Towards 3D VR-Sketch to 3D Shape Retrieval

Ling Luo[1,2] Yulia Gryaditskaya[1,2] Yongxin Yang[1,2] Tao Xiang[1,2] Yi-Zhe Song[1,2]

[1]SketchX, CVSSP, University of Surrey [2]iFlyTek-Surrey Joint Research Centre on Artificial Intelligence

Abstract

Growing free online 3D shapes collections dictated research on 3D retrieval. Active debate has however been had on (i) what the best input modality is to trigger retrieval, and (ii) the ultimate usage scenario for such retrieval. In this paper, we offer a different perspective towards answering these questions – we study the use of 3D sketches as an input modality and advocate a VR-scenario where retrieval is conducted. Thus, the ultimate vision is that users can freely retrieve a 3D model by air-doodling in a VR environment. As a first stab at this new 3D VR-sketch to 3D shape retrieval problem, we make four contributions. First, we code a VR utility to collect 3D VR-sketches and conduct retrieval. Second, we collect the first set of 167 *3D VR-sketches on two shape categories from ModelNet. Third, we propose a novel approach to generate a synthetic dataset of human-like 3D sketches of different abstract levels to train deep networks. At last, we compare the common multi-view and volumetric approaches: We show that, in contrast to 3D shape to 3D shape retrieval, volumetric point-based approaches exhibit superior performance on 3D sketch to 3D shape retrieval due to the sparse and abstract nature of 3D VR-sketches. We believe these contributions will collectively serve as enablers for future attempts at this problem. The VR interface, code and datasets are available at* `https://tinyurl.com/3DSketch3DV`.

1. Introduction

3D model retrieval has become an important topic due to a growing number of free online 3D repositories. It finds applications in CAD design, 3D printing, 3D animation and movies production, where the time required to model a 3D shape can be strongly reduced by relying on retrieval from existing 3D shape collections. Various input modalities have been tried – images [15] and rough 2D sketch [33] at first, with latest research focusing on 3D to 3D, i.e., using 3D scans [1, 5] or existing 3D shapes [31, 25, 22, 7, 10, 11, 13, 24, 26, 32].

Despite great strides made, two salient questions still remain (i) what constitutes the best input modality to initiate 3D retrieval, and (ii) under what usage scenario can such retrieval be best facilitated. For the former, 2D sketches and images are both in 2D, therefore can not reach the level of details desired, and the 2D-3D domain gap can also be counter-intuitive. The 3D-based paradigm on the other hand dictates existing 3D models to be readily available, which to some degree forms a "chicken-and-egg" problem (i.e., where/how to source the input model at the first place). As for the latter, apart from 2D sketches which are freely defined by the user, all other usage scenarios do not offer flexibility in terms of the input desired – one can not easily alter an image, not to mention a 3D model/scan.

In this paper, we offer a new perspective on 3D shape retrieval – we advocate the use of 3D sketches as a new input modality. This new 3D-VR sketch to 3D shape paradigm not only enables detailed retrieval as per the common 3D model to 3D model setting, but also simultaneously offers a degree of flexibility found elsewhere in 2D sketch-based retrieval. Our ultimate vision is as follows: with a specific 3D model in mind, one emerges into a VR environment, roughly sketch out the model using handheld controllers, press retrieve and then relevant models would start populating the VR environment – for any downstream tasks the user may desire (e.g., VR shopping, 3D printing).

Our first contribution is therefore coding a VR environment for the said purpose. With this VR environment, we collected the first human VR-sketch dataset. 10 users with no artistic background were hired to produce a total of 167 3D VR-sketches from the two categories from ModelNet: chairs and bathtubs. Some examples are shown in Figure 1. Since the collection process is both time- and cost-sensitive, we additionally propose the first 3D model to 3D sketch generator, and construct a synthetic dataset of 3D sketches. A key trait of our generator is that it can produce 3D sketches of different abstraction levels, effectively simulating that found in real-human sketches. Through training a series of deep 3D VR-sketch to 3D retrievals model using the real and synthetic datasets, we drive out a few important insights (i) retrieval performance drops with an increasing abstraction level, and (ii) models trained with synthetic sketches can already reach a decent performance level when

2475-7888/20/$31.00 ©2020 IEEE
DOI 10.1109/3DV50981.2020.00018

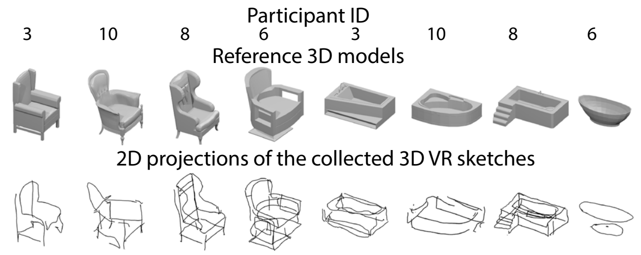

Figure 1: Our collected VR sketches exhibit variability of sketching styles and levels of details.

tested on human sketches.

At last, we experiment with different shape/sketch representations, and state-of-the-art losses commonly used in 3D model retrieval [14, 13, 32] but re-purposed for our problem. We focus on investigating the domain gap between 3D shapes and 3D sketches (see Figure 1 for an example), due to the two key properties of sketches: *sparsity*: full 3D models versus sparse sketched lines, and *abstraction*: 3D models are geometrically perfect, while sketches are subject to deformations. In particular, we examine multi-view versus point-based 3D representations, and show that the later is more robust to both sparsity and an increased level of abstractness. We further propose an architecture with a reconstruction path to explicitly allow for change in abstractness.

