Robotics and Autonomous Systems 116 (2019) 64–79

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Learning 3D local surface descriptor for point cloud images of objects in the real-world

Ju-Hwan Seo, Dong-Soo Kwon *

Daejeon, Republic of Korea

HIGHLIGHTS

- New point cloud dataset including images of identical scenes at three resolutions.
- A study over algorithms that convert point cloud into voxel representation.
- Feature learning by using domain-adversarial Siamese convolutional neural network.
- Intensive evaluation of surface descriptors is performed on public datasets.

ARTICLE INFO

Article history:
Received 7 September 2018
Received in revised form 25 January 2019
Accepted 18 March 2019
Available online 22 March 2019

Keywords:
3D local surface descriptor
RGB-D sensor
Point cloud
Convolutional neural network

ABSTRACT

Surface descriptors, which represent the surface characteristics of an image numerically, are the fundamental elements in many vision applications. Although traditional surface descriptors that are handcrafted or learned using machine learning techniques have been applied in many different vision applications, some difficulty remains in handling large amounts of noise and variance in 3D data. To resolve this difficulty, recent studies have applied deep learning techniques for the development of surface descriptors. Unlike other techniques based on the complete 3D CAD model or pre-known mesh information of the object, we consider the constraint of the robotic applications in which the information mentioned above is difficult to preload. In this paper, we propose a new 3D surface descriptor that does not require any pre-loaded topological information of the objects or a mesh construction, which may occasionally fail with new or previously unknown objects. Further, we propose a voxel representation that is adaptive to the density of the points, resolving the problem of varying densities of the point cloud data. Finally, we adopt domain-adversarial learning that leads a network to learn the features discriminative for similarity measurements while remaining invariant to different point densities. We gathered approximately 5,000 point-cloud images of objects along with their position and orientation information. We then constructed approximately half a million pairs of point clouds indicating the identical and different parts of the objects, which are labeled as true and false, respectively. The dataset of constructed pairs was used for the learning of 3D surface descriptors using a Siamese convolutional neural network (SCNN) with a domain-adversarial characteristic. The results indicate that the proposed descriptor outperforms other descriptors.

1. Introduction

Many robots operate in structured environments where complete information on the surroundings is provided. This information includes objects, maps, and users. However, as robots expand their field of operation from industry to general society, and their work from task execution to interaction with their surroundings, the operation of robots in newly faced and unstructured environments is becoming a key issue in the field of robotics.

One of the key functions for robots in unstructured environments is related to how they perceive objects that they need to interact with. This includes vision-related issues such as object detection, tracking, and model reconstruction.

For these perception-related issues, the surface or texture descriptors used for matching identical parts of the target objects in images taken from different perspectives are regarded as an important part of the system and have long been an area of study. In the research field related to 2D images, many descriptors [1–3] have been introduced and widely used in various applications.

Similar to the features used for image-based applications, robot systems in 3D environments require descriptors that can be used in 3D environments. The development of 3D surface descriptors, however, is more complex and highly variant because the operations involve three additional axes (one for the position and two for the orientation). Furthermore, 3D sensors commonly

* Corresponding author.
E-mail address: kwonds@kaist.ac.kr (D.-S. Kwon).

https://doi.org/10.1016/j.robot.2019.03.009
0921-8890/

present more noise than 2D sensors, thus increasing the difficulty of developing robust surface descriptors.

A common method to reduce noise is converting a point cloud into a mesh [4,5]. Three-dimensional scanned data occasionally present holes or non-smooth surfaces depending on the conditions of the environment, such as the lighting, illumination, and material properties of the target objects. In such cases, a mesh construction that can fill in holes [6] or smooth the surface [7] can be helpful. In addition to noise reduction, this mesh construction is expected to fill in the lacking topology information in a point cloud. However, differences between a point cloud and its converted mesh may exist, such as incorrectly connected edges and incorrectly filled-in holes, resulting in incorrect surface matching. In particular, without pre-loaded object information, robots may only be able to detect a partial surface of a newly faced object, often leading to an inaccurate mesh construction on partial surfaces.

Therefore, we focus on the development of a surface descriptor that can be directly applied to point cloud data. Motivated by recent successes in the direct learning of features from data using deep learning techniques [8–10], we applied a 3D Siamese convolutional neural network (SCNN) that is capable of coping with such variances in 3D data.

In addition, we address the practical problem of variations in the density of the point cloud for the same surface depending on the resolution of the RGB sensor used for 3D sensing and the distance between the objects and sensor. Fewer points are used to represent the surface of an object located farther from the camera, thus increasing the difficulty to characterize the surface.

We compared the proposed surface descriptor with other descriptors, such as a point feature histogram (PFH) [11], fast point feature histogram (FPFH) [12], rotation-invariant feature transform (RIFT) [13], spin image [14], signatures of histograms of orientations (SHOT) [15], point fair feature (PPF) [16], 3D histogram of point distribution (3DHoPD) [17], local voxelized structure (LoVS) [18], local feature statistics histogram (LFSH) [19], ensemble of shape functions (ESF) [20], viewpoint feature histogram [21], clustered viewpoint feature histogram [22], and 3DMatch [23], which can be applied to point cloud data directly. We found that the proposed descriptor exhibits the best performance when compared with the abovementioned descriptors even for a dataset gathered in a different environment.

2. Related works

2.1. Hand-crafted local surface descriptors in 3D

Hand-crafted local surface descriptors for registration are proposed by many researchers with their own perspectives and ideas. A qualitative review on the detectors and descriptors related to 3D registration methods can be found in [24] and [25]. Among the reviewed descriptors, we focused on those that are applicable to point cloud images. In addition, we added recent advancements based on our own survey.

As discussed in [24], many descriptors are histogram based. Rusu et al. [11] proposed a PFH feature based on the idea of using a histogram of quadruplets, which represents the relations between points in an area of interest of a point cloud image. A year later, the authors proposed the use of FPFH [12], which is a simplified version of PFH with a lower computational complexity. Lazebnik et al. [13] introduced a RIFT descriptor that characterizes a surface near a point of interest by assigning neighbor points to a histogram bin according to their distance and angular position relative to the point of interest. The resulting histogram is used for surface matching. Johnson et al. [14] introduced the spin-image descriptor, which attaches object-oriented coordinates to a surface and characterizes them based on the coordinates. Therefore, it has the advantage of a description of surfaces invariant to a change in viewpoint.

Huang et al. [26] proposed a point cloud matching method based on 3D self-similarity for 3D points achieved from different perspectives or at different instants in time. The authors developed a new 3D local-feature descriptor that can efficiently characterize distinctive signatures of surfaces including the surface normal, curvature, and photometric information. Salti et al. [15] introduced a SHOT descriptor that encodes the histograms of the surface normals within the first-order differential entities. The authors aimed at a more favorable balance between the descriptive power and robustness by concerning both major approaches in the development of surface descriptors, which are signature- and histogram-based.

Prakhya et al. [17] proposed 3DHoPD, which aims at a low computational cost while securing robustness to noise and a competitive performance. It utilizes both the keypoint position in the new 3D space and a histogram of the neighborhood point distribution. Buch et al. [16] proposed the use of a PPF descriptor, which utilizes a histogram of the rotational components that describe the relations between the center point and a neighbor point. Yang et al. [19] proposed the use of an LFSH descriptor that encodes the statistical properties of the local depth, point density, and angles between the surface normals to describe the local shape geometries. The authors of the LFSH descriptor also introduced another descriptor, called triple orthogonal local depth images (TOLDI) [27], which computes the descriptions by concatenating three local depth images captured from three orthogonal views in the local reference frame and rotational contour signature (RCS) [28] that utilizes the 2D contour of a 3D-to−2D projected image of the local point cloud. Quan et al. introduced two hand-crafted descriptors. The first is named a LoVS descriptor [18] and establishes a cubic volume at the keypoint; the cubic volume is then divided into several voxels with binary values depending on their point occupancy. The second is named a rotational silhouette map (RSM) [29] and utilizes multiple silhouette maps generated by rotating and projecting a 3D model multiple times. Guo et al. [30] proposed rotational projection statistics (RoPS) descriptor that utilizes a rotational invariant local reference frame for each point and computes feature vector by projecting 3D points onto 2D planes. Unlike others we surveyed, a mesh should be given to compute RoPS descriptor.

We also studied some global descriptors. Although their main task, describing the whole shape of an object, is slightly different from local descriptors, they also can be used for describing the local surface of an object. Wohlkinger et al. proposed ESF descriptor [20] which is a combination of three shape functions for the point cloud's distances, angles, and area. Rusu et al. [21] introduced VFH descriptor which is made up by a viewpoint direction component and an extended FPFH component. Aldoma et al. [22] introduced CVFH descriptor that divides the object into smooth regions, then compute a VFH for every region to resolve the problem that VFH descriptor is not robust to occlusion. The authors of CVFH also introduced Oriented, Unique and Repeatable CVFH (OUR-CVFH) [31] that expands their previous descriptor by adding a unique reference frame to secure robustness. Whereas above-mentioned descriptors focus on the relations between points, Rodriguez-Sanchez et al. introduced SCurV [32] that exploits flatness, concavity, and convexity of surfaces.

Most studies related to surface descriptors share the common idea that similarities in the spatial distance, orientation, color, point distribution, and surface normal are crucial in associating points or parts from two different images. However, the surface descriptors employed by these studies differ based on which

feature they should be invariant to. RIFT focuses on invariance to a rigid transformation, whereas a spin image focuses on a viewpoint invariance. Therefore, the surface descriptor that exhibits the best performance for a certain application can vary depending on the environment and characteristics of the target application. This is natural because the characteristics of certain target applications cannot be fully considered in the development of such surface descriptors. To overcome this limitation, we focus on learning the features directly from the data rather than considering which characteristic the surface descriptor should be invariant to, or which feature is important when associating the identical parts of an object.

2.2. Learning descriptor

2.2.1. Learning by traditional machine learning methods

Litman et al. [33] introduced bag-of-features shape descriptors that return positive values if paired input shapes are similar, and vice versa. They used a traditional learning method, i.e., optimizing the loss function, with a training dataset including paired inputs labeled as positive if they are similar, and negative if they are dissimilar. Similar to this approach, Gang Hua et al. [34] introduced another descriptor with a different objective function that represents the ratio of variance between non-similar and similar pairs. The weights of the projection matrix that maps raw data into a feature space are trained by optimizing the objective function. Consequently, similar image pairs are close together in the projected feature space, whereas non-similar pairs are far apart.

Although descriptors trained using traditional machine learning methods present advantages compared to hand-crafted descriptors, traditional machine learning methods have a limitation that they cannot deal with large amounts of data [35].

2.2.2. Learning by deep learning methods

Many recent studies on the development of surface descriptors have utilized a convolutional neural network (CNN), which is known to have better generalization power than traditional methods. Moreover, a CNN has been successfully used in many computer vision fields, not only for 2D applications, but also for 3D applications such as 3D shape recognition [36].

