \in \{0.1, 0.2, 0.3\}$. Table 3 lists the performance of 3DFeat [53], FCGF [10] and Mink. with various normalization methods. When evaluating the registration performance, we reduce the maximum times of

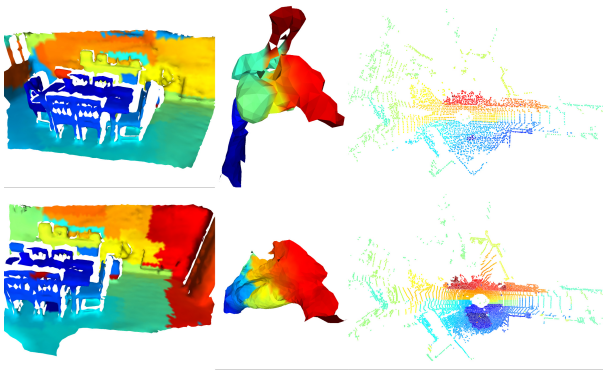

Figure 3. **Visualization of 3D feature descriptions produced with B-NHN.** The displayed sample pairs with mismatched resolutions are from the 3DMatch, clinical, and KITTI datasets. Features were generated by Mink.+B-NHN. For display purposes, each vertex of the meshes in the figure was assigned with the output feature embedding of the spatially closest point in the corresponding point clouds. The feature descriptions were mapped to scalar values with UMAP [35] and displayed in the JET colormap.

validation in RANSAC from 10000 in [10] to 1000 for the ease of experiments. This results in a slight increase in error for all methods. As in Table 3, Mink.+B-NHN outperforms all comparison methods. The notable feature-match recall performance difference between Mink.+BN and FCGF, we believe, is mainly due to the different loss design, where we use RR loss, described in 3.2, for network training instead of hardest contrastive (HC) loss [10]. Potentially, this could also result from the mild point density variation within a downsampled point cloud, since we observe that HC loss performed inferior to RR loss in all resolution-mismatch benchmarks. Because RANSAC [17] is robust to outliers, as in Table 3, the differences of registration performance of all normalization techniques with Mink. are negligible.

Resolution-mismatch benchmark. In this benchmark, we downsampled the point cloud with a random grid size $\in [0.15, 1.5]$ m for training. Feature-match recall was used as the evaluation metric. During evaluation, the target edge lengths were 0.15 m and 0.075ϕ m for a sample pair. ϕ was set to 9 numbers $\in [1, 5]$; for each ϕ value, all testing sample pairs were evaluated. The inlier distance threshold τ_1 is set to 0.15ϕ m to allow for more error in coarser-resolution samples. Results are shown in Table 4.

4.5. Nasal Cavity

Finally, we evaluate 3D descriptors for the task of video-CT registration [4], using our dataset of nasal cavity volumes. This corresponds to one of our envisioned target applications where the resolution-mismatch problem is inevitable without additional prior knowledge. Because large amounts of ground truth data are difficult to obtain for real nasal cavities, the dataset has been built from simulation using a statistical shape model of the nasal cavity, as described

Method	*3DFeat	FCGF	Mink.+BN	˜+IN	˜+BIN	˜+NHN	˜+B-NHN
FMR($\tau_2 = 0.1$)	N/A	0.810	0.912	0.908	0.923	0.928	**0.933**
FMR($\tau_2 = 0.2$)	N/A	0.395	0.569	0.589	0.681	0.775	**0.793**
FMR($\tau_2 = 0.3$)	N/A	0.117	0.076	0.065	0.114	0.241	**0.308**
RRE(°)	0.57	0.283	0.234	**0.232**	0.235	0.243	0.244
STD(°)	0.46	0.314	0.236	**0.205**	0.250	0.238	0.248
RTE(cm)	25.90	8.05	6.48	6.51	6.64	6.63	**6.40**
STD(cm)	26.20	7.76	6.13	5.54	6.07	6.36	**5.48**
Success rate(%)	95.97	**99.10**	98.92	98.92	98.74	98.92	98.92

Table 3. **Evaluation of performance in the KITTI standard benchmark.** The results of 3DFeat are reported in [53]. We evaluated the performance of FCGF using the pre-trained model provided by [10] with grid size 30 cm, the same one as all experiments to its right in this table. STD stands for the standard deviation of the term above it and symbol "˜" represents "Mink.". For B-NHN, α_1 and α_2 are 0.04 ± 0.04 and 0.06 ± 0.04, respectively.

