Shape-Based Autotagging of 3D Models for Retrieval

Ryutarou Ohbuchi, Shun Kawamura

Graduate School of Medicine and Engineering, University of Yamanashi,
4-30 Takeda, Kofu-shi, Yamanashi-ken, 400-8511, Japan
ohbuchiAT yamanashi.ac.jp, g09mk007AT yamanashi.ac.jp

Abstract. This paper describes an automatic annotation, or autotagging, algorithm that attaches textual tags to 3D models based on their shape and semantic classes. The proposed method employs *Manifold Ranking* by Zhou et al, an algorithm that takes into account both local and global distributions of feature points, for tag relevance computation. Using Manifold Ranking, our method propagates multiple tags attached to a training subset of models in a database to the other tag-less models. After the relevance values for multiple tags are computed for tag-less points, the method selects, based on the distribution of feature points for each tag, the threshold at which the tag is selected or discarded for the points. Experimental evaluation of the method using a text-based 3D model retrieval setting showed that the proposed method is effective in autotagging 3D shape models.

Keywords: 3D geometric modeling, content based retrieval, automatic annotation, manifold ranking, text tags, semantic retrieval.

1 Introduction

Specification of a query is one of the most fundamental issues in retrieving multimedia objects such as images and 3D geometric models. These data objects may be searched either by example(s) or by text(s), each approach having its own advantages and disadvantages. To retrieve 3D models, a query for the *Query-By-Example* (QBE) approach could be sketches in either 2D [12] or 3D of the desired shape, an example 3D model, or a photograph of a real-world object having the desired shape. Most of the published work on 3D model retrieval used the QBE approach [2, 5, 6, 7, 9, 10, 12, 14]. While the QBE approach is quite powerful and useful in many application scenarios, it does have drawbacks. For example, it is often impractical to find a 3D model that have a shape similar enough to the desired one. Or, a user may find sketching a complex shape difficult. Capturing semantic aspect of the desired shape is also difficult with the QBE approach. Previous work used either single-class learning via relevance feedback [6, 7] or off-line multi-class learning [10] for semantics.

An alternative, *Query-By-Text* (QBT) approach, employs a text as a query, as in text-based search engines. While the QBT is disadvantageous in directly specifying a

desired shape, it has an advantage in describing semantic content of the desired shape. For example, a word "mammal" could be used to retrieve both dolphin and cat that have very different geometric shapes. The issue in QBT of 3D models is that most of the 3D models are not associated with text tags, and that adding consistent tags manually to a large number of 3D models can be quite difficult. An automatic or semi-automatic approach, e.g., propagation of tags from a set of tagged models to a large number of tag-less models, is necessary.

The authors know of only two examples, one by Zhang et al [16] and the other by Goldfeder et al [3] that studied autotagging of 3D models. The method by Zhang et al [16] associates a list of attributes, not text tags, with a 3D model. The method employs biased kernel regression to propagate probabilities of the attributes from the manually tagged source models to the tag-less models. For retrieval, the method treats the list of probabilities as a feature vector, and computes distances among feature vectors. Thus, it is not strictly a QBT method. The method by Goldfeder et al [3] is a QBT approach, in which text tags are propagated from tagged 3D models to tag-less 3D models. To propagate tags, it uses local structure of the shape feature space via nearest neighbor search. For experiments, they used a set of 192,343 mesh-based 3D models having text tags in a snapshot of *Google 3D Warehouse* (*G3W*). The G3W is an evolving set of a large number of 3D models contributed by users, and it contains errors and inconsistencies in its tags. Unfortunately, the G3W snapshot they have used is not available for use at this time. Goldfeder et al evaluated tagging performance of their method by using a retrieval task, by comparing retrieval performance of their auto-tagging based QBT algorithm with that of a QBE algorithm that uses 3D models as its queries. The paper [3] reported that the QBT performed equal or better than the QBE algorithm they have compared against.

In this paper, we propose a novel off-line semi-supervised algorithm for autotagging 3D model collections that exploits both *local and global* distribution of tagged and tag-less shape features by means of the *Manifold Ranking* (MR) algorithm by Zhou, et al. [17, 18]. From a set of shape features extracted from a set of tagged 3D models that shares a same text tag, the MR algorithm diffuses a relevance rank value indicating the likelihood of tag-less models to have the tag. This MR process is performed once for each of keyword to determine a set of tags a tag-less model should receive. To discard an unreliable tag attached to the tag-less model, compactness of the distribution of the tagged models sharing the tag is taken into consideration. An evaluation of the proposed algorithm by using a 3D model retrieval scenario showed that a QBT retrieval method by using the text tags added by the method significantly outperforms a QBE retrieval method that used 3D shape feature comparison

2 Autotagging 3D Models via Multiple Manifold Ranking

Our algorithm, whose outline is shown in Figure 1, consists of the following three steps. Details of the steps will be described in the following subsections.