In 3D ShapeNets [37], a method that converts a 3D mesh into a binary tensor with a size of 30 x 30 x 30 was proposed. The voxels are assigned a value of 1 if they are inside the mesh, and a value of 0 if they are outside the mesh. These tensors are input into a typical 3D CNN, and the network returns the result of the object classification. Similarly, VoxNet [38] integrates a volumetric occupancy grid representation and a 3D CNN. Their representation assumes an ideal beam sensor model. The status of the voxels, hit or pass through, is computed using 3D ray tracing. Given this information, three different occupancy grids are implemented. In addition, 3D-A-Nets [39] introduces the concept of a multilayer dense representation (MDR) that utilizes stacks of 2D images extracted by slicing a 3D volumetric shape along a specified axis. Huang et al. [40] proposed local descriptor using multi-view convolutional networks. Their proposed network utilizes multiple 2D rendered images of 3D man-made object models as an input to compute a numerical surface description. Feng et al. [41] further expanded this multi-view-based approach by considering the intrinsic hierarchical correlation between views, which is called a group-view convolutional neural network (GVCNN). Deng et al. introduced PPFNet [42], which captures the global context across all local patches in a scene and utilizes the context in computing the final local description. Qi et al. proposed PointNet [43] & PointNet++ [44], which takes the point cloud itself as input for the object classification task. In addition, the networks can be used for a semantic segmentation by utilizing the local and global features from an entire scene simultaneously. Yew et al. introduced 3DFeat-Net [45], which has a three-branch Siamese structure with a triplet loss. To avoid difficulty in collecting a manual annotation of the point matching, the authors utilized GPS-tagged 3D point cloud datasets collected using a Lidar sensor in an outdoor environment. Dewan et al. [46] also applied 3D Lidar scan data for training a Siamese network with a dense block architecture [47]. Zeng et al. introduced 3DMatch [23], which learns local descriptors from the RGB-D Scene Reconstruction database. The authors emphasized the importance of learning descriptors from real-world databases and exhibits a better performance than other approaches [12,14].

Although the above-mentioned descriptors have a commonality in the use of deep learning approaches, their main focuses and scopes are different. First, many of them [37–39] are related to object classification tasks, which require a global description of the shape, for 3D CAD models. Multiview-based approaches [40, 41] aim for a local description, which is similar to our approach, but they also require complete 3D models, which are difficult to construct in advance for most robotic applications. In addition, acquiring multiple views at the same time is a difficult condition to satisfy for most robotic applications. The remaining descriptors have no above-mentioned issues, but some [42–44] utilize global information for a local description, and others [45,46] aim for data acquired by a Lidar sensor in an outdoor environment. Consequently, we concluded that 3DMatch [23] is the most relevant study to our own.

However, applying 3DMatch to data acquired for robotic applications incurs an issue in which the original data representation still requires surface information because the values of the voxels are assigned by computing the minimum distance to the surface. Although they provided a method that converts a point cloud into a representation using source codes [48], this method is based on a simple nearest-point distance-based method and lacks a detailed analysis for the conversion. In addition, their model is trained and analyzed using voxels constructed from 3D mesh data, and not a point cloud alone. This can lead to results not yet optimized for voxels converted from only a point cloud.

2.3. Domain-adversarial learning

Another issue we hope to tackle in the use of deep learning approaches for a surface description is related to a domain shift [49]. Deep networks are known to be able to learn feature representations that are generally useful, especially when trained with sufficiently large datasets. In practice, however, a good performance on large training datasets cannot always guarantee a good performance on a real implementation because of a phenomenon known as a domain shift [49]. Fine-tuning is a known solution to this issue. However, it requires a sufficiently large amount of labeled training data acquired from an environment where implementations are applied, which is a difficult and expensive process.

To resolve this issue, domain adaptation techniques have recently been applied to deep networks. Ganin et al. [50] proposed a domain-adversarial neural network (DANN) that has a gradient reversal layer and an additional domain-classifier at the end of the feature extraction layer. Then, minimizing a loss value related to a main task makes the feature layers more discriminative for the main task while maximizing the loss value related to a domain classifier, making the feature layers more indiscriminative for domain classification. Consequently, the final resultant features can be applied well to different domains. Based on the idea of DANN [50], Pei et al. [51] expanded DANN to a multiple source domain adversarial neural network (MDANN). Besides these theoretical descriptions and their implementation on well-known

datasets such as MNIST [52], the effects of domain-adversarial learning have also been shown in applications such as person re-identification [53] and language understanding [54].

Developing a descriptor using deep networks for robotic applications incurs a similar issue. First, acquiring a sufficiently large pose-annotated image dataset from the implementation environment is a challenging endeavor [23,45]. Second, the characteristics of the data acquired from an implementation environment are varied, depending on the sensors, target objects, sensing distances, and other factors. This means that the domain difference can be large for the same task of describing the object surface. Consequently, a good performance of a descriptor using deep networks trained with a certain dataset cannot guarantee a good performance on other datasets, for instance, images acquired from the target implementation environment of an anonymous developer. For these reasons, we implemented and examined the effects of domain-adversarial learning on a surface describing task.

2.4. Our approach

Overall, the goal of our research is to develop a new local surface descriptor that can be used for objects in 3D environments. The key differences between our method and related approaches are that we consider practical problems when using local surface descriptors in robotic applications. Namely, we focus on developing a new local surface descriptor that does not require pre-analyzed surface information or a mesh construction, and is therefore directly applicable to a point cloud. In addition, we address the problem of varying densities in which the number of points representing certain surfaces differs depending on the conditions of the robotic applications, such as the sensor resolution and the distance between the sensor and object. The first thing we attempted for the problem is density-adaptive voxel representation. The second one is domain-adversarial learning on images of the same objects with a different point density was attempted.

The contributions of our work can be summarized as follows:

- We introduce a new database including information on the surfaces of the objects, as well as their position and orientation. A Kinect v2 sensor was used, and the database includes images of identical scenes at three different resolutions (SD, qHD, and HD). (This database will be opened to the public. Samples can be seen at https://github.com/tclhri.)
- We examine various methods for converting a point cloud into a voxel representation. The proposed voxel representation of a point cloud is adaptive to its point density to resolve the problem of varying densities of point clouds caused by the difference in resolution of the RGB sensor, and the distance between an object and a camera.
- We propose a new algorithm for measuring the point density. From the experiment results, we found that the proposed algorithm exhibits a better performance than the nearest-neighbor point-distance algorithm.
- We applied domain-adversarial learning to descriptor learning to induce a network to learn domain-invariant (in this study, invariant to varying point densities) features. The results of an ablation study indicate that the performance of the descriptor is better with domain-adversarial learning.
- Finally, the proposed surface descriptor outperforms other descriptors including PFH, FPFH, RIFT, Spin Image, SHOT, PPF, 3DHoPD, LoVS, LFSH, ESF, VFH, CVFH, and 3DMatch.

The results of our study are presented in the following sections.

Table 1
Database information. The numbers indicate how many scenes are collected for each object.

Training				Test	
Beanie	180	Bluestick	198	Blackbox	120
Bottle	192	Case	198	Blanket	108
Cup	222	Glove	210	Book	120
Graydoll	78	Handcream	198	Handdrill	152
Headphone	204	Muffler	222	Joystick	117
Nucbox	270	Pillow	204	Microphone	152
Purpledoll	150	Shoes	288	Staplers	126
Slipper	270	Slipper2	258	Thermos	117
Sprayer	312	Umbrella	198	Towel	123
Cup	204	Woodblock	204	Wooden head	120
Woodblock2	192	Xtionbox	234		
		In Total	4743	In Total	1255

3. Database construction

Because our objective is to develop a surface descriptor that can indicate whether two paired point clouds are similar in a supervised manner, we need a training dataset. This dataset must include pairs of two-point cloud images and labels indicating whether they are similar. However, manual labeling requires a huge amount of labor; hence, we need a method to construct point cloud image pairs and label them automatically.

To achieve this, we first prepared a flat board with 12 AR markers and collected point cloud data of scenes including an object on the board. The center points of the 12 AR markers in the camera coordinates were tracked using AR marker tracking provided by [55]. Consequently, a single scene datum consists of a point cloud image and the positions of the 12 AR markers. All scene images were taken from various camera angles and distances to secure the variances in the dataset.

Quantitative information of the collected scene images is summarized in Table 1. For the images of each object, one-third were in SD (512 x 424), one-third were in qHD (960 x 540), and one-third were in HD (1920 x 1080) resolutions. Example images of objects in the training and test datasets are shown in Fig. 1.

From the randomly chosen two point cloud scenes, the amount of transformation between the 12 AR marker positions in the two images can be obtained by applying singular value decomposition (SVD). By applying the transformation acquired through SVD, the surface in the second image can be aligned to the surface in the first image. Aligned images are used to extract the true and false pairs, which indicate the identical and different surfaces of the objects, the entire process of which is illustrated in Fig. 2.

In this random-based pair-set construction, we excluded the points outside the board and the points on the flat board to prevent a surface that is not a part of an object from being included in the pair dataset. In addition, we considered the situation in which certain parts of an object in the first image could not be seen in the second image because the images were obtained from different perspectives. In this case, empty voxels in the second image can be assigned as a true pair of voxels in the first image. To avoid these pairs affecting the training process, a pair was not set if it had an empty voxel.

Using the above-described approach, we constructed five types of dataset. The first, which is the only dataset used for the training of a network, consists of point cloud images of 22 objects captured using a Kinect v2 sensor. For each object, 10,000 true pairs and 10,000 false pairs were constructed such that the dataset had 440,000 pairs. The second and third types consist of point cloud images of ten objects captured using Kinect v2 and Kinect v1 sensors. For each object, 1,000 true pairs and 1,000 false pairs were constructed, resulting in 20,000 pairs. These two datasets were used only for evaluating the networks. Note that,

Fig. 1. Objects in training/test dataset.

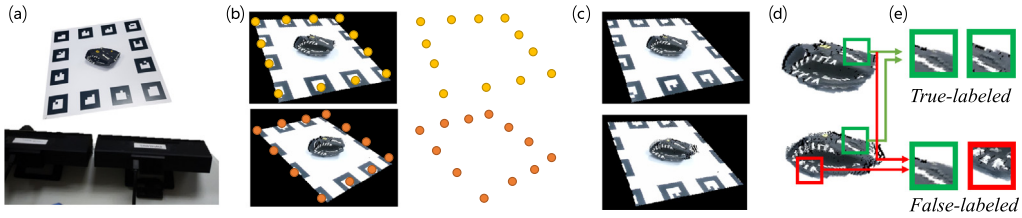


Fig. 2. The overall process of database construction. (a) Point cloud image acquisition. (b) Get the position & orientation of the object from 12 AR markers. (c) Align two images by computing the transformation matrix by applying SVD on 12 points in two images (d) Plane is segmented out (e) True-labeled & False-labeled voxel pair construction.

to assess the performance for novel objects, the objects from the test dataset were not included in the training dataset. Moreover, the third dataset is utilized to evaluate whether the developed surface descriptor can work well with images captured by other sensors. The fourth is a pair dataset using the same approach for images from the TUW database [56]. This database is publicly available and contains point cloud images of objects with pose labels. Specifically, the TUW database includes images of seventeen objects and we constructed 34,000 pairs. This dataset is utilized to assess whether the developed surface descriptor can work well with images captured by other sensors in other environments. Lastly, we constructed a pair dataset using the same approach for images from the Stanford 3D scan repository [57]. The images in the Stanford dataset is acquired by scan devices. Therefore, it may present different characteristics to data of our interest. However, we concluded to use this dataset because performance evaluation on fundamentally different dataset can be beneficial for further analysis. It includes images of five objects and we constructed 5,000 true-labeled pairs and 5,000 false-labeled pairs.

4. Voxel representation

Point cloud data acquired using an RGB-D camera comprise a set of points represented in the form {x, y, z, r, g, b}. This representation is not suitable for use as input data for a neural network because its form depends on the number of points in the point cloud. For this reason, we need to convert them into a suitable form, namely, a 3D voxel, to deal with the network while preserving the characteristics of the data.

First, we attached a voxel grid to the interest point of a point cloud image. The smallest unit of the voxel grid is 0.001 m, and the size of the voxel is 30 x 30 x 30, such that the size of a single voxel in the real coordinates is 0.03 m x 0.03 m x 0.03 m.

When we extracted points in a 0.03 m x 0.03 m x 0.03 m sized volume, we found that only approximately 100 points were included in a single voxel, which is a relatively small number considering there are 30 x 30 x 30 voxels. Typical methods such as LoVS [18] that assign a non-zero value if a point is located in a voxel, and zero otherwise, can lead to an overly sparse voxel. Indeed, nearly 26,900 voxels remained empty in this case, as shown in Fig. 3.(b). In addition, there are empty spaces between points and they may affect describing the surfaces. This is not desirable because only a small portion of voxels was used to describe the surface of the objects, and it is difficult for a network to learn repeating patterns from them.