ϕ	$\{1, 1.5, 2\}$			$\{2.5, 3, 3.5\}$			$\{4, 4.5, 5\}$		
τ_2	.05	.10	.20	.05	.10	.20	.05	.10	.20
FCGF [10]	0.939	0.873	0.371	0.944	0.846	0.410	0.945	0.844	0.450
Mink.+BN	0.915	0.643	0.069	0.927	0.760	0.205	0.941	0.802	0.315
Mink.+IN	0.926	0.796	0.229	0.947	0.869	0.461	0.960	0.905	0.556
Mink.+BIN	0.920	0.706	0.100	0.931	0.778	0.249	0.936	0.820	0.361
Mink.+NHN	0.951	0.919	0.685	0.966	0.929	0.804	0.973	0.934	0.814
Mink.+B-NHN	**0.956**	**0.923**	**0.755**	**0.969**	**0.936**	**0.849**	**0.977**	**0.948**	**0.874**

Table 4. **Evaluation of feature-match recall in the KITTI resolution-mismatch benchmark.** For B-NHN, α_1 and α_2 are 0.01 ± 0.06 and 0.07 ± 0.07, respectively.

in Sec. 4. In video-CT registration, only part of the entire surface in the CT scans can be observed using an endoscope. Therefore, nasal passages were manually segmented from the mean model of the nasal cavity to get the indices of vertices in the statistical shape model that belong to nasal passages. Fig. 3 shows an example of a pair of the whole nasal cavity and the right nasal passage. During training, full-range rotation and partial cropping with cropping ratio $\in [0, 0.5]$ were applied. As in Sec. 4.3, ACVD [49] was applied to remesh the data with random target edge length $\in [2, 20]$ mm. As the operation in Sec. 4.3, a grid downsampling is applied to the point cloud after remesing, with a grid size equal to the sampled target edge length. This is to build a sparse 3D volume for Mink. to process. During evaluation, the target edge lengths were 2 mm and ϕ mm for a sample pair. ϕ was set to 9 numbers $\in [1, 5]$; for each ϕ value, 1000 testing sample pairs with random mode weights, full-range rotation, and partial cropping with ratio $\in [0, 0.5]$ were evaluated. We set the inlier distance threshold to $\tau_1 = 2\phi$ mm to allow for more error in coarser-resolution samples. Regarding the mode weight sampling, the first 10 mode weights were uniformly sampled $\in [0, 2.5]$, $[2.5, 3]$, and $[-3, 0]$ standard deviation of PCA results during training, validation, and evaluation, respectively. Table 5 lists the feature-match recall performance of different methods.

5. Discussion

Although NHN and B-NHN perform well on the tasks above, some limitations remain in their use. Compared to global normalization techniques, NHN is sensitive to the sparsity of 3D volume data because of the variability of local neighborhood statistics, an issue which B-NHN mitigates by incorporating a weighted portion of global statistics. This limitation becomes more prominent when applied to sparse voxels, where the local neighborhood is often empty. We also observe that the optimization of α_1, α_2 in B-NHN is prone to get trapped in local optima. We observe that directly optimizing over all parameters in the mismatch-resolution setting yields worse performance compared with the 2-stage training strategy described in Sec. 4.1, a specialized optimization technique could be worth exploring. For now, we do not have a thorough understanding of why (B-)NHN performs well in the evaluated data variation, resolution mismatch. We aim to investigate more deeply whether the increased performance is indeed caused by the aspects described in Sec. 3.1. If all convolution operations involved are changed to group-wise convolution, B-NHN-Conv reduces to LCN [38] under the following conditions: 1) the α_1 and α_2 are fixed to zero; 2) the convolution kernel weights $\boldsymbol{W}$ are fixed to the identity mapping of the input voxel centered within the receptive field of the kernel, to the output voxel. B-NHN-Conv reduces to BN when fixing α_1 and α_2 to one and kernel weights $\boldsymbol{W}$ to the identity mapping. The BN statistics $\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}_{\mathcal{B}}$ can be replaced with those in IN, LN, etc, in which cases B-NHN-Conv reduces similarly.

