SketchDesc: Learning Local Sketch Descriptors for Multi-view Correspondence

Deng Yu, Lei Li, Youyi Zheng, Manfred Lau, Yi-Zhe Song, *Senior Member, IEEE*, Chiew-Lan Tai, Hongbo Fu†

***Abstract*—In this paper, we study the problem of multi-view sketch correspondence, where we take as input multiple freehand sketches with different views of the same object and predict as output the semantic correspondence among the sketches. This problem is challenging since the visual features of corresponding points at different views can be very different. To this end, we take a deep learning approach and learn a novel local sketch descriptor from data. We contribute a training dataset by generating the pixel-level correspondence for the multi-view line drawings synthesized from 3D shapes. To handle the sparsity and ambiguity of sketches, we design a novel multi-branch neural network that integrates a patch-based representation and a multi-scale strategy to learn the pixel-level correspondence among multi-view sketches. We demonstrate the effectiveness of our proposed approach with extensive experiments on hand-drawn sketches and multi-view line drawings rendered from multiple 3D shape datasets.**

***Index Terms*—Multi-view sketches, correspondence learning, multi-scale, patch-based descriptor.**

I. Introduction

Sketching as a universal form of communication provides arguably the most natural and direct way for humans to render and interpret the visual world. While it is not challenging for human viewers to interpret missing 3D information from single-view sketches, multi-view inputs are often needed for computer algorithms to recover the underlying 3D geometry due to the inherent ambiguity in single-view sketches. A key problem for interpreting multi-view sketches of the same object is to establish semantic correspondence among them. This problem has been mainly studied for 3D geometry reconstruction from careful engineering drawings in orthographic views [1]. The problem of establishing semantic correspondence from rough sketches in arbitrary views (e.g., those in Figure 1 (middle and bottom)) is challenging due to the abstraction and distortions in both shape and view, and is largely unexplored. Addressing this problem can benefit various applications, for example to design an interactive interface for users with little training in drawing to create 3D shapes using multi-view sketches.

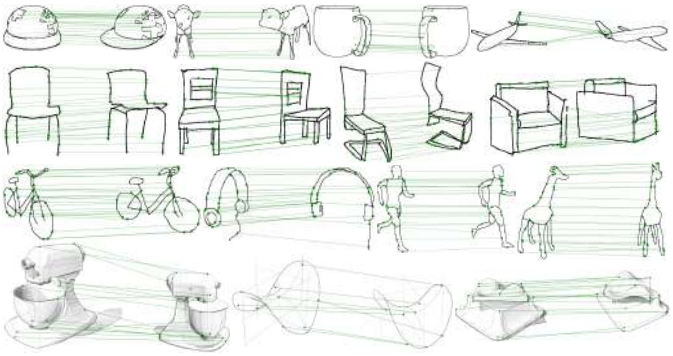


Fig. 1. Multi-view sketch correspondence results by SketchDesc on line drawings synthesized from 3D shapes (top) and freehand sketches (middle and bottom).

In this work, we take the first step to learn to establish the semantic correspondence between freehand sketches depicting the same object from different views. This demands a proper shape descriptor. However, traditional descriptors like Shape Context [2] and recent learning-based patch descriptors [3]–[5] are often designed to be invariant to 2D transformations (e.g. rotation, translation, and limited distortion) and cannot handle large view changes (e.g. with view disparity greater than 30 degrees). Such descriptors, especially used for the applications of stereo matching [6]–[10] and image-based 3D modeling [11], [12], heavily exploit the features containing textures and shadings of images for inferring similarities among different points or image patches, and thus are not directly applicable to our problem. This is because sketches only contain binary lines and points, exhibiting inherent sparsity and ambiguity.

We observe that human viewers can easily identify corresponding points from sketches with very different views. This is largely because human viewers have knowledge of sketched objects in different views. Our idea is thus to adapt deep neural networks previously used for learning patch-based descriptors to learn descriptors for corresponding patches from multi-view sketches.

Training deep neural networks require a large-scale dataset of sketch images with ground truth semantic correspondence.

This work was partially supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 11206319, CityU 11212119, CityU 11237116, HKUST16210718), and the City University of Hong Kong (Project No. 7005176 (SCM)). Youyi Zheng was partially supported by the China Young 1000 Talent Program and the Fundamental Research Funds for the Central Universities.

D. Yu, M. Lau, and H. Fu are with the School of Creative Media, City University of Hong Kong. H. Fu is the corresponding author of this paper. E-mail: {deng.yu@my., manfred.lau@, hongbofu@}cityu.edu.hk

L. Li and C.-L. Tai are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. E-mail: {llibb, taicl}@cse.ust.hk

Y. Zheng is with the State Key Lab of CAD&CG, Zhejiang University. E-mail: youyizheng@zju.edu.cn

Y.-Z. Song is with SketchX, CVSSP, University of Surrey. E-mail: y.song@surrey.ac.uk

Unfortunately, such training datasets are not available. On the other hand, manually collecting multi-view hand-drawn sketches and labeling the ground-truth correspondence would be a demanding task. Following previous deep learning solutions for 3D interpretation of sketches [13]–[15], we synthesize the multi-view line drawings from 3D shapes (e.g. from ShapeNet [16]) by using non-photorealistic rendering. Given the synthesized dataset of multi-view line drawings as sketches, we project the 3D vertices of 3D models to multi-view sketches to get ground-truth correspondences (Figure 2). This data generation pipeline is to emphasize the correspondence from 3D shapes and force deep networks to learn the valuable 3D correspondences among 2D multi-view sketches.

We formulate the correspondence learning problem into a metric function learning procedure and build upon the latest techniques for metric learning and descriptor learning [3]. To find an effective feature descriptor for multi-view sketches, we combine a patch-based representation and a multi-scale strategy (Figure 3) to address the abstraction problem of sketches (akin to Sketch-A-Net [17]).

We further design a multi-branch network (Figure 4) with shared-weights to process the patches at different scales. The patch-based representation helps the network specify the local features of a point on a sketch image by embedding the information of its neighboring pixels. Our multi-scale strategy feeds the network with the local and global perspectives to learn the distinctive information at different scales.

The multi-scale patch representation allows the use of ground-truth correspondences that are away from sketch lines as the additional training data. This not only improves the correspondence accuracy but also enables correspondences inside the regions of sketched objects (e.g., the cup in Figure 1), potentially benefiting applications like sketch-based 3D shape synthesis. We evaluate our method by performing multi-view sketch correspondence and pixel-wise retrieval tasks on a large-scale dataset of synthesized multi-view sketches based on three shape repositories: ShapeNet, Princeton Segmentation Benchmark (PSB) [18] and Structure Recovery [19] to show the effectiveness of our proposed framework. We also test our trained network on freehand sketches (Figure 1) drawn by volunteers and collected from the OpenSketch dataset [20], which shows its robustness against shape and view distortions.

Our contributions are summarized as follows:

- To the best of our knowledge, we are among the first to study the problem of multi-view sketch correspondence.
- We introduce a multi-branch neural network to learn a patch-based descriptor that can be used to measure semantic similarity between patches from multi-view sketches. We demonstrate the improved accuracy for sketch correspondence achieved by using our descriptor compared to other existing descriptors.
- We collect a large-scale dataset consisting of 6,852 multi-view sketches with ground-truth correspondences of 587 shapes spanning 18 object categories, and will release the dataset to the research community.

II. Related Work

We review the literature that is closely related to our work, namely, the approaches for correspondence establishment in images, and the approaches for sketch analysis using deep learning technologies.

A. Correspondence Establishment for Images

Image-based Modeling and Stereo Matching. Image-based modeling often takes as input multiple images of an object [12], [21] or scene [11], [22] from different views, and aims to reconstruct the underlying 3D geometry. A typical approach to this problem is to first detect a sparse set of key points, then adopt a feature descriptor (e.g., SIFT [23]) to describe the patches centered at the key points, and finally conduct feature matching to build the correspondence among multi-view images. Stereo matching [6], [7] takes two images from different but often close viewpoints and aims to establish dense pixel-level correspondence across images. Our problem is different from these tasks in the following ways. First, the view disparity in our input sketches is often much larger. Second, unlike natural photos, which have rich textures, our sketches have more limited information due to their line-based representation.

Local Image Descriptors. Local image descriptors are typically derived from image patches centered at points of interest and designed to be invariant to certain factors, such as rotation, scale, or intensity, for robustness. Existing local image descriptors can be broadly categorized as hand-crafted descriptors and learning-based descriptors. A full review is beyond the scope of this work. We refer the interested readers to [24] for an insightful survey.

Classical descriptors include, to name a few, SIFT [23], SURF [25], Shape Context [2], and HOG [26]. The conventional local descriptors are mostly built upon low-level image properties and constructed using hand-crafted rules. Recently, learning-based local descriptors produced by deep convolutional neural networks (CNNs) [3]–[5] have shown their superior performance over the hand-crafted descriptors, owing importantly to the availability of large-scale image correspondence datasets [27], [28] obtained from 3D reconstructions. To learn robust 2D local descriptors, extensive research has been dedicated to the development of CNN designs [3], [29]–[32], loss functions [4], [33]–[37] and training strategies [38]–[40]. The above methods, however, are not specially designed for learning multi-view sketch correspondence. The work of GeoDesec [40] shares the closest spirit to ours and employs geometry constraints from 3D reconstruction by Structure-from-Motion (SfM) to refine the training data. However, SfM heavily depends on the textures and shadings in the image domain and is not suitable for sketches.

Multi-scale Strategy for Descriptors. Shilane and Funkhouser [41] computed a 128-D descriptor at four scales in spherical regions to describe distinctive regions on 3D shape surfaces. Recently, Huang et al. [42] proposed to learn 3D point descriptors from multi-view projections with progressively zoomed out viewpoints. Inspired by these methods, we design a multi-scale strategy to gather local

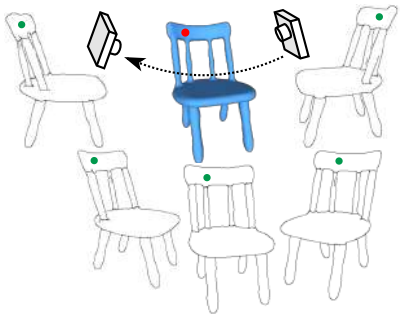


Fig. 2. Illustration of synthesized multi-view sketches (of size 480 × 480), with corresponding pixels (in green) projected from the same vertex (in red) on a 3D shape.