In summary, our contributions include: (i) a new perspective on 3D model retrieval, where 3D VR-sketches are used for the first time to conduct retrieval, (ii) a dataset of human 3D VR-sketches, collected using a purpose-built VR environment, (iii) an approach to generate a synthetic 3D sketch with a variable level of abstractness, (iv) comprehensive evaluations using recent 3D shape retrieval models re-purposed to the task of 3D sketch-based retrieval, to drive out insights, plus a novel regularization track to address the sketch-model domain gap.

2. Related work

3D shape representation. When dealing with 3D shape retrieval from a single image or a 3D shape, existing work is divided into two large groups based on the shape representation used: view-based [31, 25, 22, 7, 10, 11, 13] or volumetric. The volumetric representation can be further broken down to point-cloud [29, 24, 26, 32] or voxel-based [36, 23, 3, 19]. In this work we show that on the task of 3D sketch to 3D model retrieval point-cloud representation beats multi-view approaches due to better handling of the sparsity of 3D sketches.

2D image- and 3D shape-based retrieval. The retrieval problem in a multi-class setting is closely related to the shape classification problem, where the intermediate embedding of an image or a 3D shape is used for retrieval. A vanilla approach for multi-class shape classification is to use a softmax cross-entropy loss [13, 20]. Others works specifically tackle retrieval [31, 14], where the triplet loss and its variants [34, 38, 35, 14] have become a common standard. Amongst these, [14] represents the state-of-the-art for 3D model retrieval. It combines triplet and center losses [35], to solve for their respective drawbacks by using the class center as a positive sample. In this work we evaluate both the triplet and the triplet center losses on the task of 3D sketch-based retrieval and combine them with an additional Reconstruction Loss, tailored to the 3D sketch-based retrieval problem.

Due to a domain gap between images and 3D shapes, or target and query 3D models being from different distributions, Siamese [38, 27] or Heterogeneous [37, 15, 20] network architectures can be more beneficial for a certain problem. In this work, we compare these two types of architectures for our multi-view baseline, and show that due to a strong domain gap between a 3D sketch and a 3D shape, the Heterogeneous architecture by far beats the Siamese one.

3D sketch-based 3D model retrieval. So far there was very little work on 3D sketch-based retrieval [17, 16, 37, 8], especially in recent years. Most of them work with sketches collected using Microsoft Kinect, which not only has limited tracking accuracy, but the collection interface is also counter-intuitive, where 3D sketching is performed while visualizing 2D projections. As a result, sketches collected mostly have low fidelity and exhibit less details. Our VR-sketches are completely different: (i) visualization and sketching are both performed in 3D, and (ii) the use of the latest VR technology offers a high precision. Together, they ensure our VR-sketches are of high-fidelity and rich in details, thus more fitting for retrieval. A notable exception is the work by Giunchi et al. [8], yet they address a different problem of 3D model retrieval for dense-color VR sketches (and optionally base 3D shapes), while we target at a more simplistic and abstract shape representation – a sparse set of single-color lines. Note that we can not find the said dataset [17, 16, 37] in an open access, thus can not offer a direct comparison. Our NGVNN [13] baseline is however already superior to the state-of-the-art used by [37] and [8].

Non-photorealistic rendering (NPR). NPR is an old graphics and vision problem, for a detailed overview of existing methods for generating 2D NPR rendering we refer the interested reader to a recent report [2]. The only attempt to produce the representation that resembles a 3D sketch automatically form a 3D shape was proposed by Li et al. [17] as a concatenation of the shape views from six canonical viewpoints. They use this representation to show that the performance of their non-learning method on outline to 3D model retrieval task significantly outperforms the performance of the 3D sketch to 3D model retrieval. This experiment indicates that such simple shape representation is not sufficient as training data for 3D sketch-based 3D shape retrieval. Our experiments support that: The network trained on sketches with higher abstractness values significantly outperforms the network trained on clean sketches. To the best of our knowledge, we are the first to propose a method to generate abstract 3D sketches from 3D models.

3. Datasets

Collecting a full dataset of 3D human sketches is a labor intensive task. We collect a small dataset of human sketches that we use to guide the synthetic dataset generation. We as well use it as a test set to validate that the network trained on the proposed synthetic data generalizes well to human sketches. To obtain the training data, we reside to a common strategy of generating a synthetic training data.

3.1. Selected shapes

For training and testing our sketch-based retrieval models we use the repaired clean manifold meshes[1] [4] from ModelNet10. We use all 10 classes from ModelNet10, with the total of 3958 shapes. We split the dataset into train, validation and test sets, which contain 2847, 317 and 792 shapes, respectively, ensuring a proportional split of shapes of each class between three sets.

3.2. Human 3D sketch dataset

Task. We are targeting 3D sketches created by novices, which can be viewed as an equivalent of the quick 2D sketches from the QuickDraw dataset [12]. To enable the usage of the collected sketches for fine-grained retrieval evaluation we built a dataset of paired sketches and 3D models. We experimented with a setting, where one can observe a 3D model in VR for an unlimited duration of time, and then is asked to sketch from memory. We observed that under this scenario the participants were sometimes omitting features important to accurately testing fine-grained retrieval, and instead let the participants to sketch over a reference 3D model (Figure 2). We provide in the supplemental a qualitative evaluation of the sketches from memory and the retrieval performance on such inputs. To mimic the level of detail that can be expected from the 3D sketches from the imagination, we opted to use wide ribbon lines, but do not pose any constraints on sketching style or level of detail.

Participants. We selected 139 chairs and 28 bathtub shapes from ModelNet10's test set, and hired 10 participants, who have no art background or 3D sketch experience. The shapes were randomly divided into 10 subsets, consisting from 13-14 chairs and 2-3 bathtubs each. Each participant was assigned with one of these subsets. There is no duplicated shapes between subsets.

