Improving Style Similarity Metrics of 3D Shapes

Kapil Dev* Lancaster University
Kwang In Kim Lancaster University
Nicolas Villar Microsoft Research Cambridge
Manfred Lau† Lancaster University

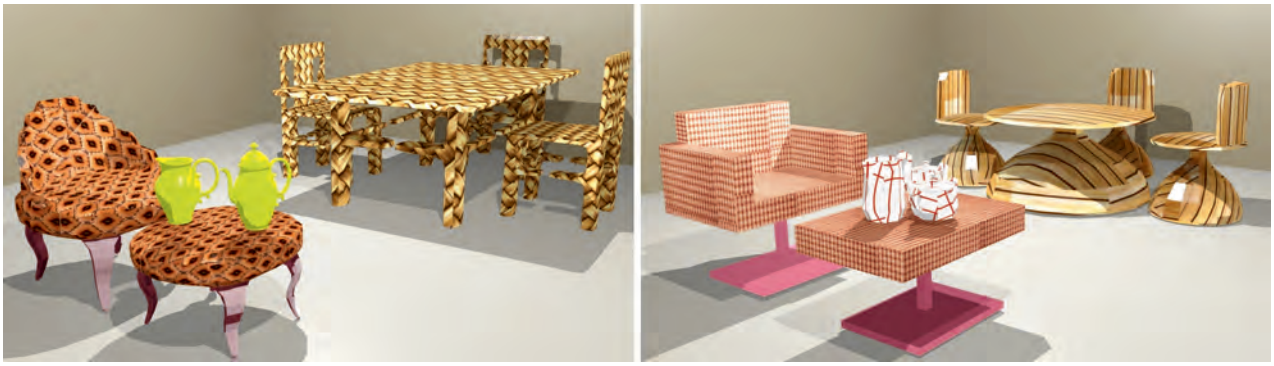

Figure 1: Online 3D shape repositories contain shapes with both color and texture data. We consider these characteristics along with geometric features to define a style metric between different 3D shapes. The images above show two examples of 3D scene composition created with our style similarity metric that considers geometry, color, and texture.

ABSTRACT

The idea of style similarity metrics has been recently developed for various media types such as 2D clip art and 3D shapes. We explore this style metric problem and improve existing style similarity metrics of 3D shapes in four novel ways. First, we consider the color and texture of 3D shapes which are important properties that have not been previously considered. Second, we explore the effect of clustering a dataset of 3D models by comparing between style metrics for individual object types and style metrics that combine clusters of object types. Third, we explore the idea of user-guided learning for this problem. Fourth, we introduce an iterative approach that can learn a metric from a general set of 3D models. We demonstrate these contributions with various classes of 3D shapes and with applications such as style-based similarity search and scene composition.

Index Terms: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric Algorithms

1 INTRODUCTION

Metric learning was introduced in the machine learning community for computing distance functions [11]. The concept of metric learning has recently been applied to various problems in computer graphics for computing distance functions that correspond to style similarity metrics for 2D clip art [7], infographics [18], and 3D shapes [13, 14]. Our work is inspired by these previous works and we focus on style similarity metrics of 3D shapes in this paper. We start with the same overall framework of computing 3D shape features, collecting human preferences of style data with crowdsourcing, and learning a style similarity distance function between pairs of 3D shapes. In contrast, we improve existing work with four novel contributions.

*e-mail: kapil.saini@hotmail.com
†e-mail:manfred.lau@gmail.com

Graphics Interface Conference 2016
1-3 June, Victoria, British Columbia, Canada

Our first contribution is in considering the color and texture of 3D shapes in addition to their geometry within the style similarity metric. To the best of our knowledge, the state-of-the-art previous works [13, 14] focus only on geometric features. We compute additional features corresponding to the color and texture of each 3D model, and study the role of these features towards the style metric in addition to geometric features such as curvatures, shape distributions, and shape diameter functions. We hypothesize that color and texture features would be dominant over geometric features. We test this hypothesis by observing the relative values of the set of learned weights that correspond to the features, and by observing the results of style-based similarity searches of 3D models with colors and textures.

The second contribution considers learning style metrics of 3D models by clustering them. If the 3D shapes represent multiple object types, we can cluster them and learn style metrics in different ways. We can have a metric for each pair of object types (e.g. chairs $\rightarrow$ tables, forks $\rightarrow$ spoons) which we hypothesize will be more accurate, but there will be N^2 metrics if there are N object types. We can also build clusters of object types (e.g. "furniture" for chairs and tables, "cutlery" for forks and spoons). We will have K^2 metrics if there are K clusters, which can be much smaller if K is much smaller than N. However, we hypothesize that these metrics will be less accurate as they combine 3D shapes of different types. While Liu et al. [13] compared between learning from triplets (i.e. data generated from queries) of two object types (e.g. $X \rightarrow Y$, and $Y \rightarrow X$) and all triplets, they did not explore further the clustering of object types that we propose. Our "clustering" is intuitive as the 3D models are typically divided into high-level clusters such as cutlery and then more specific object types such as forks, spoons, and knives.

The third contribution is to explore the idea of user-guided learning for the style metric problem. We experiment with learning both generic and user-guided metrics of style similarity. The generic metric is based on the crowdsourced style matching preferences, while a user-guided metric is based only on one user's style preferences. If a user is not satisfied with the search results from a crowdsourced metric, our interface allows the user to provide information

(e.g. re-rank the results) and create new training data for learning a user-guided style similarity metric. We hypothesize that this user-guided concept can be useful for learning personalized metrics for different users.

