Deformation-Aware 3D Model Embedding and Retrieval

Mikaela Angelina Uy[1], Jingwei Huang[1], Minhyuk Sung[2], Tolga Birdal[1], and Leonidas Guibas[1]

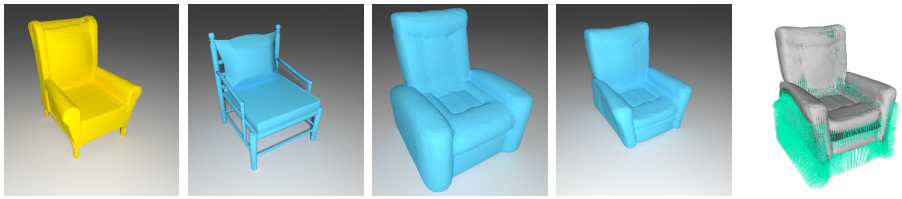
[1] Stanford University [2] Adobe Research

Abstract. We introduce a new problem of *retrieving* 3D models that are *deformable* to a given query shape and present a novel deep *deformation-aware* embedding to solve this retrieval task. 3D model retrieval is a fundamental operation for recovering a clean and complete 3D model from a noisy and partial 3D scan. However, given a finite collection of 3D shapes, even the closest model to a query may not be satisfactory. This motivates us to apply 3D model deformation techniques to adapt the retrieved model so as to better fit the query. Yet, certain restrictions are enforced in most 3D deformation techniques to preserve important features of the original model that prevent a perfect fitting of the deformed model to the query. This gap between the deformed model and the query induces *asymmetric* relationships among the models, which cannot be handled by typical metric learning techniques. Thus, to retrieve the best models for fitting, we propose a novel deep embedding approach that learns the asymmetric relationships by leveraging location-dependent egocentric distance fields. We also propose two strategies for training the embedding network. We demonstrate that both of these approaches outperform other baselines in our experiments with both synthetic and real data. Our project page can be found at deformscan2cad.github.io.

Keywords: 3D Model Retrieval, Deformation-Aware Embedding, Non-Metric Embedding.

1 Introduction

A fundamental task in 3D perception is the 3D reconstruction, where the shape and appearance of a real object are captured into digital form through a scanning process. The result of 3D scanning is usually imperfect, due to sensor noise, outliers, motion blur, and scanning pattern artifacts. Despite the advances in robust techniques for fusing scans [47,9,65,17,16,30], the quality of the produced 3D shapes can be far from what is desired. Recently, we are witnessing growing efforts to replace the observed noisy, cluttered and partial scans with clean geometry, such as artist-created CAD models [39,6,5,15]. In this way, eventually, an entire scene can be virtualized into a set of 3D models that are free of noise, partiality, and scanning artifacts – while maintaining the semantically valid structure and realistic appearance. One straightforward way to achieve this goal



(a) Query (b) Closest (c) Retrieved (d) Deformed (e) Deform. Flow

Fig. 1. Example of deformation-aware 3D model retrieval. Given a query (a), the closest 3D model in terms of Chamfer distance has distinct geometric and semantic differences. The model retrieved with our framework (c) better fits the query *after* deformation (d). The deformation flow is visualized in (e).

is to replace the sensor data by a known CAD model *retrieved* from an existing repository.

Unfortunately, such retrieval is only viable when there is an almost exact match between the scan and the model. Given the tremendous variety of real 3D shapes, it is implausible to expect that a CAD model in the repository can exactly match the input or the user's desire – even with the recent advent of large-scale 3D repositories [10,61,62,58]. The closest shape in the database from the query might still have subtle but semantically important geometric or structural differences, leading to an undesirable gap in various settings (e.g., the difference in global structure in Fig. 1 (b)). To reduce such differences, we propose to retrieve a CAD model (Fig. 1 (c)) with similar structure to the query, so that we can apply a deformation operation to fit the query (Fig. 1 (d)) better than the closest shape (Fig. 1 (b)). One challenge is to efficiently retrieve such a CAD model especially given that the deformation requires significant time to compute. In light of this, we propose an efficient *deformation-aware* 3D model retrieval framework that finds a 3D model best matching the input *after* a deformation. Such an approach of joint retrieval and fitting can help more closely reconstruct the target with the same initial pool of 3D models (Fig. 1 (d)) while maintaining retrieval efficiency.

A key issue in this deformation-aware retrieval is in dealing with the *scope* of the deformation of each 3D model. Since the goal of 3D model retrieval is to take advantage of the high fidelity and fine structure of shape representation of man-made models, it is desired to maintain such beneficial properties in the deformation. The long literature of 3D shape deformation has also stemmed from this preservation intent and has investigated diverse ways of constraining or regularizing the deformation; for example, making a smooth function of the deformation with a coarse set of control points [51,36,35,34,41,64] or having per-edge or per-face regularization functions preserving local geometric details, given a mesh representation of a model [57,40,42,32,56]. Such constraints/regularizations aim to ensure production of *plausible* variations without losing the original 3D model's features – although they simultaneously confine the scope of deformation and prevent it from exactly matching the target. Thus, given a function deforming a *source* model to match the *target* under appropriate regularizations, we consider

the notion of the *fitting gap*, defined as the difference between the *deformed* source model and the target.

We introduce a novel deep embedding technique that maps a collection of 3D models into a latent space based on the fitting gap as characterized by a given deformation function. Our embedding technique is agnostic to the exact nature of the deformation function and only requires values of the fitting gap for sampled pairs of the source and target models in training. Due to the *asymmetric* nature of the fitting gap and the lack of a *triangle inequality*, the embedding cannot be accomplished with typical metric learning techniques [26,11,50]. Hence, we propose a novel approach, learning a location-dependent *egocentric* anisotropic distance field from the fitting gaps and suggest two network training strategies : one based on margin loss and the other based on regression loss. In test time, given a query shape, the retrieval can be performed by computing the egocentric distance from all 3D models in the database and finding the one that gives the smallest distance.

In our experiments with ShapeNet [10] dataset, we demonstrate that our framework outperforms all the other baseline methods and also that the second regression-based training strategy provides consistently better performance across different categories of the shapes. We also test our framework with queries of 3D scans and images. In the case of real 3D scans, our outputs show even a smaller average fitting gap when compared with human selected 3D models.

In summary, our contributions are:

- defining a new task, that of retrieving a 3D CAD model in a *deformation-aware* fashion;
- introducing a novel *asymmetric* distance notion called *fitting gap*, measuring shape difference after deforming one model toward the other;
- formulating an *egocentric anisotropic distance field* on a latent embedding space so as to respect the asymmetry of the fitting gap;
- proposing two deep network training strategies to learn the said embedding;
- demonstrating that our framework outperforms baselines in the experiments with ShapeNet and presenting results for 3D object reconstruction in a real scan-to-CAD application as well as an image-to-CAD scenario.

2 Related Work

3D Model Deformation. 3D model deformation has been a decades-long problem in geometry. Given a shape represented with a mesh, the problem is defined as finding the best positions of vertices in a way that the new shape fits a target while preserving local geometric details.