[1] https://github.com/lei65537/Visual_Driven_Mesh_Repair

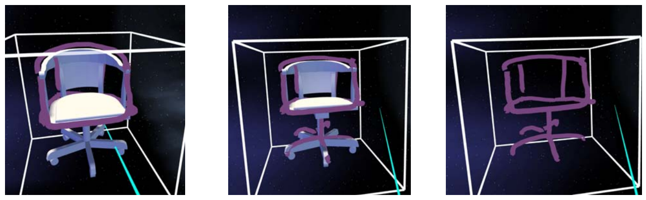
Figure 2: Our VR sketching environment allows to load the reference 3D model, and the user is asked to sketch around it. At any moment the user can hide the 3D model.

Interface. Although there are general-purpose VR painting and design softwares that enable users to draw directly in 3D (such as Google's Tilt brush[2], and Facebook's Quill[3]), they do not serve our purpose in full: (i) we would like to record detailed stroke-based information, (ii) we require the option of displaying a reference model for data collection, and (iii) we want a fresh code base for any additional functionalities in the future (e.g., sketch-based 3D editing). We implemented our custom 3D sketching environment based on Oculus Rift platform and Unity engine.

3.3. Synthetic training data generation

As a first step towards obtaining a synthetic sketch representation of a 3D shape, we extract detailed curve-networks with the method of Gori et al. [9]. This method is designed to produce a curve network that preserves well shape details, which are, though, uncommon for human novices sketches (Figure 3 b,c). We observe that novices not only omit small details, but tend to represent thin volumetric details with single lines and moreover often omit subset of feature lines (Figure 3 b). To obtain human sketch appearance, we focus on two aspects: level of detail and mechanical inaccuracies. We first perform details filtering and lines consolidation, followed by lines filtering. We then break the long curve chains into shorter strokes, to which we apply a set of local and global transformations (Figure 3 d).

3.3.1 Curves network extraction.

FlowRep method [9] requires an input to be a curvature-aligned quad-dominant mesh. For processing efficiency, we simplify original triangular meshes by decimation before converting them into quad-dominant meshes using Blender, but the quality of conversion does not always comply with the requirement of FlowRep. Thus, in this stage, we filtered nearly 20% of original dataset which failed to be processed by FlowRep. This can be alleviated in the future by replying on more advanced quad-meshing algorithms [21].

[2] https://www.tiltbrush.com/
[3] https://www.quill.com/

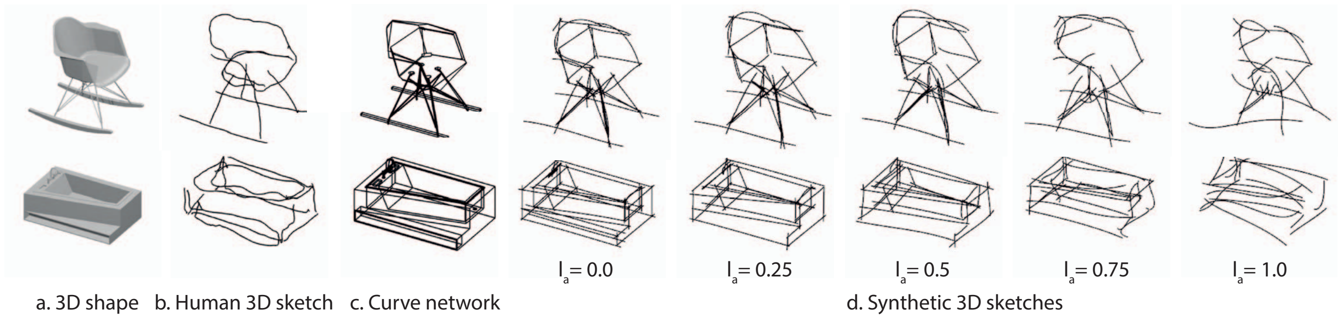

Figure 3: For a subset of shapes from the two categories from ModelNet10: chairs and bathtubs (a), we collect human novices sketches (b). We generate a synthetic dataset of 3D sketches for the 10 shape categories from ModelNet10. We first generate detailed curve networks with FlowRep [9] (c) and then apply a set of detail and stroke filtering steps to mimic the appearance of human sketches, where the appearance is controlled by an abstractness parameter l_a (d), defined in Section 3.3.3.

FlowRep allows to control the level of detail of the output curve network by adjusting the values of the descriptiveness threshold d_{max}. We, nevertheless, found little impact of this parameter on the output networks, as illustrated in Figure 4. We set this parameter to 20° to generate our synthetic dataset, following their default value.

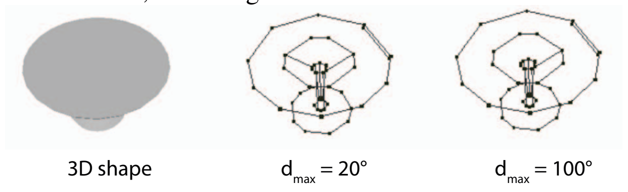

Figure 4: We found little effect of the descriptiveness threshold parameter d_{max} in FlowRep [9] on most of the shapes from ModelNet10 dataset.

3.3.2 Details filtering and consolidation.

The code by Gori et al. [9] returns networks of curves in a form of chained edges of an input mesh. We first break these chains into several if the angle between two consequent edges is smaller than 135 degrees. We then remove all the chains those length is smaller than 10% of the smallest of height and width of the bounding box of the original shape (d_{min}). We re-sample all the chains using Ramer-Douglas-Peucker algorithm [28] with an accuracy parameter set to $0.02\, d_{min}$. Finally, we iteratively go over all the chains and compute for each chain the closest, tangentially aligned chain. If the distance between such two chains is smaller than 5% of the largest length of the two chains, the chains are substituted by their aggregate curve. These steps allow to remove small details and close to each other lines. Yet, the simplified curve networks contain many more lines than can be found in majority of 3D sketches by novices.

3.3.3 Abstraction.

In this section we describe our approach to obtain a 3D sketch with different levels of detail and mechanical inaccuracies, which we jointly refer to as a level of sketch abstractness l_a. We constraint l_a to be between 0 and 1.

