3D Shape Retrieval Using a Single Depth Image from Low-cost Sensors

Jie Feng[1] Yan Wang[2] Shih-Fu Chang[1,2]
[1]Department of Computer Science, Columbia University
jiefeng@cs.columbia.edu
[2]Department of Electrical Engineering, Columbia University
{yanwang, sfchang}@ee.columbia.edu

Abstract

Content-based 3D shape retrieval is an important problem in computer vision. Traditional retrieval interfaces require a 2D sketch or a manually designed 3D model as the query, which is difficult to specify and thus not practical in real applications. With the recent advance in low-cost 3D sensors such as Microsoft Kinect and Intel Realsense, capturing depth images that carry 3D information is fairly simple, making shape retrieval more practical and user-friendly. In this paper, we study the problem of cross-domain 3D shape retrieval using a single depth image from low-cost sensors as the query to search for similar human designed CAD models. We propose a novel method using an ensemble of autoencoders *in which each autoencoder is trained to learn a compressed representation of depth views synthesized from each database object. By viewing each autoencoder as a probabilistic model, a likelihood score can be derived as a similarity measure. A domain adaptation layer is built on top of autoencoder outputs to explicitly address the cross-domain issue (between noisy sensory data and clean 3D models) by incorporating training data of sensor depth images and their category labels in a weakly supervised learning formulation. Experiments using real-world depth images and a large-scale CAD dataset demonstrate the effectiveness of our approach, which offers significant improvements over state-of-the-art 3D shape retrieval methods.*

1. Introduction

Content-based 3D shape retrieval is an important topic in computer vision. It provides rich geometry information compared with 2D images and has important applications in 3D content organization and exploration, 3D model editing and printing, augmented reality, and even self-driving cars. Traditional 3D retrieval has focused on the settings of a CAD model database and various forms of query, including 2D sketches and CAD models. Recent developments in

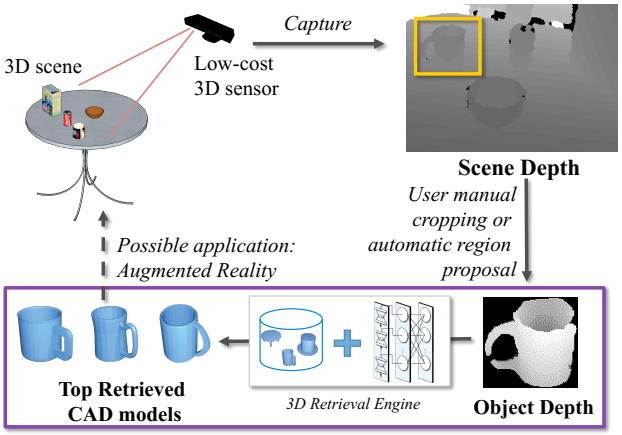


Figure 1. 3D shape retrieval using one depth image from a low-cost sensor. The purple rectangle shows the scope of this paper.

low-cost 3D sensors such as Microsoft Kinect and the Intel Realsense camera have introduced a new setting to 3D retrieval, i.e. using the user-captured depth image as the query (as shown in Figure 1). This new setting allows general users without professional knowledge of CAD modeling or sketching to effortlessly obtain 3D models that are visually similar to physical objects. This motivates many new exciting applications. For example, a user can search for a CAD model that is similar to his cup, and then virtually move the cup around through augmented reality glasses. With the potential proliferation of these 3D sensors on mobile devices, shape retrieval and its derived applications can be as accessible and enjoyable as taking a picture.

While offering advantages when compared with traditional settings, 3D shape retrieval that is based on low-cost sensors also has its unique challenges. First, the captured 3D shape is usually incomplete due to self-occlusion. Algorithms for RGBD registration such as KinectFusion [8] can alleviate this problem by combining multiple depth images. However, in most cases, it is still impractical to scan many angles of an object, in addition to the requirement of computational cost to run the algorithm. Second, the inexpensive cost of the sensors is accompanied by a compromise

of higher sensor noise, especially comparing with higher-end LiDAR sensors or desktop 3D scanners. This makes the retrieval problem especially difficult, considering that the CAD models in the database are manually designed and free of sensor noise. Third, despite the fact that much effort has been invested in 3D features, adequate feature representation is still lacking for shape retrieval, especially when handling an incomplete query that could be captured from arbitrary viewpoints. In this paper, we address the challenges associated with the cross-domain 3D retrieval problem, with input from the low-cost 3D sensors and clean 3D models as the search targets.

As a recent effort to tackle the above challenges, Wang *et al.* [28] adopts an approach of first reconstructing a 3D model from the depth inputs as the query, then extracting 3D features from the reconstructed model, and finally using a Regression Tree Field to perform the retrieval. Although their approach shows promising performance, the use of 3D features has shortcomings, especially when only the noisy depth image from a single view is available. As an example, Figure 2 shows a depth map of a chair and the reconstructed 3D model from it. The inaccurate measurement near the object boundary significantly distorts the shape of the recovered 3D model, and thus makes 3D-based approaches more difficult. Also, good retrieval performance heavily relies on high-quality 3D local features, which requires significant efforts to properly engineer. As an alternative to 3D feature-based methods, the view-based approach is much less sensitive to the sensory noise and can benefit from a large body of mature matching techniques that have been developed recently. Furthermore, to avoid dependency on ad-hoc feature engineering, we leverage the successful representation learning paradigm, using neural networks to learn a discriminative, yet compact, representation directly on the depth images synthesized from 3D models in the database.