Our fourth contribution is to introduce an iterative approach that can learn a metric from a general set of 3D models. The motivation is that previous work constructs the crowdsourcing queries either randomly which can lead to many queries that provide irrelevant data [13], or by manually placing all 3D models into carefully-constructed groups in order to generate useful queries [14]. We thereby develop an approach where an initial set of queries is used to learn a metric, which is then iteratively used to generate further queries and metrics. We hypothesize that this iterative process can generate useful queries without tedious manual processing.

We demonstrate the above four contributions with various classes of 3D shapes (e.g. furniture, tableware, and cutlery) and build tools to show the applications of style-based similarity search and 3D scene composition. We obtain empirical results to test our hypothesis in each case. Our results will help to improve the development of style similarity metrics of 3D shapes.

1.1 Related Work

Style Similarity Metrics and Analysis. There has been a recent interest in research in style-based similarity metrics of both 2D and 3D content. The idea is to compute a style-based distance function between pairs of 2D or 3D objects. In the case of 2D content, such distance functions have been developed for 2D clip art [7], font selection [15], and infographics [18].

In the case of 3D content, the state-of-the-art methods for computing style similarity metrics of 3D shapes [13, 14] are most closely related to our work. Liu et al. [13] construct part-aware feature vectors for predicting style compatibility between 3D furniture models from different object classes. Lun et al. [14] create a style-similarity measure based on geometric elements of the 3D shapes. Our work is different in the four contributions described above. In particular, we give more details here on how data is collected in previous methods that is different from our approach. While Liu et al. [13] learn a distance metric from randomly generated crowdsourcing queries, Lun et al. [14] learn it from crowdsourcing queries that are selected from groups of 3D models that are manually pre-classified. However, if the number of 3D models is large, constructing queries (i.e. in the form of triplets of 3D models) at random can result in a large number of queries for which humans do not provide consistent responses. Such queries do not provide useful information for the learning process. On the other hand, Lun et al. [14] constructs most of the queries by first manually placing the 3D models into meaningful groups, from which more useful queries can be generated. However, a manual pre-processing of the 3D models is required. In addition, these queries typically have obvious responses such that the crowdsourcing step seems unnecessary. In this paper, we thereby take an iterative approach to learn a metric, similar to an adaptive selection method to generate queries [21]. Our iterative approach does not generate random queries (except in the first iteration) and does not require a manual pre-grouping of the 3D models.

There has also been recent work on performing style analysis in shapes. For example, Li et al. [12] perform general analysis of curve styles in a set of shapes. However, they only address curve styles along 2D silhouette profiles and do not consider structural or 3D shape styles.

3D Shape Retrieval. Our work is related to the area of 3D shape retrieval [1, 5, 9] as one of our applications is in style-based search of 3D shapes. The key difference of our work is in the "style-based" aspect.

Personalized Content Retrieval and User-Guided Learning. Personalized information retrieval [17] involves learning a user specific model of perceived relevance to present reordered search results. The concept of relevance feedback in 3D shape retrieval provides an efficient way to give a user more control over the process of retrieval and exploration of 3D shapes. For instance, Gong et al. [8] build a user-friendly interface for 3D object retrieval using relevance feedback, while Gao et al. [6] employ relevance feedback to allow users to perform active exploration of large 3D datasets. There also exists previous work in personalized image search [19, 10]. For example, CueFlik [4] allows users to re-rank search results for the problem of image search, and to create their own rules which can then be used to improve the search. Our work explores the idea of user-guided learning to learn user-guided style metrics for 3D shapes.

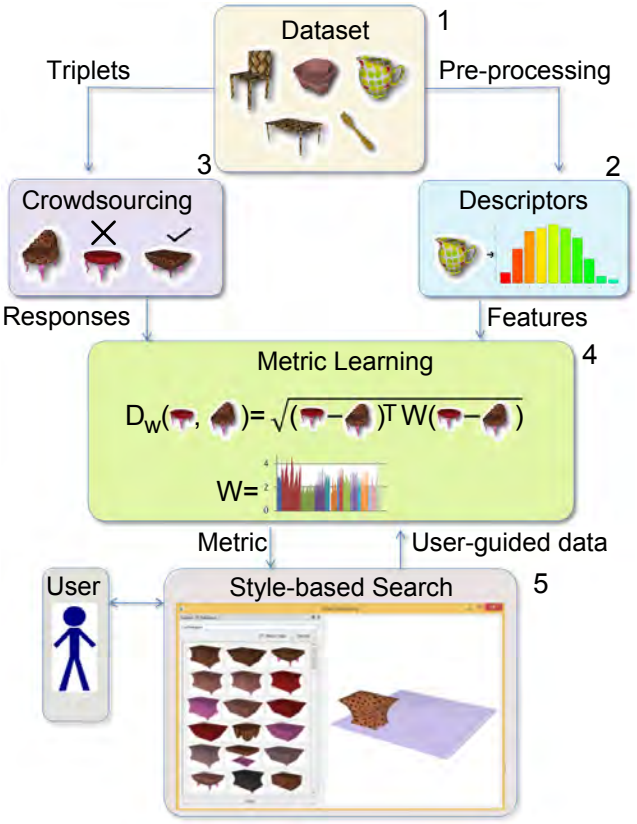

Figure 2: Overview of our framework. Our novelty is in considering color/texture features (step 2), exploring the learning of metrics with different clusters of 3D shapes (all steps), user-guided learning (steps 4 and 5), and an iterative approach (steps 3 and 4).