ϕ	$\{1, 1.5, 2\}$			$\{2.5, 3, 3.5\}$			$\{4, 4.5, 5\}$		
τ_2	.30	.50	.70	.30	.50	.70	.30	.50	.70
FCGF [10]	0.125	0.030	0.002	0.543	0.446	0.315	0.775	0.566	0.511
Mink.+BN	0.078	0.016	0.002	0.499	0.475	0.380	0.526	0.500	0.500
Mink.+IN	0.000	0.000	0.000	0.313	0.146	0.000	0.500	0.500	0.435
Mink.+BIN	0.294	0.178	0.109	0.731	0.551	0.496	0.893	0.671	0.535
Mink.+NHN	0.620	0.373	0.204	0.982	0.876	0.630	0.986	0.925	0.777
Mink.+B-NHN	**0.645**	**0.409**	**0.237**	**0.990**	**0.906**	**0.688**	**0.992**	**0.949**	**0.834**

Table 5. **Evaluation of feature-match recall on the clinical dataset of nasal cavity.** For B-NHN, α_1 and α_2 are 0.001 ± 0.064 and 0.055 ± 0.107, respectively.

6. Conclusion

In this work, we have confronted the challenge of resolution mismatch in 3D point correspondence estimation. To do so, we proposed a new type of normalization, Batch-Neighborhood Normalization, and showed that it increases the robustness of 3D descriptors to point density variation. In empirical experiments, our method surpasses the performance of state-of-the-art models in the resolution mismatch setting and performs favorably in the standard benchmarks. Based on the method design and experiment results, we believe B-NHN is adaptable to other domains that employ convolution, including 2D CNNs and graph neural networks, and likely suitable for other types of data variation. These areas provide interesting directions to explore for future work.

References

[1] Y. Aflalo, A. Dubrovina, and R. Kimmel. Spectral generalized multi-dimensional scaling. *International Journal of Computer Vision*, 118(3):380–392, 2016. 2

[2] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization, 2016. 2

[3] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[4] S. D. Billings, A. Sinha, A. Reiter, S. Leonard, M. Ishii, G. D. Hager, and R. H. Taylor. Anatomically constrained video-ct registration via the v-imlop algorithm. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, editors, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2016*, pages 133–141, Cham, 2016. Springer International Publishing. 5, 7

[5] D. Boscaini, J. Masci, S. Melzi, M. M. Bronstein, U. Castellani, and P. Vandergheynst. Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. In *Computer Graphics Forum*, volume 34, pages 13–23. Wiley Online Library, 2015. 2

[6] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *Advances in neural information processing systems*, pages 3189–3197, 2016. 2

[7] D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, and D. Cremers. Anisotropic diffusion descriptors. In *Computer Graphics Forum*, volume 35, pages 431–441. Wiley Online Library, 2016. 2

[8] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 2

[9] O. Burghard, A. Dieckmann, and R. Klein. Embedding shapes with green's functions for global shape matching. *Computers & Graphics*, 68:1–10, 2017. 2

[10] C. Choy, J. Park, and V. Koltun. Fully convolutional geometric features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8958–8966, 2019. 2, 5, 6, 7, 8

[11] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle, et al. The cancer imaging archive (tcia): maintaining and operating a public information repository. *Journal of digital imaging*, 26(6):1045–1057, 2013. 5

[12] H. Deng, T. Birdal, and S. Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 602–618, 2018. 1, 2

[13] 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*, pages 195–205, 2018. 1, 2, 6

[14] N. Donati, A. Sharma, and M. Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8592–8601, 2020. 2

[15] S. Ehsan, A. F. Clark, N. U. Rehman, and K. D. McDonald-Maier. Integral images: Efficient algorithms for their computation and storage in resource-constrained embedded vision systems. *Sensors*, 15(7):16804–16830, 2015. 6

[16] D. Eynard, E. Rodola, K. Glashoff, and M. M. Bronstein. Coupled functional maps. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 399–407. IEEE, 2016. 2

[17] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. 6, 7

[18] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 1, 2, 5

[19] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5545–5554, 2019. 1, 2

[20] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5540–5549, 2019. 6

[21] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. 2

[22] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan. Trisi: A distinctive local surface descriptor for 3d modeling and object recognition. In *GRAPP/IVAPP*, 2013. 1