(1) **Shape feature extraction:** Extract a shape feature from the input 3D model.

(2) **Tag propagation:** For each tag, propagate, in the shape feature space, the likelihood of a 3D model having the tag from the tagged "training example" 3D models to the tag-less models. The tag likelihood propagation is performed by using the Manifold Ranking (MR) algorithm by Zhou, et al [17, 18]. Repeat the MR-based tag likelihood propagation for all the tags.
(3) **Tag selection:** A tag-less model now has multiple likelihood values for multiple tags. Select, based on the mutual distance of the tagged and tag-less models, the most likely tags to be attached to the model.

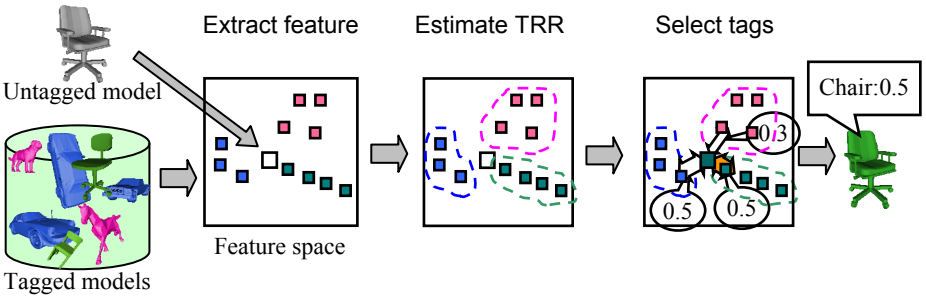

Fig. 1. The method estimates a *Tag Relevance Rank* (TRR), a likelihood of a model having a text tag, by Manifold Ranking [17, 18]. Estimation of TRR is performed for every tag. Then most relevant of the tags are selected for a 3D model by using tag-specific threshold values.

2.1 Feature extraction

So far as a shape descriptor, or a *feature*, of a 3D model is a vector, it can be used in the proposed autotagging algorithm. A feature for comparing 3D shapes is required to have a set of invariances. Typically, certain invariance to shape representation, invariance to geometrical transformation up to similarity transformation, and invariance to geometrical and topological noise are expected. One might need to compare a CAD model defined as a 3D solid by using curved surface patches with a polygon soup model or a point set model. Invariance against three rotational degrees of freedom is not trivial to achieve [14]. Topological and geometrical noises, such as holes, variation in vertex connectivity in a mesh, displacements of vertices, need to be tolerated. Depending on an application, additional invariance, such as invariance against joint articulation, may be required.

For the experiment described in this paper, we used our extension of Wahl's *Surflet Pair Relation Histograms* (*SPRH*) [15], a shape feature that has nice invariance properties, readily available, and has reasonable retrieval performance. The Windows 32bit executable of our extension of the SPRH is available at our web site [8]. The SPRH was originally developed for oriented point set models acquired by Laser range scanners. We added a step to convert a surface based model to an oriented point-set model by using *quasi*-Monte Carlo sampling of surfaces [10]. As a result, the extended SPRH is able to accept popular surface-based shape representations such as polygon soup and closed polygonal mesh. The SPRH feature is a joint histogram of

quadruples (δ, α, β, γ), in which δ is the distance and α, β, and γ are the angles between a pair of oriented points. By default, each of the four quantities has 5 bins, making a SPRH feature a $5^4 = 625D$ vector. The SPRH is invariant to similarity transformation without requiring pose normalization. It is also insensitive to various noises, such as variation in mesh tessellation or connectivity, holes, or small vertex displacements.

2.2 Tag propagation and selection

Our method uses the *Manifold Ranking* (MR) [17, 18] algorithm to estimate the relevance of a text tag to tag-less 3D models. The MR algorithm is a graph-based learning algorithm that can be used either in unsupervised, supervised, or semi-supervised modes. Our method uses the MR in semi-supervised mode. Intuitively, the MR resembles to solving a diffusion equation on an irregular connectivity mesh in a high-dimensional feature space. The mesh is created by connecting, by their proximity, the feature points in the feature space. For example, given the 625D feature of the SPRH, the mesh is embedded in a 625D space with its feature points having 625D coordinate. In our autotagging framework, the mesh for the MR is created from union of feature vectors of the tagged as well as tag-less 3D models. Typically, a text tag is associated with multiple tagged (or "source") 3D models so that a MR process would have multiple sources for diffusion. Given *N* text tags, or keywords, the proposed algorithms runs *N* such MR processes to diffuse *Tag Relevance Rank* (*TRR*) value for the *N* tags onto all the tag-less 3D models. At the equilibrium of solving the diffusion equation iteratively, the higher the diffused TRR value, the higher the likelihood of the tag-less model having the tag.