1.2 Overview

Figure 2 shows an overview of our approach. We collect 3D models from online sources (step 1). We then compute various shape descriptors or features (including color/texture features) for each 3D model (step 2). We generate queries containing triplets of 3D models and place them on Amazon Mechanical Turk to collect crowdsourced data regarding style preferences of the 3D shapes (step 3). The features and collected data are then used to compute a style similarity measure with an iterative approach (step 4). The style metric can be used in various applications, including style-based search of 3D models (step 5). An individual user can re-rank the models in our interface according to their style preferences, and this information can then be used to compute a user-guided style metric.

2 Dataset and Mesh Features

2.1 Dataset

We collected 3D models from the following sources: 3D Warehouse, Threeding.com, Thingiverse, Lun et al. [14], and ShapeNet [2]. The object (Figure 3) types can be categorized into "tableware" (teapots, sugar bowls, creamers), "cutlery" (knives, spoons, forks), "living" room furniture (sofas, coffee tables), and "dining" room furniture (chairs, tables).

Figure 3: Example 3D models. The rows correspond to dining room furniture, cutlery, living room furniture, and tableware.

We collected 3D models in two ways. First, we specifically build a set of shapes such that we can easily construct queries (Figure 4) where the user can specify whether his/her style preferences of 3D shapes is more dependent on geometry or color/texture. For example, in the "living" room furniture example in Figure 4, "A" matches with "B" more in its geometry aspects while "A" matches with "C" more in its color/texture. Understanding whether users prefer the geometry or color/texture aspects more is important for our analysis of style similarity. Hence we map a selected set of textures onto a set of 3D shapes. Our choice of texture images is inspired by commonly-used patterns in real life. For example, dining room furniture mainly uses wooden shades and textures while living room furniture uses fabric shades and textures. We have 17 models x 5 textures for each type of tableware (i.e. 17x5 teapots, 17x5 creamers, 17x5 sugar bowls), 21x7 for each type of cutlery (i.e. 21x7 spoons, 21x7 forks, 21x7 knives), 18x7 for each type of living room furniture (i.e. 18x7 sofas and 18x7 coffee tables), and 21x7 for each type of dining room furniture (i.e. 21x7 chairs and 21x7 tables). Second, we downloaded a general set of 3D models from the ShapeNet [2] online repository. For the "living" and "dining" categories, we have the following types and numbers of models: chairs (37), tables (35), sofas (35), and coffee tables (27). These models have much variety in both their geometry and color/texture.

2.2 Geometric Features

We compute shape descriptors on the dataset of 3D models to obtain a 2728-dimensional feature vector for each 3D model ($\mathbf{x}$). Before computing features, all models are oriented in the same direction and scaled to have similar proportions within each object type. We use an over-complete set of features and let the learning decide the relative importance of each feature.

We aim to capture both global and local shape properties. The features are not new on their own. Please refer to previous work [3, 14, 16, 20] for details of them. We compute histograms of the following (with the number of histogram bins in brackets): shape distribution (128), curvature (gauss, mean, max, min: 128 each), shape diameter (128), light field descriptor (470), voxel gradient (192), voxel gradient direction (128), silhouette centroid distances (192), silhouette Fourier descriptor (57), silhouette Zernike moments (108), silhouette D2 descriptor (192), silhouette gradient (192), silhouette gradient direction (96), and shape histogram (192). For these geometric features, there are a total of 2587 dimensions in the feature vector.

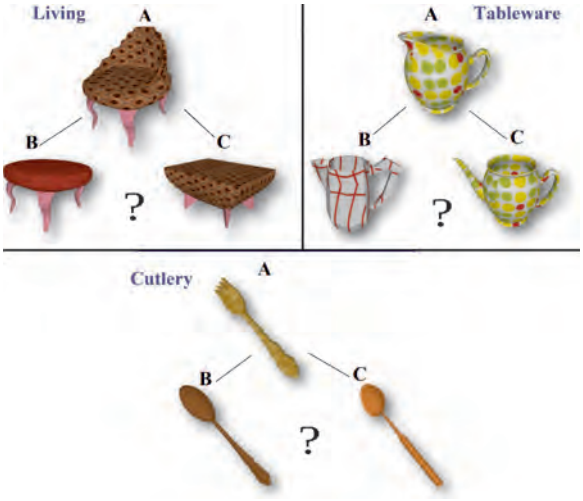

Figure 4: We specifically build parts of our dataset to test whether users prefer to match the style of 3D shapes based on geometry, color/texture, or both. We show some examples in various categories. For the "living" category, B is more similar to A than C in geometry but is less similar in color/texture. For the "tableware" category, C is more similar to A than B in both geometry and color/texture. For the "cutlery" category, both B and C are different from A in their geometry and color/texture.

The first three features above are computed on a dense uniformly-sampled version of a model's surface, which allows for avoiding artifacts caused by the triangulation of a shape. For the other features above, the model is voxelized ($300 \times 300 \times 300$), and silhouettes along the x, y, and z directions are obtained by projecting the voxel space onto the three axes respectively.

2.3 Color and Texture Features

We compute the following features (inspired by [7]) to capture color and texture properties: average HSV of the top five dominant colors (3), hue histogram (32), saturation histogram (32), value histogram (32), and local binary patterns (42). Our feature vector has a total of 141 dimensions of these color/texture features. To maintain uniformity, all texture images are resized to 512×512. A 3D shape may have more than one texture. For example, a 3D model of a chair may consist of a wooden texture for its seating area and a steel texture for its legs. We handle multiple textures by computing the same features above for each texture and combining their histograms.