Autoencoders have been shown to be a simple, yet powerful, model achieving state-of-the-art performance in many problems including object recognition [13], face recognition [17], and action recognition [14]. They fit well in our retrieval problem and we can train autoencoders for the synthesized depth images from the 3D models. Instead of learning a single autoencoder to represent all 3D models, we propose to learn an object-specific autoencoder for each model and take a generative probabilistic perspective, similar to those in [1][10], to measure the similarity between the noisy sensory query and each 3D model. Finally, due to the inherent difference between sensor depth images and synthesized depth images used to train the autoencoder, the cross-domain issue also needs to be addressed.

Specifically, as shown in Figure 3, we propose a neural network architecture called *ensemble of autoencoders* to tackle the challenging problems of 3D shape retrieval using a single depth view from a low-cost 3D sensor. This is inspired from successful ideas using an ensemble of weak classifiers in detection and recognition [18][29][25]. For each 3D object model in the database, a contractive autoencoder [20] is trained using synthesized depth images to represent the view distribution of the object model. This object-specific autoencoder serves as a model to estimate a probability score that a depth image is generated by the corresponding object. An ensemble of autoencoders is then constructed to produce this score for each database object. To handle the cross-domain issue, we build a domain adaptation layer on top of the ensemble structure to learn the domain difference between the synthesized and sensor-captured depth images in a weakly supervised way.

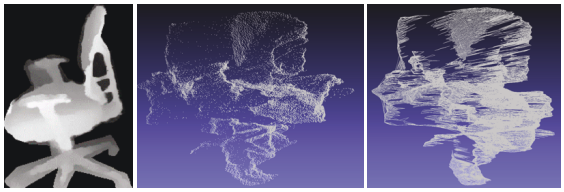


Figure 2. An example depth map of a chair (left), the reconstructed point cloud (middle), and the mesh model (right).

By utilizing autoencoders to model the view distribution of each object, our approach not only eliminates the need of specialized feature engineering, but also enhances robustness against viewpoint variance. The adaptation process explicitly addresses the cross-domain issue between training images in the 3D model database and query inputs captured by noisy sensors, which provides significant performance gains according to our experiments. The overall architecture is easy to scale and train, making it suitable for our retrieval task. Experiments on popular datasets demonstrate that the proposed approach has significantly better retrieval performance and computational efficiency compared with state-of-the-art 3D retrieval approaches and baselines.

In summary, our contributions include:

- proposing a novel approach for the cross-domain 3D shape retrieval problem, using a unified neural network architecture with a domain adaptation design;
- the first attempt to learn a depth feature representation from an ensemble perspective, to the best of the authors' knowledge;
- demonstrating the efficacy and efficiency of the proposed approach with extensive experiments.

2. Related Work

2.1. Content-based 3D Shape Retrieval

Just as 2D local descriptors play a critical role in content-based image retrieval, many 3D local descriptors have also been proposed to describe the local geometry of 3D models for shape retrieval. Spin Images proposed by Johnson et al. [9] project neighboring vertices to a local coordinate

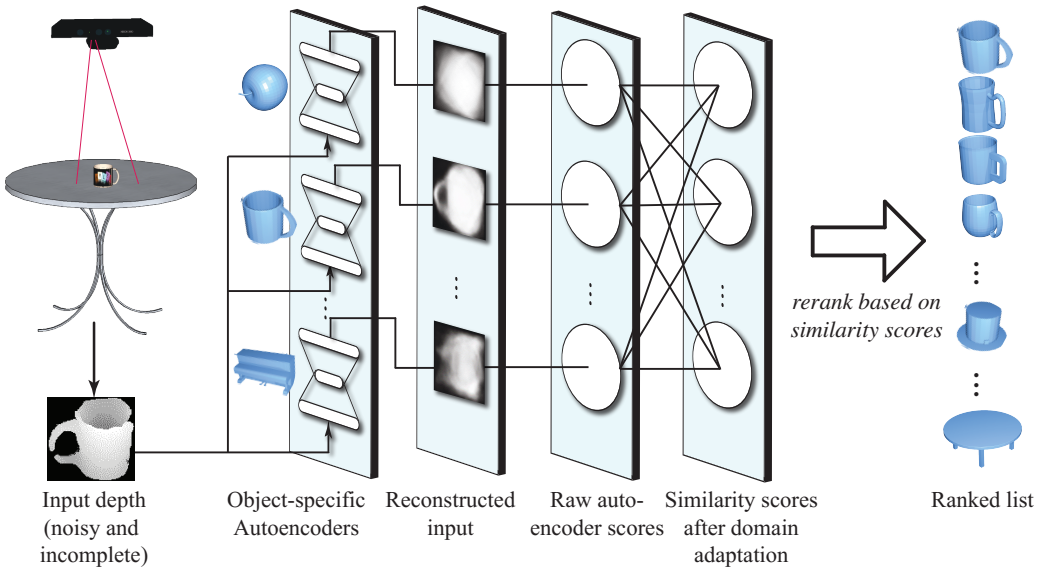


Figure 3. The architecture of our retrieval system, from a neural network perspective. The 3D model next to each autoencoder indicates the CAD model from which the autoencoder is trained.

system, forming a 2D density histogram as the feature. Spin Image is invariant to isometric deformation, but is sensitive to scale change. Heat Kernel Descriptor [19] offers certain non-rigid matching capabilities, using the Laplace-Beltrami operator. Inspired by successful 2D local descriptors, 3D extension of these descriptors has also been introduced for shape retrieval, including 3D SIFT [24] and 3D HOG [31]. These 3D local descriptors are usually aggregated to form an object-level feature vector. After extracting local descriptors from an object, the Bag-of-Words(BoW) model is widely used to aggregate these descriptors into a histogram representation, and then distance metrics such as ℓ_1, ℓ_2 or intersection are adopted for retrieval.