To resolve this issue, we applied point expansion methods that interpolate empty spaces between points. Consequently, a single point does not occupy a single voxel but multiple voxels depending on its size, as depicted in Fig. 3.

In this expansion process, another issue occurs in that a small expansion may still cause empty spaces between points, and an overly large expansion may cause inefficiency and a deterioration of the discriminativeness. Therefore, the amount of expansion needs be determined based on the resulting point density measurement.

Studies related to this issue have been conducted. In 3DMatch [23], a nearest-neighbor point distance-based method for converting a point cloud into a voxel grid was developed. This method assigns the value of a voxel by computing the distance to the closest point. If a voxel has a point inside it, or sufficiently close, the voxel has a value closer to 1. If a voxel presents a

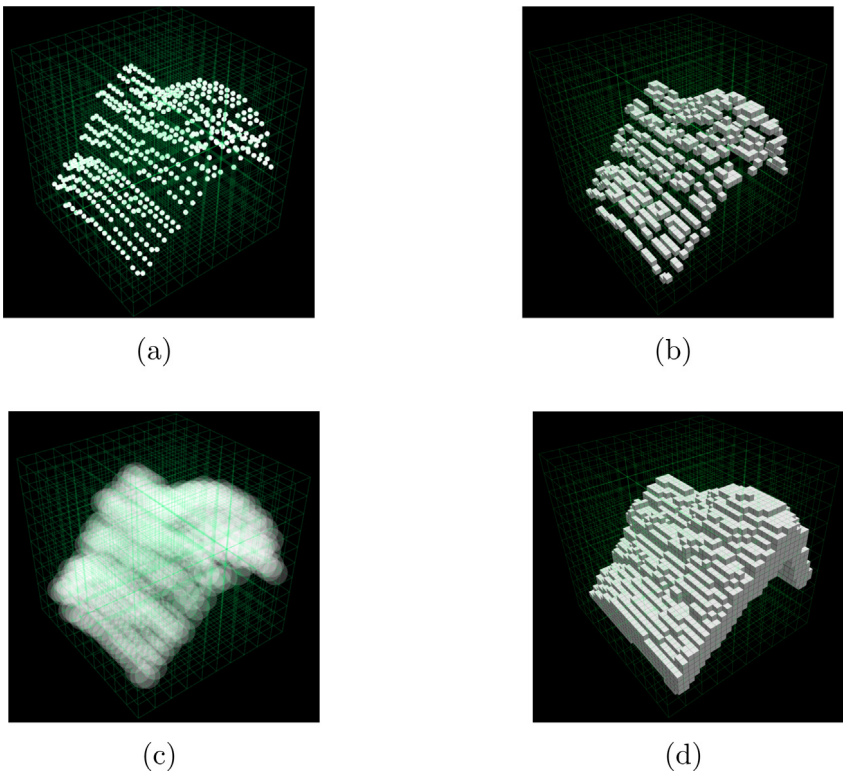


Fig. 3. Illustration of point expansion method and the corresponding voxel representation (a) point cloud without point expansion, (b) voxel occupancy of (a), (c) point cloud with point expansion, (d) voxel occupancy of (c).

distance to the closest point higher than the average of the nearest-neighbor point distance, then the value of the voxel is truncated to zero.

Although this is simple and intuitive, it can occasionally lead to unintended results, particularly when handling a point cloud that has a different resolution on each axis. When the distances between points along a certain axis are smaller than those along other axes, the nearest-neighbor point distance will result in biased values. An example of this is shown in Fig. 4(b), which considers point distances in the x-axis as dominant.

In this study, we propose a new method for measuring the point density. The basic idea is to find the closest point in four directions, namely, +x, −x, +y, and −y, in the XY space. The distances in each direction are then measured. By doing so, the measurement is less biased for a certain axis. Furthermore, unlike the nearest-neighbor point distance-based approach, it is possible to measure the axial resolutions. The detailed algorithm is summarized in Algorithm 1, and a visual representation of the method is shown in Fig. 4.

The reason why we choose the XY space rather than the XZ or YZ space is that the nearest relations between points in the XY space are preserved in the 3D space because a point cloud is constructed by assigning depth values to the pixels of RGB images that lie in the XY space. In Fig. 4, we visualize how the nearest relations between points in the XY, XZ, and YZ spaces are associated with the relations between points in a 3D space. As indicated in the figure, the XZ and YZ spaces show undesirable links, whereas XY does not.

In addition to the new point-density measuring method, we examined the effects of two point-expansion methods. The first is a spherical expansion that expands a point isotropically, i.e., the amounts of expansion in each axis are identical. The second is an ellipsoidal expansion that expands a point anisotropically, i.e., the amounts of expansion in each axis are different. Detailed

Algorithm 1 Proposed Point Density Measuring Algorithm

```
mX,mY,mZ ← 0,   average distance in x, y, z axis
N ← The number of points in a voxel
n ← 0
while n ≤ N do
    px, py, pz ← The position of point i
    qx, qy, qz ← The position of the closest point
        from point i in XY plane in +x direction
    mX ← mX + |px − qx|
    mZ ← mZ + |pz − qz|

    qx, qy, qz ← The position of the closest point
        from point i in XY plane in -x direction
    mX ← mX + |px − qx|
    mZ ← mZ + |pz − qz|

    qx, qy, qz ← The position of the closest point
        from point i in XY plane in +y direction
    mY ← mY + |py − qy|
    mZ ← mZ + |pz − qz|

    qx, qy, qz ← The position of the closest point
        from point i in XY plane in -y direction
    mY ← mY + |py − qy|
    mZ ← mZ + |pz − qz|

    n ← n+1
end while
mX ← max(0.001,mX/2N)
mY ← max(0.001,mY/2N)
mZ ← max(0.001,mZ/4N)
return mX, mY, mZ
```

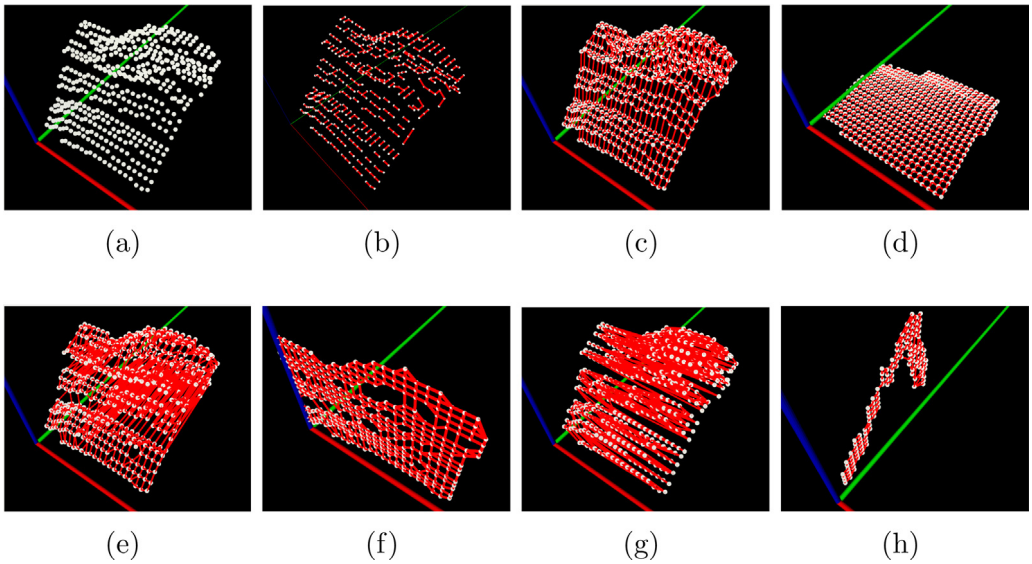


Fig. 4. (a) Point cloud in a voxel. (b) Visualization of the nearest neighbor points. The majority of links are in a certain direction. (c) Visualization of the proposed method. Links are constructed between neighbors in four directions. (d) Visualization of the nearest points in the XY plane. (e,f) Visualization of the nearest points in the XZ plane and 3D space. (g,h) Visualization of the nearest points in the YZ plane and 3D space. *red: *x*-axis, green: *y*-axis, blue: *z*-axis . (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

algorithms for both methods are described in Algorithms 2 and 3.

Finally, we tested seven types of representations to determine the most effective one. The first is a raw representation in which only a 1^3 mm^3 voxel that has a point in it can have a value of 1, while the other values remain at zero. This voxel representation is the same as that of LoVS [18]. The second is a spherical expansion, which is described in Algorithm 2, using the nearest-neighbor point distance method. Note that this voxel representation is identical to that of 3DMatch [23]. The third is a spherical expansion using the proposed point-density measuring method. A comparison between the second and third types indicates whether the proposed point-density measuring method can lead to a better performance. The fourth is an ellipsoidal expansion, which is described in Algorithm 3, using the proposed point-density measuring method. This is included in a candidate set to evaluate whether an expansion of different amounts in each axis can lead to a better performance. The fifth, sixth, and seventh types are binary versions of the second, third, and fourth representations. They only have binary values of zero or 1 in the voxels. Binarization of the voxels is expected to simplify the patterns and lead to easier training, although with a sacrifice of the descriptiveness. In addition, it is advantageous in memory efficiency. Whether binarization is a good choice will be examined experimentally.

All voxel representations are shown in Fig. 5 and summarized in Table 2 using abbreviations that will be used in the following sections. Note that an ellipsoidal expansion with the nearest-neighbor point distance method is out of the candidate set because the method cannot measure the axial distances.

Algorithm 2 Spherical Expansion

if Nearest Neighbor Point Distance **then**
  nnDist ← average nearest point distance
else if Proposed Method **then**
  mX,mY,mZ ← average distance in x, y, z axis
    acquired by Algorithm 1
end if
for $i \in \{1, \ldots, 30\}$ **do**
  for $j \in \{1, \ldots, 30\}$ **do**
    for $k \in \{1, \ldots, 30\}$ **do**
      $V = \{vx, vy, vz\}$ ← The center of Voxel(i,j,k)
      $W = \{wx, wy, wz\}$ ← The position of the
        closest point from V
      $D = \{(vx\text{-}wx),(vy\text{-}wy),(vz\text{-}wz)\}$
      if Nearest Neighbor Point Distance **then**
        $Voxel(i,j,k) \leftarrow max(0, 1\text{-}\frac{\|D\|_2}{nnDist})$
      else if Proposed Method **then**
        $r \leftarrow \sqrt{mx^2 + my^2 + mz^2}$
        $Voxel(i,j,k) \leftarrow max(0, 1\text{-}\frac{\|D\|_2}{r})$
      end if
    end for
  end for
end for

Table 2
There are seven representations; R: raw, SN: Sphere Nearest Neighbor, SNB: Sphere Nearest Neighbor binary, SP: Sphere proposed, SPB: Sphere proposed binary, EP: Ellipsoid proposed, EPB: Ellipsoid proposed binary.