3 Collecting Style Similarity Information

This section describes the process of collecting data from humans about the style similarity of 3D shapes. Since it is difficult for humans to provide absolute similarity values (for example, to provide a real number to say how stylistically similar a chair model is to a table model), we ask humans to provide relative values. We differentiate between crowdsourced data collection with many users and user-guided data collection.

3.1 Crowdsourced Data Collection

We collect data by gathering the preferences of a large number of humans by posting tasks on Amazon Mechanical Turk. This idea

is similar to previous work [7, 13, 14] and we describe our process here for completeness. The key is to collect data in the form of triplets where we have three objects (A, B, C) and A is more similar in style to B than C. To collect such triplets, we create queries where a human is presented with a 3D model of one object type X and six models of object type Y. Figure 5 shows some example queries. The task is to identify which two of the six of type Y are more similar in style to the model of type X. For each task, we get eight triplets. If we let the two preferred type Y be Y_1 and Y_2 and the rest be Y_3 to Y_6, the eight triplets are of the form $(X, Y_1, Y_3 - Y_6)$ and $(X, Y_2, Y_3 - Y_6)$.

Figure 5: Four example HIT tasks. For each task, users were asked to select two pairs of models out of the six that are more similar in style compared to the others. Users were instructed to compare the following to make their decision: number of parts and their arrangement, color, texture, dimensions of parts and the overall shape, and curviness of parts and the overall shape.

We post these tasks as HITs (Human Intelligence Tasks) on Amazon Mechanical Turk. Each HIT contains 25 tasks and we paid $0.15 for each HIT. We can choose the 3D models in these tasks manually or with an iterative approach (Section 4.2). We generate tasks with various pairs of object types, as indicated in Figure 6. Each human "Turker" is initially given written instructions and an example task with the responses (two pairs) already chosen by us. We ask Turkers to specifically pay attention to the overall shape, shape of parts, color, and texture before providing their preferences. For the crowdsourced data collection, we had 220 users and collected 48,000 triplets.

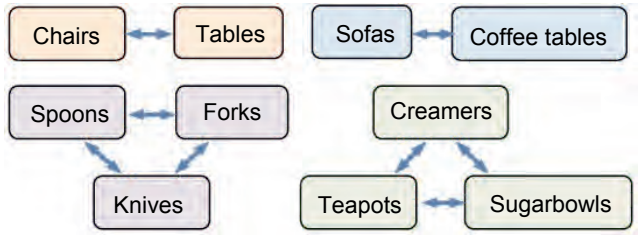

Figure 6: Object types in HIT tasks. Pairings of 3D model types for which crowdsourcing queries were generated.

3.2 User-Guided Data Collection

We also investigate to see if we can learn user-guided metrics and hence we collect data from individuals. We do not use Mechanical Turk here as it can be difficult to require a specific Turker to work on many HITs to collect the needed data (as many Turkers would do just one HIT). Hence we have users who directly use our tools in our lab to collect personalized data. We have two ways to collect such data and we use both of them and combine the data.

First, we built a tool to allow users to specify their own preferences by interactively re-ranking search results. The idea is that if a user is not satisfied with the results from the crowdsourced metric, he/she can re-rank the results to generate training data which can then be used to learn a user-guided metric. The tool (Figure 7 top) allows a user to visualize all 3D models of an object type on the left scrollable panel, where the models can be ranked according to their style similarity to a selected 3D model in the current environment on the right. A user is initially asked to perform a search with the tool using the crowdsourced metric. The user can then re-arrange the ranked results based on his/her preferences of how well they match in style with a model selected in the current environment. The user is asked to specifically place the ten closest match at the top since we use them to generate triplets data. The user interface consists of dragging and dropping the images of the 3D models interactively with the mouse to re-order them. If there is a long list of 3D models in the scrollable panel, the user can also move the mouse cursor over a 3D model and press a key on the keyboard to move it to the top of the ranking. After re-arranging the models, the user clicks a button to generate new triplets according to the ranking. The triplets are of the form (A, B, C) where A is the selected model in the environment, B is one of the top ten ranked models, and C is one of the other models (not ranked as top ten). Such triplets indicate that for the selected model A, the model B is more similar in style to it than C. This process generates $10 * (n - 10)$ triplets where n is the number of models we have for the object type being ranked.

Second, we provide another tool (Figure 7 bottom) for the user to generate triplets data. For this tool, the user can choose two object types X and Y, and the system randomly chooses one model of type X and six models of type Y. The user chooses two of the six and the system generates eight corresponding triplets (as described in the HIT tasks above).

For this user-guided data collection process, we had eight users who collected data for various object types. Each user generated just over 30,000 triplets and took about 45 minutes.