[23] Q. Huang, F. Wang, and L. Guibas. Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014. 2

[24] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. 2, 4, 6

[25] M. Khoury, Q.-Y. Zhou, and V. Koltun. Learning compact geometric features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 153–161, 2017. 1, 2

[26] A. Kovnatsky, M. M. Bronstein, A. M. Bronstein, K. Glashoff, and R. Kimmel. Coupled quasi-harmonic bases. In *Computer Graphics Forum*, volume 32, pages 439–448. Wiley Online Library, 2013. 2

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 2, 3

[28] O. Litany, T. Remez, E. Rodola, A. Bronstein, and M. Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5659–5667, 2017. 2

[29] R. Litman and A. M. Bronstein. Learning spectral descriptors for deformable shape correspondence. *IEEE transactions on pattern analysis and machine intelligence*, 36(1):171–180, 2013. 2

[30] X. Liu, Y. Zheng, B. Killeen, M. Ishii, G. D. Hager, R. H. Taylor, and M. Unberath. Extremely dense point correspondences using a learned feature descriptor. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5

[31] Z. Liu, H. Hu, Y. Cao, Z. Zhang, and X. Tong. A closer look at local aggregation operators in point cloud analysis. *ECCV*, 2020. 6, 7

[32] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery. 5

[33] P. Luo, J. Ren, Z. Peng, R. Zhang, and J. Li. Differentiable learning-to-normalize via switchable normalization. In *International Conference on Learning Representations*, 2019. 2

[34] S. Lyu and E. P. Simoncelli. Nonlinear image representation using divisive normalization. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 2

[35] L. McInnes, J. Healy, N. Saul, and L. Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. 7

[36] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5115–5124, 2017. 2

[37] H. Nam and H.-E. Kim. Batch-instance normalization for adaptively style-invariant neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 2558–2567. Curran Associates, Inc., 2018. 2, 6

[38] A. Ortiz, C. Robinson, D. Morris, O. Fuentes, C. Kiekintveld, M. M. Hassan, and N. Jojic. Local context normalization: Revisiting local normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 3, 6, 8

[39] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5099–5108. Curran Associates, Inc., 2017. 6, 7

[40] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. Partial functional correspondence. In *Computer Graphics Forum*, volume 36, pages 222–236. Wiley Online Library, 2017. 2

[41] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009. 1

[42] S. Salti, F. Tombari, and L. Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, 2014. 1

[43] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry. How does batch normalization help optimization? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 2483–2493. Curran Associates, Inc., 2018. 2

[44] J. Schult, F. Engelmann, T. Kontogianni, and B. Leibe. Dualconvmesh-net: Joint geodesic and euclidean convolutions on 3d meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 6

[45] W. Shao, T. Meng, J. Li, R. Zhang, Y. Li, X. Wang, and P. Luo. Ssn: Learning sparse switchable normalization via sparsestmax. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[46] A. Sinha, A. Reiter, S. Leonard, M. Ishii, G. D. Hager, and R. H. Taylor. Simultaneous segmentation and correspondence improvement using statistical modes. In M. A. Styner and E. D. Angelini, editors, *Medical Imaging 2017: Image Processing*, volume 10133, pages 377 – 384. International Society for Optics and Photonics, SPIE, 2017. 5

[47] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 6, 7

[48] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016. 2, 6

[49] S. Valette, J. M. Chassery, and R. Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):369–381, 2008. 5, 6, 8

[50] T. Windheuser, M. Vestner, E. Rodola, R. Triebel, and D. Cremers. Optimal intrinsic descriptors for non-rigid shape analysis. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014. 2

[51] Y. Wu and K. He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 2

[52] J. Yang, Q. Zhang, Y. Xiao, and Z. Cao. Toldi: An effective and robust approach for 3d local shape description. *Pattern Recognition*, 65:175–187, 2017. 1

[53] Z. J. Yew and G. H. Lee. 3dfeat-net: Weakly supervised local 3d features for point cloud registration. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision – ECCV 2018*, pages 630–646, Cham, 2018. Springer International Publishing. 6, 7, 8

[54] 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*, pages 1802–1811, 2017. 1, 2, 5, 6