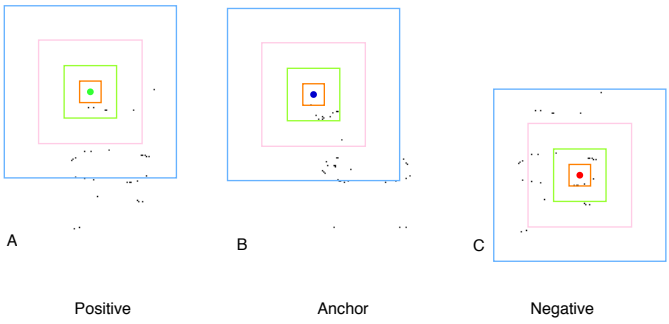


Fig. 3. Illustration of our multi-scale patch-based representation. Given a 480×480 sketch input and a pixel of interest, we use multi-scale (32×32, 64×64, 128×128, and 256×256) patches to capture the neighborhood of the pixel. The positive, anchor and negative patches are formed as a training triplet.

and global context to locate corresponding points in sketches across views (Figure 3). Different from [42], which focuses on rendered patches of 3D shapes, our work considers patches of sparse line drawings with limited textures as input. To reduce the network size and computation, we adopt a smaller input scale of 32×32 [3] rather than 224×224 in [42]. In addition, Huang et al. [42] used three viewpoints for 3D point descriptors, while our work extracts descriptors of points in sketches drawn under a specific viewpoint for correspondence establishment across a larger range of views.

B. Deep Learning in Sketch Analysis

With the recent advances in deep learning techniques, a variety of deep learning based methods have been proposed for sketch analysis tasks such as sketch synthesis [43], face sketch-photo synthesis [44]–[46], sketch recognition [17], sketch segmentation [47], and sketch retrieval [48], [49]. Among these works, [45] and [46] are the most relevant to our work, and they aim to model the correspondence between face photos and face sketches. However, different from our task to exploit the pixel-level correspondence between sketches, they explore the image-level mapping between face photos and sketches. In general, none of these existing solutions can be directly applied to our task. To make deep learning possible in sketch analysis, there exists multiple large-scale datasets of sketches including the TU-Berlin [50], QuickDraw [43], and Sketchy [48] datasets. However, they do not contain multi-view sketches and thus cannot be used to train our network. A sketch is generally represented as either a rasterized binary-pixel image [48] or vector sequences [43], [47] or both [49]. Since it is difficult to render line drawings from 3D shapes as a sequence of well-defined strokes, we use rasterized binary-pixel images to represent our training data and input sketches.

Multi-view Sketch Analysis. Multi-view sketches are often used in sketch-based 3D modeling [13]–[15], [51], [52]. Early sketch-based modeling methods (e.g., [51]) require precise engineering drawings as input or are limited by demanding mental efforts, requiring users to first decompose a desired 3D shape into parts and then constructing each part through careful engineering drawings [1]. To alleviate this issue, several recent methods [13]–[15], [52] leverage learning-based frameworks (e.g., GAN [53]) to obtain the priors from training data and then infer 3D shapes from novel input multi-view sketches (usually in orthographic views, namely, the front, back, and side views). However, these methods often process individual multi-view sketches in separate branches and do not explicitly consider the semantic correspondence between input sketches. Recently, a richly-annotated dataset of product design sketches, named OpenSketch, was presented by [20], which contains around 400 sketches of 12 man-made objects. However, the limited data in OpenSketch is insufficient for training deep neural networks. In our work, we synthesized 6,852 sketches for 18 shape categories to learn local sketch descriptors.

SketchZooms by Navarro et al. [54] is a concurrent work and studies the sketch correspondence problem with a similar deep learning based solution. Our work is different from SketchZooms as follows. To generate training data for cross-object correspondence, the used 3D shapes need to be semantically registered together in advance in SketchZooms, while our work is more fine-grained as it explores training data generation from individual 3D shapes for cross-view correspondence. SketchZooms follows [42], [55] and adapts AlexNet [56] (40M parameters) with the final layer replaced with a view pooling layer. In contrast, our designed framework has a smaller size (1.4M parameters), and its high performance shown in our experiments paves a way for our method to be more easily integrated into mobile or touch devices.

III. Methodology

Terminology. In the next sections, we differentiate between the word "points" for shapes, sketch images, and sketch lines as follows. We refer to the points on the surface of a 3D shape as *vertices*; the points on a 2D sketch image as *pixels* and the points exactly on the sketch lines as *points*.

For input sketches represented as rasterized images, a key problem to semantic correspondence learning is to measure the difference between a pair of pixels in a semantic way. This is essentially a metric function learning problem where a pair of corresponding pixels has a smaller distance than a pair of non-corresponding pixels in the metric space. Formally, it can be expressed as follows:

$$D_m\langle p_{c1}, p_{c2}\rangle < D_m\langle p_{c1}, \widetilde{p}_{nc}\rangle, \tag{1}$$

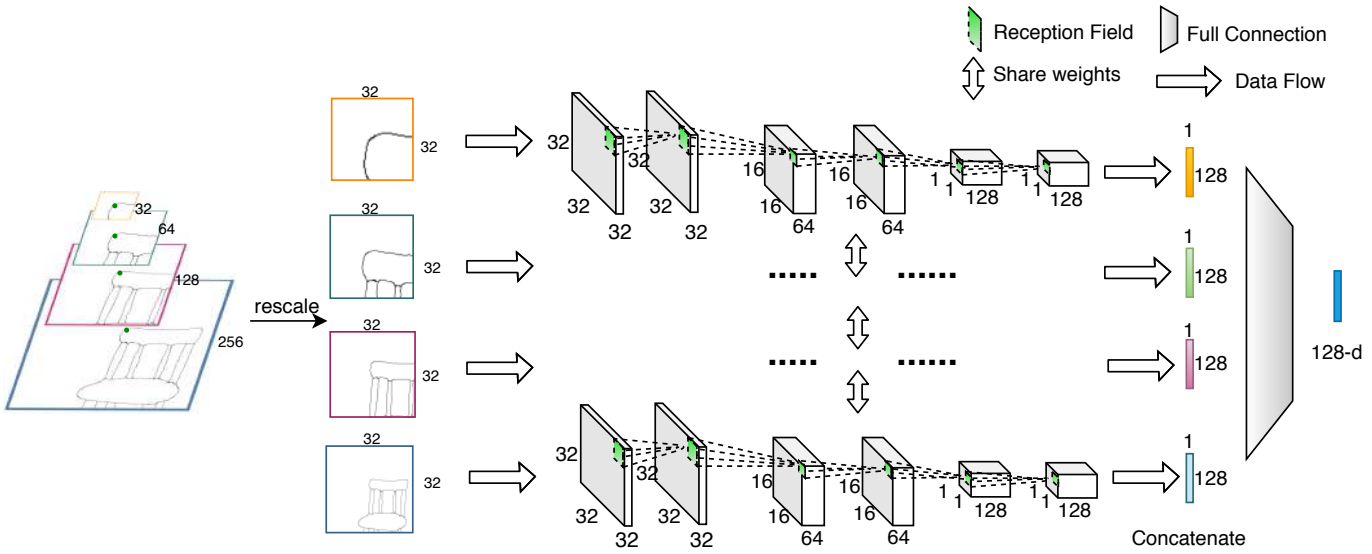


Fig. 4. The architecture of SketchDesc-Net. Our input is a four-scale patch pyramid (32×32, 64×64, 128×128, 256×256) centered at a pixel of interest on a sketch, with each scale rescaled to 32×32. Given the multi-scale patches, we design a multi-branch framework with shared weights to take as input these rescaled patches. The dashed lines represent the data flow from an input patch to an output descriptor. For the kernel size and stride in our network, we adopt the same settings as [3]. Finally, the output as a 128-D descriptor embeds features from the four scales by concatenation and full connection operations.

where $D_m\langle\cdot,\cdot\rangle$ is a metric function, and p_{c1} and p_{c2} represent corresponding pixels from different sketches (e.g., the pair of anchor and positive pixels in Figure 3) while p_{c1} and $\widetilde{p}_{nc}$ are a pair of non-corresponding pixels (e.g., the pair of anchor and negative pixels in Figure 3).

We follow [3] to build the metric function D_m by learning a sketch descriptor (SketchDesc) with a triplet loss function. Basically we optimize the loss function in Equation 2, which minimizes the distance between pairs of corresponding pixels and maximizes the distance between pairs of non-corresponding pixels. Since sketch images are rather sparse, we adopt a multi-scale patch-based representation, that is, multi-scale patches centered at a pixel of interest in a sketch. We resort to deep CNNs, which own the superior capability of learning discriminative visual features from sufficient training data. As illustrated in Figure 4, our designed network (Section III-B) takes a multi-scale patch as input and outputs a 128-D distinctive descriptor.

To train the network, we first synthesize line drawings of a 3D shape from different viewpoints as multi-view sketches. We then generate the ground-truth correspondences by first uniformly sampling points on the 3D shape and then projecting them to the corresponding multiple views. We will discuss the process of data preparation in more detail in the next section.

A. Data Preparation

Multi-view Sketches with Ground-truth Correspondences. We follow a similar strategy in [55], [57], [58] to synthesize sketches from 3D shapes. Specifically, we first render a 3D shape with the aligned upright orientation to a normal map under a specific viewpoint, and then extract an edge map from the normal map using Canny edge detection. We adopt this approach instead of the commonly used suggestive contours [59], because the latter is suited for high-quality 3D meshes and cannot generate satisfactory contours from poorly-triangulated meshes (e.g., airplanes and rifles in the Structure Recovery dataset [19]). Hidden lines of the edge detection results are removed. In our implementation, each sketch is resized to a 480×480 image. As mentioned in [50], most humans are not faithful artists and create sketches in a casual and random way. Unlike [42], [54] using limited views, to better accommodate the shape and viewpoint variations in freehand sketches, we sample viewpoints on the upper unit viewing hemisphere (in an elevation angle of 15∼45 degrees) at every 15 or 30 degrees in the azimuth angle for rendering each 3D model.

By projecting each vertex to the corresponding views, we naturally construct ground-truth correspondences (with the projections from the same vertices) among synthesized multi-view sketches, as illustrated in Figure 2. We do not consider hidden vertex projections (invisible under depth testing). If the projections of a vertex are visible only in less than two different views, this vertex is not considered in ground-truth correspondences. Following this strategy, we can generate from 28K to 60K ground-truth pairwise correspondences from each 3D shape. Our synthesized correspondence dataset for training and testing are derived from 6,852 multi-view sketches distributed over 18 shape categories of existing shape datasets [16], [18], [19] (Section IV). We will make our dataset publicly available.