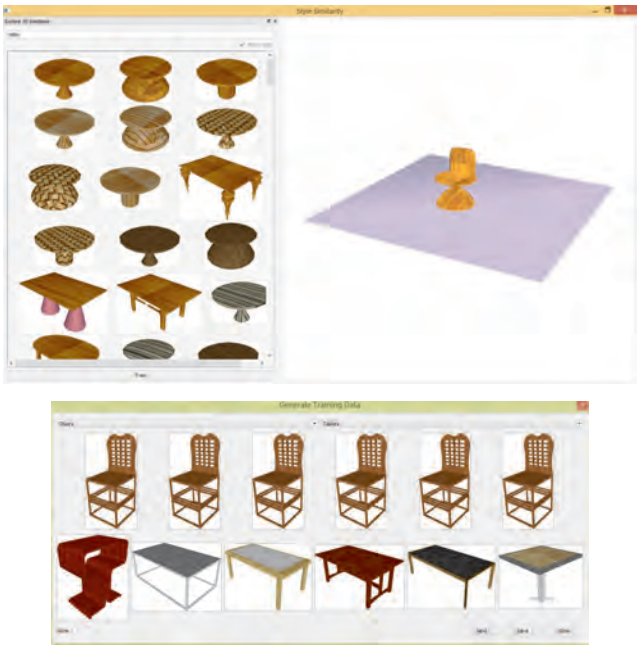

Figure 7: User-Guided Data Collection. Top: A tool for the user to specify style preferences. Right window shows a 3D environment. Left window shows a list of 3D models of an object type. This list can be ranked based on the style similarity compared to the selected model on the right. The user can interactively drag and drop these models to re-rank them to specify their own style preferences, and then the metric can be re-trained. Bottom: Another tool for the user to generate as many additional triplets as he/she wishes. It uses a format that is similar to the HIT tasks.

4 Learning Similarity Metric

In this section, we describe our framework for learning a style similarity metric with the feature vector and similarity data described in Sections 2 and 3. The framework is based on metric learning and is inspired by previous methods [7, 18]. Our work takes an iterative approach to compute a style metric.

4.1 Style Metric Computation

We use a metric learning approach to compute a distance between two 3D models based on their style similarity. Let $\mathbf{x}$ and $\mathbf{y}$ be the feature vectors for two 3D models, and we wish to compute the distance between them:

$$d_{\mathbf{W}}(\mathbf{x},\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^T \mathbf{W}(\mathbf{x}-\mathbf{y})} \quad (1)$$

The learning formulation and solution to solve for $\mathbf{W}$ is the same as in previous approaches [7, 18], and hence we do not repeat the details here but refer the reader to the previous works.

4.2 Iterative Approach

We take an iterative approach to learn a metric and the idea is to gradually build a better $\mathbf{W}$ matrix. We take the steps in Algorithm 1 for each pair of object types X and Y, with $\mathbf{x}_i$ and $\mathbf{y}_i$ denoting the feature vectors of X_i and Y_i respectively. The $random\_pick\_from(X)$ function returns a random shape from object type X, and the function $learn\_matrix\_using\_triplets(\mathbf{S})$ returns the weight matrix using the set of triplets $\mathbf{S}$ and the corresponding feature vectors.

Algorithm 1 Iterative learning algorithm

```
1:  procedure ITERATIVE–LEARN
2:  W_0 = identity matrix or random matrix
3:  for p=1:N
4:      S = ∅
5:      for q=1:M
6:          X_i = random_pick_from(X)
7:          Y_i = argmin_Y d_{W_{p-1}}(x_i, y)
8:          Y_j = random_pick_from(Y \ Y_i)
9:          S = S ∪ {(X_i, Y_i, Y_j)}
10:     W_p = learn_matrix_using_triplets(S)
11: end procedure
```

We post HITs on Amazon Mechanical Turk to collect data and learn the weight matrix in each iteration of Algorithm 1. We can either stop the iterative process after a fixed number of iterations or until the accuracy starts to decrease. We compute the prediction accuracy of a metric learned with a set of triplets by performing five-fold cross validation on them.

The reliability of Turkers was an issue when collecting crowdsourced data. For each HIT, we have 5 tasks out of 25 as control questions to check the quality of the responses. We only accept a HIT if 80% or more of the control questions match with our responses. This is similar to the idea of control questions in previous work [7], and these control questions have clear answers that are meant to check if Turkers are realistically attempting the questions or just randomly selecting answers. In each iteration, we keep reposting the rejected HITs (which can be done by new Turkers) until we get the desired number of HITs.

We noticed that the HIT rejection rate tends to be high in the initial iterations (as high as 60% in some cases). This is because the initial iterations produce essentially "random" triplets (i.e. Y_i and Y_j being random due to the initial $\mathbf{W}$). Hence it was difficult for Turkers to provide good responses without paying proper attention and many of them gave responses that seem random. As we progress towards more iterations, the learned $\mathbf{W}$ matrix becomes more effective and the triplets become less "random."

5 Results and Applications

We present the results towards each of our four contributions. We use the applications of style similarity based 3D model search and 3D scene composition to demonstrate our work.

5.1 Color and Texture Features

We learn the weight matrices for various object categories with the iterative approach. We experimented with both diagonal and full matrices and empirically observe no significant differences. Hence we choose to learn diagonal matrices and plot the log of the diagonal values (Figure 8) which correspond to the relative importance of the feature values. The plots show that color-related features consistently dominate over the geometry features, and this is true across the four categories. The other pattern we observe in these plots is that there is consistency in the 2700+ feature values again across the four categories, which shows that our method is robust. These results are observed for our data collection which includes 3D shapes we manually textured and general 3D shapes downloaded from online datasets.

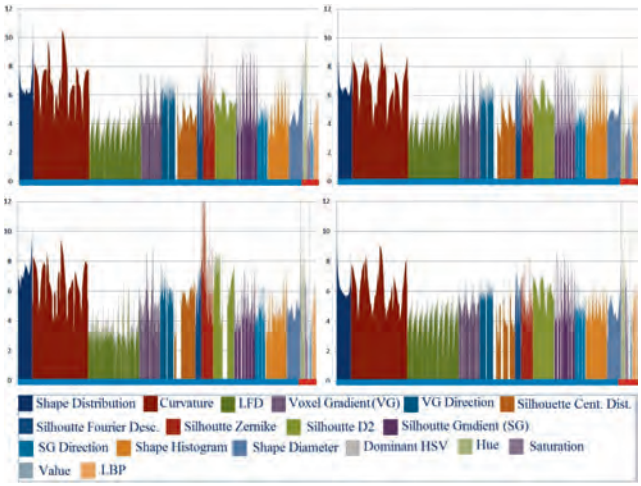

Figure 8: Log plots of the learned weights for (from top left) "dining", "living", "tableware", and "cutlery" categories. The weights correspond to features in the feature vector, in the order described in Section 2. There are 13 geometric features (blue bar on bottom of each plot) and 5 color/texture features (red bar).