Another major direction for 3D shape retrieval is view-based methods, which project each 3D object model to a collection of 2D *view images*, from which regular image features are extracted to describe the 3D model. View matching is then performed to compute similarity scores. The view images are usually silhouettes or textureless depth images, since most 3D CAD models are not textured. The view-based approach can benefit from sophisticated image processing techniques and is therefore usually more discriminative for retrieval than 3D local descriptors. Chen et al. [4] proposed a feature representation for 3D models called Light Field Descriptor (LFD) that creates 10 silhouettes from the vertices of a dodecahedron and computes Zernike moments and Fourier descriptors for each image. Daras et al. [5] proposed the Compact Multi-View Descriptor (CMVD) method, which integrates multiple features from the binary and depth images to describe a 3D model. Bo et al. [3] have proposed to learn a Kernel Descriptor for RGBD images, which demonstrates promising results on instance and category recognition on several RGBD datasets.

2.2. Representation Learning for 3D Models

Despite the significant progress gained from the aforementioned approaches for 3D shape retrieval, most of them still rely heavily on manually-designed features. Recent years have witnessed unprecedented advancement in representation learning for computer vision, especially deep learning. Although color images remain the major focus, some efforts have been invested to extend these techniques to 3D model processing. Wu et al. [30] introduced a convolutional Deep Belief Network (DBN) to infer the 3D structure and semantics behind a depth image, outperforming existing approaches on various tasks including shape classification and 2.5D recognition. Leng et al. [15] have proposed the use of a stacked local convolutional autoencoder to learn deep representation for 3D object models, and demonstrated significant improvement over state-of-the-art retrieval methods. Our approach is motivated by the power of feature learning but differs greatly from these approaches in learning an ensemble of neural networks to produce view feature and proposes a domain adaptation layer specifically for our cross-domain retrieval problem.

3. Approach

In our approach, the captured noisy depth image is first input to an ensemble of autoencoders, each of which then estimates the probability the input depth image is generated by the underlying autoencoder. The output score values of the autoencoders are then fed to the layer of domain adaptation before ranking the 3D object models in the database. The entire framework is shown in Figure 3.

3.1. Modeling Depth View Distribution

3.1.1 Object-specific Autoencoders

Autoencoders are neural networks aiming to reconstruct the input itself. They first use an encoding function $h(\cdot)$ to transform the input data $\boldsymbol{x}$ to a hidden representation:

$$h(\boldsymbol{x}) = \phi(W_h^T \boldsymbol{x} + \boldsymbol{b}_h). \quad (1)$$

Here, $\phi(\cdot)$ is an activation function, which may have various forms, such as linear activation, sigmoid, and rectified linear unit (ReLU). Then, the reconstruction is computed from a decoder $g(\cdot)$:

$$g\big(h(\boldsymbol{x})\big) = \phi\big(W_r^T h(\boldsymbol{x}) + \boldsymbol{b}_r\big), \quad (2)$$

where W_r is often set to W_h^T. Therefore, we use $W = W_h = W_r^T$ to simplify the notation. The training process aims to minimize the average reconstruction error for a set of training data $\chi = \{\boldsymbol{x_i}\}_{i=1}^N$, with a regularization term to prevent converging to the trivial solution of identity function:

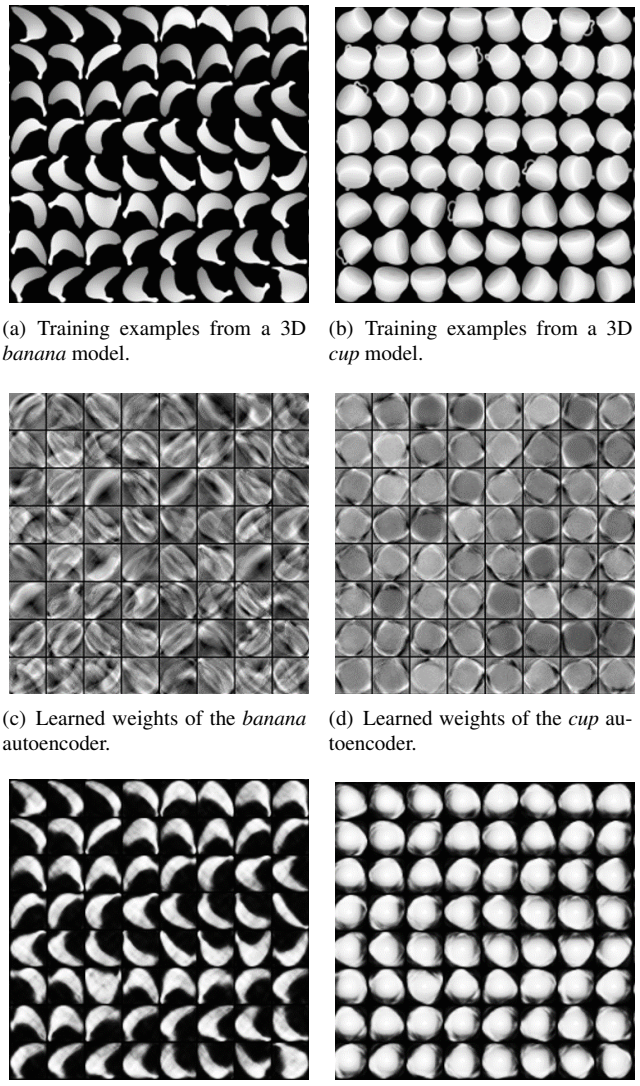
$$\underset{W,\boldsymbol{b}_h,\boldsymbol{b}_r}{\operatorname{argmin}} \frac{1}{N} \sum_i D\big(\boldsymbol{x_i}, r(\boldsymbol{x_i})\big) + R(\chi). \quad (3)$$