Previous work has introduced various ways of formulating the regularization conserving the local geometric details, which are mainly classified into three categories. The first is so-called *free-form* [51,36] approaches. These methods use the voxel grids of the volume enclosing the surface as control points and define a smooth deformation function interpolating weights from the control points to the mesh vertices. The second are *cage-based* approaches [35,34,41,64], which

take the control points not from voxel grids but from a coarse scaffold mesh surrounding the input. The last is *vertex-based* approaches [57,40,42,32,56]. In these methods, the objective function for the optimization is directly defined with the mesh vertex positions, which describe geometric properties that should be preserved, such as mesh Laplacian [57,40] or local rigidity [42,32,56].

Recently, neural networks also have been applied to these three (free-form [69,28,33,38], cage-based [68], and vertex-based [63,24]) approaches of the 3D shape deformation. The purposes of leveraging neural networks in the deformation vary, including: better handling partiality in the target [28], finding per-point correspondences in an unsupervised way [24], enabling taking data in other modalities (e.g., color images or depth scans) as input [63,33,38], correlating shape variations with semantic meanings [69], and deformation transfer [68].

In this work, we propose a deformation-aware retrieval technique that can employ any of the deformation methods introduced above as a *given* function for generating plausible deformations. We assume that, based on the regularization of preserving geometric properties, the given deformation function *guarantees* the plausibility of the deformed 3D model while minimizing the fitting distance.

Retrieval via Deep Embedding. With deep embedding, retrieval problems have been formulated in diverse ways depending on their characteristics.

A significant progress has been made on learning *similarity metrics*, after Chopra *et al*. [13] and Hadsell *et al*. [26] introduced pioneering work for Siamese network architecture and contrastive loss. Given positive and negative samples of the query, the contrastive loss is defined as pulling and pushing the positive and negative samples in the embedding, respectively. While the contrastive loss is often defined with two separate losses for each of them [52,60], in the retrieval, considering *relative* distances can engage more flexibility in the embedding function. Thus, later work has more exploited margin losses [27,21], coupling positive and negative samples and pushing the distance between them to be greater than a threshold. Researchers have also verified that the performance can be improved with a better strategy of triplet sampling, such as hard negative mining [50,52] that takes the farthest positive and the closest negative samples at each time. In Sec. 3.2, we introduce an embedding approach incorporating techniques above, although our problem is fundamentally different from the metric learning due to *asymmetry*. Thus, we focus on dealing with the asymmetry.

Another direction is *graph embedding*, which is more general in terms of handling asymmetric relationships (when considering *directed* graphs). The basic goal of the graph embedding is to represent adjacencies among nodes in the graph with similarity in the embedding space. Thus, it can be formulated as regressing the existence or weight of edges [2,66]. However, recent work focuses more on learning high-order proximity, with the assumption that *neighbors of neighbors are neighbors*, and leverages ideas of random walk in the embedding [48,25,19]. This *transitivity* assumption, however, is *not* guaranteed to hold in our problem. In Sec. 3.3, we introduce our second embedding approach following the idea of similarity regression but without exploiting the random walk procedure.

Although metric learning has been previously adapted for 3D point sets [18], it is shown that non-metric learning is able to generate a more complex, accurate and perceptually meaningful similarity model [22,59]. While similarity search on non-metric spaces is widespread in the classical retrieval systems [54,12,55,53,46], simultaneous learning and non-metric embedding of deep features is still an open problem. In this paper, we address this gap for the particular problem of deformation-aware 3D shape retrieval.

3 Deformation-Aware Embedding

We propose an efficient deformation-aware retrieval framework which retrieves a 3D model from the database that can best match the query shape through deformation — in the context of the deformation, we will also use the terms *source* and *target* for the *database* and *query* shapes, respectively. For the framework, we develop a deep embedding technique that maps a given collection of 3D models $\mathbf{X}$ into a latent space based on a given notion of *distance after deformation*. While in principle any shape can be deformed to any other shape under the same topology, such notion of *fitting gap* emerges from the consideration of constraints or regularizations in the deformation. A 3D model, which can be easily converted into a mesh, typically has delicate geometric structure that faithfully describes sharp edges and smooth surfaces. Thus, in the deformation of meshes, previous research has paid attention to preserve the fine geometric structure and proposed a variety of techniques regularizing the deformation – in ways to maintain mesh Laplacian [57,40], local rigidity [42,32,56], and surface smoothness [51,36]. Such regularizations, however, obviously may limit the scope of deformation of each 3D model, meaning that a model may not exactly reach the other target shape via deformation. Thus, given a pair of the source and target models $\mathbf{s}, \mathbf{t} \in \mathbf{X}$ and a deformation function $\mathcal{D} : (\mathbf{X} \times \mathbf{X}) \rightarrow \mathbf{X}$ warping the source shape $\mathbf{s}$ to best match the target shape $\mathbf{t}$ under the regularizations, we define the fitting gap $e_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ from $\mathbf{s}$ to $\mathbf{t}$ as how much the deformed source shape $\mathcal{D}(\mathbf{s}; \mathbf{t})$ *deviates* from the target shape $\mathbf{t}$:

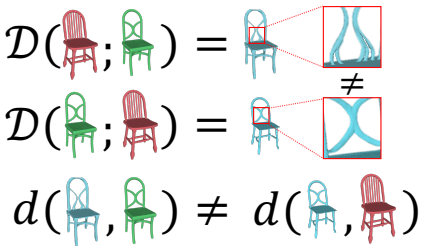


Fig. 2. *Fitting gap is asymmetric.* The four bars of the red chair can deform close to the two bars of the green chair, achieving a small fitting gap. However, it is harder to deform the green chair into the red chair as we cannot split two bars into four, hence resulting in a larger fitting gap.

$$e_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) = d(\mathcal{D}(\mathbf{s}; \mathbf{t}), \mathbf{t}), \tag{1}$$

where $d : (\mathbf{X} \times \mathbf{X}) \rightarrow [0, \infty)$ is a function measuring the difference between two 3D models. In other words, the fitting gap is *shape difference after the source deformation* (0 means perfect fitting). Considering the given deformation function $\mathcal{D}$ as a black-box, our goal of the embedding is to build a latent space reflecting

the fitting gap characterized by the deformation function so that given a query (target) $\mathbf{t}$, the 3D model $\hat{\mathbf{s}} \in \mathbf{X} \setminus \{\mathbf{t}\}$ that gives the smallest fitting gap $e_{\mathcal{D}}(\hat{\mathbf{s}}, \mathbf{t})$ can be retrieved for downstream applications.