Let us assume propagation of a tag from a set of tagged 3D models to the other 3D models without the tag. Let $\chi = \{x_1, \cdots, x_s, x_{s+1}, \cdots, x_t, x_{t+1}, \cdots, x_n\}$ be a set of n features in a m-dimensional feature space $\mathbb{R}^m$, in which first $s+t$ points from x_1 to x_t are the feature points for the tagged models. Among the tagged points, points from x_1 to x_t are the *source set*, and the point from x_{s+1} to x_t are the *probe set*. This splitting of the tagged points is done to estimate a TRR threshold value *th*, which decides if a tag should be attached to a tag-less point or not. The source set and the probe set is of equal size, and are drawn randomly from the set of $s+t$ points sharing the same tag. To put it briefly, by performing TRR diffusion within a set of models sharing a tag, the tightness of the distribution of the tag is estimated to compute the *th*. Details on tag selection will be explained below. The rest of the points from x_{t+1} to x_n are the tag-less ones for which the TRR values need be computed. Let $d : \chi \times \chi \to \mathbb{R}$ denote a distance metric on χ, e.g., *L1-norm* or *Cosine* distance, that assigns a pair of points x_i and x_i a distance $d(x_i, x_j)$. Let $f : \chi \to \mathbb{R}$ be a ranking function that assigns each x_i a ranking score f_i, thereby forming a rank vector $\mathbf{f} = [f_1, \cdots, f_n]^T$. Let the n-dimensional binary-valued vector $\mathbf{y} = [y_1, \cdots, y_n]^T$ be a label vector, in which $y_i = 1$ for the tagged ("source") point and $y_i = 0$ for the rest, i.e., points to which tags are assigned.

We first create the affinity matrix $\mathbf{W}$ where $\mathbf{W}_{ij}$ indicates the similarity between samples x_i and x_j

$$\mathbf{W}_{ij} = \begin{cases} \exp\left(-\dfrac{d(x_i, x_j)}{\sigma}\right) & if \quad i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The distance metric $d(x_i, x_i)$ used in forming $\mathbf{W}$ affects ranking performance. We will compare several distance measures for their performance in Section 3.1. The positive parameter σ defines the radius of influence. Note that $\mathbf{W}_{ii} = 0$ since there is no ark connecting a point with itself. The matrix $\mathbf{W}$ is positive symmetric. A normalized *graph Laplacian* $\mathbf{S}$ is then defined as;

$$\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}} \tag{2}$$

where $\mathbf{D}$ is a diagonal matrix in which $\mathbf{D}_{ij}$ equals to sum of *i*-th row of $\mathbf{W}$, that is, $\mathbf{D}_{ij} = \sum_j \mathbf{W}_{ij}$ Then the ranking vector $\mathbf{f} = [f_1, \cdots, f_n]^T$ can be estimated by iterating the following until convergence;

$$\mathbf{f}^{(t+1)} = \alpha\mathbf{S}\mathbf{f}^{(t)} + (1-\alpha)\mathbf{y} \tag{3}$$

As the equilibrium is reached, each tag-less point would have a TRR value associated with them, indicating the likelihood of the point having the tag. We call the ranking vector at the equilibrium $\mathbf{f}^*$.

The algorithm performs the tag propagation process describe above *n* times, once for each of the *n* tags. After all the tags have been diffused, a point in the feature space would have *n* positive TRR values corresponding to the *n* tags. A TRR value f_i indicates how relevant the tag is to the point *i*. The system must now decide what subset of the *n* tags should be attached to the point. To decide, our method employs a simple proving to estimate the cut-off threshold *th* of the TRR for each tag. As described above, the method randomly splits the set of $s+t$ tagged points sharing the same tag into two (roughly) equal-sized subsets, *the source set* and *the probe set*. After the MR, the TRR values of the points in the probe set, that is, the points from x_{s+1} to x_t in the set of features χ, are used to estimate a threshold *th* by using the following equation;

$$th = \min_{s+1 \leq i \leq t}(f_i) \tag{4}$$

The threshold *th* is computed for each tag. If the TRR $f_i \geq th$ at a tag-less point, the tag *i* is attached to the point. The tag is also given the TRR value f_i as its confidence value at the point. In general, a point may be associated with multiple tags if more than one tag has *TRR* values greater than or equal to their respective threshold values. As a reviewer pointed out, the use of minimum to compute the threshold *th* is debatable, as it is not robust against outliers. However, given small size of some of the classes in the benchmark dataset, e.g., just 4 models per class and 2 models in its prove set, we could not come up with a better method that is statistically robust.