Multi-scale Patch Representation. We represent each visible projection of a 3D vertex on sketch images with a patch-based representation centered at the corresponding pixel to capture the distinctive neighboring structures (Figure 3). To better handle the sparsity and the lack of texture information in sketches, we adopt a multi-scale strategy. Given a 480×480 sketch image, we employ a four-scale representation (i.e., 32×32, 64×64, 128×128, and 256×256) for a pixel, as illustrated in Figure 3.

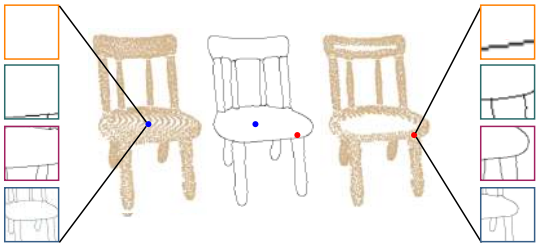


Fig. 5. An illustration of two sampling mechanisms (Left: OR-sampling; Right: AND-sampling) for training data. OR-sampling generates more challenging data (blue) than AND sampling (red).

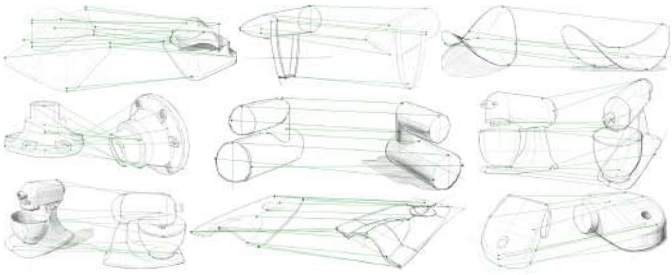


Fig. 6. Sketch correspondence in the OpenSketch dataset computed with SketchDesc descriptors. Note that even with limited ground-truth correspondence in the training set of OpenSketch, SketchDesc can still establish a robust correspondence for the multi-view sketches.

The multi-scale patch-based representation allows us to sample ground-truth correspondences inside sketched objects, and not necessarily on sketch lines. This significantly increases the number of ground-truth correspondences for training. However, we do require valid information existing in a multi-scale patch representation. We have tried two sampling mechanisms: 1) *OR-sampling*: a multi-scale patch is valid if the patch is non-empty at any of the scales (Figure 5 left); 2) *AND-sampling*: a multi-scale is valid if the patch is non-empty at every scale (Figure 5 right). The former generates valid multi-scale patches at almost every visible vertex projection, since the patch at scale 256×256 is often non-empty given its relatively large scale. In contrast, the AND-sampling generates valid multi-scale patches only near to sketch lines. We will compare the performance of these two sampling mechanisms in Section IV.

Before feeding these patches into our multi-branch network, we rescale all the patches to 32×32 (i.e., the smallest scale) by bilinear interpolation (Figure 4). Below we describe our network architecture in more detail.

B. Network Architecture

We design a network architecture to learn local descriptors for measuring the semantic distance between a pair of pixels in multi-view sketches. As illustrated in Figure 4, our network has four branches to process the set of four scaled input patches. The four branches share the same architecture and weights in the whole learning process. Each branch receives a 32×32 patch and outputs a 128×1 (i.e., 128-D) feature vector which is then further fused at the concatenation layer and the final fully-connected layer. Note that due to the shared weights among the branches, the multi-branch structure does not increase the number of parameters. Our network produces a 128-D descriptor as output, which will be later used for sketch correspondence and pixel-wise retrieval in Section IV.

C. Objective Function

To train our network, we employ the random sampling strategy of [3] to assemble the triplets (Figure 3) in a training batch. We define the output descriptors of a triplet as (f_a, f_p, f_n), where f_a is the descriptor vector of an anchor pixel in one sketch, and f_p and f_n represent the descriptors of the corresponding pixel (corresponding to the same vertex in a 3D shape as the anchor pixel) and a non-corresponding pixel. The non-corresponding pixel can be selected from either the same sketch or in the other views. With the descriptor triplets, we adopt the triplet loss [60] to train the network. The triplet loss, given in Equation 2, aims to pull closer the distance between a pair of corresponding pixels (f_a, f_p) and push away the distance between a pair of non-corresponding pixels (f_a, f_n) in the metric space.

$$L_{triplet} = \frac{1}{n}\Sigma_{i=1}^{n} \max\left(0, d\left(f_{a_i}, f_{p_i}\right) - d\left(f_{a_i}, f_{n_i}\right) + m\right), \tag{2}$$

where n is the number of triplets in a training batch, $(f_{a_i}, f_{p_i}, f_{n_i})$ denotes the i-th triplet, $d(\cdot,\cdot)$ measures the Euclidean distance given two descriptors, and the margin m is set to 1.0 in our experiments.

IV. Experiments

We conducted extensive experiments on two sketch datasets: our synthesized multi-view sketch dataset and the OpenSketch dataset [20]. For our synthesized dataset (Figure 10), we utilize three existing 3D shape repositories: the Structure-Recovery database [19], Princeton Segmentation Benchmark (PSB) [18], and ShapeNet [16]. Table I shows some information about the three shape repositories, including the selected categories, the number of 3D shapes per category, and the number of views used per category. The shapes in each category were manually selected based on the criterion of increasing shape diversity and decreasing the redundant and repeated shapes. For the OpenSketch dataset, it has around 400 sketches of 12 man-made objects, which were drawn by professional product designers. Different from the synthesized sketches, the OpenSketch data contains abundant annotations of additional shadings, skeletons and auxiliary lines (shown in Figure 1 (bottom) and Figure 6).

In addition to the hand-drawn sketches in OpenSketch, we also performed evaluation on a set of more freely-drawn sketches. Several participants were invited to create freehand sketches on a touchscreen, after observing a 3D shape for a fixed amount of time. Each participant was given three salient views of each shape that ordinary users would be familiar with. Figure 1 (two middle rows) shows some representative results of sketch correspondence (i.e., chair, bicycle, and fourleg). Note that to avoid visual clutter, for each freehand sketch pair, we randomly selected 20∼50 pairs of matched

	Structure-Recovery						PSB									ShapeNet							
Category	Airplane	Bycicle	Chair	Fourleg	Human	Rifle	Airplane	Bust	Chair	Cup	Fish	Hand	Human	Octopus	Plier	Airplane	Bag	Cap	Car	Chair	Earphone	Mug	Pistol
Shapes	54	12	27	16	16	25	20	20	20	20	20	20	20	20	20	38	35	35	39	23	28	34	25
Views	11	12	12	12	12	12	11	12	12	11	12	11	12	11	11	11	12	12	12	12	12	12	12

TABLE I
OBJECT CATEGORIES AND THE NUMBER OF SHAPES AND VIEWS USED IN OUR DATASET.

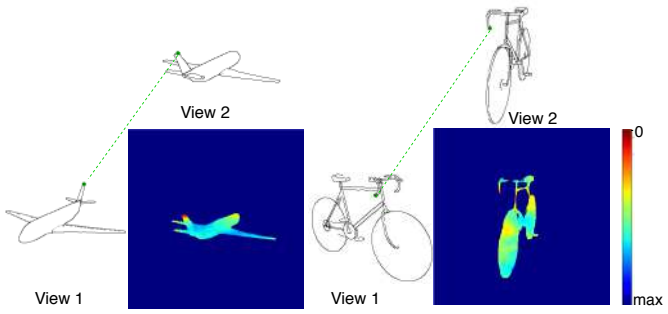


Fig. 7. For a given pixel inside a sketched object under View 1, we find a corresponding point in the other sketch under View 2 by computing a distance map through our learned descriptor.

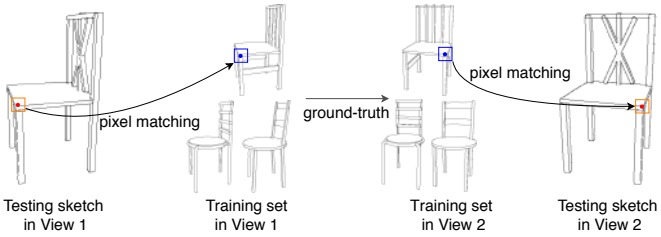


Fig. 8. The pipeline of a retrieval-based baseline, which directly utilizes the ground-truth correspondence in the training set. For illustration, we show the top-3 sketches retrieved from the training set by pixel matching.

correspondences computed by nearest neighbor search with SketchDesc (see Section IV-B1). In the supplementary material, we provide more results of the computed correspondences among hand-drawn multi-view sketches.

A. Implementation Detail

We implemented our network with the PyTorch [61] framework and used the Xavier initialization [62]. We train our network for each object category with a data splitting ratio of $8:1:1$ (training : validation : testing). All multi-view sketches are rendered to the size of 480×480. The batch size is set to 64. Our network is trained on an NVIDIA RTX 2080Ti GPU and optimized by the Adam [63] optimizer ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) with a learning rate of $1e^{-3}$. The number of iteration epochs in our experiments is set to 100.

B. Performance Evaluation

To verify the effectiveness of our proposed network, we design two evaluation tasks: multi-view sketch correspondence and pixel-wise retrieval. We compare our approach with the existing learning-based descriptors, including LeNet [64], L2-Net [3], HardNet [4], SOSNet [5] and AlexNet-based view pooling [42], [54] (AlexNet-VP in short). Note that we reimplemented LeNet, L2-Net, HardNet, SOSNet, and AlexNet-VP following their original configurations. For AlexNet-VP, we use the same multi-scale patch inputs as our network. The training input to all other networks is a single-scale 32×32 patch. For a fair comparison, we only consider pixels that pass the AND-sampling criteria, that is, the 32×32 patch of each multi-scale input should be non-empty.

1) Sketch Correspondence: In this task, we validate the performance of the learned descriptors in finding corresponding pixels in pairs of multi-view sketches in the testing set. Given a pair of testing sketches, for each pixel in one sketch, we compute its distances to all pixels in the other sketch (see distance visualization in Figure 7). We consider it as a successful matching if the pixel with the shortest distance is no further than 16 pixels (half of the smallest patch scale) away from the ground-truth pixel. Note that not all pixels in a sketch image have ground-truth correspondences (only among those projected).

To clarify the dissimilarity between the training set and the testing set, we designed a retrieval-based baseline (illustrated in Figure 8) in the task of sketch correspondence.