Note that such definition of the fitting gap does not guarantee *symmetry* given arbitrary deformation function $\mathcal{D}$: $\exists\, \mathbf{s}, \mathbf{t} \in \mathbf{X}$ s.t. $e_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) \neq e_{\mathcal{D}}(\mathbf{t}, \mathbf{s})$; a counterexample can be found as shown in Fig. 2. Moreover, any notion of transitivity such as directional triangular inequality ($e_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) + e_{\mathcal{D}}(\mathbf{t}, \mathbf{u}) \leq e_{\mathcal{D}}(\mathbf{s}, \mathbf{u})$) is not guaranteed. For both reasons, the fitting gap is not a *metric*. The only properties of metrics that are satisfied with the fitting gap are the following two:

1. (Non-negativity) $e_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) \geq 0$ for every $\mathbf{s}, \mathbf{t} \in \mathbf{X}$.
2. (Identity) $e_{\mathcal{D}}(\mathbf{t}, \mathbf{t}) = 0$ for every $\mathbf{t} \in \mathbf{X}$. [1]

Non-negativity holds since d in Eq (1) is a distance function. For identity, we assume that the given deformation function $\mathcal{D}$ satisfies $\mathcal{D}(\mathbf{t}, \mathbf{t}) = \mathbf{t}$ (making no change when the source and target are the same), and thus $e_{\mathcal{D}}(\mathbf{t}, \mathbf{t}) = d(\mathcal{D}(\mathbf{t}), \mathbf{t}) = d(\mathbf{t}, \mathbf{t}) = 0$. A family of such bivariate functions is often called pseudosemimetrics [8] or premetrics [3]. Embedding based on such a notion has been underexplored.

Next, we illustrate how we encode the fitting gap among 3D models on a latent embedding space (Sec. 3.1) and then propose two strategies of training our embedding network (Sec. 3.2 and 3.3).

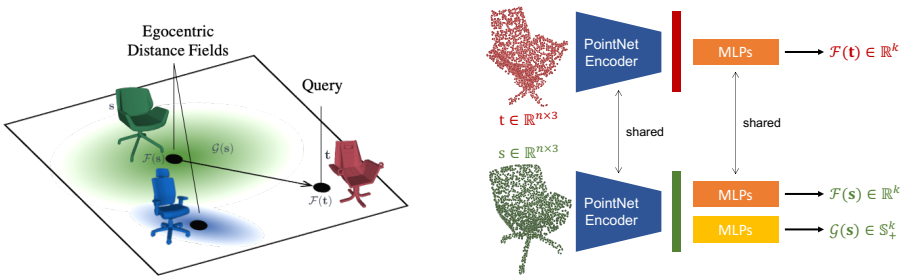
3.1 Embedding with Egocentric Distances

Consider an embedding network $\mathcal{F} : \mathbf{X} \to \mathbb{R}^k$ that maps each 3D model in $\mathbf{X}$ to a point in a k-dimensional latent space. The key in our embedding is to allow the network to properly encode *asymmetric* relationships among 3D models described with the fitting gap while satisfying the properties including non-negativity and identity. Given this, in addition to mapping a 3D model $\mathbf{s} \in \mathbf{X}$ to a point in the embedding space, we propose another network $\mathcal{G} : \mathbf{X} \to \mathbb{S}^k_+$ that predicts an *egocentric* anisotropic distance field for each 3D model, represented with a $k \times k$ positive-semidefinite (PSD) matrix. Analogous to Mahalanobis distance [45], we define the egocentric distance function $\delta : \mathbf{X} \times \mathbf{X} \to [0, \infty)$ given the target and *observer* 3D models $\mathbf{t}, \mathbf{s} \in \mathbf{X}$ as follows:

$$\delta(\mathbf{t}; \mathbf{s}) = \sqrt{(\mathcal{F}(\mathbf{t}) - \mathcal{F}(\mathbf{s}))^T \, \mathcal{G}(\mathbf{s}) \, (\mathcal{F}(\mathbf{t}) - \mathcal{F}(\mathbf{s}))}. \tag{2}$$

Although it is a common practice to employ Mahalanobis distance in metric learning [43,7,37], we do *not* learn a metric. Hence, we propose to vary the PSD matrix (the inverse covariance matrix in the Mahalanobis distance) depending on the *observer* shape so that it can characterize the fitting gap of the observer

[1] This is *not* exactly the same with the property of metrics, *identity of indiscernibles*, meaning the two-way identity ($e_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) = 0 \Leftrightarrow \mathbf{s} = \mathbf{t}$). We cannot guarantee that $e_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) = 0 \Rightarrow \mathbf{s} = \mathbf{t}$ from our definition of $e_{\mathcal{D}}$. Nevertheless, this is not necessary in the retrieval problem.



(a) (b)

Fig. 3. (a) Visual illustration of our embedding space and egocentric distance. **(b)** Our Siamese network architecture. The PointNet [49] encoder branches to two MLPs: the embedding network $\mathcal{F}$ and the egocentric distance field network $\mathcal{G}$. Both the embedding vector $\mathcal{F}(\mathbf{s})$ and the distance field $\mathcal{G}(\mathbf{s})$ are predicted for the source $\mathbf{s}$, while only the embedding vector $\mathcal{F}(\mathbf{t})$ is predicted for the target $\mathbf{t}$. These are used to calculate for their asymmetric fitting gap.

shape over the latent space. We remark that, in retrieval, each model in the database that can be *retrieved* becomes an *observer* when computing the distance from the query to the model since we deform the retrieved 3D model to fit the query (see Fig. 3(a)). Also, note that the function δ satisfies non-negativity (since $\mathcal{G}(\mathbf{s}) \succeq 0$) and identity ($\forall \mathbf{s},\ \delta(\mathbf{s};\mathbf{s}) = 0$).

When considering the goal of retrieval (Sec. 3), our desire is to learn a egocentric distance function δ that satisfies for every t that

$$\underset{\mathbf{s}\in\mathbf{X}\setminus\{\mathbf{t}\}}{\operatorname{argmin}}\, e_{\mathcal{D}}(\mathbf{s},\mathbf{t}) = \underset{\mathbf{s}\in\mathbf{X}\setminus\{\mathbf{t}\}}{\operatorname{argmin}}\, \delta(\mathbf{t};\mathbf{s}). \tag{3}$$

Since it is practically impossible to compute the deformation function $\mathcal{D}$ for all ordered pairs of 3D models in $\mathbf{X}$ due to the intensive computation time, we leverage the inductive bias of neural networks generalizing the prediction to unseen data points. Thus, in network training, we select a fixed size subset of models $\mathbf{X_t} \setminus \{\mathbf{t}\} \subset \mathbf{X}$ for every model $\mathbf{t} \in \mathbf{X}$ and only use the set of source-target pairs $\{(\mathbf{s},\mathbf{t}) \,|\, \mathbf{s} \in \mathbf{X_t}, \forall \mathbf{t} \in \mathbf{X}\}$ in the training while precomputing the fitting gap. In the following subsections, we introduce two training strategies with different loss functions. The difference between these two strategies is also analyzed with experiments in Sec. 5.