In the example of Figure 2, tagged points belonging to the tag "car" has the tightest distribution, with the similarity threshold *th*=0.8. The TRR of the "car" tag at the tag-less point in question is 0.5, which is below the threshold *th*=0.8. Thus, the "car" tag is not attached to the point. Similarly, the "animal" tag is not attached to the point. The "chair" tag, on the other hand, has the *TRR*=0.5, while the threshold for the "chair" tag is *th*=0.4. Thus, the point is associated with the tag "chair".

3. Experiments and Results

As the research on autotagging, or automatic annotation of 3D models just started, there is no established benchmark for an objective evaluation. Ideally, one should use a large enough corpus, something similar to the snapshot of G3W containing 190k models used by Goldfeder et al, for tagging. Unfortunately, the G3W snapshot used by Goldfeder et al is not available for us. In addition to the database, one also needs a method to evaluate the tagging performance numerically and objectively.

We resorted to using a well accepted benchmark for QBE 3D model retrieval, the *Princeton Shape Benchmark* (PSB) [13] for our evaluation. The 1,814 model PSB contains two equal-sized subsets, the *train set* and *test set*, each consisting of 907 models. We regard class labels of the PSB train set as text tags, and transfer the tags from the train set to the test set. We then regarded the retrieval performance derived under the QBT framework as the performance index of the autotagging method.

The PSB has a 4-level class hierarchy, from the most detailed "Base" level classes to the most abstract "Coarse 3" level classes containing only two classes; "natural" and "manmade". Of these four levels, we use the Base level classes only in the following experiments. The Base level class itself has four internal levels of hierarchy. If leaf nodes of the Base level are counted, there are about 90 classes in the Base level of both train and test set. Note that some of the classes in the Base class of the train set do not have their counterparts in the Base level of the test set, and vice versa. If both leaf and non-leaf nodes of the Base level classes are counted there are about 130 classes. (To be exact, there are 129 classes in the train set and 131 classes in the test

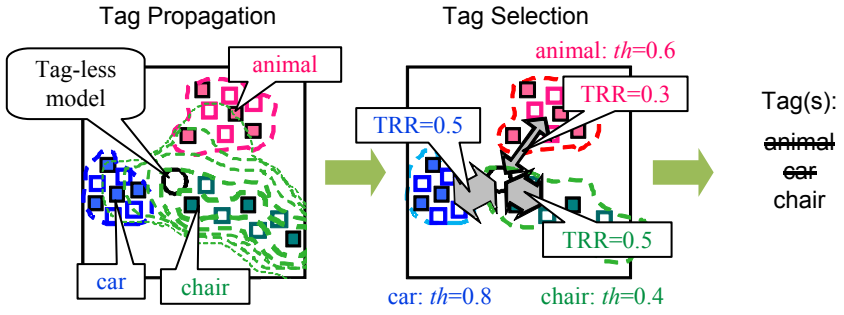

Fig. 2. A tag is attached to a feature if its *Tag Relevance Rank* (*TRR*) propagated from features with the tag exceeds the threshold *th* specific to the tag. The threshold is the estimated minimum TRR of the *prove set* of features having the tag.

set.) Note also that the size of a class at the leaf-node of the Base class can be quite small, ranging from just four per class to a couple of dozen per class.

As the feature vector, we used the SPRH by Wahl [15] modified to accept polygon soup and polygonal mesh models found in the PSB. Note that the performance of the SPRH is only modest by current state of the art. Here we do not aim for the highest retrieval performance, but we want to evaluate our autotagging algorithm by using a QBT framework.

3. 1. Distance measures for similarity matrix

This set of experiments compares, using the retrieval performance, the autotagging performance of various distance measures used in forming the affinity matrix **W** in equation (1).

For this experiment, we used 63 class labels that are common among the train and test sets of the PSB. These 63 classes are selected from the Base level, and include both leaf and non-leaf nodes in the Base level. These 63 classes correspond to 811 models in the train set and 823 models in the test set. We used the 3D models in the train set as the tagged models, and propagated their class labels to the models in the test set.