For a given pixel on one sketch in View 1 of a testing pair, we perform the pixel matching operation (by nearest neighbor search) to get the most similar pixel among all the training sketches under View 1. Note that the pixel matching operation takes as input the patches (32×32) centered on the pixels. With the matched pixel, we utilize the ground-truth cross-view correspondence in the training data to find its corresponding pixel in View 2 (i.e., the same view as the other sketch in the testing sketch pair). Finally, we perform the pixel matching again to find the most similar pixel on the testing sketch in View 2. Note that we employ the conventional image matching method [26] for the pixel matching operation.

The performance of our approach and other competing methods on several representative categories is reported in Table II. We report the averaged success rate as the correspondence accuracy. We observe that our network outperforms its learning-based and retrieval-based competitors. Due to the lack of texture in sketch patches, the frameworks designed for image patches (i.e., L2-Net, HardNet, and SOSNet) cannot learn effective descriptors to match the corresponding pixels in multi-view sketches. Our descriptor also surpasses those based on LeNet and AlexNet-VP. Figure 9 gives some qualitative comparisons between different approaches. Since 32×32 patches are required to be non-empty, the testing pixels are near sketch lines. We observe that the descriptors learned by

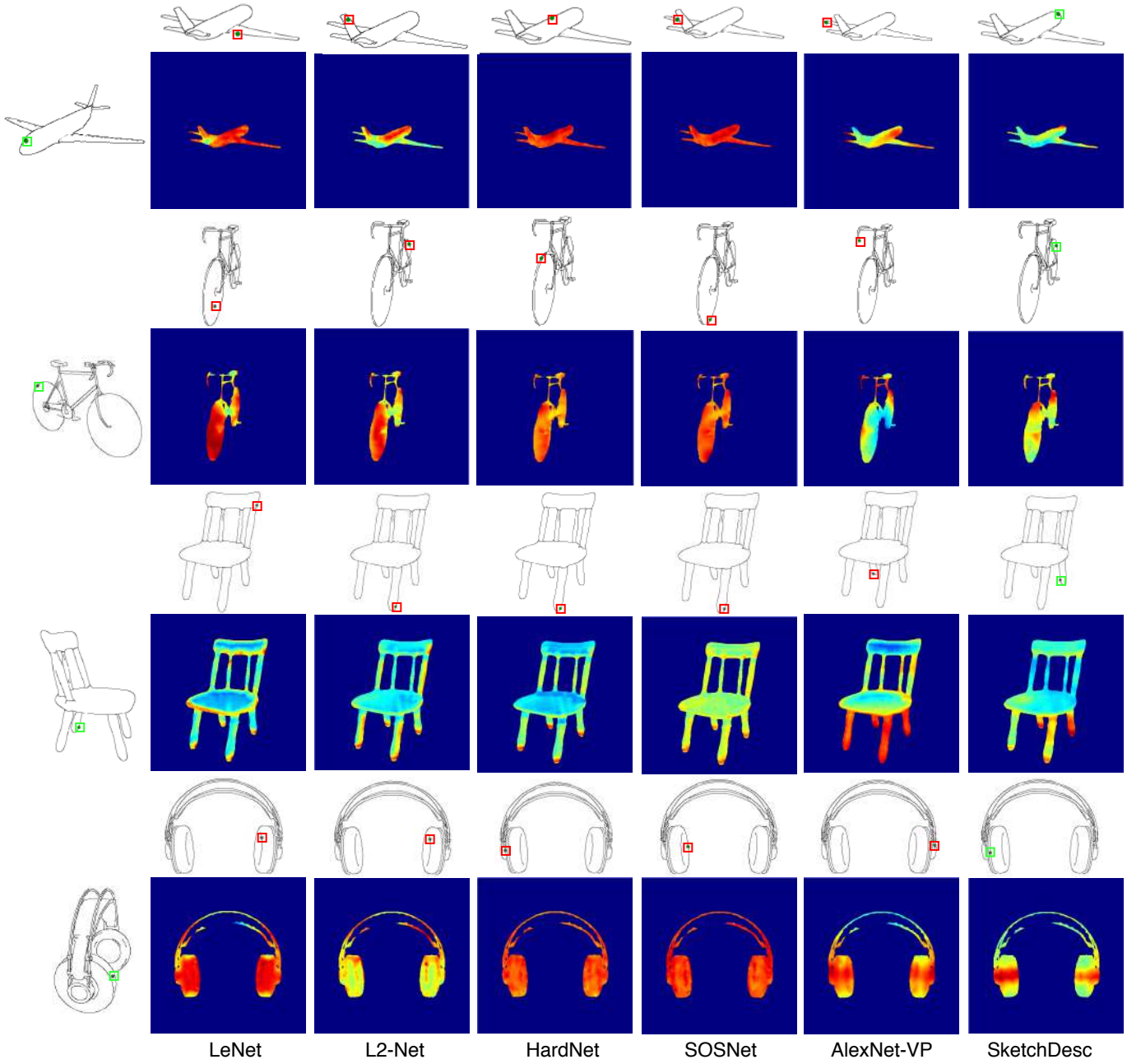


Fig. 9. Sketch correspondence results by computing the distance maps with different approaches. Correct and wrong matching results are marked as green and red boxes, respectively.

LeNet, HardNet, L2-Net, and SOSNet are less discriminative, and are thus not effective in finding corresponding pixels among multi-view sketches. AlexNet-VP uses the max-pooling layer, making it difficult to focus on local details for robust descriptors, as discussed in [4]. As a consequence, AlexNet-VP produces more ambiguous distance maps (e.g., the chair example in Figure 9) compared to our method. The retrieval-based baseline employs the ground-truth from the training set, but it still fails to build reliable correspondences for multi-view sketches. In fact, the retrieval-based baseline has the worst performance. It is mainly because of the incapability of the image-based descriptor [26] on dealing with the highly similar local patches with limited textures in multi-view sketches, and also because of the large disparity between multi-view sketches in the training set and testing set (Figure 10). In addition, it is challenging to use this baseline in practice with hand-drawn sketches, due to the required view prior and its large running time to infer the correspondences for a pair of multi-view sketches (3.15 minutes vs 8.76 seconds by our method on average).

Figure 1 shows successful matchings randomly selected from all successful matchings on the synthesized multi-view sketches (top) and the hand-drawn sketches (middle). In Figures 1(bottom) and 6, we also show that SketchDesc can

		base.	LeNet	L2Net	HardNet	SOSNet	AlexNet-VP	SketchDesc
Structure-Recovery	Airplane	0.14	0.18	0.24	0.21	0.19	0.36	**0.54**
	Bicycle	0.28	0.38	0.39	0.42	0.40	0.66	**0.71**
	Chair	0.27	0.40	0.40	0.39	0.39	0.51	**0.56**
	Fourleg	0.27	0.29	0.34	0.32	0.28	0.52	**0.66**
	Human	0.21	0.28	0.38	0.40	0.35	0.64	**0.73**
	Rifle	0.27	0.43	0.52	0.53	0.50	0.67	**0.76**
	avg_acc	0.24	0.33	0.38	0.38	0.35	0.56	**0.66**
PSB	Airplane	0.07	0.18	0.23	0.13	0.10	0.52	**0.72**
	Bust	0.20	0.33	0.22	0.23	0.20	0.31	**0.45**
	Chair	0.22	0.29	0.27	0.24	0.26	0.44	**0.60**
	Cup	0.26	0.22	0.25	0.19	0.22	0.35	**0.57**
	Fish	0.29	0.30	0.36	0.35	0.31	0.63	**0.64**
	Human	0.20	0.35	0.35	0.38	0.34	0.54	**0.79**
	Octopus	0.08	0.10	0.11	0.09	0.07	0.24	**0.28**
	Plier	0.09	0.10	0.09	0.06	0.14	0.57	**0.73**
	avg_acc	0.18	0.23	0.24	0.21	0.21	0.45	**0.62**
ShapeNet	Airplane	0.21	0.39	0.44	0.40	0.35	0.67	**0.69**
	Bag	0.11	0.17	0.14	0.15	0.14	0.31	**0.38**
	Cap	0.17	0.21	0.25	0.26	0.22	0.55	**0.75**
	Car	0.13	0.17	0.21	0.21	0.21	0.55	**0.73**
	Chair	0.16	0.19	0.20	0.18	0.19	0.42	**0.47**
	Earphone	0.12	0.15	0.18	0.16	0.14	0.25	**0.42**
	Mug	0.21	0.19	0.20	0.17	0.17	0.32	**0.39**
	Pistol	0.23	0.30	0.34	0.33	0.34	0.66	**0.71**
	avg_acc	0.17	0.22	0.25	0.23	0.22	0.47	**0.57**

TABLE II
SKETCH CORRESPONDENCE ACCURACY (I.E., THE AVERAGE SUCCESS RATE) FOR THE DIFFERENT METHODS. THE BEST RESULTS IN EACH OBJECT CATEGORY ARE IN BOLDFACE.

		LeNet	L2-Net	HardNet	SOSNet	AlexNet-VP	SketchDesc
Structure-Recovery	Airplane	0.33	0.42	0.45	0.40	0.45	**0.68**
	Bicycle	0.37	0.40	0.44	0.39	0.60	**0.84**
	Chair	0.39	0.45	0.42	0.42	0.70	**0.82**
	Fourleg	0.33	0.45	0.41	0.30	0.75	**0.82**
	Human	0.20	0.44	0.40	0.27	0.74	**0.92**
	Rifle	0.56	0.56	0.57	0.54	0.76	**0.83**
	avg_map	0.36	0.45	0.45	0.39	0.67	**0.82**
PSB	Airplane	0.26	0.26	0.11	0.11	0.42	**0.68**
	Bust	0.01	0.29	0.29	0.26	0.51	**0.64**
	Chair	0.33	0.48	0.42	0.46	0.75	**0.86**
	Cup	0.05	0.08	0.04	0.06	0.22	**0.50**
	Fish	0.19	0.28	0.23	0.21	0.58	**0.73**
	Human	0.13	0.53	0.52	0.46	0.79	**0.93**
	Hand	0.08	0.37	0.37	0.28	0.52	**0.64**
	Octopus	0.38	0.29	0.29	0.46	0.67	**0.71**
	Plier	0.18	0.36	0.21	0.18	0.79	**0.86**
	avg_map	0.18	0.33	0.28	0.27	0.58	**0.73**
ShapeNet	Airplane	0.14	0.22	0.15	0.09	0.37	**0.55**
	Bag	0.17	0.20	0.15	0.14	0.53	**0.69**
	Cap	0.16	0.24	0.19	0.17	0.59	**0.71**
	Car	0.20	0.19	0.22	0.22	0.56	**0.67**
	Chair	0.11	0.15	0.14	0.140	0.51	**0.67**
	Earphone	0.25	0.30	0.25	0.20	0.49	**0.83**
	Mug	0.05	0.06	0.06	0.08	0.28	**0.34**
	Pistol	0.29	0.35	0.35	0.36	0.69	**0.82**
	avg_map	0.17	0.21	0.19	0.17	0.50	**0.66**

TABLE III
PIXEL-WISE RETRIEVAL PERFORMANCE FOR THE DIFFERENT METHODS.

be further utilized to infer some correspondences of multi-view sketches in the OpenSektch dataset. Due to the limited ground-truth correspondence in OpenSketch, we show all the successful matchings. More correspondence results (successful and unsuccessful matchings) of multi-view sketches can be found in the supplementary material.