3.2 Margin-Loss-Based Approach

We first propose our margin-loss-based approach, inspired by previous weakly supervised learning work [4]. We leverage on having a notion of positive (deformable) and negative (not deformable) candidates for each query shape. For a query (target) shape $\mathbf{t}$, we define a positive set $\mathbf{P_t} = \{\mathbf{s} \in \mathbf{X_t} \,|\, e_{\mathcal{D}}(\mathbf{s},\mathbf{t}) \leq \sigma_{\mathbf{P}}\}$ and a negative set $\mathbf{N_t} = \{\mathbf{s} \in \mathbf{X_t} \,|\, e_{\mathcal{D}}(\mathbf{s},\mathbf{t}) > \sigma_{\mathbf{N}}\}$ of the 3D models based on the

thresholds $\sigma_{\mathbf{P}}$ and $\sigma_{\mathbf{N}}$ ($\sigma_{\mathbf{P}} < \sigma_{\mathbf{N}}$). In training, we sample triplets $(\mathbf{t}, \mathbf{P}'_{\mathbf{t}}, \mathbf{N}'_{\mathbf{t}})$ by taking random subsets $\mathbf{P}'_{\mathbf{t}} \subset \mathbf{P}_{\mathbf{t}}$ and $\mathbf{N}'_{\mathbf{t}} \subset \mathbf{N}_{\mathbf{t}}$ and define the loss as follows:

$$\mathcal{L}_M\left(\mathbf{t}, \mathbf{P}'_{\mathbf{t}}, \mathbf{N}'_{\mathbf{t}}\right) = \frac{1}{\mathbf{N}'_{\mathbf{t}}} \sum_{\mathbf{n} \in \mathbf{N}'_{\mathbf{t}}} \left[\max_{\mathbf{p} \in \mathbf{P}'_{\mathbf{t}}} \left(\delta\left(\mathbf{t}; \mathbf{p}\right)\right) - \delta\left(\mathbf{t}; \mathbf{n}\right) + m\right]_+, \tag{4}$$

where $[\ldots]_+$ denotes the hinge loss [14] and m is a margin parameter. This is in contrast to the loss of Arandjelovi *et al.* [4] where the best/closest positive is taken to handle false positives. The intuition for our loss is that the distance from the query to the furthest positive candidate should always be pulled closer than any of the negative candidates.

3.3 Regression-Based Approach

We also propose another training strategy that uses a regression loss instead of defining the positive and negative sets. Since we only need to learn *relative* scales of the fitting gap $e_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ for each query $\mathbf{t}$ in retrieval, inspired by Stochastic Neighbor Embedding (SNE) [29], we first convert the fitting gap into a form of *probability* as follows:

$$p(\mathbf{s}; \mathbf{t}) = \frac{\exp\left(-e^2_{\mathcal{D}}(\mathbf{s}, \mathbf{t})/2\sigma^2_{\mathbf{t}}\right)}{\sum_{\mathbf{s} \in \mathbf{X}'_{\mathbf{t}}} \exp\left(-e^2_{\mathcal{D}}(\mathbf{s}, \mathbf{t})/2\sigma^2_{\mathbf{t}}\right)}, \tag{5}$$

where $\mathbf{X}'_{\mathbf{t}} \subset \mathbf{X}_{\mathbf{t}}$ is a randomly sampled fixed size subset and $\sigma_{\mathbf{t}}$ is a pre-computed constant for each shape $\mathbf{t}$, which is determined in a way to satisfy the following condition (which is based on Shannon entropy) [44]:

$$\log_2 \tau = -\sum_{\mathbf{s} \in \mathbf{X}'_{\mathbf{t}}} p(\mathbf{s}; \mathbf{t}) \log_2(p(\mathbf{s}; \mathbf{t})), \tag{6}$$

where τ is a perplexity parameter determining the extent of the neighborhood. Note that we regress the probabilities $p(\mathbf{s}; \mathbf{t})$ since we do not have access to the entire distribution of $p(\cdot; \mathbf{t})$ but only to the models in the subset $\mathbf{X}'_{\mathbf{t}}$ for each $\mathbf{t}$. This is contrast to SNE which seeks to fully match the given and predicted distributions. We similarly convert the learned asymmetric distance $\delta(\mathbf{t}; \mathbf{s})$ into a form of probability:

$$\hat{p}(\mathbf{s}; \mathbf{t}) = \frac{\delta^2(\mathbf{t}; \mathbf{s})}{\sum_{\mathbf{s} \in \mathbf{X}'_{\mathbf{t}}} \delta^2(\mathbf{t}; \mathbf{s})}. \tag{7}$$

The following $l1$-distance is finally defined as regression loss:

$$\mathcal{L}_R(\mathbf{t}, \mathbf{X}'_{\mathbf{t}}) = \frac{1}{\mathbf{X}'_{\mathbf{t}}} \sum_{\mathbf{s} \in \mathbf{X}'_{\mathbf{t}}} |\hat{p}(\mathbf{s}; \mathbf{t}) - p(\mathbf{s}; \mathbf{t})|. \tag{8}$$

4 Implementation Details

In our implementation, we first convert 3D CAD models into meshes to compute deformation. In order to cope with the multiple connected components of the CAD models in the deformation, we particularly convert the CAD models into *watertight* meshes using the method of Huang *et al.* [31]. We further simplify them with a mesh decimation technique [23] for efficient computation. For the deformation function $\mathcal{D}$, we use a simplified version of ARAP deformation [56] — refer to the supplementary for the details. For computing the distance between shapes and feeding the 3D shape information to the network, we also generate point clouds from the meshes by uniformly sampling $2,048$ points. The distance function $d(\mathbf{x}, \mathbf{y})$ (see Eq (1)) measuring the shape difference between two 3D models $\mathbf{x}, \mathbf{y} \in \mathbf{X}$ is defined as average two-way Chamfer distance (CD) between the point sets resampled on the meshes $\mathbf{x}, \mathbf{y}$ following previous work [20,1,63,68]. We remark that our embedding framework does *not* require any specific type of the deformation function, and in the embedding, the given deformation function is only used to precompute the fitting gap between two shapes in the sampled pairs. See Sec. 2 for more options of the deformation function.

Network Architecture for $\mathcal{F}$ and $\mathcal{G}$ and training details. Fig. 3(b) illustrates our network design. We build a Siamese architecture taking a pair of source `s` and target `t` point clouds with PointNet [49] encoder (the earlier part until the *maxpool* layer) as our shared encoder. The outputs after the *maxpool* then pass through two separate branches of *MLP*; one is $\mathcal{F}$ that predicts the location in the k-dimensional latent embedding space, and the other is $\mathcal{G}$ that predicts the egocentric distance field (the PSD matrix, see Sec. 3.1). In $\mathcal{G}$, we predict a *positive diagonal* matrix as our PSD matrix using a sigmoid activation and adding $\epsilon = 1e^{-6}$. We use $k = 256$ for most of our experiments but also demonstrate the effect of varying the dimension of the latent space in the supplementary material.