Figure 9 shows the results of style similarity based search with our style metric. The top five search results for each query 3D model show that while both geometry and color/texture are important, color/texture is considered first when attempting to match style before geometry is considered. This is true across the different types of shapes that are shown, which again include 3D shapes we manually textured and general 3D shapes downloaded online. In the second row (right column) of Figure 9, the red sofa happens to match well with the query model as their curvatures are similar. In Figure 1, we use our style similarity metric with our search tool to compose 3D scenes. As the 3D models that are preferred by the crowdsourced metric are placed at the top of the search results, it is easier to find models that match in style with a selected shape.

Hence we have empirical evidence to support our hypothesis that color/texture features dominate over geometry features, in the plots of weights and in the style based search results.

5.2 Clustering of Object Types

Table 1 shows the accuracy results for different pairings of object types and clusters. We do not take the iterative approach here to ensure that the randomness does not affect the results. We instead cre-

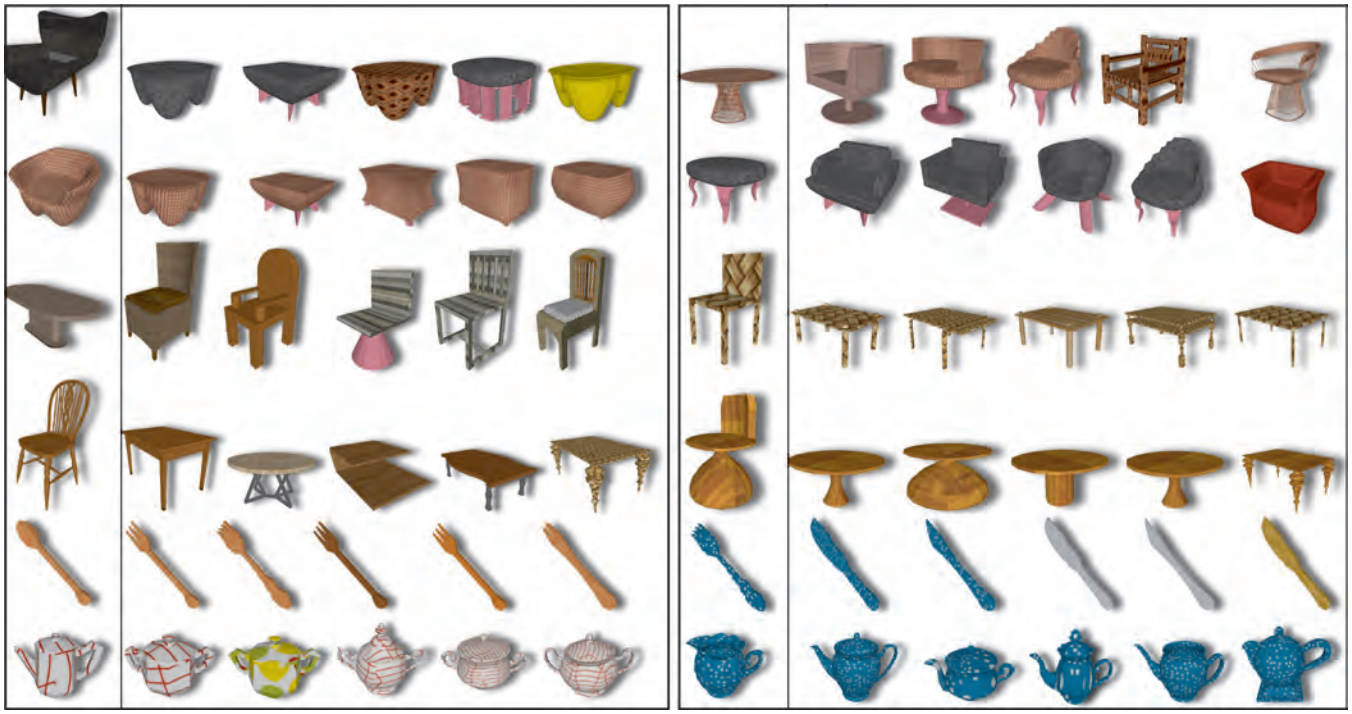

Figure 9: Style similarity based sample search results with our crowdsourced metric. There are two columns of results. First model in each column is the query model which is followed by top five models that best match in style with the query.

ated HITs manually to cover the range of 3D models in each object type. We took 5 HITs with acceptable responses (after control questions) for each pair of object types, and generated a total of 6040 triplets. For the clustering into groups, the idea is that chairs/tables can be a "furniture" cluster and forks/spoons can be a "cutlery" cluster. Since the models in our dataset have already been labeled (e.g. as chairs, forks), we manually cluster them into higher-level categories (e.g. chairs/tables is "furniture"). We combine the collected triplets from the separate types to create the triplets data for the clusters.