2) Multi-view Pixel-wise Retrieval: We further design a multi-view corresponding pixel retrieval task. Given multi-view sketches synthesized from multiple shapes, we uniformly sample a set of pixels (1000∼1600 pixels) on one sketch and search in the other sketches for the corresponding pixels which are from the same shape (in different views). We use the descriptors computed from the compared networks as queries and adopt the Mean Average Precision (MAP) [65] to measure the retrieval performance. Let $\boldsymbol{y} = (y_1, \ldots, y_n) \in \{-1, +1\}^n$ be the labels of a ranked list of pixels returned for a pixel query, with -1 and +1 indicating negative and positive match, respectively. Then the *precision* at rank i is given by[1] $P_i(\boldsymbol{y}) = \sum_{k=1}^{i}[y_k]_+ / \sum_{k=1}^{i}|y_k|$ and the *average precision* (AP) is given by $AP(\boldsymbol{y}) = \sum_{k:y_k=+1} P_k(\boldsymbol{y}) / \sum_{k=1}^{N}[y_k]_+$. Finally, given N_q as the number of total query pixels, the mean average precision (MAP) is computed by $MAP = \sum_{y=1}^{N_q} AP_y / N_q$. Experiments are performed category-wise in Structure-Recovery, PSB, and ShapeNet. To further report the performance on a dataset, we adopt avg_map which averages the MAP over all the tested categories.

[1]Here $[z]_+ = \max\{0, z\}$.

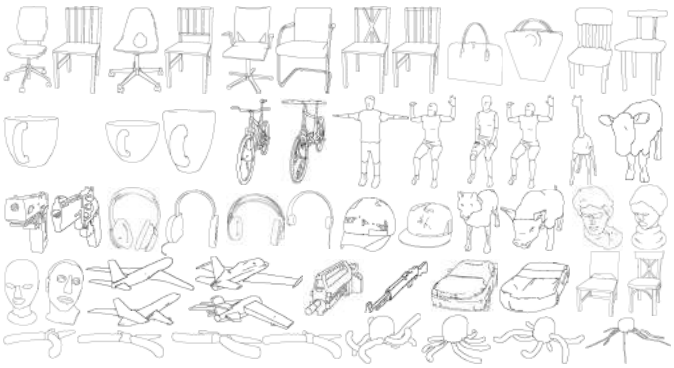


Fig. 10. The pairs of a testing sketch (left) and its most similar training sketch (right) retrieved from the training set under the same view, utilizing the image matching method [26].

Table III shows the results given by the compared methods. SketchDesc achieves the best performance among all the learned descriptors. Our learned descriptor surpasses the image-based descriptors of L2-Net, HardNet, and SOSNet by a large margin. For AlexNet-VP, it achieves a closer but still lower performance compared with our method.

3) Cross-dataset Validation: Considering the limited data in the testing set, we further demonstrate the generalization ability of SketchDesc with a cross-dataset validation.

For the three shape datasets (Structure-Recovery, PSB, and ShapeNet), we take two of them as the training set and the remaining one as the testing set. To reduce the effects of imbalanced data distribution, we report the performance on a more balanced and overlapping category (Chair), which has 27, 20, and 23 shapes in each of the three datasets respectively (Table I). We report the performance of the sketch correspondence task and the multi-view pixel-wise retrieval task in Table IV and Table V respectively. We observe that SketchDesc still shows an overwhelming superiority over its competitors. There is only a slight drop (0.02 on average) in the cross-dataset performance of the sketch correspondence task, compared to Table II. However, the cross-dataset performance of the pixel-wise retrieval task shows a noticeable degradation (0.15 on average), compared to Table III. This is mainly because that the pixel-wise retrieval task has a much larger search space (multi-view sketches of multiple shapes) than the sketch correspondence task (paired multi-view sketches of a single shape), and tends to lead to more mismatches due to ambiguity. In general, the robustness and effectiveness of SketchDesc are further verified by the large and unseen testing data.

4) View Disparity: To further show the robustness of different learned descriptors against the degree of view disparity, given the same input testing pixels in one sketch, we visualize how the quality of correspondence inference

Methods	Structure-Recovery&ShapeNet / PSB	PSB&ShapeNet / Structure-Recovery	Structure-Recovery&PSB / ShapeNet
LeNet	0.27	0.35	0.21
L2-Net	0.23	0.40	0.26
HardNet	0.24	0.39	0.27
SOSNet	0.24	0.40	0.26
AlexNet-VP	0.26	0.39	0.26
SketchDesc	**0.55**	**0.49**	**0.54**

TABLE IV
CROSS-DATASET (CHAIR) PERFORMANCE OF DIFFERENT METHODS IN THE SKETCH CORRESPONDENCE TASK. THE TRAINING SETS ARE LABELED WITH UNDERLINES.

Methods	Structure-Recovery&ShapeNet / PSB	PSB&ShapeNet / Structure-Recovery	Structure-Recovery&PSB / ShapeNet
LeNet	0.07	0.01	0.04
L2-Net	0.16	0.35	0.13
HardNet	0.19	0.32	0.14
SOSNet	0.17	0.35	0.15
AlexNet-VP	0.54	0.40	0.24
SketchDesc	**0.73**	**0.64**	**0.54**

TABLE V
CROSS-DATASET (CHAIR) PERFORMANCE OF DIFFERENT METHODS IN THE PIXEL-WISE RETRIEVAL TASK. THE TRAINING SETS ARE LABELED WITH UNDERLINES.

Different scales	Structure-Recovery	PSB	ShapeNet
32^2	0.47	0.31	0.20
64^2	0.64	0.44	0.45
128^2	0.75	0.65	0.61
256^2	0.76	0.66	0.62
$64^2 + 128^2 + 256^2$	0.81	0.69	0.66
$32^2 + 128^2 + 256^2$	0.81	0.67	0.65
$32^2 + 64^2 + 256^2$	0.79	0.68	0.64
$32^2 + 64^2 + 128^2$	0.77	0.67	0.61
$32^2 + 64^2$	0.70	0.57	0.48
$32^2 + 64^2 + 128^2$	0.77	0.67	0.61
$32^2 + 64^2 + 128^2 + 256^2$	**0.82**	**0.73**	**0.66**

TABLE VI
THE PERFORMANCE OF USING DIFFERENT SCALE COMBINATIONS AS INPUTS TO SKETCHDESC-NET IN THE PIXEL-WISE RETRIEVAL TASK.

	Structure-Recovery	PSB	ShapeNet
W/o shared weights	0.80	0.65	0.63
W/ shared weights	**0.82**	**0.73**	**0.66**

TABLE VII
THE PERFORMANCE OF OUR METHOD WITH OR WITHOUT SHARED-WEIGHTS.

changes with the increasing view disparity. As shown in Figure 11, SketchDesc shows a more stable performance of correspondence inference than the competitors. Please note that the ground-truth corresponding pixels might become invisible in certain views, and all the learned descriptors could not distinguish the visibility of corresponding pixels. Nevertheless, SketchDesc still produces the most reasonable results.

C. Ablation Study

In this subsection, we validate the effectiveness of the key components of our method with ablation studies.

Multi-scale Strategy. The designed multi-scale patch-based representation ($32\times32, 64\times64, 128\times128, 256\times256$) plays an essential role in our method. We first show how different scales can influence the performance of the learned descriptors. We test our network with increasingly more scales and evaluate its performance on the pixel-wise retrieval task. We employ the average MAP (Mean Average Precision) metric over the whole dataset. Quantitative results are shown in Table VI. We can see that the features from the larger scales are more discriminative than the features from smaller scales by ablating one of the multiple scales in Table VI (rows 5 - 8). As we remove the larger scale, the poorer performance of SketchDesc is witnessed on all three datasets. Multiple scales are better than a single scale. A representative visual comparison is shown in Figure 12. It is found that as larger scales are involved, the ambiguous regions (bright yellow regions) on the feet, legs and backs of the camel are gradually rejected. In other words, with the multi-scale patches as inputs, our network can enjoy not only a more precise local perception but also a global perspective.

Shared Weights. In our method, the multi-scale patches are processed by a shared-weight scheme. To verify its effectiveness, we perform a comparison on the pixel-wise retrieval task with an unshared-weight network structure. The comparison results are reported in Table VII. It is found that the shared-weight structure in our network achieves higher accuracy. The improvement is even more significant on PSB. The results confirm a similar design choice of shared-weight structure used in existing studies like [42] and [54].

Training Data Generation. We evaluate the performance of two patch sampling mechanisms on the task of pixel-wise retrieval: OR-sampling and AND-sampling (Section III-A). It can be found from Table VIII that OR-sampling achieves a significantly better performance. This is because the OR-sampling leads to a significantly larger dataset for training our network.

D. Limitations and Discussions

Our method has the following limitations. First, although our method could generalize well to unseen sketches or even hand-drawn sketches of the same objects, when the viewpoint differs from the examples in the training set drastically, our method could fail. This is a common generalization problem for any learning-based methods. Increasing the training data could help yet in the cost of additional training burden. We use a rather simple method to sample viewpoints for preparing the training data. A more careful view selection might be made by adopting best-view selection methods [66]. Additionally, our method is currently designed for multi-view correspondences of rigid objects. If the object undergoes articulation or non-rigid deformations (e.g. people dancing), our method may not perform well. We consider this as an intriguing future work to explore.