In training, we further randomly downsample the point clouds with $2,048$ points to $1,024$ points for memory efficiency (but the entire $2,048$ points are used in baseline methods). We set the minibatch size as 8 for the query $\mathbf{t}$, and $|\mathbf{P}'_\mathbf{t}| = 2$ and $|\mathbf{N}'_\mathbf{t}| = 13$ for the margin-loss-based and $|\mathbf{X}'_\mathbf{t}| = 15$ for the regression-based approaches. We use Adam optimizer with a learning rate of 0.001 and train for 350 epochs for all cases.

Remark that the resolution of the 3D model in retrieval is not affected by the resolution of input point clouds fed into our network. A 3D model in any resolution can be retrieved in their original format.

5 Results

We present our experimental evaluation to demonstrate the advantage of our embedding framework for deformation-aware 3D model retrieval. We also showcase

Table 1. Quantitative results of retrievals. See Sec. 5 for baselines and evaluation metrics. The numbers multiplied by $1e^{-2}$ are reported. Bold is the smallest, and underscore is the second smallest. Our retrieval results give smaller *after*-deformation distances $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ while the *before*-deformation distances $d^m(\mathbf{s}, \mathbf{t})$ are large.

	Method	Table		Chair		Sofa		Car		Plane	
		Top-1	Top-3	Top-1	Top-3	Top-1	Top-3	Top-1	Top-3	Top-1	Top-3
Mean $d^m(\mathbf{s}, \mathbf{t})$	Ranked CD	**4.467**	**3.287**	**4.412**	**3.333**	**3.916**	**2.985**	**2.346**	**1.860**	**2.530**	**1.540**
	AE	4.867	3.491	4.710	3.473	4.223	3.178	2.579	1.942	3.045	1.789
	CD-Margin	4.875	3.449	4.750	3.518	3.087	4.151	2.525	1.905	2.801	1.655
	Ours-Margin	6.227	4.026	5.664	3.889	4.825	3.400	2.962	2.142	3.442	1.885
	Ours-Reg	5.955	3.979	5.751	3.981	5.091	3.628	3.119	2.263	3.436	1.976
Mean $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$	Ranked CD	2.095	1.284	1.937	1.186	1.450	0.886	1.138	0.716	**1.199**	0.569
	AE	2.180	1.292	1.991	1.196	1.521	0.887	1.214	0.753	1.392	0.634
	CD-Margin	2.362	1.373	2.134	1.242	1.587	0.909	1.249	0.773	1.315	0.620
	Ours-Margin	2.127	1.251	1.915	1.144	1.420	0.835	1.226	0.747	1.300	0.586
	Ours-Reg	**1.969**	**1.129**	**1.752**	**1.054**	**1.338**	**0.788**	**1.112**	**0.681**	**1.199**	**0.529**

applications of our approach in two real scenarios: Scan-to-CAD (Sec. 5.2) and Image-to-CAD[2].

Baselines. We compare the proposed margin-loss-based (*Ours-Margin*, Sec. 3.2) and regression-based (*Ours-Reg*, Sec. 3.3) approaches with three retrieval baselines (we also compare with more baselines in the supplementary):

1. Ranked by Chamfer Distance (*Ranked CD*): This retrieves the closest 3D models by Chamfer Distance (CD), which is our distance function d in Sec. 4.
2. Autoencoder (*AE*): This learns an embedding space by training a point cloud autoencoder as defined in [1]. The dimension of the latent space is 1024, which is larger than that of our space.
3. Chamfer Distance Triplet (*CD-Margin*): This baseline is the same with *Ours-Margin* (Sec. 3.2) except for that the distance for the hinge loss is defined as the Euclidean distance over the latent space instead of our egocentric asymmetric distance. The positive and negative candidates are sampled by taking 20 closest models ordered by CD and random 50 models, respectively.

The Chamfer Distance Triplet (*CD-Margin*) is trained in the same way with our margin-loss-based approach (*Ours-Margin*) described in Sec. 3.2; the minibatches are generated with 8 queries and 2 positive and 13 negative random candidates for each of them. We use a margin value $m = 0.5$ for *CD-Margin* and also normalize the latent codes to have a unit $l2$-norm as done in FaceNet [50].

Note that neither of the three baselines above leverage the information about *deformability*, meaning how a 3D model can be deformed to fit the query.

Evaluation Metrics. To avoid sampling bias of the point clouds, in the evaluations, we measure the distance between two shapes as a two-way *point-to-mesh* Chamfer distance; we also use a denser point cloud including $50k$ uniformly sampled points in this case. We denote this new distance function

[2] Due to space restrictions we present results of Image-to-CAD in our supplementary material.

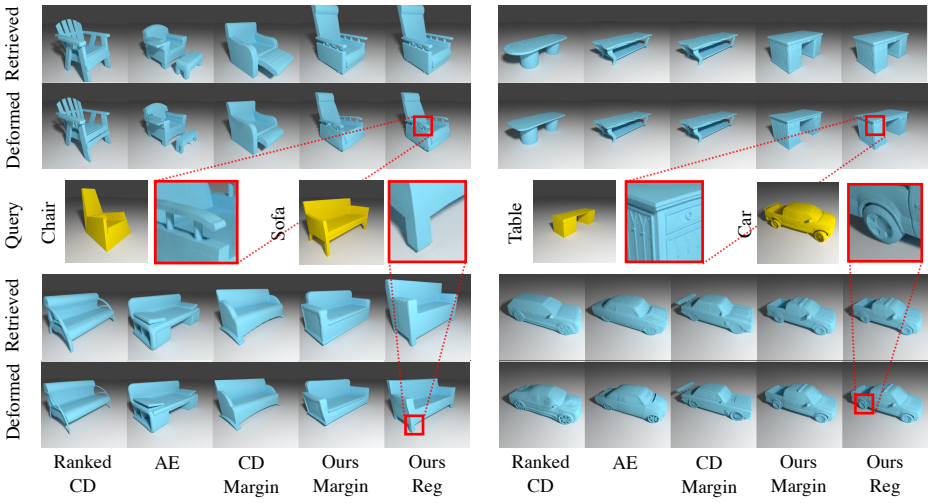


Fig. 4. Visualization of retrieval followed by deformation on ShapeNet. Our network is able to retrieve models that better fit after deformation despite having large geometric distances initially. Notice the big back part of the retrieved chair and the thick seat of the retrieved sofa, attributes that are not identical to the query. Yet, these parts properly fit the target after deformation. Our network is also able to retrieve a sofa with legs and a car with a trunk that are present in the desired targets. Moreover, our deformation-aware *retrieval & deformation* approach also allows us to preserve fine-details of the source model post-deformation as shown in the zoomed in regions. See supplementary for more results.

as $d^m : (\mathbf{X} \times \mathbf{X}) \rightarrow [0, \infty)$ and the accompanying fitting gap function as $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t}) = d^m(\mathcal{D}(\mathbf{s}; \mathbf{t}), \mathbf{t})$. Also, for simplicity, we will use the notations $d^m(\mathbf{s}, \mathbf{t})$ and $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ as the source-target distances *before* and *after* the source deformation in the rest of the paper. We report the mean of these numbers for the best among top-N retrieved models; $N = 1, 3$ are reported. For a partial scan input, we use a *one-way* point-to-mesh distance; see Sec. 5.2 for the details.