Here $r(\boldsymbol{x}) = g\big(h(\boldsymbol{x})\big)$ is the reconstructed version of $\boldsymbol{x}$, and $D(\cdot,\cdot)$ is a distance function that is usually mean squared error or cross-entropy. $R(.)$ is a regularizer, which can be ℓ_1 norm on $h(\boldsymbol{x_i})$ (sparse autoencoder (SAE) [13]), a denoising function (denoising autoencoder (DAE) [27]), Frobenius norm on Jacobian matrix of $h(\boldsymbol{x_i})$ (contractive autoencoder (CAE) [20]). With a proper activation function and regularization, an autoencoder is able to learn a robust and useful feature representation for the input data [1][20][27].

We represent each 3D object model o_i as a collection of depth images $\{\boldsymbol{x_i}\}$[1] rendered from various locations on its view sphere. The sampling is done uniformly from each rotation axis (pitch, yaw, roll) of the object. For each axis, we divide it into 15 uniform segments, which yields $15 \times 15 \times 15 = 3375$ total number of views for each object. Each view is cropped to fit the bounding box of the object, and resized to a fixed scale grayscale image. Example depth images are shown in Figure 4(a) and Figure 4(b).

A contractive autoencoder with mean squared error and sigmoid activation is learned using the synthesized depth images from each object as training data. The autoencoder can be considered as a compact representation of the depth view distribution given the 3D object. We refer to it as object-specific autoencoder AE_i for object o_i. The training of the autoencoder is accomplished by Stochastic Gradient Descent (SGD). Figure 4 illustrates two example autoencoders trained on a banana model and a cup model. The visualized weights are the learned weights of the encoding layer and they show that important shape and depth characteristics are indeed captured by the autoencoders. From the reconstruction results, it is apparent that the trained autoencoder is capable of describing the depth views for the specific training object, e.g. the banana, while the autoencoder that learned from the cup cannot properly recover the depth views for the banana. Such discriminative nature further justifies our approach using an individual autoencoder for each object.

[1]We use the notation $\boldsymbol{x_i}$ here because it is also the training data of the autoencoders.



(a) Training examples from a 3D *banana* model.

(b) Training examples from a 3D *cup* model.

(c) Learned weights of the *banana* autoencoder.

(d) Learned weights of the *cup* autoencoder.

(e) *Banana* reconstruction from *banana* autoencoder.

(f) *Banana* reconstruction from *cup* autoencoder.

Figure 4. Two example object-specific autoencoders.

3.1.2 View-Object Similarity

The reconstruction error has been used as a measure of the fitness of an input sample to an autoencoder. However, for

a regularized autoencoder, this is usually not a good scoring metric, especially when comparing among different autoencoders [10][26]. Some recent works have investigated the data generating distribution learned by an autoencoder, and have shown that certain regularized autoencoders like DAE or CAE can be interpreted as probabilistic models [1][10]. More specifically, [10] treats the reconstruction process in the autoencoder training as a dynamic system and derives a potential energy function as:

$$E(\boldsymbol{x}) = \int_{\infty}^{\boldsymbol{x}} h(\boldsymbol{t})d\boldsymbol{t} - \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{b}_r\|_2^2 + \text{const.} \quad (4)$$

Here, the integral is well-defined because $h(\boldsymbol{t})$ can be shown as a gradient field. The constant depends on the boundary conditions of the dynamic system, which are unknown. The exact form of potential energy depends on the specific choice of activation function. In the case of sigmoid activation, the potential energy function can be written as:

$$E(\boldsymbol{x}) = \sum_k \log\left(1+\exp(\boldsymbol{w}_k^T\boldsymbol{x}+\boldsymbol{b}_h^k)\right) - \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{b}_r\|_2^2 + \text{const}, \quad (5)$$

where k is the index of the hidden node in the autoencoder. This potential energy[2] is identical to the free energy of the corresponding Restricted Boltzmann Machine (RBM), which is more limited and more difficult to train than an autoencoder. This "energy" view equips an autoencoder with the ability to estimate the likelihood of an input sample that will serve, in our case, as the similarity between a query view and an object model.