	Point expansion	Point density measurement	Binarization
R	N	–	Y
SN	Spherical	Nearest neighbor	N
SNB	Spherical	Nearest neighbor	Y
SP	Spherical	Proposed(Algorithm 1)	N
SPB	Spherical	Proposed(Algorithm 1)	Y
EP	Ellipsoidal	Proposed(Algorithm 1)	N
EPB	Ellipsoidal	Proposed(Algorithm 1)	Y

5. Learning 3D surface descriptor

Using a constructed database and a proposed adaptive voxel representation, we trained a network with two characteristics: a ① Siamese structure and ② domain-adversarial learning. The first, a Siamese-structured network, has two branches that share the same architecture and weight values. Paired inputs are separated during the first stage, and each input is then fed into a branch of the network. Because the two branches have the same weight values, two input images are mapped into the same feature space. The contrastive loss function at the end of the network allows the network to learn a feature space that shortens the distance

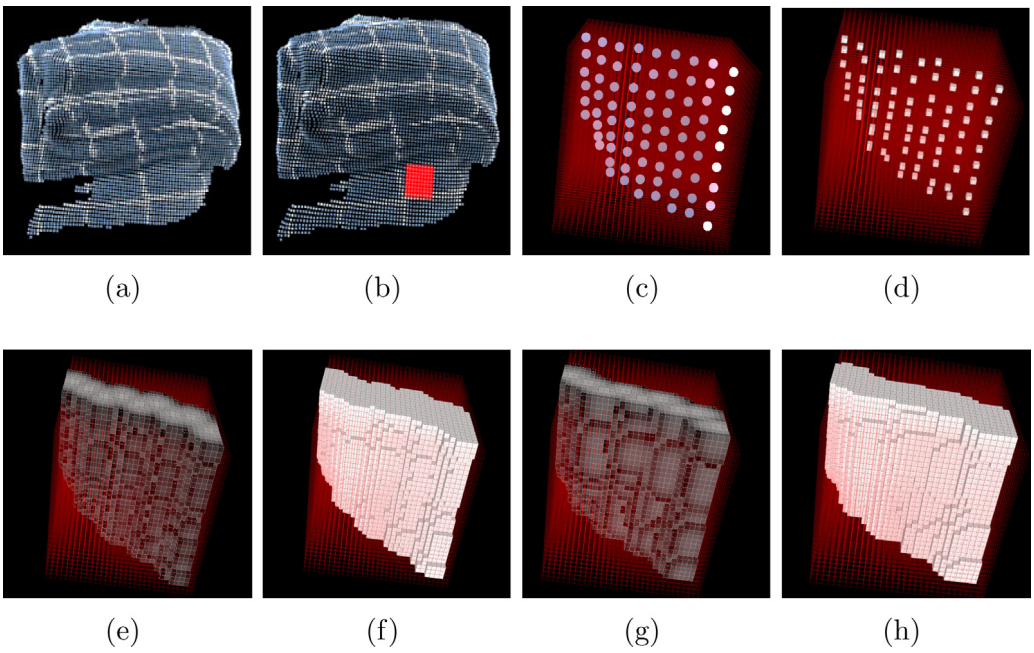


Fig. 5. (a) a point cloud. (b) An interest area is assigned. (c) points in the interest area. (d) voxel representation without point expansion. (e) spherical expansion. (f) spherical expansion with binarization (g) ellipsoidal expansion. (h) ellipsoidal expansion with binarization. *In this example image, ellipsoidal expansion shows thin and less bumpy surface.

Algorithm 3 Ellipsoidal Expansion

```
mX,mY,mZ ← average distance in x, y, z axis
for i ∈ {1, ..., 30} do
    for j ∈ {1, ..., 30} do
        for k ∈ {1, ..., 30} do
            V = {vx, vy, vz} ← The center of Voxel(i,j,k)
            W = {wx, wy, wz} ← The position of the
                               closest point from V
            Scale ← ∛( (√(mx²+my²+mz²))³ / (mx·my·mz) )   .... *
            {mX, mY, mZ} ← Scale · {mX, mY, mZ}
            D = {(vx − wx)/mX, (vy − wy)/mY, (vz − wz)/mZ}
            Voxel(i,j,k) ← max(0, 1-||D||2)
        end for
    end for
end for
```

*Scaling is implemented to make the volume of a sphere with a radius $\sqrt{x^2+y^2+z^2}$ be the same as the one of an ellipsoid with $\{x, y, z\}$ as the radiuses of the principal axes

between two similar inputs and lengthens the distance between different inputs. In a contrastive loss function, the label (true or false) is represented by the symbol y, the Euclidean distance between the feature vectors extracted by each branch is d, N denotes the batch size, and *margin* is a hyperparameter that controls how far dissimilar pairs should be.

The second is a domain-adversarial characteristic. A domain classifier that aims to discriminate whether inputs are from SD, qHD, or HD images is added to the end of the convolutional layers. Typical cross entropy with the softmax loss function is used to optimize the classifier. Because the purpose of the network is to learn a feature representation that is discriminative to a surface description while invariant to the domain-difference, the feature layers are optimized in the direction of minimizing the contrastive loss while maximizing the cross-entropy loss of the domain classifier. The detailed structure and parameters used for further experiments are shown in Fig. 6. In the figure, L_s is a contrastive loss function and L_{di}s are typical cross-entropy function. The structure of the feature extraction network is 'conv3d(32,7,1,0)-conv3d(32,7,1,0)-conv3d(32,5,1,0)-conv3d(32,5,1,0)-conv3d(32,5,1,0)', where conv3d(k, n, s, p) means a convolutional layer consists of n weight filters with the size k^3, the stride parameter s, and the padding parameter p. The structure of the similarity metric network is 'fc(2048)-fc(512)-fc(256)' and the structure of the domain discriminator network is 'fc(2048)-fc(512)-fc(256)-fc(3)', where fc(n) means a fully-connected layer has n nodes.

In addition to the parameters related to network structures, there is another parameter λ that adjusts the balance between the performance on a main task and the domain-invariance. If λ is equal to zero, a network cannot learn the domain-invariant feature representation. In contrast, if λ is too high, a network becomes invariant to the domain differences, but there is a possibility of deteriorating the performance of the main task. Therefore, we checked the performance of networks using five values of λ (0.001, 0.01, 0.03, 0.05, and 0.1), the results of which are summarized in the experiment section.

The implementation details are as follows. Networks were constructed using TensorFlow [58] and trained using Nesterov's accelerated gradient (NAG) [59] with a batch size of 200. The learning parameters were 0.0001 for the initial learning rate, 0.95 for the learning rate decay per epoch, 0.99 for the momentum, and 1.0 for *margin* in the contrastive loss function. The weights of the networks were initialized using the Xavier initialization algorithm [60]. We tested several structures and hyper-parameters.

6. Experiment and discussion

We conducted four experiments. The first is a comparison between the voxel representation candidates listed in Table 2. Based on the experiment results, we will assess whether the proposed point-density measuring algorithm can help improve the performance of the surface descriptor. Moreover, we will evaluate

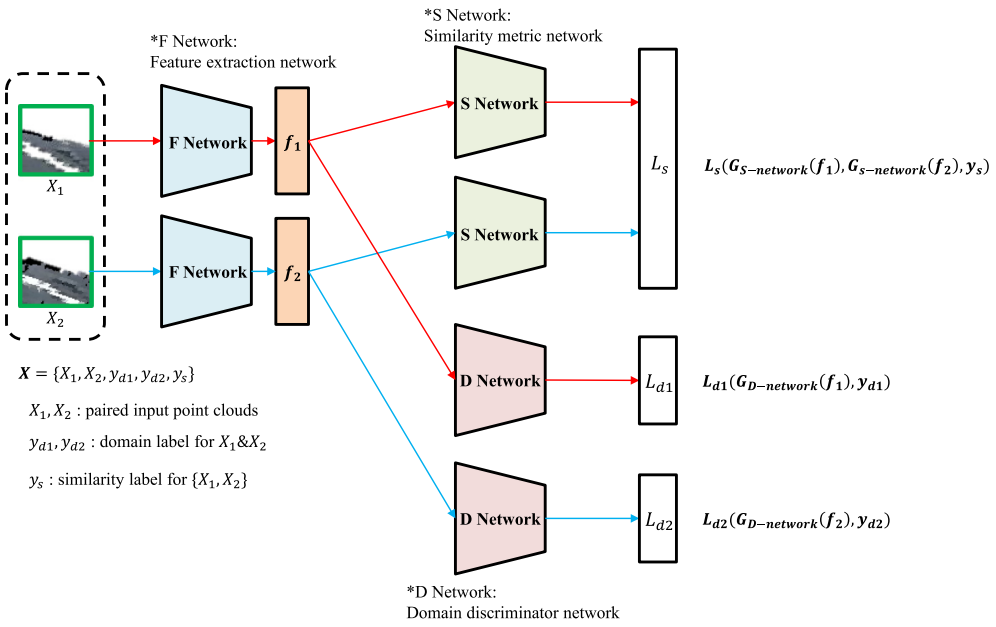


Fig. 6. The network structure used for experiments.

which point expansion method, either spherical or ellipsoidal, exhibits a better performance.

The second experiment is an ablation study on domain-adversarial learning. The purpose of the experiment is to verify whether domain-adversarial learning is effective in making a surface descriptor discriminative for a surface describing task, but invariant to the domain difference so that the performance of the network can be robust to various point density. In detail, we trained networks while varying the parameter λ.

The third experiment is a comparison with other 3D local surface descriptors, namely, PFH, FPFH, RIFT, Spin Image, SHOT, PPF, 3DHoPD, LoVS, LFSH, ESF, VFH, CVFH, and 3DMatch. The former eight descriptors, namely PFH, FPFH, RIFT, Spin Image, SHOT, ESF, VFH, and CVFH, are implemented through the PCL library [61]. The remaining descriptors were implemented using the source codes provided by the authors.

The last experiment was conducted to evaluate the robustness of the descriptors to noise. Based on the Stanford 3D repository dataset [57], which is one of public point cloud datasets, we applied Gaussian noise to the original images at three difference levels and checked the performances of the descriptors.

In the third and fourth experiments, which involves descriptors proposed from other works, an issue occurs in that most of the hand-crafted descriptors utilize a spherical-shaped interest volume, whereas LoVS, 3DMatch, and our proposed approach all utilize a cubic-shaped interest volume. Therefore, point sets in the interest volume of an interest point can be varied depending on the descriptors. For this issue, we set the size of a cubic-shaped interest volume as 0.03 m^3 and the radius of the spherical-shaped interest volume as $\frac{\sqrt{3}}{2} \cdot 0.03\ \text{m} = 0.026\ \text{m}$, which is the radius of a sphere that circumscribes the 0.03 m^3 cubic-shaped interest volume.

In the case of 3DMatch, there is another issue in that the original 3DMatch uses a voxel representation spanning 0.3 m^3 because it was initially proposed for larger indoor structures. Because applying a 0.3 m^3 sized voxel could be unfair for the small-sized objects used for this experiment, the evaluation of 3DMatch was conducted using the same scale of 0.03 m used by the proposed descriptor.

The test datasets were constructed using the method described in Section 3. They consist of pairs of point cloud images in a 0.03 m^3 volume. If paired point clouds indicate the identical part of an object, the pair is labeled true. Otherwise, the pairs are labeled false. The ratio between the amount of true- and false-labeled pairs is 50:50.

The performance of the trained surface descriptor was evaluated using the receiver operating characteristic (ROC) curves. For drawing the ROC curves, we used the Euclidean distance between two feature vectors extracted from each pair of images as a similarity score. If a descriptor is ideal or good enough, similarity scores for true-labeled pairs are close to zero, whereas those for false-labeled pairs are large. This means that the descriptor is discriminative in comparing the similarity, and that the ROC-related measurements will be good. If the performance of a descriptor is not good, the numerical difference between similarity scores for true-labeled and false-labeled pairs will not be large, which means bad discriminative power.

From the drawn ROC curves, we computed the area under the curve (AUC) and the false positive rate at a 95% true-positive rate (FPR@95), which are widely used to evaluate the surface descriptors [14–16,23,28,32,62,63]. Higher AUC values and a lower FPR@95 indicate a better description. Furthermore, we computed the highest F1-score, which is a combined measurement of the true-positive rate and false-positive rate. A higher F1-score indicates a better performance.