Level of detail (I) To reduce the number of lines we first compute the similarity matrix using a discrete Frechet distance among chains. When computing the Frechet distance we first align the strokes at one of their end points, by translating one of the strokes. We then perform the grouping according to this similarity matrix with Agglomerative Hierarchical Clustering, where the number of cluster $n_{cluster}$ is a function of l_a and the number of chains in the sketch n_{chains}: $n_{cluster} = \max(n_{chains}(1 - 0.8l_a)/2, 10)$. (II) We next for each cluster iteratively select a pair of two most distant lines in terms of their absolute positions and compute the mean distance from all the lines d_{mean} in the cluster to these two lines. We remove all the lines in a cluster for which the distance to any of the two selected lines is less than d_{mean}. (III) After getting the reduced set of chains, we break long chains into several shorter chains, which we refer to as strokes. We split each chain at vertices, where the curvatures are twice larger than the mean curvature of an original chain. We further filter out short strokes whose lengths are smaller than $0.2s_{max}$ to avoid tiny details, where s_{max} is the largest dimension of an input network.

Mechanical inaccuracies To mimic mechanical and perspective inaccuracies of human sketches we apply to each stoke a set of global and local deformations. We first apply a global translation, rotation and scaling, depending on the given level of abstraction l_a. To achieve this we deploy an auxiliary parameter t, which is randomly sampled from the range $[0, 1.5l_a]$. The rotation angle for each axes is then independently randomly sampled from the interval $[-10t, 10t]$ degrees. The scale factor for each axes is independently randomly sampled from $[1 - 0.1t, 1 + 0.1t]$. To obtain a translation vector we randomly sample from the surface of a sphere with its radius value randomly sampled from $[0, s_{max}t]$, where s_{max} is the largest dimension of an

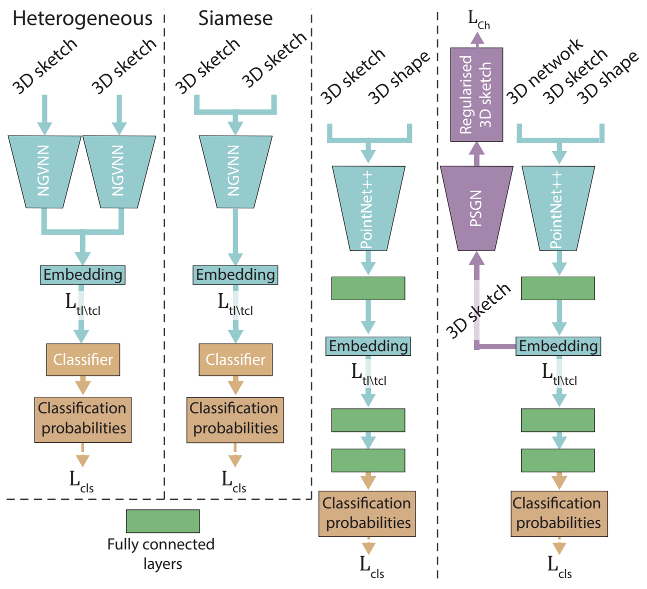

Figure 5: The two diagrams on the left demonstrate the heterogeneous and siamese network architectures with NGVNN as a backbone. The two diagrams on the right show the networks based on PointNet++, when trained as described in Section 4.2 and in Section 4.3.

input network, as before. After a global stroke deformation, we apply a random translation for each stroke vertex. The translation vector is randomly sampled from the disk with a radius r_{v_i}, which lies in the plane orthogonal to the stroke direction in the vertex v_i. The radius r_{v_i} is randomly sampled from the range $[0, 0.1 l_a l_{stroke}]$, where $l_{stroke} \in [0, 1]$ is the length of the current stroke. We extend both ends of each stroke by p which value is randomly sampled from $[0, 0.1 s_{max}]$, to reproduce human strokes appearance.

After global and local stroke deformations, we apply 3D spline interpolation, which results in smoother strokes, matching the appearance of human strokes.

For our experiments, we generate 5 synthetic datasets with 5 levels of abstraction, $[0.0, 0.25, 0.5, 0.75, 1.0]$, and obtain two mixed datasets by mixing sketches of all abstraction levels or only the sketches with 3 abstraction levels in the middle. Figure 3 c. shows example sketches obtained with different settings of l_a parameter.

4. Networks

4.1. Backbones

Multi-view representation We choose View N-Gram Network (NGVNN) [13] as a baseline for a multi-view based shape retrieval, since it was shown to outperform the alternative solutions [18, 14, 7, 31] on the 3D-shape based retrieval task on ModelNet40. We employ VGG-11 with batch normalization [30] as a backbone. The output of the penultimate layer is used as a feature embedding. We experiment with two architectures: one where the same network is used to encode the 3D sketch and 3D shape (Siamese), and one where there are two branches those weights are trained separately (Heterogeneous) (see Figure 5).

Volumetric representation Due to sparsity of a 3D sketch, among the volumetric shape representations we opted for point-based. In particular, we select PointNet++ [26], which applies hierarchical feature learning to efficiently capture local structures and to get a robust point cloud representation. We adopt their multi-scale grouping and random input dropout during training (MSG+DP) strategy, which is more robust to a variable sampling density on an input point cloud due to an adaptively selected neighborhood's size. We use the same three-level hierarchical network with three fully connected layers as in the original paper and use the output of the first fully connected layer as a feature embedding. We use the Siamese network for this model.

We train both PointNet++ and NGVNN from scratch for a fair comparison. The feature vector is a 512-dimensional vector in both cases.