	Chairs	Tables	Forks	Spoons
Chairs	-	66.73	34.52	50.44
Tables	73.29	-	41.67	47.52
Forks	64.25	51.19	-	80.19
Spoons	38.87	61.17	61.38	-

	Chairs and Tables	Forks and Spoons
Chairs	66.73	42.24
Tables	73.29	42.99
Forks	61.94	80.19
Spoons	50.16	61.38

	Chairs and Tables	Forks and Spoons
Chairs and Tables	73.15	42.65
Forks and Spoons	51.83	72.21

	Chairs, Tables, Forks and Spoons
Chairs, Tables, Forks, and Spoons	56.84

Table 1: Cross-validation percentages for different pairings of object types and clusters. We learn metrics for X $\rightarrow$ Y, where X (and Y) is the type or cluster in each row (and column).

Observing the results from Table 1, we see that the percentages for some object types (e.g. chairs and tables) are comparable to the results with the iterative approach (shown below). We intentionally compared across different object types here (e.g. forks $\rightarrow$ tables) and hence some pairings give low percentages as it may be difficult to compare between some object types. This does not affect what we aim to show: the tradeoff between learning metrics for specific object types versus clusters of object types.

We observe that the percentages of the clustered pairings are somewhat averaged from the percentages of the separated pairings. We hypothesized that the clustered metrics would be less accurate, as they may be mixing object types that are quite different. However, our empirical results show no clear consensus of whether the metrics from specific object pairings or clustered pairings is better.

	Chairs and Tables	Forks and Spoons
Chairs and Tables	72.30	40.68
Forks and Spoons	52.12	71.10

	Chairs, Tables, Forks and Spoons
Chairs, Tables, Forks, and Spoons	56.36

Table 2: We randomly take half of the original triplets (compared to Table 1) in each of these five cases and re-calculate the cross-validation percentages.

Since we combine the triplets data to learn a style metric during this "clustering" process, we also tested whether the number of triplets would have been a variable that affects the percentages (i.e. more triplets data may lead to a higher percentage). Table 2 shows the results where we randomly take half of the triplets in each case and re-calculate the percentage. These results show that the number of triplets does not affect the percentage.

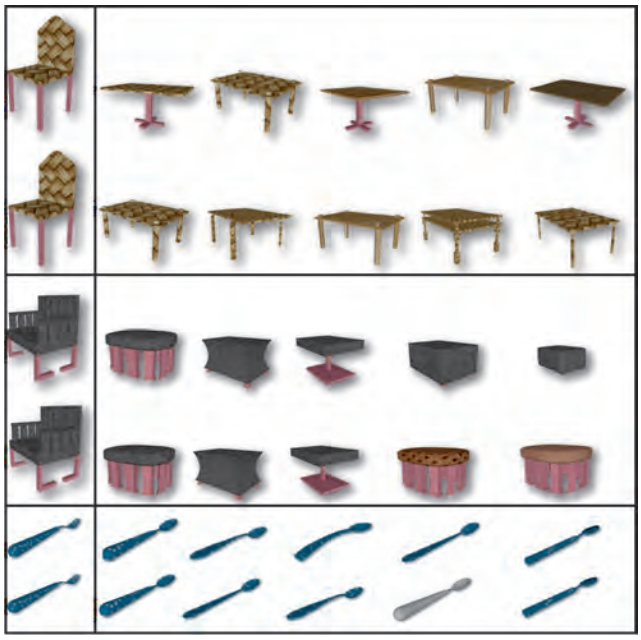

Figure 10: Comparison between crowdsourced (first row in each case) and user-guided (second row in each case) results for "dining", "living", and "cutlery" categories. Each row first shows the query model and then the top five ranked models from style similarity based search.

5.3 User-Guided Style Similarity Metrics

Figure 10 shows that the user-guided results are interestingly somewhat different from the crowdsourced results. For example, the color and shape of the legs of the furniture pieces are different between the two results. For the "cutlery" example, the crowdsourced results mainly match with the color/texture features, while the user-guided results include a spoon that has very similar geometry (i.e. round-shaped handle) but different color/texture. The individual user in this case was attentive to the geometry of the cutlery in addition to their color/texture. These results demonstrate that we can learn a style metric for individual users that is different from the crowdsourced metric, providing evidence for our hypothesis that the user-guided concept can be beneficial in some cases. However, since the overall color and shape preferences among different people are still mostly consistent, the differences in the user-guided metrics may only be subtle.

5.4 Iterative Learning Approach

For each category of shapes, we started with posting randomly generated triplets. We collected the same number of triplets for each iteration, and we have between 1600 and 2000 of triplets for each of our four object categories (in each iteration). The accuracies achieved with the metric learned on such triplets in each iteration are shown in Figure 11. As the first iteration represents the non-iterative method since it is the same as randomly generating triplets as done in previous work, we can compare between the percentages for the non-iterative method and the iterative method (our last iteration). The percentages increased from 69% to 87% for "dining", from 65% to 85% for "living", from 57% to 82% for "cutlery", and from 49% to 86% for "tableware". These results support our hypothesis that the iterative process can generate useful HITs, and can avoid having to randomly generate triplets [13] or to manually group the 3D models in advance [14]. We found that stopping after a fixed number of three iterations worked well, and this was consistent across the four object categories.