5.1 Experiments on ShapeNet [10]

We experiment with four classes in ShapeNet [10] dataset: *Table*, *Chair*, *Sofa* and *Car*. We train/evaluate the networks per class with the training/test splits of Yang *et al*. [67]. In the evaluations, we take all models in the test set as queries and retrieve 3D models from the same test set but except for the query. For our training data, we precompute the fitting gap $e_{\mathcal{D}}$ (Sec. 3). To obtain source-target pairs, we sample 100 source models for every target $\mathbf{t} \in \mathbf{X}$, i.e. $|\mathbf{X_t}| = 100$, which consist of the 50 closest models by the distance d in Sec. 3.1 (not including $\mathbf{t}$ itself) and another 50 random models. We use $\sigma_P = 3.5e^{-4}, 3e^{-4}, 2e^{-4}$, and $1.2e^{-4}$ and $\sigma_N = 7.5e^{-4}, 6e^{-4}, 4e^{-4}$, and $2e^{-4}$ for the table, chair, sofa, and car classes, respectively, and $m = 10$ for our margin-loss-based approach. We use $\tau = 5$ for all classes in our regression-based approach.

Table 2. Ranking evaluations with 150 models per query. The models are randomly selected and sorted by $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ (the query is not included). All results are for the top-1 retrieval results of each method. The numbers multiplied by $1e^{-2}$ are reported. Bold is the smallest, and underscore is the second smallest.

Method	Table			Chair			Sofa			Car			Plane		
	Mean d^m	Mean $e^m_{\mathcal{D}}$	Mean Rank	Mean d^m	Mean $e^m_{\mathcal{D}}$	Mean Rank	Mean d^m	Mean $e^m_{\mathcal{D}}$	Mean Rank	Mean d^m	Mean $e^m_{\mathcal{D}}$	Mean Rank	Mean d^m	Mean $e^m_{\mathcal{D}}$	Mean Rank
Ranked-CD	**6.24**	3.20	12.53	**5.65**	2.61	11.37	**4.73**	1.87	14.07	**2.75**	1.31	12.0	**1.83**	**1.26**	**5.53**
AE	6.95	3.11	11.69	6.08	2.61	10.21	5.19	1.91	14.43	3.09	1.39	14.55	2.60	1.68	17.91
CD-Margin	6.77	3.19	12.55	6.02	2.72	13.24	5.07	1.93	15.76	3.02	1.48	18.94	2.36	1.56	12.27
Ours-Margin	8.89	2.88	8.86	7.15	2.37	8.15	5.83	1.67	9.09	3.61	1.34	12.95	2.65	1.48	10.67
Ours-Reg	8.59	**2.71**	**7.05**	7.39	**2.24**	**6.32**	6.23	**1.62**	**7.91**	3.80	**1.24**	**7.80**	2.64	1.42	8.96

We report mean $d^m(\mathbf{s}, \mathbf{t})$ and mean $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ for all methods in Tab. 1. When observing the results of our two methods, margin-loss-based (*Ours-Margin*) and regression-based (*Ours-Reg*) approaches, the distance *before* deformation ($d^m(\mathbf{s}, \mathbf{t})$) is farther than the baselines, but the distance *after* deformation ($e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$) is smaller. Such a result is shown consistently in all classes, particularly for *Ours-Reg* results. This indicates that our methods can discover 3D models that can better align with the query shape through the given deformation operation.

Also, *Ours-Reg* achieves better results than *Ours-Margin* consistently for all classes. The advantage of *Ours-Reg* is that it can discriminate among all models in $\mathbf{X_t}$, while *Ours-Margin* can only consider *inter*-relationships across the positive set $\mathbf{P_t}$ and the negative set $\mathbf{N_t}$ but not *intra*-relationships in each set. Hence, *Ours-Reg* can achieve a better understanding of the overall distribution of the data. *Ours-Margin* also has a trade-off for the thresholds of $\sigma_{\mathbf{P}}$ and $\sigma_{\mathbf{N}}$; too tight thresholds may result in overfitting, and too loose thresholds can make the two sets less distinguishable. We empirically found good thresholds for each class, but finding the optimum thresholds is a very time-consuming task requiring an extensive binary search.

In Fig. 4, we visualize some examples of retrieved models and their deformations given query models. The models retrieved by our methods have large distance before deformation but better fit after deformation compared with the results of other methods. For example, the chair and sofa retrieved by our methods as shown in Fig. 4 have bigger back parts than the queries, but they become smaller properly after operating the deformation. Our network is able to be agnostic to the details that are easy to recover via deformations such as chair body size, and table leg thickness. It rather retrieves based on the overall shape of the model that can be deformed to fit the desired target. On the other hand, the small geometric details can be inherited from the retrieved model and be preserved during deformation. It is also noticeable that our retrieval is more structurally faithful as we observe the presence of legs in the retrieved sofa or the trunk of the car that are essential for valid deformation.

We also report the *rank* of retrieved models when we sort the test models based on $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$. Since it is computationally extremely expensive to compute the deformation for all pairs in our dataset, for each 3D model, we randomly

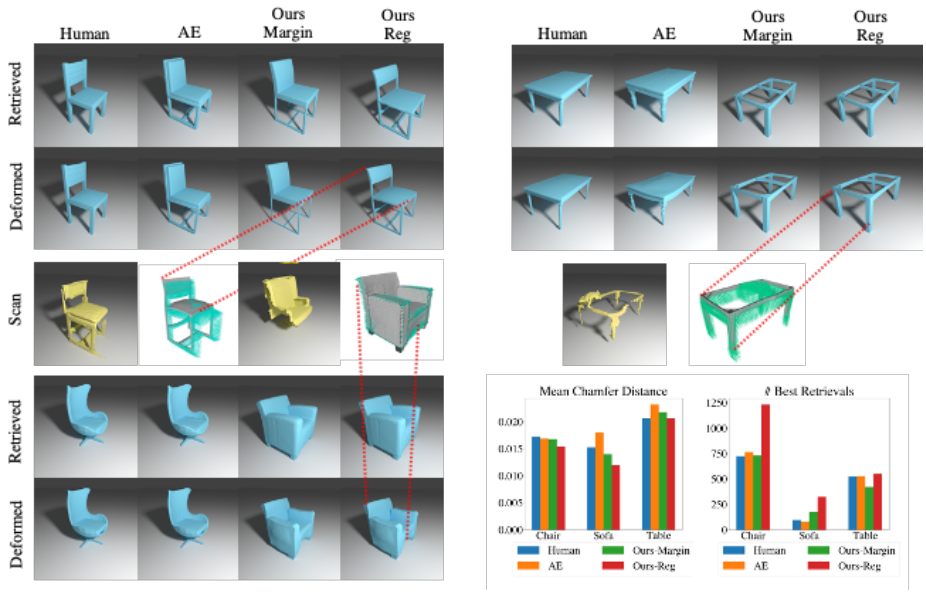


Fig. 5. (**Top row and bottom left**) Qualitative results of Scan-to-CAD. (**Bottom right**) Quantitative Results: We compare different retrieval methods on the Scan2CAD [6] dataset. The left chart shows the mean fitting errors and the right chart shows the number of best candidates retrieved by different methods. *Ours-Reg* achieves the minimum overall fitting errors and the maximum number of best retrievals among all categories compared with other methods. See supplementary for more results.