4.2. Losses

We compare the performance of the two backbone 3D representations with the optimal sampling and rendering strategies, when trained with the classification loss $\mathcal{L}_{cls}$, versus training with a combinations of the classification loss and the triplet loss $\mathcal{L}_{tl}$ [34], or the combination of the classification loss and the triplet center loss $\mathcal{L}_{tcl}$ [14]:

$$\mathcal{L} = \lambda_{cls}\mathcal{L}_{cls} + \lambda_{tl\backslash tcl}\mathcal{L}_{tl\backslash tcl}, \tag{1}$$

we set $\lambda_{cls} = 1$ and $\lambda_{tl} = \lambda_{tcl} = 0.1$.

The goal of both triplet loss and triplet center loss is to obtain a better structured embedding space. The input for the cross-entropy loss $\mathcal{L}_{cls}$ are the class probabilities, while the input for the triplet loss and triplet center loss are features extracted by the encoders (Figure 5).

Triplet loss The goal of the triplet loss is to ensure that the anchor-negative distances are larger than the anchor-positive distances by a given margin. For the triplet loss $\mathcal{L}_{tl}$, we use both 3D sketches and 3D shapes as anchors. 3D shapes and 3D sketches sharing the same class as the anchor are treated as positive, while those from different classes are negative samples. The triplet loss for the i-th triplet is:

$$\mathcal{L}^i_{tl} = \max\{0, d^i_{pos} - d^i_{neg} + m_{tl}\}, \tag{2}$$

where d^i_{pos} is the anchor-positive distance and d^i_{neg} is the anchor-negative distance, m_{tl} is the margin. As a distance

d we use the squared euclidean distance, where the feature space is normalized to a unit hypersphere. We choose the margin value by calculating the average difference between d_{pos} and d_{neg} of the validation set of the model trained with the classification loss only. We set margin for NGVNN baseline to $m_{tl} = 2.0$ and for PointNet++ to $m_{tl} = 1.8$.

Triplet center loss Triplet center loss [14] combines the strength of the triplet loss and the center loss. It avoids the problem of triplet hard mining by taking as a positive the learned center of the shape's class. As a negative sample it takes the closest center of some other class:

$$\mathcal{L}^i_{tcl} = \max\{0, d(f^i, c_{y^i}) - \min_{j \neq y^i} d(f^i, c_j) + m_{tcl}\}, \quad (3)$$

where y^i is a class label of the anchor 3D sketch/shape, f^i is the embedding of an anchor and c_k is the center of the class k in the embedded space.

Given the data imbalance among classes of the used 3D shape dataset, we employ a balanced batch sampler. For each batch, we first randomly select $k = 8$ classes from all K classes, then we select $p = 1$ 3D sketch-shape pairs for each of the k class. For the efficiency, we use an on-the-fly hard negative triplet mining to calculate the triplet loss rather than generating the triplets beforehand.

The final loss $\mathcal{L}_{tl/tcl}$ is the mean of all triplet (center) losses within a batch: $\mathcal{L}_{tl\backslash tcl} = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}^i_{tl\backslash tcl}$, where N is the number of all valid triplets.

4.3. Regularization branch

To account for a variation of human styles, we aim at pushing the 3D sketch embedding towards style invariance by introducing a regularization branch. We propose the architecture with an additional decoder branch (Figure 5), based on the reconstruction point-based architecture (PSGN) by Fan et al. [6], and only use 2 fully connected layers to form the reconstruction branch. As a reconstruction loss, we use the Chamfer distance $\mathcal{L}_{Ch}$ between an input 3D sketch with a given level of abstraction and the 3D curve network obtained as described in Section 3.3.2. The full loss is

$$\mathcal{L} = \lambda_{cls}\mathcal{L}_{cls} + \lambda_{tl\backslash tcl}\mathcal{L}_{tl\backslash tcl} + \lambda_{Ch}\mathcal{L}_{Ch}, \quad (4)$$

where $\lambda_{cls} = 1$ and $\lambda_{tl\backslash tcl} = 0.1$, as before. We choose $\lambda_{Ch} = 8$ for TCL and $\lambda_{Ch} = 12$ for TL.

5. Evaluations

We adopt the following common evaluation measures to evaluate the retrieval performance: Mean Average Precision (mAP), Normalized Discounted Cumulative Gain (NDCG), Nearest Neighbor (NN), which evaluate the ability of a model to discriminate between shape classes, and Top-k accuracy, which measure how many of the retrieval tasks have a ground-truth within top k retrieved results.

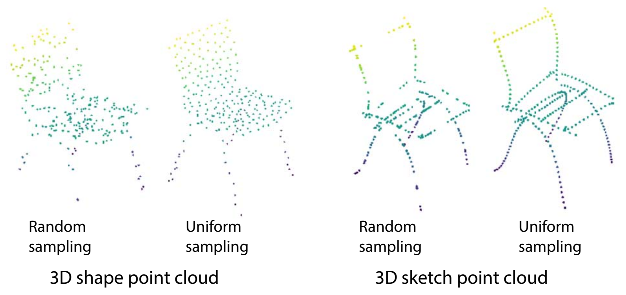

Figure 6: Visual comparison of random and uniform sampling strategies. To clearly show the difference, we set the number of points to 256 for this visualization.

5.1. Sampling and rendering strategies

We experiment with two rendering styles for NGVNN [13] and two sampling strategies for PointNet++ [26].

3D sketch rendering for multi-view network. We generate 12 224×224 orthographic views of each 3D shape and 3D sketch by placing 12 virtual cameras around it every 30 degrees, as was proposed by Lee et al. [15]. The cameras are elevated 30 degrees from the ground plane. For both 3D shapes and 3D sketches we experiment with two types of rendering styles: Phong Shading and depth maps. For 3D sketches we represent each line as a 3D tube.