Sampling Mechanism	Structure-Recovery	PSB	ShapeNet
AND-sampling	0.79	0.68	0.59
OR-sampling	**0.82**	**0.73**	**0.66**

TABLE VIII
THE PERFORMANCE OF TWO SAMPLING MECHANISMS FOR DATA PREPARATION ON THE TASK OF MULTI-VIEW PIXEL-WISE RETRIEVAL. NOTE THAT DIFFERENT SAMPLING MECHANISMS ARE ONLY USED TO GENERATE THE TRAINING DATA.

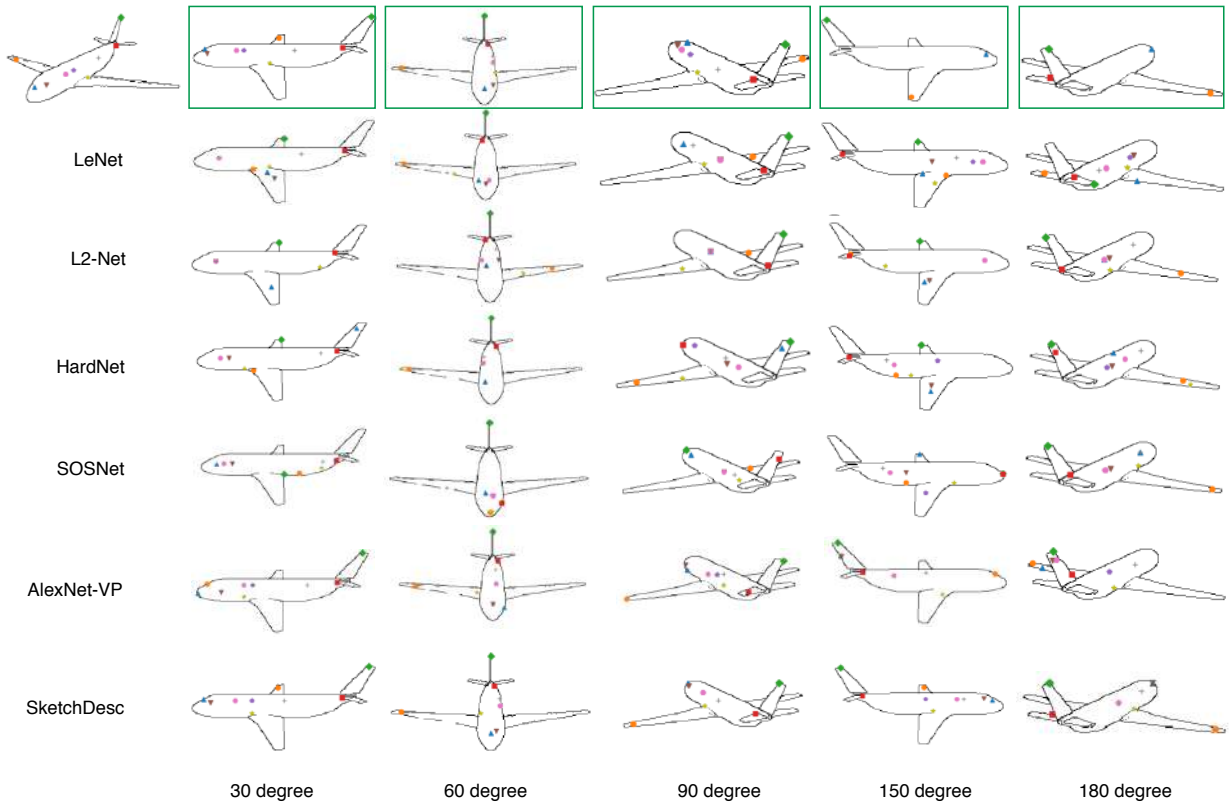


Fig. 11. Performance of different methods with increasing view disparity (30, 60, 90, 150, and 180 degrees). Given some anchor pixels on the sketched object at top-left, we show the corresponding pixels computed by the different methods. The ground-truth correspondences are labeled with green boxes.

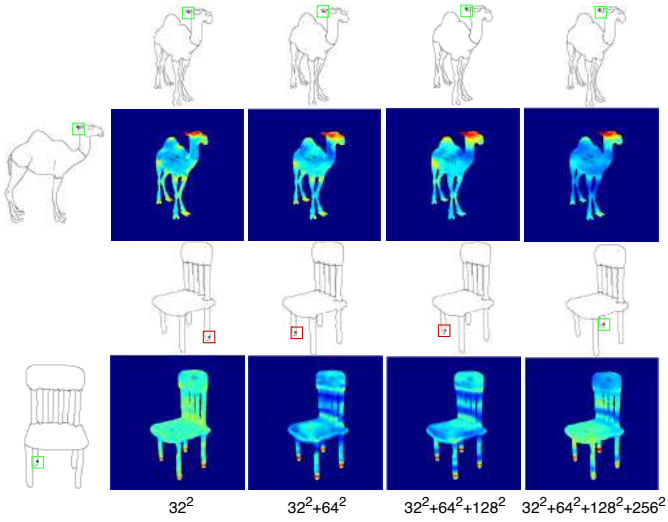


Fig. 12. Visualization of different multi-scale choices. The distance maps show the distances from the highlighted point in the left sketch to all the pixels in the other sketch.

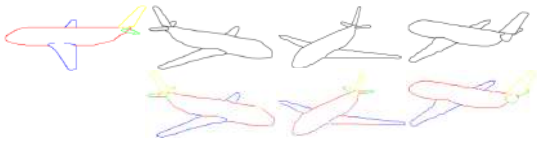


Fig. 13. Sketch segmentation transfer. The top row are the inputs: one segmented sketch and several unlabeled sketches. The bottom row are the outputs: sketches with point-wise labels after graph-cut postprocessing. With SketchDesc, we can transfer the labels among multi-view sketches with the computed correspondence.

V. Applications

A. Sketch Segmentation Transfer

Sketch segmentation is a challenging task that demands plenty of human-labeled training data [57], [67]. Noris et al. [68] proposed a scribble-based UI for user-guided segmentation of sketchy drawings, which still requires substantial human efforts. With SketchDesc, we show that segmentation labels can be easily transferred across multi-view sketches (Figure 13). Specifically, we use SketchDesc to produce 128-d descriptors for every point on the sketches. With the correspondences established by the descriptors, we transfer the segmentation from one labeled sketch to other views. As shown in Figure 13, although there are some distortions in the hand-drawn sketches, we can still obtain reasonable segmentation results through transfer.

B. Multi-view Image-based Correspondence

Image-based descriptors [3]–[5] mostly rely on the information of textures in patches to build the correspondence among multi-view images. If the input images lack discriminative texture details (Figure 14), the image-based methods (e.g. the state-of-art SOSNet [5]) may fail to extract robust descriptors

Fig. 14. Correspondence matching among multi-view images of different objects.

with their single-scale input. Here we show that our sketch-based descriptor SketchDesc can also be extended to this correspondence establishment among multi-view photos (Figure 14) under such challenging situations even with no textures. Here we use edge maps as a proxy, that is, we first convert a photo to its edge map format and then apply SketchDesc to obtain local descriptors for matching corresponding points in different views. Figure 14 shows that SketchDesc can infer a reasonable correspondence among multi-view images of different objects on mere edge maps, indicating the potential of our proposed SketchDesc in the image domain. The reason why SketchDesc outperforms SOSNet is because the multi-scale strategy in our method gives more confidence with the global and local perspectives. This further emphasizes the importance of th multi-scale idea in both the image and sketch domains. On the other hand, while we believe SketchDesc can assist existing photo correspondence techniques especially when images do not have rich textures, we do not claim that SketchDesc is a general solution for photo correspondence.

VI. Conclusions

In this paper, we have introduced a deep learning based method for correspondence learning among multiple sketches of an object in different views. We have proposed a multi-branch network that encodes contexts from multi-scale patches with global and local perspectives to produce a novel descriptor for semantically measuring the distance of pixels in multi-view sketch images. The multi-branch and shared-weights designs help the network capture more feature information from all scales of sketch patches. Our data preparation method provides the ground truth effectively for training our multi-branch network. We believe the generated data can benefit other applications. Both qualitative and quantitative experiments show that our learned descriptor is more effective than the existing learning-based descriptors. In the future, it would be interesting to exploit more neighboring information and learn the per-point features in a joint manner.

References

[1] L. Governi, R. Furferi, M. Palai, and Y. Volpe, "3d geometry reconstruction from orthographic views: A method based on 3d image processing and data fitting," *Computers in Industry*, vol. 64, no. 9, pp. 1290–1300, 2013.

[2] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," in *NIPS*, 2001, pp. 831–837.

[3] Y. Tian, B. Fan, and F. Wu, "L2-Net: Deep learning of discriminative patch descriptor in euclidean space," in *CVPR '17*, 2017, pp. 661–669.

[4] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," in *Advances in Neural Information Processing Systems*, 2017, pp. 4826–4837.

[5] Y. Tian, X. Yu, B. Fan, F. Wu, H. Heijnen, and V. Balntas, "Sosnet: Second order similarity regularization for local descriptor learning," in *CVPR '19*, 2019, pp. 11 016–11 025.

[6] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1582–1599, 2008.

[7] F. Tombari, S. Mattoccia, L. Di Stefano, and E. Addimanda, "Classification and evaluation of cost aggregation methods for stereo correspondence," in *CVPR '08*, 2008, pp. 1–8.

[8] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang, "A deep visual correspondence embedding model for stereo matching costs," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 972–980.

[9] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *CVPR '16*, 2016, pp. 5695–5703.

[10] G. Yang, J. Manela, M. Happold, and D. Ramanan, "Hierarchical deep stereo matching on high-resolution images," in *CVPR '19*, 2019, pp. 5515–5524.

[11] T. Tung, S. Nobuhara, and T. Matsuyama, "Complete multi-view reconstruction of dynamic scenes from probabilistic fusion of narrow and wide baseline stereo," in *ICCV '09*. IEEE, 2009, pp. 1709–1716.

[12] S. Liu and D. B. Cooper, "Ray markov random fields for image-based 3d modeling: Model and efficient inference," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 1530–1537.

[13] G. Nishida, I. Garcia-Dorado, D. G. Aliaga, B. Benes, and A. Bousseau, "Interactive sketching of urban procedural models," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 130, 2016.

[14] J. Delanoy, M. Aubry, P. Isola, A. A. Efros, and A. Bousseau, "3d sketching using multi-view deep volumetric prediction," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 1, no. 1, p. 21, 2018.