3.2. Ensemble of Autoencoders

3.2.1 Ensemble Structure

Given the potential energy interpretation described above, each object-specific autoencoder is able to provide a score using Equation (5) to represent how likely an input depth view is generated by the corresponding object model. However, directly using the scores for ranking involves two challenges. First, due to the unknown constant in Equation (5), the likelihood scores from different autoencoders are not directly comparable. Second, depth images captured from low-cost 3D sensors usually exhibit quite different appearances compared with those synthesized from clean, human-designed 3D models. Common situations include noisy and low fidelity depth value, holes, and missing parts due to objects located beyond the sensor's distance range. Since our object-specific autoencoders are trained with synthesized depth images, directly applying the learned autoencoders on sensor depth images leads to inaccurate predictions.

To address these issues, we treat the output scores from object-specific autoencoders as the middle-level feature and add a processing layer on top to predict the final ranking scores so that the object model most similar to the query view will receive the highest score. We call this layer Domain Adaptation Layer (DAL).

[2]We use autoencoder energy or score interchangeably in this paper.

3.2.2 Domain Adaptation by Multi-class Classification

We treat this adaptation process as a multi-class classification in which each database object is a class. In our case, the softmax classifier is used. The goal of the classifier is to predict the most similar database object given the autoencoder scores of a sensor depth image as feature vector. After training the classifier, the prediction score for each object can then be used for ranking. However, it is very challenging to obtain training data in which every object in the database has sensor depth images from similar physical objects. To tackle this problem, we propose a two-step training process. First, we pre-train the classifier using synthesized depth images from each object model; this is equivalent to score calibration tailored for synthesized views. Second, we apply weakly supervised learning to fine-tune the classifier with sensor depth images. The first step is fairly straightforward to implement; therefore, we will focus on the second step.

In most cases, we have knowledge about the category of query images and database objects instead of the connection between a query image and its most similar object, which could be highly subjective. The idea is to treat the most similar object for a query from the same category as a hidden variable and to convert our object classifier to a category classifier using existing category labels.

More specifically, we denote the category label of a query view $\boldsymbol{x_i}$ and an object o_k as $y(\boldsymbol{x_i})$ and $y(o_k)$, respectively. The raw output score vector from autoencoders is $s(\boldsymbol{x_i})$, and the most similar object of $\boldsymbol{x_i}$ is denoted as $o_{\boldsymbol{x_i}}$, which is unknown. The only supervision we have is $y(\boldsymbol{x_i}) = y(o_{\boldsymbol{x_i}})$, i.e. a query depth image and its most similar object model must come from the same category. Our goal is to train an *object* classifier given only the *category* labels. Inspired by multiple instance learning, we represent the category likelihood for the query image using the maximum object likelihood from the same category. The mathematical formulation for the learning objective is depicted in Equation (6) using negative log-likelihood.

$$-\log L = -\sum_i \log p(y(\boldsymbol{x_i})|s(\boldsymbol{x_i})), \quad (6)$$

$$p(y(\boldsymbol{x_i})|s(\boldsymbol{x_i})) = \max_{y(o_k)=y(\boldsymbol{x_i})} p(o_k|s(\boldsymbol{x_i})), \quad (7)$$

$$p(o_k|s(\boldsymbol{x_i})) = \frac{\exp(\boldsymbol{u_k^T}s(\boldsymbol{x_i}))}{\sum_k \exp(\boldsymbol{u_k^T}s(\boldsymbol{x_i}))} \quad (8)$$

where $p(o_k|s(\boldsymbol{x_i}))$ is the probability of $\boldsymbol{x_i}$ belonging to the kth object. The goal is then to find the optimal $\boldsymbol{u_k}$ for each object to minimize Equation (6).

Due to the presence of the max function, the objective is not directly differentiable. As in [2], we use the Noisy-OR (NOR) model to approximate the max function:

$$p(y(\boldsymbol{x_i})|s(\boldsymbol{x_i})) = 1 - \prod_{y(o_k)=y(\boldsymbol{x_i})} (1 - p(o_k|s(\boldsymbol{x_i}))) \quad (9)$$

This approximation ensures that if one object yields high probability, the corresponding category will get high probability, and the value is also bounded within $[0, 1]$. Since the NOR model is differentiable, we can adopt gradient descent for optimization. This formulation is general to query images from a subset of categories as the database objects in which only the weights relevant to objects from the query image categories are adjusted.

3.2.3 Ensemble Training

The entire ensemble architecture with the domain adaptation layer is essentially a multi-layer neural network in which end-to-end learning can be performed for a specific task and dataset. However, to make the system more scalable and efficient to train, we first pre-train all object-specific autoencoders and the domain adaptation layer by using only synthesized depth images with their corresponding 3D object models. The entire architecture can subsequently be fine-tuned using sensor depth images. The pre-trained autoencoders can be trained in parallel and reused when new objects are added to the ensemble and only the domain adaptation layer needs to be retrained.