6.1. Evaluation of voxel representations

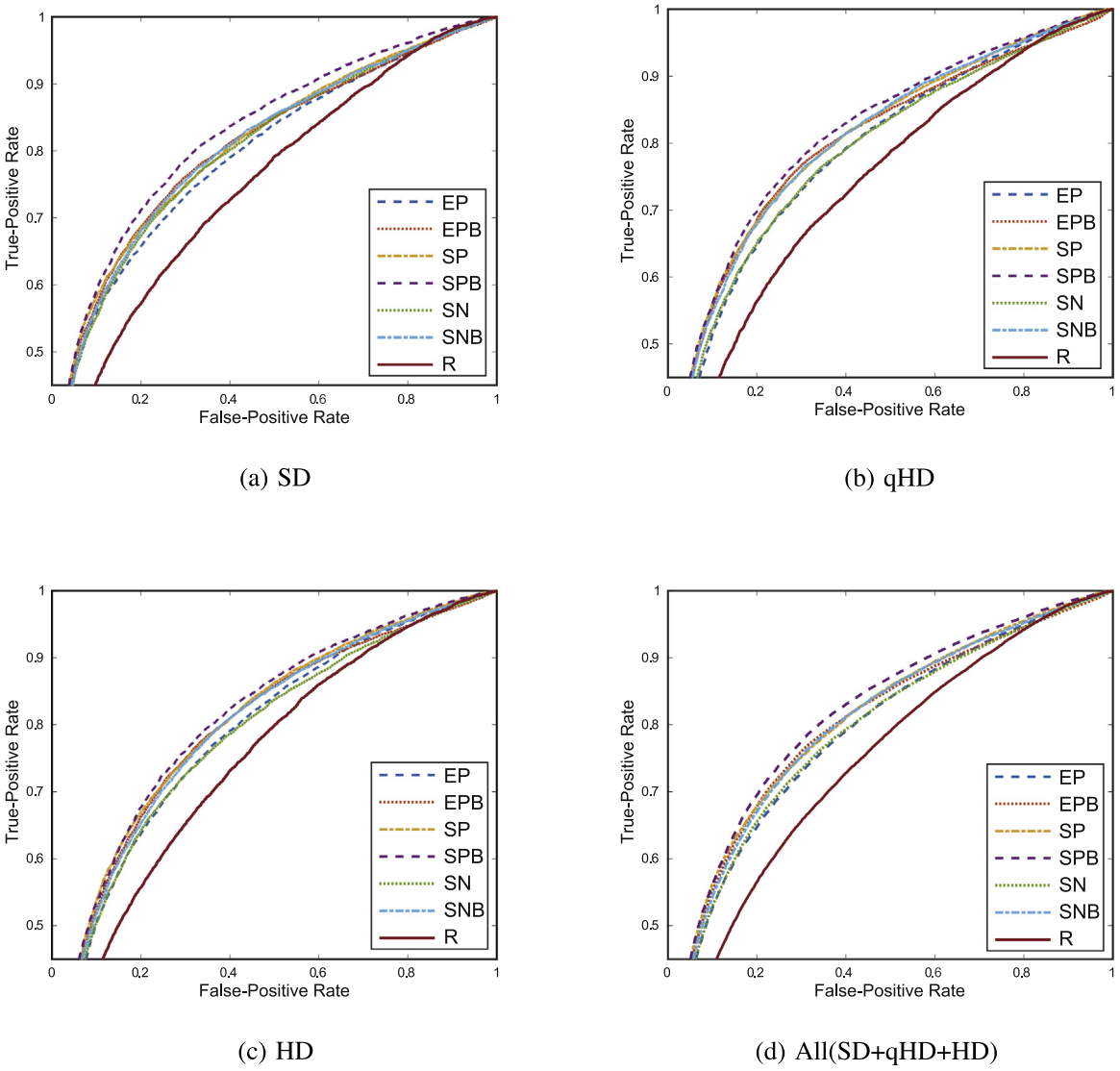
The evaluation results of the first experiment are summarized in Fig. 7 and Table 3.

Q. Which point density measuring method is better?

The comparisons between SN & SP and SNB & SPB indicate the answer to this question. In total, there are 24 cases: two comparisons (SN & SP and SNB & SPB) x three criteria (AUC, F1-score, and FPR-95) x four test datasets. For all the cases, the proposed method exhibits a better performance than the other approaches. On average, the improvement of the proposed method is 0.0157 (1.57%) for AUC, 0.0108 (1.08%) for the F1-score, and 0.0339 (3.39%) for FPR-95.

Q. Which point expansion method is better?

As the results show, all types of expansion method achieve a better performance than non-expansion methods. As mentioned



(a) SD (b) qHD

(c) HD (d) All(SD+qHD+HD)

Fig. 7. Performance evaluation of voxel representations. Each graph corresponds to the test data (a) SD, (b) qHD, (c) HD, (d) All(SD+qHD+HD).

Table 3
Numerical measurements of voxel representations. There are four test dataset; S: SD resolution, q: qHD resolution, H: HD resolution, A: All(SD+qHD+HD). The bolded text means the best.

		R	SN	SNB	SP	SPB	EP	EPB		R	SN	SNB	SP	SPB	EP	EPB
	AUC	0.7479	0.8027	0.8061	0.8090	**0.8267**	0.7962	0.8065		0.7401	0.7862	0.8055	0.8073	**0.8166**	0.7880	0.8017
S	F1-score	0.6910	0.7323	0.7369	0.7325	**0.7549**	0.7226	0.7392	q	0.6916	0.7249	0.7385	0.7373	**0.7496**	0.7234	0.7426
	FPR95	0.8184	0.7991	0.7975	0.7936	**0.7494**	0.8155	0.8126		0.8265	0.8321	0.7938	0.7808	**0.7676**	0.8022	0.8289
	AUC	0.7449	0.7823	0.7968	0.8036	**0.8104**	0.7863	0.7977		0.7438	0.7901	0.8025	0.8066	**0.8176**	0.7901	0.8019
H	F1-score	0.6995	0.7201	0.7336	0.7339	**0.7422**	0.7228	0.7350	A	0.6935	0.7248	0.7356	0.7341	**0.7485**	0.7224	0.7386
	FPR95	0.8115	0.8086	0.7771	0.7648	**0.7501**	0.7850	0.8086		0.8190	0.8150	0.7908	0.7801	**0.7561**	0.8011	0.8152

earlier, and as depicted in Fig. 3(a), these results can be interpreted as the networks facing more difficulty when learning repeating patterns with no point expansion.

Moreover, whether a spherical expansion is better than an ellipsoidal expansion is confirmed through a comparison between SP & EP and SPB & EPB. Again, there are 24 cases: two comparisons x three criteria x four test datasets. In all cases, a spherical point expansion exhibits a better performance. On average, the amounts of difference between spherical and ellipsoidal expansions are 0.0162 (1.62%) for AUC, 0.0108 (1.08%) for the F1-score, and 0.0408 (4.08%) for FPR95.

The proposed point-measuring method enabled us to test an ellipsoidal point expansion. In this case, however, the method showed a worse performance. We suppose that this phenomenon emerges because the axial density is not equal over all areas of a point cloud. As a result, it is less representative compared to a spherical expansion. As shown in Fig. 4, some areas require a z-axis dominant expansion, whereas another area requires a y-axis dominant expansion, although both areas belong to the same point cloud. For deeper research into this phenomenon, the implementation and experiments regarding the effects of an ellipsoidal expansion based on the axial density near the points of interest, rather than for the entire area, will constitute future work.

Table 4
Ablation study on domain-adversarial learning. The bolded text means the best.

λ		w/o DAL	0.001	0.01	0.03	0.05
S	AUC	0.8267	**0.8388**	0.8366	0.8257	0.8228
	F1-score	0.7549	**0.7706**	0.7655	0.7525	0.7505
	FPR95	0.7494	**0.7413**	0.7415	0.7427	0.7502
q	AUC	0.8166	0.8129	**0.8237**	0.8168	0.8161
	F1-score	0.7496	0.7425	**0.7534**	0.7470	0.7443
	FPR95	0.7676	0.7739	0.7623	0.7686	**0.7590**
H	AUC	0.8104	0.8076	**0.8157**	0.8102	0.8055
	F1-score	0.7422	0.7405	**0.7462**	0.7398	0.7407
	FPR95	0.7501	0.7485	**0.7433**	0.7537	0.7538
A	AUC	0.8176	0.8198	**0.8252**	0.8175	0.8145
	F1-score	0.7485	0.7501	**0.7545**	0.7452	0.7449
	FPR95	0.7561	0.7554	**0.7492**	0.7552	0.7563

Q. Does binarization of voxel help to achieve better performance?

This third question can be answered by comparing SN & SNB, SP & SPB, and EP & EPB. Interestingly, for 12 comparison cases for each criterion, binarization helped achieve better AUC values in 12 cases, a better F1-score in 12 cases, and a better FPR95 in 9 cases. On average, binarization helped achieve improvements of 0.0118 (1.18%) for AUC, 0.0137 (1.37%) for the F1-score, and 0.0108 (1.08%) for FPR95.

As mentioned earlier, binarization is expected to simplify the patterns although at a sacrifice in detail. As shown in Fig. 5(e) and (g), frontal surfaces show both white areas where the voxel value is high and black areas where the voxel value is low. This leads to bumpy surfaces, whereas binarization leads to more flattened surfaces, as shown in Fig. 5(f) and (h). Because these bumpy surfaces are caused by a point expansion method, not by a characteristic of the real surface of an object, in most cases, we interpret these experiment results as the binarization possibly helping smooth the voxel surfaces and weakening the effects of un-intentionally generated cracks, thereby leading the network to extract more proper features.

Based on the result of the first experiment, further experiments were conducted using a voxel representation SPB, which shows the best performance.

6.2. Ablation study on domain-adversarial learning

To evaluate the contribution of domain-adversarial learning for a surface description, we compared the performances of networks with and without domain-adversarial learning. We tested five values of λ to find the optimal balance between the surface description and domain-invariance. The results are summarized in Table 4.

As Table 4 shows, domain-adversarial learning with $\lambda = 0.01$ achieves a better performance than the other cases. In further experiments, for a comparison with other descriptors and when evaluating the robustness to noise, we used a model trained using $\lambda = 0.01$.

6.3. Evaluation of the proposed and other surface descriptors

Q. Can the proposed descriptor outperform than others?

In this experiment, we compared the performance of the proposed descriptor with other 3D surface descriptors, namely, PFH, FPFH, RIFT, Spin Image, Shot, PPF, 3DHoPD, LoVS, LFSH, ESF, VFH, CVFH, and 3DMatch. The results are summarized in Fig. 8 and Table 5. Because evaluating the performance of descriptors on Kinect v2 dataset, that is used for training ours, is unfair, we also prepared three different datasets such as Kinect v1, TUW, and Stanford. Those are not used in the training process.

Along six test datasets excluding Stanford dataset, the proposed descriptor generally shows better performance. Whereas the higher performance than hand-crafted descriptors can be explained in terms of the difference between handcrafted and deep learning descriptors, the higher performance than 3DMatch requires a further explanation. We believe there are mainly three differences between our descriptor and 3DMatch.

First, our descriptor is trained using voxel representations converted from a point cloud alone, whereas 3DMatch is trained using voxels converted from mesh data, which include surface information. Because the shapes are different between voxels converted from a mesh and those converted from a point cloud alone, even with the conversion algorithms provided by 3DMatch, the 3DMatch network may not yet be optimized for voxels converted from a point cloud alone. Second, 3DMatch extracts points in a 0.3 m x 0.3 m x 0.3 m volume and converts them into a 30 x 30 x 30 voxel, which are the same dimensions as ours. Because this volume is quite large, and includes more points than ours, which extracts points in a 0.03 m x 0.03 m x 0.03 m area, our network has more experience regarding sparse point sets. Finally, the point-density measurement methods are different. 3DMatch is based on the nearest-neighbor point distance, whereas our descriptor is based on the method described in Algorithm 1. As shown through our experiment, the proposed density measurement method can help improve the performance of the surface descriptor.