[15] C. Li, H. Pan, Y. Liu, X. Tong, A. Sheffer, and W. Wang, "Robust flow-guided neural prediction for sketch-based freeform surface modeling," in *ACM Transactions on Graphics*. ACM, 2018, p. 238.

[16] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas *et al.*, "A scalable active framework for region annotation in 3d shape collections," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, p. 210, 2016.

[17] Q. Yu, Y. Yang, F. Liu, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Sketch-a-net: A deep neural network that beats humans," *International Journal of Computer Vision*, vol. 122, no. 3, pp. 411–425, 2017.

[18] X. Chen, A. Golovinskiy, and T. Funkhouser, "A benchmark for 3d mesh segmentation," in *ACM Transactions on Graphics*, vol. 28, no. 3. ACM, 2009, p. 73.

[19] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu, "Structure recovery by part assembly," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 180, 2012.

[20] Y. Gryaditskaya, M. Sypesteyn, J. W. Hoftijzer, S. Pont, F. Durand, and A. Bousseau, "Opensketch: a richly-annotated dataset of product design sketches," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, p. 232, 2019.

[21] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, "Building rome in a day," *Communications of the ACM*, vol. 54, no. 10, pp. 105–112, 2011.

[22] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," in *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3. ACM, 2006, pp. 835–846.

[23] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[24] G. Csurka and M. Humenberger, "From handcrafted to deep local invariant features," *CoRR*, vol. abs/1807.10254, 2018.

[25] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European Conference on Computer Vision*. Springer, 2006, pp. 404–417.

[26] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR '05*, 2005.

[27] S. Winder and M. Brown, "Learning local image descriptors," in *Proc. IEEE CVPR*, June 2007.

[28] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "HPatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *Proc. IEEE CVPR*, 2017.

[29] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "MatchNet: Unifying feature and metric learning for patch-based matching," in *Proc. IEEE CVPR*, 2015, pp. 3279–3286.

[30] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *CVPR '15*, June 2015.

[31] X. Zhang, F. X. Yu, S. Kumar, and S.-F. Chang, "Learning spread-out local feature descriptors," in *ICCV '17*, 2017.

[32] X. Wei, Y. Zhang, Y. Gong, and N. Zheng, "Kernelized subspace pooling for deep local descriptors," in *CVPR '18*, 2018.

[33] V. Kumar B G, G. Carneiro, and I. Reid, "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," in *Proc. IEEE CVPR*, 2016.

[34] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proc. IEEE CVPR*, 2016.

[35] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks," in *Proc. BMVC*, 2016, pp. 119.1–119.11.

[36] M. Keller, Z. Chen, F. Maffra, P. Schmuck, and M. Chli, "Learning deep descriptors with scale-aware triplet networks," in *CVPR '18*, 2018.

[37] D. Mishkin, F. Radenovic, and J. Matas, "Repeatability is not enough: Learning affine regions via discriminability," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 284–300.

[38] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in *ICCV '15*, 2015, pp. 118–126.

[39] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker, "Universal correspondence network," in *NIPS*, 2016.

[40] Z. Luo, T. Shen, L. Zhou, S. Zhu, R. Zhang, Y. Yao, T. Fang, and L. Quan, "Geodesc: Learning local descriptors by integrating geometry constraints," in *ECCV '18*, 2018, pp. 168–183.

[41] P. Shilane and T. Funkhouser, "Distinctive regions of 3d surfaces," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 2, p. 7, 2007.

[42] H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V. G. Kim, and E. Yumer, "Learning local shape descriptors from part correspondences with multiview convolutional networks," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 1, p. 6, 2018.

[43] D. Ha and D. Eck, "A neural representation of sketch drawings," *arXiv preprint arXiv:1704.03477*, 2017.

[44] N. Wang, D. Tao, X. Gao, X. Li, and J. Li, "A comprehensive survey to face hallucination," *International journal of computer vision*, vol. 106, no. 1, pp. 9–30, 2014.

[45] M. Zhu, J. Li, N. Wang, and X. Gao, "A deep collaborative framework for face photo–sketch synthesis," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 10, pp. 3096–3108, 2019.

[46] M. Zhu, N. Wang, X. Gao, J. Li, and Z. Li, "Face photo-sketch synthesis via knowledge transfer." in *IJCAI*, 2019, pp. 1048–1054.

[47] L. Li, C. Zou, Y. Zheng, Q. Su, H. Fu, and C.-L. Tai, "Sketch-r2cnn: An attentive network for vector sketch recognition," *arXiv preprint arXiv:1811.08170*, 2018.

[48] P. Sangkloy, N. Burnell, C. Ham, and J. Hays, "The sketchy database: learning to retrieve badly drawn bunnies," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 119, 2016.

[49] P. Xu, Y. Huang, T. Yuan, K. Pang, Y.-Z. Song, T. Xiang, T. M. Hospedales, Z. Ma, and J. Guo, "Sketchmate: Deep hashing for million-scale human sketch retrieval," in *CVPR '18*, 2018, pp. 8090–8098.

[50] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *ACM Trans. Graph.*, vol. 31, no. 4, pp. 44–1, 2012.

[51] A. Rivers, F. Durand, and T. Igarashi, "3d modeling with silhouettes," *ACM Trans. Graph.*, vol. 29, no. 4, Jul. 2010.

[52] Z. Lun, M. Gadelha, E. Kalogerakis, S. Maji, and R. Wang, "3d shape reconstruction from sketches via multi-view convolutional networks," in *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 67–77.

[53] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, X. Bing, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *International Conference on Neural Information Processing Systems*, 2014.

[54] P. Navarro, J. I. Orlando, C. Delrieux, and E. Iarussi, "Sketchzooms: Deep multi-view descriptors for matching line drawings," *arXiv preprint arXiv:1912.05019*, 2019.

[55] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *ICCV '15*, 2015, pp. 945–953.

[56] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[57] L. Li, H. Fu, and C.-L. Tai, "Fast sketch segmentation and labeling with deep learning," *IEEE computer graphics and applications*, vol. 39, no. 2, pp. 38–51, 2018.

[58] K. Xu, K. Chen, H. Fu, W.-L. Sun, and S.-M. Hu, "Sketch2scene: sketch-based co-retrieval and co-placement of 3d models," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, p. 123, 2013.

[59] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, "Suggestive contours for conveying shape," in *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3. ACM, 2003, pp. 848–855.

[60] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[61] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, 2019, pp. 8026–8037.

[62] X. Glorot and 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.

[63] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[64] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[65] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *CVPR*, 2017.

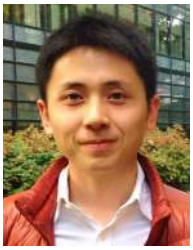
[66] C. H. Lee, A. Varshney, and D. W. Jacobs, "Mesh saliency," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 659–666, 2005.

[67] R. G. Schneider and T. Tuytelaars, "Example-based sketch segmentation and labeling using crfs," *ACM Trans. Graph.*, vol. 35, no. 5, Jul. 2016.

[68] G. Noris, D. Sỳkora, A. Shamir, S. Coros, B. Whited, M. Simmons, A. Hornung, M. Gross, and R. Sumner, "Smart scribbles for sketch segmentation," in *Computer Graphics Forum*, vol. 31, no. 8. Wiley Online Library, 2012, pp. 2516–2527.

Deng Yu is pursuing the Ph.D. degree at the School of Creative Media, City University of Hong Kong. He received the B.Eng. degree and the Master degree in computer science and technology from China University of Petroleum (East China). His research interests include computer graphics and data-driven techniques.

Lei Li is working toward the Ph.D. degree at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. He received the B.Eng. degree in software engineering from Shandong University. His research interests include computer graphics and data-driven techniques.



Youyi Zheng is a Researcher at the State Key Lab of CAD&CG, College of Computer Science, Zhejiang University. He obtained his Ph.D. from the Department of Computer Science and Engineering at Hong Kong University of Science and Technology, and his M.Sc. and B.Sc. degrees in Mathematics, both from Zhejiang University. His research interests include geometric modeling, imaging, and human-computer interaction.

Manfred Lau is an Assistant Professor in the School of Creative Media at the City University of Hong Kong. His research interests are in computer graphics, human-computer interaction, and digital fabrication. His recent research in the perception of 3D shapes uses crowdsourcing and learning methods for studying human perceptual notions of 3D shapes. He was previously an assistant professor in the School of Computing and Communications at Lancaster University in the UK, and a post-doc researcher in Tokyo at the Japan Science and Technology Agency - Igarashi Design Interface Project. He received his Ph.D. degree in Computer Science from Carnegie Mellon University, and his B.Sc. degree in Computer Science from Yale University. He has served in the program committees of the major graphics conferences including Siggraph Asia.

Yi-Zhe Song is a Reader of Computer Vision and Machine Learning at the Centre for Vision Speech and Signal Processing (CVSSP), where he directs the SketchX lab. Previously, he was a Senior Lecturer at the Queen Mary University of London, and a Research and Teaching Fellow at the University of Bath. He obtained his PhD in 2008 on Computer Vision and Machine Learning from the University of Bath, and received a Best Dissertation Award from his MSc degree at the University of Cambridge in 2004, after getting a First Class Honours degree from the University of Bath in 2003. He is a Senior Member of IEEE, and a Fellow of the Higher Education Academy. He is a full member of the review college of the Engineering and Physical Sciences Research Council (EPSRC), the UK's main agency for funding research in engineering and the physical sciences, and serves as an expert reviewer for the Czech National Science Foundation.

Chiew-Lan Tai is a Professor at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. She received the B.Sc. degree in mathematics from University of Malaya, the M.Sc. degree in computer and information sciences from National University of Singapore, and the D.Sc. degree in information science from the University of Tokyo. Her research interests include geometry processing, computer graphics, and interaction techniques.

Hongbo Fu is a Professor with the School of Creative Media, City University of Hong Kong. He received the B.S. degree in information sciences from Peking University, and the Ph.D. degree in computer science from Hong Kong University of Science and Technology. He has served as an Associate Editor of The Visual Computer, Computers & Graphics, and Computer Graphics Forum. His primary research interests include computer graphics and human computer interaction.