sample 150 other models and use them for the ranking; $e^m_{\mathcal{D}}(\mathbf{s}, \mathbf{t})$ and $d^m(\mathbf{s}, \mathbf{t})$ are precomputed for them. The results in Sec. 5.1 show the mean rank of the top-1 retrieval results out of the selected 150 models. *Our-Reg* and *Our-Margin* achieve the best and the second best mean ranks compared with the baseline methods in all classes except for cars; most of the car models are structurally similar to each other. *Our-Reg* still provides the best mean rank for cars.

5.2 Scan-to-CAD

We also evaluate our method for the real scan-to-CAD conversion problem [6]. We use our models trained on ShapeNet [10] and directly evaluate our performance on the Scan2CAD [6] dataset. Scan2CAD provides partial 3D scans of indoor scenes, which are segmented to each object instance. We normalize and align the object scans to the ShapeNet canonical space using the 9DoF alignment provided in [6]. We use our embedding for retrieval and then apply the deformation function (Sec. 4) to the retrieved CAD to fit the scan. Similar to previous works [63,68], we reflect the scan about the vertical symmetric plane before fitting. Our evaluation is performed on three common categories: chairs, sofas, and tables. We compare different methods including human annotations defined by the dataset [6], the

autoencoder, our margin-loss-based retrieval, and our regression-based retrieval. We exclude 149 extremely partial scans (2% among all scans) which cover less than 10% of the regions (after reflection) of human-selected shapes, as these shapes are too incomplete to be recognizable.

Similar to the ShapeNet evaluation, we also measure *point-to-mesh* Chamfer distance that is uniformly and densely sampled with $50k$ points. However, we evaluated on one-way CAD to scan fitting, since our goal is to fit the complete CAD models to all observed regions in the partial scans. The fitting error of *Ours-Reg* is the least in all categories as shown on the left chart on the bottom right of Fig. 5. We show a clear advantage for the chairs and the sofas. For tables, we visualized and found that Scan2CAD tables are quite similar to each other, so many potential candidates can fit the scan relatively well with our chosen deformation function (Sec. 4).

We also report the number of the best-retrieved models among the evaluated methods as shown in the right chart on the bottom-right corner of Fig. 5. The number of best candidates with *Ours-Reg* is significantly higher than the baselines and even compared with human-selected models on chairs and sofas.

Fig. 5 (top row and bottom left) shows the qualitative comparison for different methods in three categories. We see that *Ours-Reg* retrieves models that are more similar to that of the target scan. For example, ours is the only method that retrieves the model with the same connectivity of parts compared to the target as shown by the legs of the chair and the top-less table scan. For the sofa example, we retrieve a model with a large distance but which is similar in shape to the observed regions of the scan, hence the deformed model fits the scan better than other methods. By deforming the CAD model, we additionally preserve important CAD features including sharp edges and corners.

6 Conclusion

We proposed a *deformation-aware* 3D model retrieval framework that enables finding a 3D model best matching a query after the deformation. Due to the feature-preserving regularizations in most deformation techniques, 3D models generally cannot exactly match the query through deformation but induce a *fitting gap*. This gap, describing the *deformability* one to the other, is *asymmetric* by definition and thus not a metric. Hence, we introduced a novel embedding technique that can encode such relationships with *egocentric* distance fields given any arbitrary deformation function and proposed two strategies for network training. We demonstrated that our approach outperforms other baselines in the experiments with ShapeNet and also presented results in scan-to-CAD and image-to-CAD applications. We plan to further investigate the relationships among 3D models defined by the deformation in the future.

Acknowledgements This work is supported by a Google AR/VR University Research Award, a Vannevar Bush Faculty Fellowship, a grant from the Stanford SAIL Toyota Research Center, and gifts from the Adobe Corporation and the Dassault Foundation.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.J.: Learning representations and generative models for 3d point clouds. In: ICML (2018) 9, 10
2. Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., Smola, A.J.: Distributed large-scale natural graph factorization. In: WWW (2013) 4
3. Aldrovandi, R., Pereira, J.: An Introduction to Geometrical Physics. World Scientific (1995) 6
4. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR (2016) 7, 8
5. Avetisyan, A., Dahnert, M., Dai, A., Savva, M., Chang, A.X., Nießner, M.: Scan2CAD: Learning cad model alignment in RGB-D scans. In: CVPR (2019) 1
6. Avetisyan, A., Dai, A., Nießner, M.: End-to-end cad model retrieval and 9dof alignment in 3d scans. In: ICCV (2019) 1, 13
7. Bellet, A., Habrard, A., Sebban, M.: A survey on metric learning for feature vectors and structured data (2013) 6
8. Buldygin, V., Kozachenko, V., Kozachenko, I., Kozacenko, J., Kozachenko, Y., Kozačenko, Û., Kozachenko, V., Kozachenko, I., Zaâc, V.: Metric Characterization of Random Variables and Random Processes. American Mathematical Society (2000) 6
9. Bylow, E., Sturm, J., Kerl, C., Kahl, F., Cremers, D.: Real-time camera tracking and 3d reconstruction using signed distance functions. In: RSS (2013) 1
10. Chang, A.X., Funkhouser, T.A., Guibas, L.J., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository (2015) 2, 3, 11, 13
11. Chechik, G., Sharma, V., Shalit, U., Bengio, S.: Large scale online learning of image similarity through ranking. Journal of Machine Learning Research (2010) 3
12. Chen, L., Lian, X.: Efficient similarity search in nonmetric spaces with local constant embedding. IEEE Transactions on Knowledge and Data Engineering **20**(3) (2008) 5
13. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: CVPR (2005) 4
14. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning (1995) 8
15. Dahnert, M., Dai, A., Guibas, L., Nießner, M.: Joint embedding of 3d scan and cad objects. In: ICCV (2019) 1
16. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: BundleFusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. In: ACM SIGGRAPH (2017) 1
17. Dai, A., Ruizhongtai Qi, C., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis. In: CVPR (2017) 1
18. Deng, H., Birdal, T., Ilic, S.: PPFNet: Global context aware local features for robust 3d point matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 195–205 (2018) 5
19. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: KDD (2017) 4
20. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: CVPR (2016) 9
21. G, V.K.B., Carneiro, G., Reid, I.: Learning local image descriptors with deep siamese and triplet convolutional networks by minimizing global loss functions. In: CVPR (2016) 4