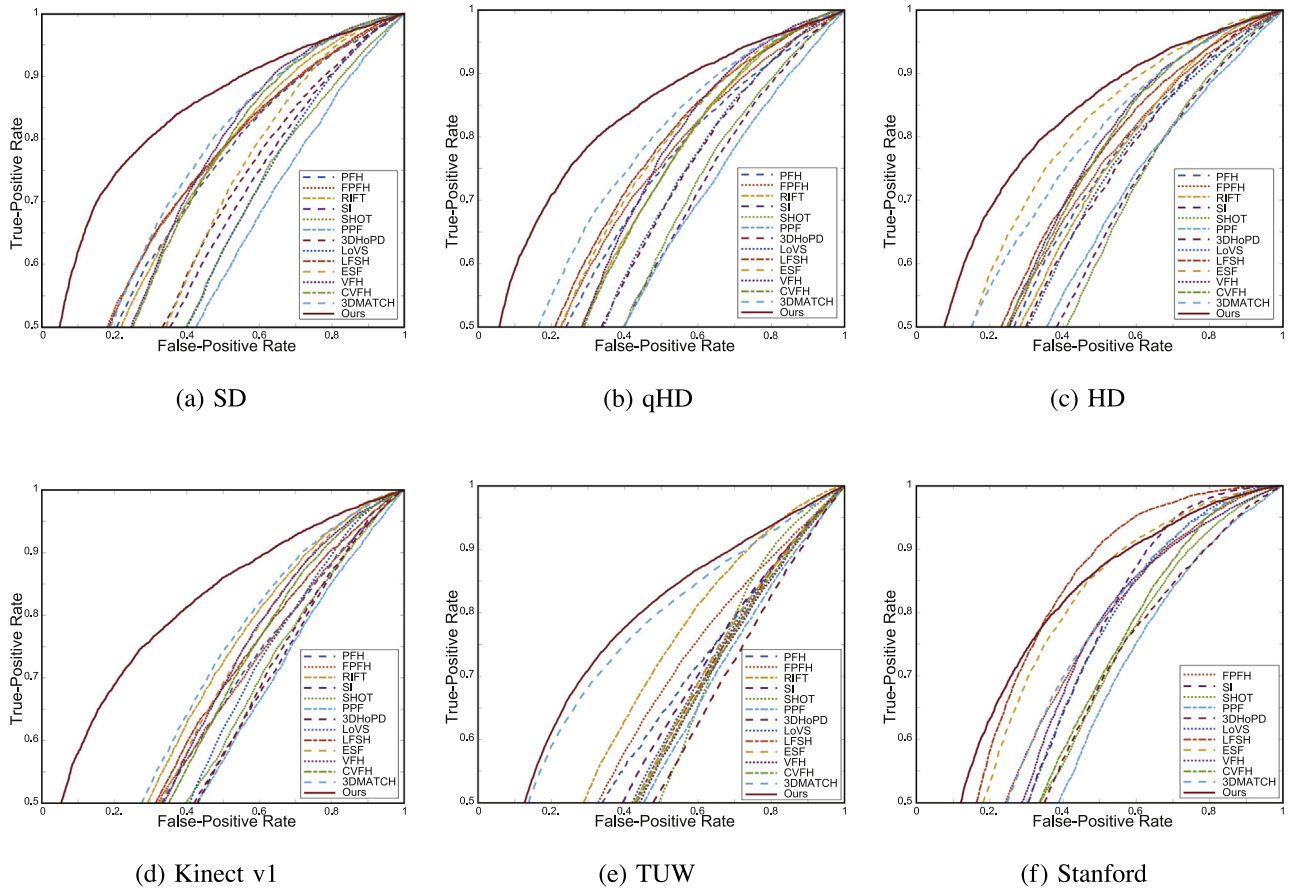
In case of Stanford dataset, LFSH and ours show comparable performances. Detailed discussion on this result will be shown after the fourth experiment.

Q. Can surface descriptors return consistent values for various point density?

Because the point density of a point cloud varies depending on the sensors and distance to an object, securing consistency under various densities allows a developer to be free from constraint, such as being forced to use a specific sensor and controlling the distance to an object. In addition, it is helpful for vision applications that often face a difference in point density between models and input images.

To determine the consistency of surface descriptors for point cloud images with various point densities, we compare the values of the evaluation criteria computed by averaging the results of the SD, qHD, and HD to those tested for a combined test dataset of SD, qHD, and HD. For eleven descriptors, namely, PFH, FPFH, RIFT, SHOT, PPF, 3DHoPD, LoVS, LFSH, ESF, VFH, CVFH, and our proposed descriptor, the results indicate that the values tested for the combined dataset are within the range of the minimum and maximum values among each SD, qHD, and HD test dataset. For example, the FPR95 values of the proposed descriptor are 0.7415 for SD, 0.7623 for qHD, 0.7433 for HD, and 0.7492 for the combined dataset, which is within the range between 0.7415 and 0.7623. In contrast, the average values for spin images and 3DMatch are less than any other values tested for the combined dataset. For example, the FPR95 values of 3DMatch are 0.7722 for SD, 0.7863 for qHD, 0.7900 for HD, and 0.8038 for the combined dataset, which are worse than in any other cases.

A further study indicates that this phenomenon is caused by different performances of a descriptor on different test datasets. We compared the threshold values exhibiting the highest F1-score for each test dataset. The average values of the similarity score for each descriptor are also listed because different score ranges lead to a different scale of the threshold values. In addition, the mean and standard deviation of the L2norm of the feature vectors are computed. As shown in Tables 6 and 7, spin image and 3DMatch exhibit relatively large differences for the SD, qHD and HD test datasets, whereas the remainders present similar statistics for each dataset. As mentioned earlier, it is desirable for a surface descriptor to show consistent values regardless of the point density of the point cloud. This result indicates that our descriptor not only works better than the others, it can provide consistent results regardless of the existence of various point densities.



(a) SD (b) qHD (c) HD

(d) Kinect v1 (e) TUW (f) Stanford

Fig. 8. Performance evaluation of 3D local surface descriptors. Each graph corresponds to the test dataset (a) SD, (b) qHD, (c) HD, (d) Kinect v1, (e) TUW and (f) Stanford dataset.

Table 5
Numerical measurements of 3D local surface descriptors. Note that PFH was not evaluated due to too high computational time issue, and RIFT was not evaluated because Stanford dataset does not provide color information of points while RIFT utilizes the information. The bolded text means the best, and underlined text means the second-best.

		PFH	FPFH	RIFT	SI	SHOT	PPF	3DHoPD	LoVS	LFSH	ESF	VFH	CVFH	3DMatch	Ours
S	AUC	0.7072	0.7241	0.7088	0.6041	0.5727	0.5543	0.6220	0.5711	0.7231	0.6264	0.7027	0.6952	0.7380	**0.8366**
	F1-score	0.6899	0.6918	0.6977	0.6737	0.6671	0.6669	0.6746	0.6730	0.6926	0.6881	0.7106	0.7051	0.7103	**0.7655**
	FPR95	0.8445	0.8402	0.7956	0.8767	0.9040	0.9218	0.8687	0.8799	0.8332	0.8154	0.7574	0.7657	0.7722	**0.7415**
q	AUC	0.6798	0.6933	0.6680	0.5790	0.5724	0.5673	0.6211	0.6164	0.7136	0.7089	0.6758	0.6680	0.7501	**0.8237**
	F1-score	0.6790	0.6842	0.6890	0.6699	0.6689	0.6667	0.6778	0.6775	0.6974	0.6959	0.6989	0.6909	0.7058	**0.7534**
	FPR95	0.8772	0.8504	0.8201	0.8936	0.8929	0.9262	0.8570	0.8573	0.8207	0.8019	0.7895	0.8084	0.7863	**0.7623**
H	AUC	0.6699	0.6775	0.6650	0.5845	0.5696	0.5967	0.6422	0.6370	0.6976	0.7617	0.6958	0.6969	0.7525	**0.8157**
	F1-score	0.6804	0.6846	0.6883	0.6683	0.6703	0.6667	0.6804	0.6762	0.6930	0.7230	0.7035	0.7008	0.7056	**0.7462**
	FPR95	0.8710	0.8486	0.8179	0.9002	0.8876	0.9124	0.8438	0.8688	0.8219	0.7540	0.7824	0.7838	0.7900	**0.7433**
A	AUC	0.6848	0.6974	0.6786	0.5761	0.5714	0.5583	0.6286	0.6016	0.7075	0.6538	0.6901	0.6845	0.7238	**0.8252**
	F1-score	0.6821	0.6865	0.6880	0.6675	0.6685	0.6667	0.6773	0.6736	0.6920	0.6864	0.7031	0.6982	0.6962	**0.7545**
	FPR95	0.8720	0.8450	0.8194	0.9039	0.8949	0.9154	0.8561	0.8703	0.8258	0.8256	0.7791	0.7905	0.8018	**0.7492**
K1	AUC	0.6165	0.6219	0.6609	0.5572	0.5648	0.5430	0.5562	0.5710	0.6300	0.6291	0.6270	0.6150	0.6700	**0.8147**
	F1-score	0.6678	0.6674	0.6857	0.6701	0.6677	0.6672	0.6682	0.6745	0.6706	0.6834	0.6800	0.6774	0.6878	**0.7399**
	FPR95	0.9077	0.9065	0.8335	0.8977	0.9149	0.9309	0.9067	0.8692	0.8872	0.8335	0.8485	0.8564	0.8312	**0.7790**
T	AUC	0.6121	0.6182	0.6612	0.5750	0.5238	0.5299	0.5150	0.5629	0.5482	0.5538	0.5432	0.5486	0.7463	**0.7591**
	F1-score	0.6667	0.6668	0.6838	0.6668	0.6713	0.6667	0.6667	0.6675	0.6667	0.6675	0.6669	0.6668	0.6994	**0.7139**
	FPR95	0.9170	0.9109	**0.8355**	0.9149	0.8822	0.9359	0.9344	0.9112	0.9147	0.9117	0.9250	0.9214	0.8547	0.8372
ST	AUC	–	0.6953	–	0.6695	0.6233	0.5274	0.6115	0.6603	0.7818	0.7532	0.6734	0.6311	0.6889	**0.7848**
	F1-score	–	0.6987	–	0.7156	0.6816	0.6677	0.6737	0.7010	**0.7543**	0.7336	0.6984	0.6929	0.7021	0.7374
	FPR95	–	0.7933	–	0.7057	0.8415	0.9195	0.8740	0.7708	**0.5942**	0.7078	0.8162	0.7925	0.7752	0.7275

6.4. Evaluation of robustness to noise

The last experiment was conducted to evaluate the robustness of the descriptors to noise. Although the test datasets we used, except for the Stanford 3D dataset, already contain noisy characteristics that appear in a real environment, and a good performance on these datasets can be evidence suggesting a robustness to noise, the experiment conducted was not a controlled experiment on noise and thus we examined the performances of

Table 6
Mean of distances between pairs in feature space, that is $\|f_1 - f_2\|_2$, and threshold values that show the highest f1-score of similarity scores for each test dataset. Spin Images and 3DMatch showed relatively big differences between test datasets while the rest including ours showed consistent results.

	SD		qHD		HD	
	Mean	Threshold	Mean	Threshold	Mean	Threshold
PFH	30.9816	16.1413	33.6848	14.8169	34.1322	12.1224
FPFH	46.2086	26.4555	51.3537	26.7371	53.0772	27.0186
RIFT	0.7240	0.4760	0.7078	0.3846	0.6947	0.4024
SI	0.2525	0.1688	0.2080	0.1129	0.1741	0.0872
SHOT	1.0405	0.4467	1.0472	0.4870	1.0308	0.4521
PPF	0.2210	0.0495	0.1691	0.0376	0.1235	0.0324
3DHoPD	0.7062	0.3641	0.7090	0.3580	0.7011	0.3365
LoVS	7.7218	6.1644	8.7518	6.7823	9.5820	7.7460
LFSH	0.5926	0.3997	0.5715	0.3694	0.4908	0.3132
ESF	0.0613	0.0418	0.0470	0.0300	0.0292	0.0204
VFH	89.4451	70.8200	89.3539	67.9288	82.6020	64.2432
CVFH	90.3188	71.4324	95.4789	70.0866	86.5523	69.1715
3DMatch	0.1843	0.0990	0.2136	0.0752	0.3610	0.1916
Ours	0.7380	0.3830	0.7624	0.3959	0.7889	0.4264

Table 7
Mean and standard deviation of L2-norm of feature vectors, that is $\|f_i\|_2$. In case of RIFT and SHOT descriptor, L2-norms of all feature vectors are 1. It leads to zero standard deviations.