4. Experiments

4.1. Experiment Settings

Datasets. In order to properly evaluate the novel problem of cross-domain 3D shape retrieval from a single depth view, we require a database consisting of CAD models, as well as queries of depth images captured by the Kinect sensor from real world scenes. We construct the CAD database from a subset of ModelNet [30]. It contains 80 categories, each of which has 20 3D models, forming a database with 1600 instances. This is larger than the widely used 3D shape datasets, e.g. Princeton Shape Benchmark [21] (907) and SHREC12 [16] (1200). For queries, we collect one set from UW RGBD object dataset [12] and the other set from NYU Depth2 dataset [22]. The UW dataset contains Kinect-captured depth images of objects on a turntable. The NYU dataset consists of depth images from cluttered scenes captured by a Kinect. Notice the objects in the CAD database and RGBD datasets are from completely different domains which is useful to demonstrate the generality of our method. Categories appearing in the query sets include everyday objects like *cup*, *box*, *chair*, etc. Because some of the categories have different names in the two datasets, e.g. *eraser* in the UW dataset is named *rubber Eraser* in ModelNet, we manually reconcile the name differences of the object labels for proper evaluation. 20 object categories were selected from UW dataset, and 80 depth images were randomly sampled for each category, with one random half for training the DAL, and the other half to form the query set. The same process was done on NYU dataset to build the second DAL training set and query set. To obtain the query objects in the depth images, an object selection process is usually involved. In a practical scenario, a user can select the query object by simple interaction. Automated selection is also possible by applying object proposal [11] or salient object detection [7]. The selection process is beyond our scope since we are focusing on the retrieval algorithm itself. We obtain the query objects from the object mask of each depth image which are given in both UW and NYU datasets. This is also generally the case in 3D shape retrieval evaluation [6] [16] [23] where the query object is already segmented.

Evaluation protocol. To evaluate a retrieval method, category labels are commonly used since it is very hard to obtain a ground truth ranked list for each object in a large database manually. Two sets of standard retrieval evaluation methods are reported in our experiments.

- Precision-Recall curve and Mean Average Precision (MAP): For a given query depth image, we compute its similarity with each object model using the network output. These models are then ranked in descending order to form a list. Precision, recall and MAP values are computed as in [23]. The final PR curve is averaged across all query points.
- First Tier (FT) and Second Tier (ST): FT measures the recall in the top K retrieval results. K is the number of database instance from the same category as the query. Similarly, ST evaluates the top 2K results to compute the measurement.

Compared approaches. Due to the lack of existing methods with the same cross-domain settings as ours, we selected the state-of-the-art and baseline approaches that address the problem settings most similar to ours.

- Random Tree Field (**RTF**) [28]: the state-of-the-art approach for cross-domain retrieval with low-cost sensors, which uses 3D local descriptors as the representation of object models. A Regression Tree Field is constructed for retrieval. We use the code from authors as it was given.
- HOG+ℓ_2 (**HOG**): a straightforward baseline algorithm for 3D shape retrieval from a single view. The HOG features are extracted from both the query and each synthesized depth image of database models. ℓ_2 norm is used to measure the distance between the query depth image and a depth image synthesized from

3D models. The shortest distance between the query and depth images of an object is assigned to the object and used for ranking the database objects at the end. The dimension of the HOG descriptor is 1116.

- Global Autoencoder (**Global AE**): a single stacked autoencoder with two encoder layers with dimensions of 500 and 300 is trained on synthesized views from all database objects. Retrieval is accomplished using ℓ_2 norm on the features from the encoder output with a dimension of 300.
- Ensemble of autoencoders without full domain adaptation layer (**Ours without DAL**): a baseline to justify the necessity of the Domain Adaptation Layer (DAL). The DAL is only trained using synthesized depth views for score calibration.
- Ensemble of autoencoders with full domain adaptation layer (**Ours**): the proposed approach. The DAL is first pre-trained using synthesized depth views and then fine-tuned using real sensor depth images.

Implementation details. The training of our architecture involves some important parameters. They play a critical role in achieving good performance. Each object depth patch is normalized to have a fixed range of depth values (0-1) to ensure distance invariance. The normalized patch maintains relative depth values thus shape geometry is not lost. After normalization, the patch is resized to 28×28 pixels and then vectorized to be input into the autoencoder. We also tried resolution of 32×32 and 48×48 but very little improvement is observed. We use 28×28 to speed up the computation. The hidden layer has 200 nodes, and the contraction coefficient is 0.01. For training autoencoders, the learning rate is 0.03, and the maximum iteration number of SGD is 200, which is generally sufficient to converge. Each object-specific autoencoder takes approximately 40 seconds to train using an NVIDIA GTX 645 GPU.

4.2. Results and Discussions

The average PR curves on UW query set are plotted in Figure 5 for the methods used in our experiment. Quantitative results for each competing method are shown in Table 1. We also report the average time cost per retrieval for each method. Comparing **RTF** with **Ours without DAL**, where both methods are only trained on CAD model data, shows that our view-based method achieves superior performance to the state-of-the-art 3D feature-based method. Given the difficulty of this task, HOG as a baseline does a reasonable job due to its high discriminative power on 2D images, but the large margin between **HOG** and **Ours without DAL** clearly shows the advantage of learned features using autoencoders over manually specified features. **Global AE** achieves good performance, especially when the recall increases. Without using a stacked autoencoder,

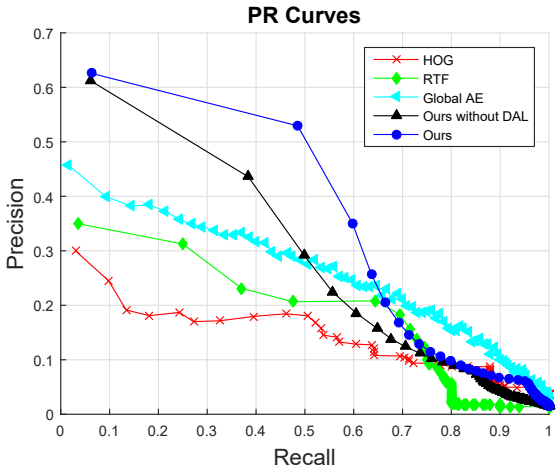


Figure 5. Precision-recall curves for our proposed methods and comparison methods on UW query set (best viewed in color).