22. Garcia, N., Vogiatzis, G.: Learning non-metric visual similarity for image retrieval. Image and Vision Computing **82** (2019) 5
23. Garland, M., Heckbert, P.S.: Simplifying surfaces with color and texture using quadric error metrics. In: Visualization (1998) 9
24. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: Deep self-supervised cycle-consistent deformation for few-shot shape segmentation. In: Eurographics Symposium on Geometry Processing (2019) 4
25. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: KDD (2016) 4
26. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR (2006) 3, 4
27. Han, X., Leung, T., Jia, Y., Sukthankar, R., Berg, A.C.: MatchNet: Unifying feature and metric learning for patch-based matching. In: CVPR (2015) 4
28. Hanocka, R., Fish, N., Wang, Z., Giryes, R., Fleishman, S., Cohen-Or, D.: ALIGNet: Partial-Shape agnostic alignment via unsupervised learning. ACM Transactions on Graphics (2018) 4
29. Hinton, G.E., Roweis, S.T.: Stochastic neighbor embedding. In: NIPS (2003) 8
30. Huang, J., Dai, A., Guibas, L.J., Nießner, M.: 3Dlite: towards commodity 3d scanning for content creation. In: ACM SIGGRAPH Asia (2017) 1
31. Huang, J., Su, H., Guibas, L.: Robust watertight manifold surface generation method for shapenet models (2018) 9
32. Igarashi, T., Moscovich, T., Hughes, J.F.: As-rigid-as-possible shape manipulation. In: ACM SIGGRAPH (2005) 2, 4, 5
33. Jack, D., Pontes, J.K., Sridharan, S., Fookes, C., Shirazi, S., Maire, F., Eriksson, A.: Learning free-Form deformations for 3D object reconstruction. In: ICCV (2018) 4
34. Joshi, P., Meyer, M., DeRose, T., Green, B., Sanocki, T.: Harmonic coordinates for character articulation. In: ACM SIGGRAPH (2007) 2, 3
35. Ju, T., Schaefer, S., Warren, J.: Mean value coordinates for closed triangular meshes. In: ACM SIGGRAPH (2005) 2, 3
36. Kraevoy, V., Sheffer, A., Shamir, A., Cohen-Or, D.: Non-homogeneous resizing of complex models. In: ACM SIGGRAPH Asia (2006) 2, 3, 5
37. Kulis, B., et al.: Metric learning: A survey. Foundations and Trends® in Machine Learning (2013) 6
38. Kurenkov, A., Ji, J., Garg, A., Mehta, V., Gwak, J., Choy, C.B., Savarese, S.: DeformNet: Free-Form deformation network for 3d shape reconstruction from a single image. In: WACV (2018) 4
39. Li, Y., Dai, A., Guibas, L., Nießner, M.: Database-assisted object retrieval for real-time 3d reconstruction. In: Eurographics (2015) 1
40. Lipman, Y., Sorkine, O., Cohen-Or, D., Levin, D., Rossi, C., Seidel, H.P.: Differential coordinates for interactive mesh editing. In: Shape Modeling Applications (2004) 2, 4, 5
41. Lipman, Y., Levin, D., Cohen-Or, D.: Green coordinates. In: ACM SIGGRAPH (2008) 2, 3
42. Lipman, Y., Sorkine, O., Levin, D., Cohen-Or, D.: Linear rotation-invariant coordinates for meshes. In: ACM SIGGRAPH (2005) 2, 4, 5
43. Liu, E.Y., Guo, Z., Zhang, X., Jojic, V., Wang, W.: Metric learning from relative comparisons by minimizing squared residual. In: ICDM (2012) 6
44. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research (2008) 8

45. Mahalanobis, P.C.: On the generalized distance in statistics. In: Proceedings of the National Institute of Science. National Institute of Science of India (1936) 6
46. Morozov, S., Babenko, A.: Non-metric similarity graphs for maximum inner product search. In: Advances in Neural Information Processing Systems (2018) 5
47. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: Real-time dense surface mapping and tracking. In: ISMAR (2011) 1
48. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: KDD (2013) 4
49. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017) 7, 9
50. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A unified embedding for face recognition and clustering. In: CVPR (2015) 3, 4, 10
51. Sederberg, T.W., Parry, S.R.: Free-form deformation of solid geometric models. In: ACM SIGGRAPH (1986) 2, 3, 5
52. Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., Moreno-Noguer, F.: Discriminative learning of deep convolutional feature point descriptors. In: ICCV (2015) 4
53. Skopal, T.: On fast non-metric similarity search by metric access methods. In: International Conference on Extending Database Technology. Springer (2006) 5
54. Skopal, T., Bustos, B.: On nonmetric similarity search problems in complex domains. ACM Computing Surveys (CSUR) **43**(4), 1–50 (2011) 5
55. Skopal, T., Lokoč, J.: Nm-tree: Flexible approximate similarity search in metric and non-metric spaces. In: International Conference on Database and Expert Systems Applications. pp. 312–325. Springer (2008) 5
56. Sorkine, O., Alexa, M.: As-rigid-as-possible surface modeling. In: Eurographics Symposium on Geometry Processing (2007) 2, 4, 5, 9
57. Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., Seidel, H.P.: Laplacian surface editing. In: Eurographics Symposium on Geometry Processing (2004) 2, 4, 5
58. Stratasys: GrabCAD community, https://grabcad.com/library 2
59. Tan, X., Chen, S., Li, J., Zhou, Z.H.: Learning non-metric partial similarity based on maximal margin criterion. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). vol. 1. IEEE (2006) 5
60. Tian, Y., Fan, B., Wu, F.: L2-Net: Deep learning of discriminative patch descriptor in euclidean space. In: CVPR (2017) 4
61. Trimble: 3D warehouse, https://3dwarehouse.sketchup.com/ 2
62. TurboSquid: TurboSquid, https://www.turbosquid.com/ 2
63. Wang, W., Ceylan, D., Mech, R., Neumann, U.: 3DN: 3d deformation network. In: CVPR (2019) 4, 9, 13
64. Weber, O., Ben-Chen, M., Gotsman, C.: Complex barycentric coordinates with applications to planar shape deformation. In: Eurographics (2009) 2, 3
65. Whelan, T., Leutenegger, S., Salas-Moreno, R.F., Glocker, B., Davison, A.J.: ElasticFusion: Dense slam without a pose graph. Robotics: Science and Systems (2011) 1
66. William L. Hamilton, Rex Ying, J.L.: Representation learning on graphs: Methods and applications. IEEE Data Engineering Bulletin (2017) 4
67. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: ICCV (2019) 11
68. Yifan, W., Aigerman, N., Kim, V., Chaudhuri, S., Sorkine-Hornung, O.: Neural cages for detail-preserving 3d deformations (2019) 4, 9, 13

69. Yumer, E., Mitra, N.J.: Learning semantic deformation flows with 3d convolutional networks. In: ECCV (2016) 4