	SD		qHD		HD	
	Mean	Std	Mean	Std	Mean	Std
PFH	69.6703	15.1646	72.7105	15.0667	72.2125	15.0322
FPFH	148.1597	8.0086	149.9206	7.1093	149.5120	7.3491
RIFT	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
SI	0.2410	0.0387	0.2486	0.0403	0.2463	0.0404
SHOT	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000
PPF	0.1711	0.0751	0.1405	0.0643	0.1158	0.0551
3DHoPD	1.7321	0.0000	1.7321	0.0000	1.7321	0.0000
LoVS	6.2291	0.9922	7.9625	1.3228	9.6643	1.4567
LFSH	1.1727	0.1634	1.1112	0.1432	1.0326	0.1051
ESF	0.1211	0.0084	0.0953	0.0118	0.0837	0.0058
VFH	127.8487	24.1483	133.1804	23.7303	129.6249	22.5481
CVFH	127.5553	23.5443	124.1460	27.0221	116.2742	27.0161
3DMatch	0.4273	0.0952	0.4847	0.1189	0.6670	0.2592
Ours	2.6271	0.8441	2.7431	0.8322	2.7192	0.8075

the descriptors while varying the amount of noise that can be presented in a point cloud.

Based on the Stanford 3D dataset, whose mean distance between points is approximately 0.0005 m, we applied Gaussian noise with three different levels, namely, 0.0001, 0.0003, and 0.0005 m. The performances of the descriptors on these noise-included datasets are summarized in Fig. 9 and Table 8.

The results reveal that descriptors such as FPFH, spin image, SHOT, ESF, VFH, CVFH, 3DMatch, and ours show robust performance on noise-included datasets. In contrary, descriptors such as PPF, 3DHoPD, LoVS, and LFSH show worse performance as the noise level increased.

6.5. Limitations

We regard that there are mainly three limitations of the proposed descriptor. The first is related to the performance on the Stanford dataset. As mentioned earlier, our goal of this study is to develop 3D local surface descriptor that can show good performance in robotic applications that often use an single affordable sensor. For this purpose, sparse point cloud images acquired by one of typical RGB-D sensors, Kinect v2, are used for the training process. Although the performance evaluation on those kinds of datasets including Kinect v2 dataset of different objects, Kinect v1 dataset, and TUW dataset supports that the goal is achieved, the proposed descriptor cannot show significant improvement on dense point cloud such as Stanford 3D scan dataset. We believe that this is because the point density of the Stanford dataset is higher than the HD dataset that has the highest density among the training datasets. Therefore, the network has no experience on such a dense point clouds. To resolve this issue, training a network with point cloud datasets that have a wider range of point density will be conducted as one of further works.

The second is related to the size of the interest volume. In this study, we set the size of interest volume to be 0.03 m^3 that might be suitable for small objects, but not for large objects or scene-level applications. Because it is desirable for descriptors to have adjustable size of the interest volume depending on target scenarios, our further study will handle this issue. Actually, different size of the interest volume to the 30 by 30 by 30 voxel representation is another causal of density variance. Because the proposed descriptor has adaptiveness to density variance, we regard the proposed descriptor may show comparable performance to other descriptors. However, it is not fully studied with experiments yet and there might be something we cannot predict.

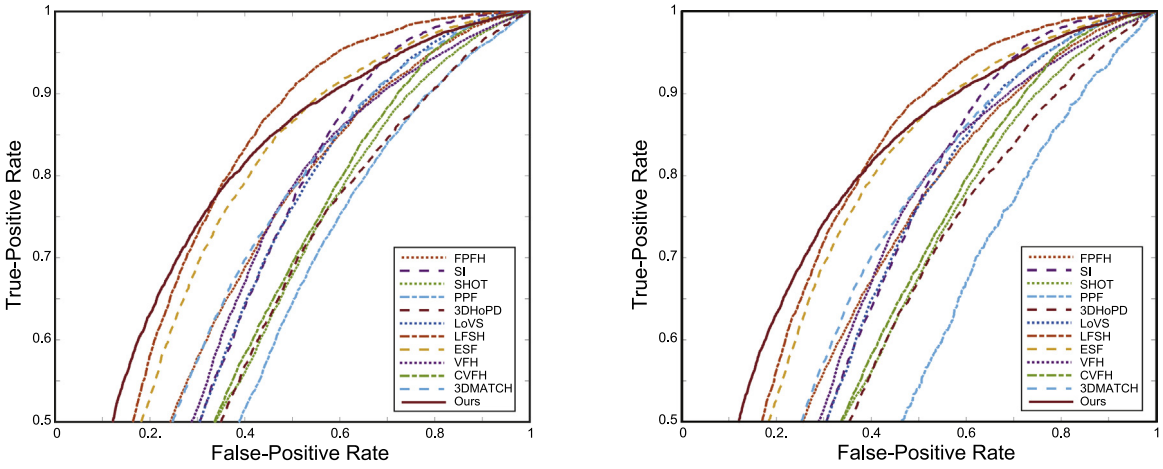
The last is related to computational cost. As many deep learning based approaches do, the computational cost is higher than hand-crafted descriptor's. Parallelization of the method and algorithm optimization will be implemented for better feasibility for real applications.

7. Conclusion

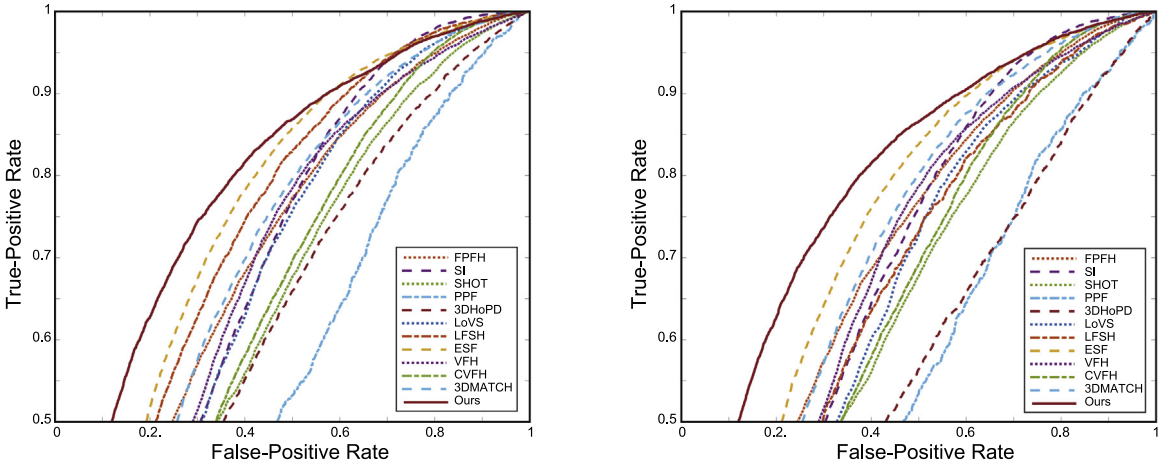
In this study, we proposed a new 3D surface descriptor for use in a point cloud. Unlike other descriptors based on the prescribed topology information or mesh construction, which is difficult to be accurate with single affordable sensors in robotic applications without human intervention, the proposed descriptor can be applied to a point cloud alone. To resolve the sparsity problem in

Table 8
Performance of 3D surface descriptors on the dataset with different noise level. The bolded text means the best, and underlined text means the second-best.

		PFH	FPFH	RIFT	SI	SHOT	PPF	3DHoPD	LoVS	LFSH	ESF	VFH	CVFH	3DMatch	Ours
N0	AUC	–	0.6953	–	0.6695	0.6233	0.5902	0.6115	0.6603	0.7818	0.7532	0.6734	0.6311	0.6889	**0.7848**
	F1-score	–	0.6987	–	0.7156	0.6816	0.6719	0.6737	0.7010	**0.7543**	0.7336	0.6984	0.6929	0.7021	0.7374
	FPR95	–	0.7933	–	0.7057	0.8415	0.8798	0.8740	0.7708	**0.5942**	0.7078	0.8162	0.7925	0.7752	0.7275
N1	AUC	–	0.6851	–	0.6693	0.6229	0.5333	0.6076	0.6598	0.7732	0.7529	0.6732	0.6305	0.6896	**0.7851**
	F1-score	–	0.6930	–	0.7148	0.6820	0.6675	0.6736	0.7026	**0.7493**	0.7331	0.6981	0.6931	0.7025	0.7374
	FPR95	–	0.8023	–	0.7126	0.8409	0.9158	0.8742	0.7712	**0.6186**	0.7056	0.8152	0.7932	0.7672	0.7276
N2	AUC	–	0.6909	–	0.6683	0.6212	0.5311	0.6035	0.6555	0.7229	0.7427	0.6723	0.6311	0.6927	**0.7847**
	F1-score	–	0.6952	–	0.7127	0.6815	0.6694	0.6724	0.6999	0.7169	0.7282	0.6982	0.6925	0.7045	**0.7380**
	FPR95	–	0.8044	–	0.7200	0.8397	0.9053	0.8763	0.7686	0.7175	**0.7143**	0.8148	0.7946	0.7703	0.7263
N3	AUC	–	0.6932	–	0.6667	0.6235	0.5277	0.5327	0.6363	0.6574	0.7257	0.6737	0.6310	0.6954	**0.7840**
	F1-score	–	0.6667	–	0.7090	0.6802	0.6668	0.6667	0.6898	0.6865	0.7203	0.6983	0.6935	0.7080	**0.7365**
	FPR95	–	0.8010	–	0.7381	0.8460	0.9243	0.9315	0.8375	0.8291	**0.7288**	0.8153	0.7930	0.7687	0.7296



(a) no noise (b) 0.0001 m



(c) 0.0003 m (d) 0.0005 m

Fig. 9. Performance of 3D local surface descriptors on the dataset with different noise level. Each graph corresponds to the noise levels (a) no noise, (b) 0.0001 m, (c) 0.0003 m, (d) 0.0005 m.

a small volume and varying point-density problem, we tested various types of voxel representation including point-density measuring methods and point-expansion methods. The performance comparison between voxel representations shows that a binarized spherical expansion exhibits a better performance than the other approaches. In addition, we adapted domain-adversarial learning to induce a network to learn density-invariant features and achieved a better performance. Finally, we conducted a performance comparison between the proposed surface descriptors and other descriptors. The results confirm that the proposed surface descriptor outperforms than the others even for images captured by other sensors, and images captured in a different environment. In addition, the proposed descriptor shows consistent results under various point densities, whereas some of the other descriptors do not.

As a next step, we will focus on improving the performance by resolving the issues summarized in limitation section. Furthermore, object model reconstruction and tracking, which are commonly used for robotic applications, will be implemented using the proposed surface descriptor.

Acknowledgments

This research was financially supported by the Ministry of Trade, Industry and Energy (MOTIE) and Korea Institute for Advancement of Technology (KIAT) through the National Innovation Cluster R&D program (P0006702 Living lab establishment and operation for verification of life safety AI service in industrial and complex living space).