Point cloud sampling. To get the point cloud representation, we first sample 10000 points from shapes and 3D sketches. For shapes we use Monte-Carlo sampling[4] and for sketches we use equidistant sampling. We then adopt two types of sampling from the initial 10000 points to obtain a sparse set of 1024 points: random sampling or uniform sampling. The uniform sampling is obtained with the farthest point sampling. The sparse sets are obtained on-the-fly and thus might differ from one iteration to another. Figure 6 shows the visualization of random and uniform sampling.

5.2. Effect of sampling and rendering.

We compare different sampling methods for PointNet++ and rendering styles for NGVNN, described in Section 5.1, when training with the classification loss and the triplet loss on the pairs of 3D shapes and 3D sketches rendered with l_a set to 0.5. As shown in Figure 8, the uniform sampling for PointNet++ and depth rendering for NGVNN perform best, so we use these settings for the rest of the experiments.

It can be seen from Figure 8 that point-based representation (dashed lines) by far outperforms the multi-view representation (solid lines) when we use the siamese architecture

[4]https://github.com/fwilliams/point-cloud-utils

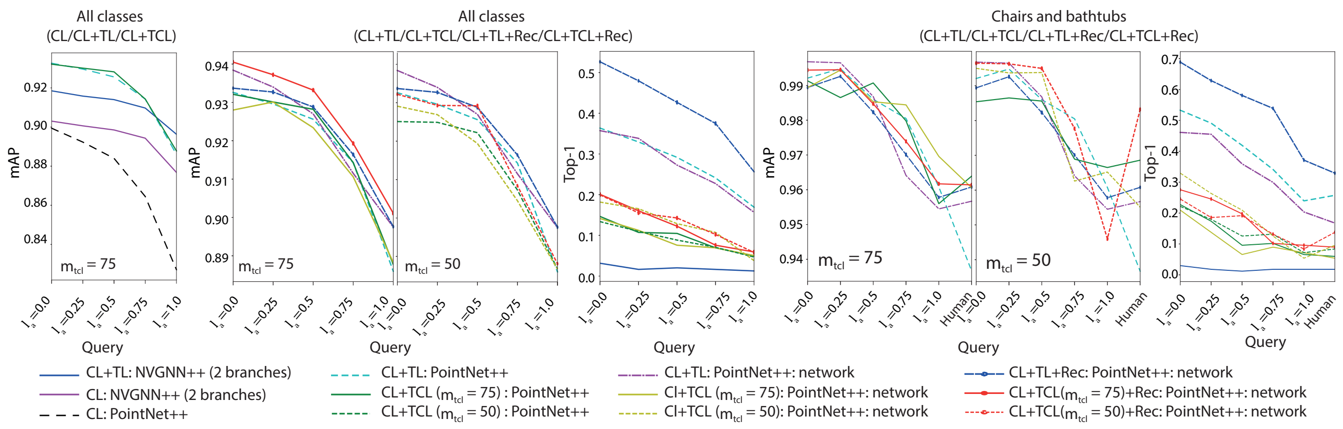

Figure 7: mAP values and Top-1 scores using models trained with the dataset of 3D sketches with $l_a = 0.5$. When we train PointNet++ with $m_{tcl} = 75$, the value that optimizes the validations synthetic dataset, then CL+TCL+Rec gives the highest mAP over all the baselines on all synthetic datasets, but has slightly lower mAP on human dataset than the network trained with CL+TCL. When $m_{tcl} = 50$, the value that optimizes the mAP on a human dataset of the network trained with CL+TCL, the reconstruction loss (Rec) boosts the mAP on human dataset from 0.969 to 0.983. ':network' means that the network, obtained as described in Section 3.3.2, is contributing to the triplets constriction.

for NGVNN. We thus for the rest of experiments use the heterogeneous architecture for NGVNN.

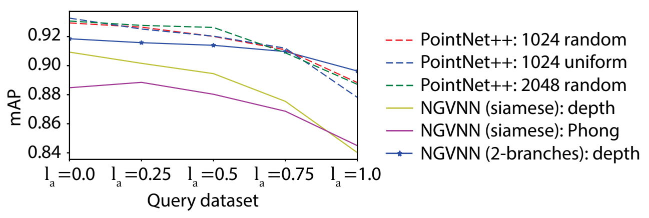

Figure 8: Comparison of sampling strategies and rendering styles, evaluated on synthetic datasets generated with different levels of abstractness l_a. All the baselines in this figure were trained using a combination: CL + TL.

5.3. Triplet loss vs triplet center loss

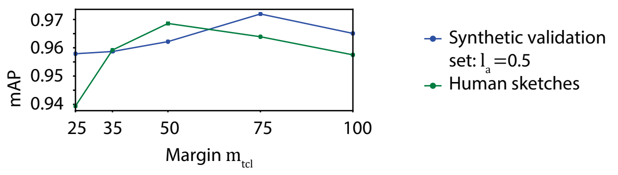

Figure 9: Evaluation of the performance of PointNet++, trained with CL+TCL, as a function of the different values of m_{tcl}, see Equation 3.

The performance of the triplet (TL) and triplet center losses (TCL) is highly dependent on the selected values of the margin. For the triplet loss we select the margin m_{tl} value as described in Section 4.2. To select the optimal value of m_{tcl}, we train PointNet++ with CL and TCL on the synthetic dataset with $l_a = 0.5$ and different values of m_{tcl} (Figure 9). Different values optimize the mAP on the synthetic, $m_{tcl} = 75$, and human, $m_{tcl} = 50$, test data. Figure 7 shows that the mAP of TL and TCL on our problem is comparable, while the Top-1 accuracy is much higher for the TL, showing that TL learns better inner-class variance and is more suitable for a fine-grained retrieval.