Supplemental Materials
SketchDesc: Learning Local Sketch Descriptors for Multi-view Correspondence

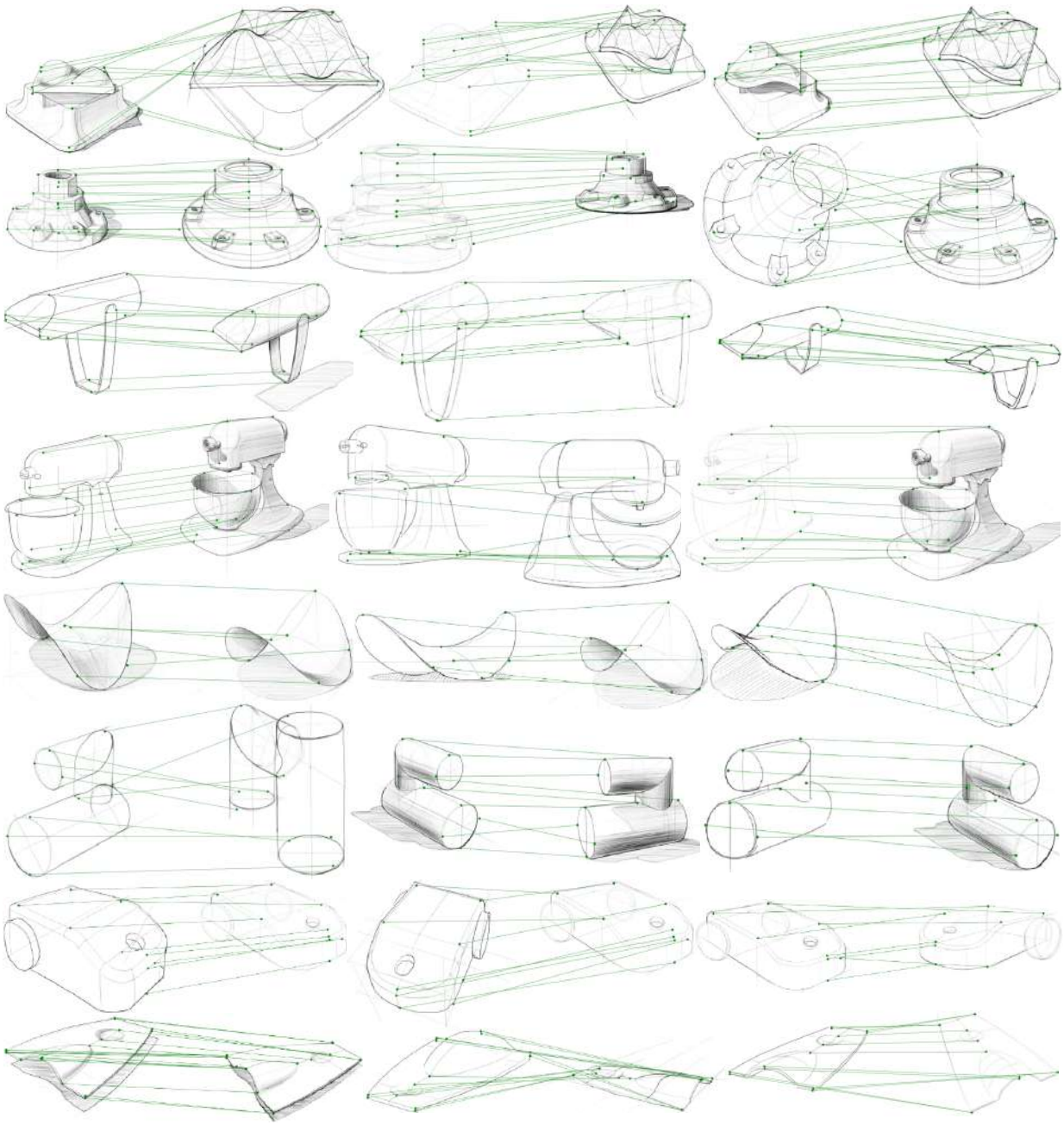


Fig. 1. Correspondence of multi-view sketches built by SketchDesc descriptors. All of the multi-view sketches are from the OpenSketch dataset.

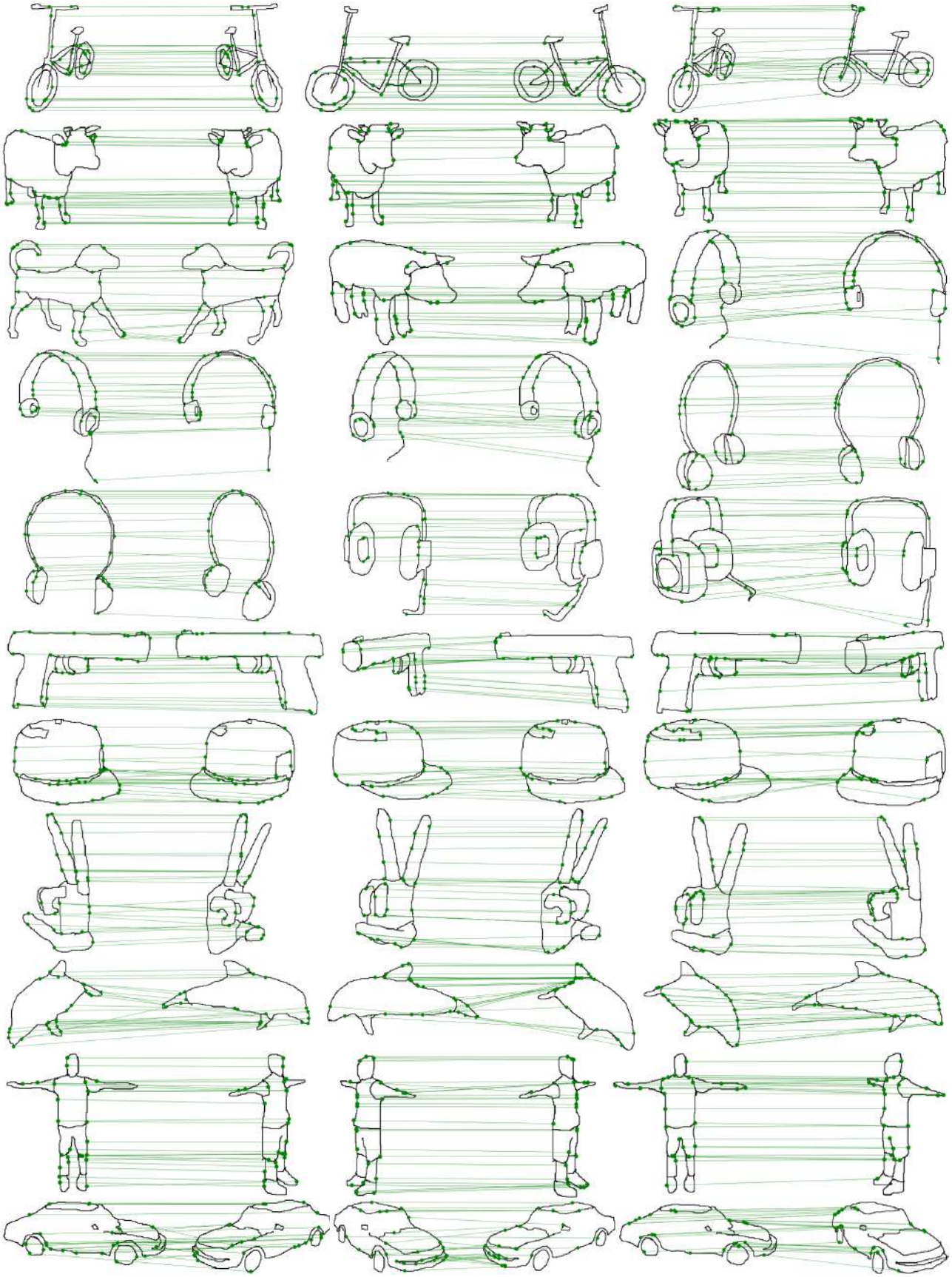


Fig. 2. Correspondence of multi-view sketches built by SketchDesc descriptors. All of these multi-view sketches are created by volunteers.

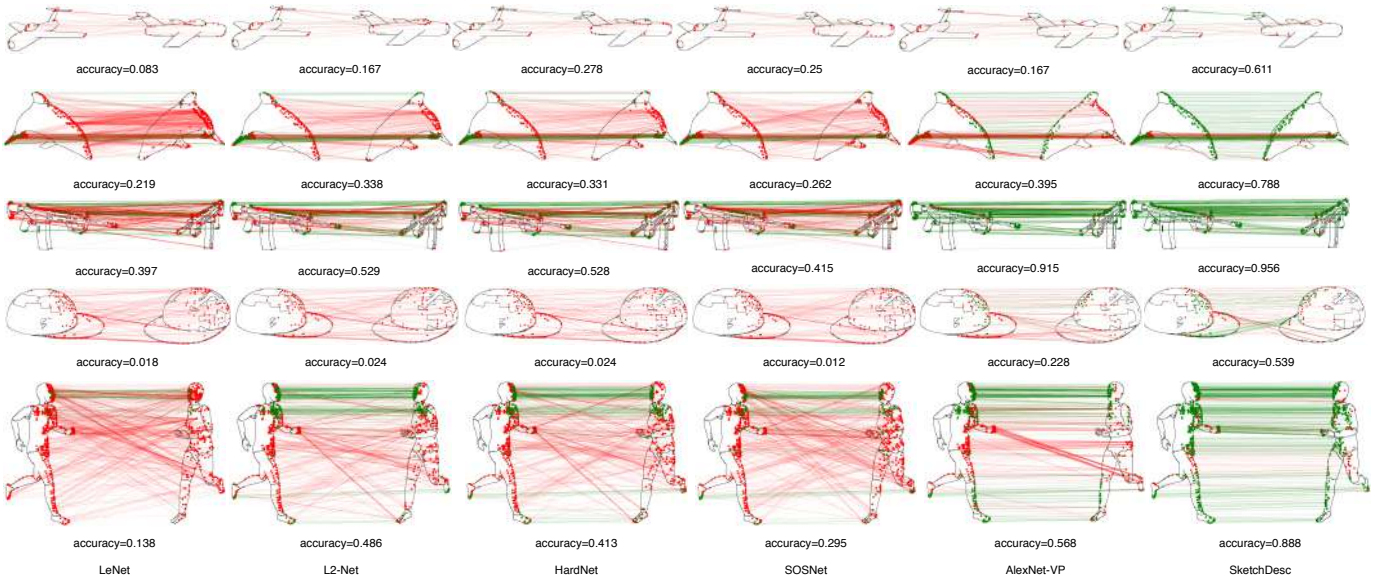


Fig. 3. Sketch correspondence for multi-view sketches (synthesized). The red lines indicate the failed matching and the green lines show the matching correspondences among multi-view sketches.