Ours without DAL achieves better MAP than **Global AE** with its First Tier accuracy approximately 12% relative gain, which indicates much better quality for top-ranked results. This demonstrates that object-specific autoencoders are more discriminative in a retrieval setting. As seen in Figure 5, the proposed object-specific autoencoders perform a little lower than the method using a global autoencoder in the very high recall range. Since our algorithm is targeted at retrieving similar object models instead of categorization, there are object models in the same category as the query but not necessarily visually similar. Thus the proposed method using object-specific autoencoders will push objects of dissimilar appearance down in the ranked list. However, in terms of the overall accuracy over the entire ranked list (based on MAP), the proposed methods are still much better than the Global AE and runs at a much high speed (as high as $40\times$ speed gain). From the results of **Ours without DAL** and **Ours**, we observe significant 7% performance gain in MAP (7% absolute gain, 27% relative gain), which demonstrates the effectiveness of our domain adaptation layer fine-tuned with sensor depth images, and illustrates the necessity to specifically address the cross-domain issue in our problem.

The same quantitative results on NYU query set is shown in Table 2. Due to the lower quality of object depth images from NYU dataset, 3D local descriptors can not be reliably extracted for **RTF** (code provided by the authors), therefore we don't report its performance here. Although the overall performance of all methods decreases compared with that on UW query set, our method still achieves the best performance among all competing methods with an obvious margin.

To understand the relative retrieval difficulties for different object categories, we present a detailed performance analysis for each object category in Table 3. The top 5 best-

performing and worst-performing categories ranked by FT are shown in the upper and lower regions of the table respectively. Top categories like *Banana* and *Cup* perform well due to their distinctive shapes and appearances, even though the input from the depth sensor may not always be of high quality. In contrast, *Cell Phone* and *Ball* perform poorly because of their somehow less discriminative shapes, making the retrieval ambiguous and encountering difficulty in distinguishing them from other similar objects, e.g. cell phone and small book, ball and orange.

Method	FT	ST	MAP	Time (s)
HOG	0.20	0.18	0.15	9.4
RTF [28]	0.32	0.22	0.18	0.27
Global AE	0.37	0.28	0.24	4.3
Ours without DAL	0.42	0.26	0.26	0.11
Ours	**0.53**	**0.30**	**0.33**	**0.11**

Table 1. Quantitative evaluation results on UW query set.

Method	FT	ST	MAP
HOG	0.12	0.15	0.13
Global AE	0.27	0.22	0.19
Ours without DAL	0.33	0.23	0.20
Ours	**0.40**	**0.26**	**0.24**

Table 2. Quantitative evaluation results on NYU query set.

For qualitative evaluation, we visualize example retrieval results in Figure 6. For each query, the color image is also shown for a better illustration, but note that only the depth image is used in our experiment. Top retrieval results are displayed by rendering the corresponding database model from a random perspective. Incorrect results are indicated with a dashed box. Our method is able to handle queries from arbitrary views and is robust to noise and obscured parts. The final two rows of Figure 6 present example failure cases in which the *bowl* retrieves *cups* as top results and the camera yields a *calculator* and *boxes* as its top results. Possible reasons are: 1) significant missing regions in the depth image, e.g. the camera screen is not perceived by Kinect sensor; 2) similar views among different objects, e.g. *bowls* and *cups* from top-down view. The algorithm nevertheless manages to accomplish its goal to locate objects with the most similar shapes based on the query view.

5. Conclusions

In this paper, we study 3D shape retrieval scenario that uses a single depth image from low-cost 3D sensors as the query. A novel approach based on an ensemble of autoencoders is presented in which an autoencoder is trained on each database model and is able to yield a likelihood measure for an input depth image. A novel domain adaptation layer is further trained to address the cross-domain issue between queries and training data to produce final ranking scores. Extensive experiments demonstrate promising performance of our approach on this challenging task.

With the fast development and deployment of low-cost 3D sensors, especially those targeting mobile devices, we anticipate wide applications of 3D shape retrieval using depth images as query. In future work, we will explore automatic query proposal using object proposal or saliency analysis, and how our ensemble architecture can be applied to other problems like 3D object recognition.

Category	FT	ST	MAP
Banana	0.95	0.48	0.87
Cup	0.94	0.48	0.85
Plate	0.94	0.48	0.85
Notebook	0.86	0.44	0.45
Box	0.67	0.36	0.50
Sponge	0.17	0.10	0.11
Camera	0.15	0.09	0.09
Keyboard	0.12	0.09	0.07
Cell Phone	0.12	0.09	0.07
Ball	0.09	0.05	0.05

Table 3. Quantitative evaluation results for the best and worst performing object categories using our method.