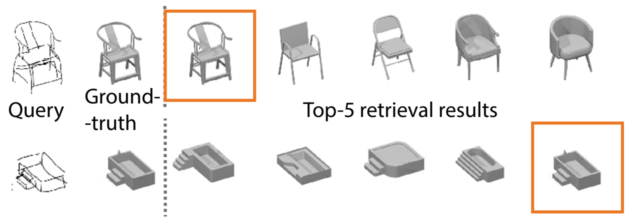

Figure 10: Example retrieval results with TL+Rec, the shapes matching the ground-truth are marked with orange boxes. Please see the supplemental for more examples.

5.4. Regularization branch

The reconstruction branch improves the mAp values for the TCL, while the mAP values for TL are similar with and without the reconstruction branch, both resulting in top performance. In Top-1 scores the regression branch significantly boosts the performance of the TL loss, resulting in the top performance on all synthetic and human datasets (Figure 7). Consistent with the observation from the previous section, TCL optimizes the inter-class variance, which is further improved with the reconstruction loss; TL optimizes the inner-class variance and strongly benefits from our reconstruction branch (Table 1).Some visual results of the retrieval are provided in Figure 10.

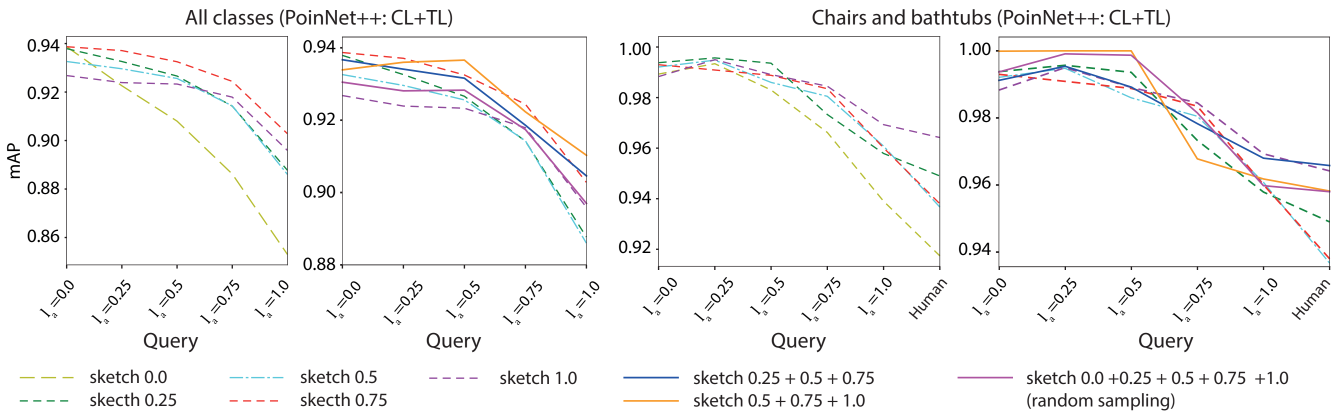

Figure 11: Comparison of the training datasets (Section 5.5). Each line corresponds to a different training dataset. For the dataset which consists of all abstractness levels for each shape we randomly selected one of the abstractness levels.

Model	mAP	NDCG	NN	Top-1
CL+TL: 3D→3D NGVNN (2-branch)	0.934	0.966	0.934	0.018
CL+TL: PointNet++	0.937	0.968	0.946	0.258
CL+TL+Rec: PointNet++	0.961	0.979	0.964	**0.329**
CL+TCL: PointNet++ (m_{tcl}=75)	0.964	0.979	0.964	0.060
CL+TCL+Rec: PointNet++ (m_{tcl}=75)	0.961	0.978	0.958	0.090
CL+TCL: PointNet++ (m_{tcl}=50)	0.969	0.981	0.964	0.084
CL+TCL+Rec: PointNet++ (m_{tcl}=50)	**0.983**	**0.990**	**0.982**	0.138

Table 1: Comparison of all the models on human sketches.

5.5. Study of data abstraction level

In this section, we use different levels of abstraction to train the PointNet++ with the combination of the classification and triplet losses (Figure 11). It can be seen that training on the cleanest sketches ($l_a = 0.0$), yellow dashed lines, generalizes the worst to the other abstraction levels and human sketches, while training with the most abstract sketches ($l_a = 1.0$) results in a stable performance, though lower than the optimal for each of abstractness levels. It gives nearly optimal performance on human sketches. Interestingly, the performance on humans sketches when training on sketches with $l_a = 0.25$ outperforms the training on sketches with $l_a = 0.5$ and $l_a = 0.75$, what might indicate that human participants either did detailed and clean sketches or highly abstract ones. Drawing more certain conclusions would though require a larger set of human sketches, which includes more participants.

5.6. Ablation study: 2D sketch vs 3D sketch

We additionally compare the performance of the 3D sketch retrieval models to a performance of a 2D sketch retrieval model. We use NGVNN as a backbone to build the 2D-sketch based shape retrieval baseline (2D→3D NGVNN), where we use NGVNN to encode 3D shape and additional branch to encode a 2D sketch, which is a common practice for 2D sketch/image based retrieval [13]. The 2D branch uses the same VGG11 backbone as NGVNN. The input for the 2D sketch branch is the view rendered with the camera azimuth angle equal to 45°. It can be seen in Figure 12 that the 2D sketch-based retrieval is outperformed by all the models for 3D sketch-based retrieval, advocating for the 3D sketch-based retrieval.

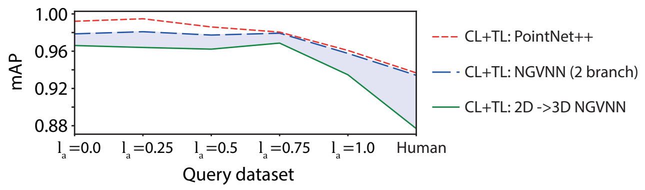

Figure 12: Comparison of the 2D sketch-based retrieval versus 3D sketch-based retrieval (Section 5.6). The baselines are trained on the dataset with $l_a = 0.5$.