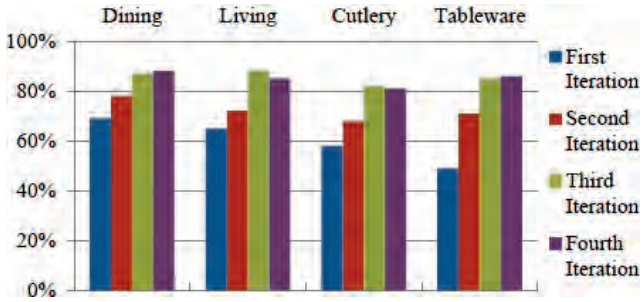

Figure 11: Cross-validation percentages for the iterative learning for "dining" (chairs→tables), "living" (sofas→coffee tables), "cutlery" (spoons→forks), and "tableware" (teapots→sugar bowls). First bar (blue) in each category shows the accuracy of the metric learned on randomly generated triplets.

6 Discussion, Limitations, and Future Work

Crowdsourcing and learning approaches have recently been successfully applied to various computer graphics problems. In particular, this approach has been applied to compute a style similarity metric for 3D models. In this paper, we further explore this problem and provide four new contributions.

In addition to the color and texture features, other properties such as construction material (e.g. glass or metal) and how textures are mapped to 3D shapes (not just the texture image) can be taken into consideration in future work when computing features.

Our "color/texture" contribution may be slightly biased to our dataset as the models were semi-manually textured. Since the variability in textures in large online datasets is different, valuable insights might be obtained in potential future work with datasets containing more variation in color, texture, and geometry.

The results from our clustering experiments are different from the results presented in Liu et al. [13] although they do not cluster the object types as we do. In their "all triplets" scenario where triplets of different pairs of object types are combined, their results show a better percentage accuracy compared to when the triplets of different objects types are separately trained to form metrics. In the case where we combined the four object types in our experiments, our results show a similar percentage accuracy compared to metrics computed with separate object types. Future work can further investigate the reason for this discrepancy. A larger number of object types may give more insightful results.

One limitation to the current learning method is that the distance function is a simple Euclidean distance metric. Hence the learned style metrics are limited in their expressive power, and more complex non-linear functions can allow the metrics to better represent human preferences.

7 Acknowledgments

We would like to thank the reviewers for their constructive comments. Kapil Dev is funded by the Microsoft Research PhD program. Kwang In Kim thanks EPSRC EP/M00533X/1.

References

[1] B. Bustos, D. A. Keim, D. Saupe, T. Schreck, and D. V. Vranić. Feature-based similarity search in 3d object databases. *ACM Computing Surveys (CSUR)*, 37(4):345–387, 2005.

[2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] 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, pages 223–232. Wiley Online Library, 2003.

[4] J. Fogarty, D. Tan, A. Kapoor, and S. Winder. Cueflik: interactive concept learning in image search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 29–38. ACM, 2008.

[5] T. Funkhouser and M. Kazhdan. Shape-based retrieval and analysis of 3d models. In *ACM SIGGRAPH 2004 Course Notes, Article No. 16*. ACM, 2004.

[6] L. Gao, Y.-P. Cao, Y.-K. Lai, H.-Z. Huang, L. Kobbelt, and S.-M. Hu. Active exploration of large 3d model repositories. *IEEE Transactions on Visualization and Computer Graphics*, 21(12):1390–1402, 2015.

[7] E. Garces, A. Agarwala, D. Gutierrez, and A. Hertzmann. A similarity measure for illustration style. *ACM Transactions on Graphics (TOG)*, 33(4):93, 2014.

[8] B. Gong, J. Liu, X. Wang, and X. Tang. Learning semantic signatures for 3d object retrieval. *IEEE Transactions on Multimedia*, 15(2):369–377, 2013.

[9] N. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, and K. Ramani. Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design*, 37(5):509–530, 2005.

[10] A. Kovashka and K. Grauman. Attribute adaptation for personalized image search. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3432–3439, 2013.

[11] B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2012.

[12] H. Li, H. Zhang, Y. Wang, J. Cao, A. Shamir, and D. Cohen-Or. Curve style analysis in a set of shapes. In *Computer Graphics Forum*, volume 32, pages 77–88. Wiley Online Library, 2013.

[13] T. Liu, A. Hertzmann, W. Li, and T. Funkhouser. Style compatibility for 3d furniture models. *ACM Transactions on Graphics (TOG)*, 34(4):85, 2015.

[14] Z. Lun, E. Kalogerakis, and A. Sheffer. Elements of style: learning perceptual shape style similarity. *ACM Transactions on Graphics (TOG)*, 34(4):84, 2015.

[15] P. O'Donovan, J. Lībeks, A. Agarwala, and A. Hertzmann. Exploratory font selection using crowdsourced attributes. *ACM Transactions on Graphics (TOG)*, 33(4):92, 2014.

[16] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3d models with shape distributions. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI 2001)*, pages 154–166. IEEE, 2001.

[17] G. Pasi. Issues in personalizing information retrieval. *IEEE Intelligent Informatics Bulletin*, 11(1):3–7, 2010.

[18] B. Saleh, M. Dontcheva, A. Hertzmann, and Z. Liu. Learning style similarity for searching infographics. In *Proceedings of the 41st Graphics Interface Conference*, pages 59–64. Canadian Information Processing Society, 2015.

[19] J. Sang, C. Xu, and D. Lu. Learn to personalized image search from the photo sharing websites. *IEEE Transactions on Multimedia*, 14(4):963–974, 2012.

[20] T. Surazhsky, E. Magid, O. Soldea, G. Elbe, and E. Rivlin. A comparison of gaussian and mean curvatures estimation methods on triangular meshes. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'03)*, volume 1, pages 1021–1026. IEEE, 2003.

[21] O. Tamuz, C. Liu, O. Shamir, A. Kalai, and S. J. Belongie. Adaptively learning the crowd kernel. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 673–680, 2011.