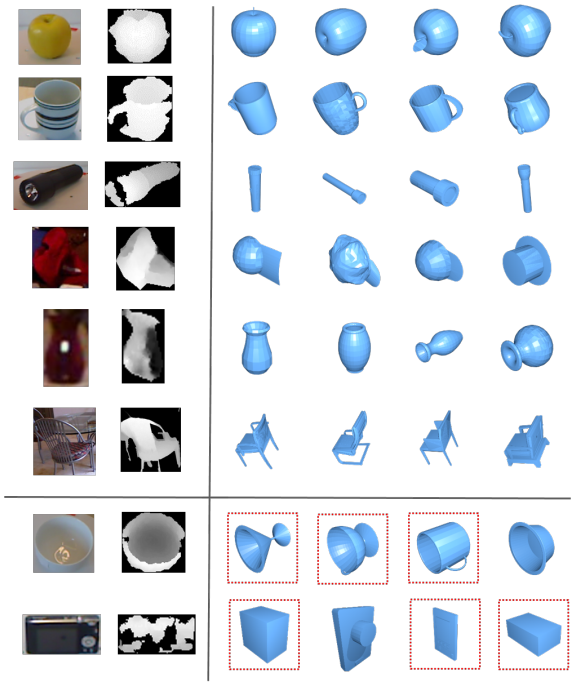


Figure 6. Example top retrieval results using our proposed method. Queries are depth images (color images are not used) from the UW dataset (row 1-3) and NYU dataset (row 4-6). From top to bottom, the queries are *apple*, *cup*, *flashlight*, *hat*, *vase*, *chair*, *bowl* and *camera*. The last two rows show failure examples, where incorrect results are highlighted with red boxes.

References

[1] G. Alain and Y. Bengio. What regularized auto-encoders learn from the data-generating distribution. *JMLR*, 15, 2014. 2, 4, 5

[2] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*. IEEE, 2009. 6

[3] L. Bo, X. Ren, and D. Fox. Depth kernel descriptors for object recognition. In *IROS*, 2011. 3

[4] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, 2003. 3

[5] P. Daras and A. Axenopoulos. A compact multi-view descriptor for 3d object retrieval. In *CBMI Workshop on Content-Based Multimedia Indexing*, 2009. 3

[6] H. Dutagaci, A. Godil, A. Axenopoulos, P. Daras, T. Furuya, and R. Ohbuchi. Shrec'09 track: querying with partial models. In *3DOR*, 2009. 6

[7] J. Feng, Y. Wei, L. Tao, C. Zhang, and J. Sun. Salient object detection by composition. In *ICCV*, 2011. 6

[8] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *UIST*, 2011. 1

[9] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *PAMI*, 21(5), 1999. 2

[10] H. Kamyshanska and R. Memisevic. On autoencoder scoring. 2013. 2, 5

[11] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *Computer Vision–ECCV 2014*, pages 725–739. Springer, 2014. 6

[12] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, 2011. 6

[13] Q. V. Le. Building high-level features using large scale unsupervised learning. In *ICASSP*, 2013. 2, 4

[14] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011. 2

[15] B. Leng, S. Guo, X. Zhang, and Z. Xiong. 3d object retrieval with stacked local convolutional autoencoder. *Signal Processing*, 2014. 3

[16] B. Li, A. Godil, M. Aono, X. Bai, T. Furuya, L. Li, R. J. López-Sastre, H. Johan, R. Ohbuchi, C. Redondo-Cabrera, et al. Shrec'12 track: Generic 3d shape retrieval. In *3DOR*, 2012. 6

[17] P. Luo, X. Wang, and X. Tang. Hierarchical face parsing via deep learning. In *CVPR*, 2012. 2

[18] T. Malisiewicz, A. Gupta, and A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011. 2

[19] M. Ovsjanikov, Q. Mérigot, F. Mémoli, and L. Guibas. One point isometric matching with the heat kernel. In *Computer Graphics Forum*, volume 29, 2010. 3

[20] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *ICML*, 2011. 2, 4

[21] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Shape modeling applications*, 2004. 6

[22] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Proc.* 2012. 6

[23] I. Sipiran, R. Meruane, B. Bustos, T. Schreck, H. Johan, B. Li, and Y. Lu. Shrec'13 track: large-scale partial shape retrieval using simulated range images. In *3DOR*, 2013. 6

[24] L. J. Skelly and S. Sclaroff. Improved feature descriptors for 3d surface matching. In *Optics East 2007*, 2007. 3

[25] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *ECCV*. 2014. 2

[26] J. Susskind, R. Memisevic, G. Hinton, and M. Pollefeys. Modeling the joint density of two images under a variety of transformations. In *CVPR*, 2011. 5

[27] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. 4

[28] Y. Wang, J. Feng, Z. Wu, J. Wang, and S.-F. Chang. From low-cost depth sensors to cad: Cross-domain 3d shape retrieval via regression tree fields. In *ECCV*, 2014. 2, 6, 8

[29] Y. Wang, R. Ji, and S.-F. Chang. Label propagation from imagenet to 3d point clouds. In *CVPR*, 2013. 2

[30] Z. Wu, S. Song, A. Khosla, X. Tang, and J. Xiao. 3d shapenet - a deep representation. In *CVPR*, 2015. 3, 6

[31] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud. Surface feature detection and description with applications to mesh matching. In *CVPR*, 2009. 3