6. Conclusion

In this work we proposed the problem of 3D VR-sketch to 3D model retrieval. We first introduced a purpose-built VR environment to collect VR-sketches and conduct retrieval. We then collected a set of 3D human sketches for a subset of two shape classes from ModelNet10. We further proposed an approach for generating synthetic 3D sketches with different levels of abstraction, and demonstrated that the methods trained on our synthetic data generalize well to human sketches. Via a series of comprehensive evaluations, we find that compared to 3D shape-based retrieval, point-based shape representation is advantageous over the multi-view representation. At last, we propose an architecture with an additional sketch regularizing branch that leads to a superior performance over all the considered baselines, demonstrating the benefit of directly tackling the abstract nature of VR-sketches. We hope to have offered a valid first stab at this new problem.

References

[1] A. Avetisyan, A. Dai, and M. Nießner. End-to-end cad model retrieval and 9dof alignment in 3d scans. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2551–2560, 2019. 1

[2] P. Bénard and A. Hertzmann. Line drawings from 3d models: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 11(1-2):1–159, 2019. 2

[3] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016. 2

[4] L. Chu, H. Pan, Y. Liu, and W. Wang. Repairing man-made meshes via visual driven global optimization with minimum intrusion. *ACM Trans. Graph. (SIGGRAPH ASIA)*, 38(6):158:1–158:18, 2019. 3

[5] M. Dahnert, A. Dai, L. J. Guibas, and M. Nießner. Joint embedding of 3d scan and cad objects. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8749–8758, 2019. 1

[6] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 6

[7] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 264–272, 2018. 1, 2, 5

[8] D. Giunchi, S. James, and A. Steed. 3d sketching for interactive model retrieval in virtual reality. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*, pages 1–12, 2018. 2

[9] G. Gori, A. Sheffer, N. Vining, E. Rosales, N. Carr, and T. Ju. Flowrep: Descriptive curve networks for free-form design shapes. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017. 3, 4

[10] A. Grabner, P. M. Roth, and V. Lepetit. 3d pose estimation and 3d model retrieval for objects in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2018. 1, 2

[11] A. Grabner, P. M. Roth, and V. Lepetit. Location field descriptors: Single image 3d model retrieval in the wild. In *2019 International Conference on 3D Vision (3DV)*, pages 583–593. IEEE, 2019. 1, 2

[12] D. Ha and D. Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations (ICLR)*, 2018. 3

[13] X. He, T. Huang, S. Bai, and X. Bai. View n-gram network for 3d object retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7515–7524, 2019. 1, 2, 5, 6, 8

[14] X. He, Y. Zhou, Z. Zhou, S. Bai, and X. Bai. Triplet-center loss for multi-view 3d object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1945–1954, 2018. 2, 5, 6

[15] T. Lee, Y.-L. Lin, H. Chiang, M.-W. Chiu, W. Hsu, and P. Huang. Cross-domain image-based 3d shape retrieval by view sequence learning. In *2018 International Conference on 3D Vision (3DV)*, pages 258–266. IEEE, 2018. 1, 2, 6

[16] B. Li, Y. Lu, F. Duan, S. Dong, Y. Fan, L. Qian, H. Laga, H. Li, Y. Li, P. Lui, et al. Shrec'16 track: 3d sketch-based 3d shape retrieval. 2016. 2

[17] B. Li, Y. Lu, A. Ghumman, B. Strylowski, M. Gutierrez, S. Sadiq, S. Forster, N. Feola, and T. Bugerin. 3d sketch-based 3d model retrieval. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 555–558, 2015. 2

[18] Z. Li, C. Xu, and B. Leng. Angular triplet-center loss for multi-view 3d shape retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8682–8689, 2019. 5

[19] K. Z. Lin, W. Xu, Q. Sun, C. Theobalt, and T.-S. Chua. Learning a disentangled embedding for monocular 3d shape retrieval and pose estimation. *arXiv preprint arXiv:1812.09899*, 2018. 2

[20] A. Liu, S. Xiang, W. Li, W. Nie, and Y. Su. Cross-domain 3d model retrieval via visual domain adaptation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 828–834. AAAI Press, 2018. 2

[21] M. Lyon, M. Campen, D. Bommes, and L. Kobbelt. Parametrization quantization with free boundaries for trimmed quad meshing. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 3

[22] Y. Ma, B. Zheng, Y. Guo, Y. Lei, and J. Zhang. Boosting multi-view convolutional neural networks for 3d object recognition via view saliency. In *Chinese Conference on Image and Graphics Technologies*, pages 199–209. Springer, 2017. 1, 2

[23] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 2

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 1, 2

[25] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016. 1, 2

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 1, 2, 5, 6

[27] Y. Qi, Y.-Z. Song, H. Zhang, and J. Liu. Sketch-based image retrieval via siamese convolutional neural network. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2460–2464. IEEE, 2016. 2

[28] N.-p. Ramer-Douglas-Peucker. Ramer-douglas-peucker algorithm. 1972. 4

[29] S. Ravanbakhsh, J. Schneider, and B. Poczos. Deep learning with sets and point clouds. *arXiv preprint arXiv:1611.04500*, 2016. 2

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[31] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 1, 2, 5

[32] M. A. Uy, J. Huang, M. Sung, T. Birdal, and L. Guibas. Deformation-aware 3d model embedding and retrieval. *arXiv preprint arXiv:2004.01228*, 2020. 1, 2

[33] F. Wang, L. Kang, and Y. Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1875–1883, 2015. 1

[34] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014. 2, 5

[35] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016. 2

[36] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 2

[37] Y. Ye, B. Li, and Y. Lu. 3d sketch-based 3d model retrieval with convolutional neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2936–2941. IEEE, 2016. 2

[38] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C.-C. Loy. Sketch me that shoe. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 799–807, 2016. 2