Appendix

For training and testing, we used a computer with Intel i7-7700K CPU @ 4.20 GHz, 16 GB main memory, and Geforce GTX 1080Ti with 12 GB memory. Tensorflow with cuda 8.0 and cudnn v5.1 is used. Point density measuring methods cost 0.532 ms with the nearest neighbor point distance method and 4.728 ms with the proposed method. Converting a point cloud into a single voxel representation costs 1.705 ms with parallel computing(cuda function). Extracting deep feature from a single voxel costs 1.504 ms with batchsize of 1. In total, about 8 ms is required for computing the proposed descriptor. For comparison, PFH costs 4.5 ms, FPFH

costs 1.1 ms, RIFT costs 0.981 ms, SpinImage costs 1.1 ms and SHOT costs 1.3 ms although they have no advantage of GPU and parallel computing. In case of 3DMatch's network structure which is more complex than ours, it costs 2.808 ms in the same implementation environment, i.e. tensorflow, with ours. All the time costs are average value. Accelerating of descriptor computation by parallelizing and using other deep learning library known to be faster than tensorflow will be proceeded as further technical works.

References

[1] D.G. Lowe, Object recognition from local scale-invariant features, in: Computer Vision, 1999. the Proceedings of the Seventh IEEE International Conference on, Vol. 2, IEEE, 1999, pp. 1150–1157.

[2] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (SURF), Comput. Vis. Image Underst. 110 (3) (2008) 346–359.

[3] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: An efficient alternative to SIFT or SURF, in: Computer Vision (ICCV), 2011 IEEE international conference on, IEEE, 2011, pp. 2564–2571.

[4] H.-Y. Wu, H. Zha, T. Luo, X.-L. Wang, S. Ma, Global and local isometry-invariant descriptor for 3D shape comparison and partial matching, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 438–445.

[5] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, J. Wan, Rotational projection statistics for 3D local surface description and object recognition, Int. J. Comput. Vis. 105 (1) (2013) 63–86.

[6] S. Salamanca, P. Merchán, E. Pérez, A. Adan, C. Cerrada, Filling holes in 3D meshes using image restoration algorithms, in: International Symposium on 3D Data Processing, Visualization, and Transmission, Vol. 2, 2008.

[7] I. Guskov, Z.J. Wood, Topological noise removal, in: 2001 Graphics Interface Proceedings: Ottawa, Canada, 2001, p. 19.

[8] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[9] S. Zagoruyko, N. Komodakis, Learning to compare image patches via convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4353–4361.

[10] J. Hu, J. Lu, Y.-P. Tan, Discriminative deep metric learning for face verification in the wild, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1875–1882.

[11] R.B. Rusu, N. Blodow, Z.C. Marton, M. Beetz, Aligning point cloud views using persistent feature histograms, in: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, IEEE, 2008, pp. 3384–3391.

[12] R.B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (FPFH) for 3D registration, in: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, Citeseer, 2009, pp. 3212–3217.

[13] S. Lazebnik, C. Schmid, J. Ponce, A sparse texture representation using local affine regions, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1265–1278.

[14] A.E. Johnson, Spin-Images: A Representation for 3-D Surface Matching, Citeseer, 1997.

[15] S. Salti, F. Tombari, L. Di Stefano, SHOT: Unique signatures of histograms for surface and texture description, Comput. Vis. Image Underst. 125 (2014) 251–264.

[16] A.G. Buch, D. Kraft, S. Robotics, D. Odense, Local Point Pair Feature Histogram for Accurate 3D Matching, BMVC, 2018.

[17] S.M. Prakhya, J. Lin, V. Chandrasekhar, W. Lin, B. Liu, 3DHoPD: A fast low-dimensional 3-D descriptor, IEEE Robot. Autom. Lett. 2 (3) (2017) 1472–1479.

[18] S. Quan, J. Ma, F. Hu, B. Fang, T. Ma, Local voxelized structure for 3D binary feature representation and robust registration of point clouds from low-cost sensors, Inform. Sci. 444 (2018) 153–171.

[19] J. Yang, Z. Cao, Q. Zhang, A fast and robust local descriptor for 3D point cloud registration, Inform. Sci. 346 (2016) 163–179.

[20] W. Wohlkinger, M. Vincze, Ensemble of shape functions for 3D object classification, in: Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on, IEEE, 2011, pp. 2987–2992.

[21] R.B. Rusu, G. Bradski, R. Thibaux, J. Hsu, Fast 3D recognition and pose using the viewpoint feature histogram, in: Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, IEEE, 2010, pp. 2155–2162.

[22] A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R.B. Rusu, G. Bradski, CAD-Model recognition and 6DOF pose estimation using 3D cues, in: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), IEEE, 2011, pp. 585–592.

[23] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, T. Funkhouser, 3DMatch: Learning local geometric descriptors from RGB-D reconstructions, in: Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, IEEE, 2017, pp. 199–208.

[24] Y. Diez, F. Roure, X. Lladó, J. Salvi, A qualitative review on 3D coarse registration methods, ACM Comput. Surv. 47 (3) (2015) 45.

[25] S. Salti, F. Tombari, L. Di Stefano, A performance evaluation of 3D keypoint detectors, in: 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on, IEEE, 2011, pp. 236–243.

[26] J. Huang, S. You, Point cloud matching based on 3D self-similarity, in: Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on, IEEE, 2012, pp. 41–48.

[27] J. Yang, Q. Zhang, Y. Xiao, Z. Cao, TOLDI: An effective and robust approach for 3D local shape description, Pattern Recognit. 65 (2017) 175–187.

[28] J. Yang, Q. Zhang, K. Xian, Y. Xiao, Z. Cao, Rotational contour signatures for both real-valued and binary feature representations of 3D local shape, Comput. Vis. Image Underst. 160 (2017) 133–147.

[29] S. Quan, J. Ma, T. Ma, F. Hu, B. Fang, Representing local shape geometry from multi-view silhouette perspective: A distinctive and robust binary 3D feature, Signal Process., Image Commun. 65 (2018) 67–80.

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

[31] A. Aldoma, F. Tombari, R.B. Rusu, M. Vincze, OUR-CVFH–Oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6DOF pose estimation, in: Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium, Springer, 2012, pp. 113–122.

[32] A.J. Rodríguez-Sánchez, S. Szedmak, J. Piater, SCurV: A 3D descriptor for object classification, in: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, Citeseer, 2015, pp. 1320–1327.

[33] R. Litman, A. Bronstein, M. Bronstein, U. Castellani, Supervised learning of bag-of-features shape descriptors using sparse coding, in: Computer Graphics Forum, Vol. 33, Wiley Online Library, 2014, pp. 127–136.

[34] G. Hua, M. Brown, S. Winder, Discriminant embedding for local image descriptors, in: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE, 2007, pp. 1–8.

[35] A. Ng, NIPS 2016 tutorial: "Nuts and bolts of building AI applications using Deep Learning" by Andrew Ng, 2016. URL https://www.youtube.com/watch?v=wjqaz6m42wU.

[36] X. Xu, D. Corrigan, A. Dehghani, S. Caulfield, D. Moloney, 3D Object recognition based on volumetric representation using convolutional neural networks, in: International Conference on Articulated Motion and Deformable Objects, Springer, 2016, pp. 147–156.

[37] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3D Shapenets: A deep representation for volumetric shapes, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1912–1920.

[38] D. Maturana, S. Scherer, Voxnet: A 3D convolutional neural network for real-time object recognition, in: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, IEEE, 2015, pp. 922–928.

[39] M. Ren, L. Niu, Y. Fang, 3D-A-Nets: 3D Deep Dense Descriptor for Volumetric Shapes with Adversarial Networks, 2017. arXiv preprint arXiv:1711.10108.

[40] H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V.G. Kim, E. Yumer, Learning local shape descriptors from part correspondences with multiview convolutional networks, ACM Trans. Graph. 37 (1) (2018) 6.

[41] Y. Feng, Z. Zhang, X. Zhao, R. Ji, Y. Gao, GVCNN: Group-view convolutional neural networks for 3D shape recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 264–272.

[42] H. Deng, T. Birdal, S. Ilic, Ppfnet: Global context aware local features for robust 3d point matching, in: Computer Vision and Pattern Recognition (CVPR), Vol. 1, IEEE, 2018.

[43] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: deep learning on point sets for 3D classification and segmentation, in: Proc. Computer Vision and Pattern Recognition (CVPR), Vol. 1, IEEE, 2017, p. 4.

[44] C.R. Qi, L. Yi, H. Su, L.J. Guibas, PointNet++: Deep hierarchical feature learning on point sets in a metric space, in: Advances in Neural Information Processing Systems, 2017, pp. 5099–5108.

[45] Z.J. Yew, G.H. Lee, 3DFeat-Net: Weakly supervised local 3D features for point cloud registration, in: European Conference on Computer Vision, Springer, 2018, pp. 630–646.

[46] A. Dewan, T. Caselitz, W. Burgard, Learning a local feature descriptor for 3D LiDAR scans, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 4774–4780.

[47] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2017, pp. 2261–2269.

[48] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, T. Funkhouser, 3DMatch Toolbox, 2017. URL https://github.com/andyzeng/3dmatch-toolbox.
[49] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, N. Lawrence, Covariate Shift and Local Learning by Distribution Matching, MIT Press, 2008.
[50] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, J. Mach. Learn. Res. 17 (59) (2016) 1–35.
[51] Z. Pei, Z. Cao, M. Long, J. Wang, Multi-adversarial domain adaptation, in: AAAI Conference on Artificial Intelligence, 2018.
[52] L. Deng, The MNIST database of handwritten digit images for machine learning research [best of the web], IEEE Signal Process. Mag. 29 (6) (2012) 141–142.
[53] L. Pang, Y. Wang, Y.Z. Song, T. Huang, Y. Tian, Cross-domain adversarial feature learning for sketch re-identification, in: 2018 ACM Multimedia Conference on Multimedia Conference, ACM, 2018, pp. 609–617.
[54] B. Liu, I. Lane, Multi-domain adversarial learning for slot filling in spoken language understanding, 2017. arXiv preprint arXiv:1711.11310.
[55] B. Morris, AR Marker tools for ROS, 2010. URL https://github.com/ar-tools/artools.
[56] A. Aldoma, T. Fäulhammer, M. Vincze, Automation of ground truth annotation for multi-view RGB-D object instance recognition datasets, in: Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, IEEE, 2014, pp. 5016–5023.
[57] M. Levoy, J. Gerth, B. Curless, K. Pull, The Stanford 3D scanning repository, 2005. URL http://www-graphics.stanford.edu/data/3dscanrep.
[58] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., Tensorflow: a system for large-scale machine learning., in: OSDI, Vol. 16, 2016, pp. 265–283.
[59] Y. Nesterov, A method for unconstrained convex minimization problem with the rate of convergence O (1/kˆ 2), in: Doklady AN USSR, Vol. 269, 1983, pp. 543–547.
[60] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
[61] R.B. Rusu, S. Cousins, 3D Is here: Point cloud library (PCL), in: Robotics and automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 1–4.
[62] M. Brown, G. Hua, S. Winder, Discriminative learning of local image descriptors, IEEE Trans. Pattern Anal. Mach. Intell. 33 (1) (2011) 43–57.
[63] X. Han, T. Leung, Y. Jia, R. Sukthankar, A.C. Berg, Matchnet: Unifying feature and metric learning for patch-based matching, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3279–3286.

Ju-Hwan Seo received his B.S. and M.S. degrees from the Korea Advanced Institute of Science and Technology (KAIST) in Daejeon, Korea, in 2012 and 2014, respectively, where he is currently working toward his Ph.D. His current research interests include computer vision and deep learning.

Dr. Dong-Soo Kwon received his B.S. degree from Seoul National University, Seoul, Korea, in 1980, the M.S degree from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1982, and the Ph.D. degree from the Georgia Institute of Technology, Atlanta, in 1991, all in mechanical engineering. From 1991 to 1995, he was a Research Staff Member with Oak Ridge National Laboratory; He is currently a Professor of mechanical engineering, the Director of the Center for Future Medical Robotics, the Director of the Human–Robot Interaction Research Center, KAIST. He is a member of the IEEE RAS AdCOM, the Korean Society of Mechanical Engineers and the International Council of Associations for Science Education. His current research interests include human–robot/computer interaction, medical robots, telerobotics, and haptics.