We compared five distances measures, that are, L^k-norm having k=0.5, 1.0, and 2.0, the *cosine* (*cos*) measure, and the *Kullback-Leibler divergence* (*KLD*). In the following equations, $\mathbf{x} = (x_i)$ and $\mathbf{y} = (y_i)$ are the feature vectors and n is the dimension of the vectors. The L^k-norm is defined by the following equation;

$$d_k(\mathbf{x},\mathbf{y}) = \left[\sum_i^n \left(\|x_i - y_i\|^k\right)\right]^{1/k} \quad (5)$$

The original paper on MR used the Euclidian distance, or L^2-norm with k=2.0, to form the affinity matrix [17, 18]. The Manhattan distance with k=1.0 is often said to perform better in higher dimension. Aggarwal, et al in [1] argued that k<1.0, e.g., k=0.5, works even better in higher dimension than k=1.0. As the cosine measure is a measure of similarity having the range $[0,1]$, we converted it to a distance using the following equation;

$$d_{\cos}(\mathbf{x},\mathbf{y}) = 1 - \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\cdot\|\mathbf{y}\|} \quad (6)$$

The KLD is sometimes referred to as *information divergence*, or *relative entropy*, and is not a distance *metric*, for it is not symmetric. We use a version of KLD that is symmetric as below;

$$d_{KLD}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} (y_i - x_i) \ln \frac{y_i}{x_i} \quad (7)$$

Figure 3 shows the retrieval performance in R-precision of various distance measures $d(x_i, x_j)$ used in forming the affinity matrix **W** of equation (1). Both the $L^{0.5}$-norm and the KLD performed equally well, finishing at the top. The L^2-norm employed in the original paper on manifold ranking [17, 18] finished the last in this case. The result will likely be different for other features and databases having different distribution of feature points. However, it is definitely worth experimenting with several different distance measures. We will use the $L^{0.5}$-norm for the experiments that follow.

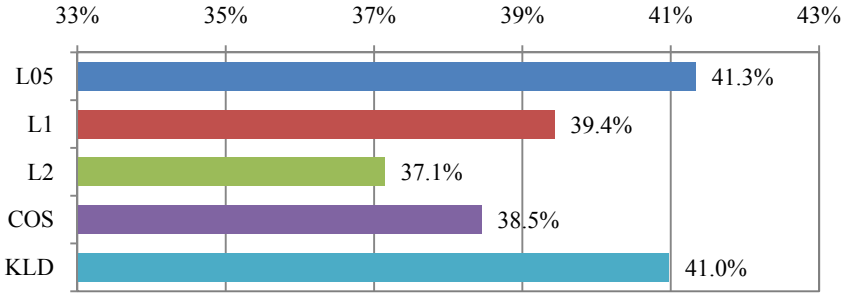

Fig. 3. Distance measures used in forming the affinity matrix **W** (Equation (1)) and retrieval performance measured in R-precision.

3.2. Query By Text v.s. Query By (3D model) Example

In this set of experiments, in an effort to quantify autotagging performance, retrieval performances of the QBT and QBE frameworks are compared. The QBE retrieval and the autotagging step for the QBT used the same shape feature, the SPRH modified by us to accept polygon-based models. As tags, we used 21 class labels of 21 leaf nodes in the Base level that are common among the PSB train and test set. This is different from the 63 class labels using in the previous experiment, which included both leaf and non-leaf nodes in the Base level of the PSB. This experiment used the leaf node only since performance evaluation in QBE framework could only use the leaf-nodes as its query. That is, it is not possible for the QBE framework to query by using a "super shape" corresponding to a super class.

Figure 4 shows the recall-precision plots for the QBT and QBE frameworks. The QBT outperformed the QBE almost everywhere except at the lowest recall. The QBE won at the lowest recall since the top ranked retrieval in the QBE is always correct, as it is the query itself. For the QBT retrieval, however, the top-ranked result may be wrong if the tagging step made mistakes.

Figure 5 compares the retrieval performance of the QBT and the QBE for when only leaf classes in the Base level are considered (case "leaf") and both leaf classes and non-leaf classes in the Base level are considered (case "all"). The R-precision values are averaged over the classes and models used in each experiment. R-precision for both QBT and QBE dropped for when non-leaf classes are included in the queries. Still, the QBT performed better than the QBE in both cases.

We observe that that the QBT retrieval performance of average R-precision=55% is on a par with some of the state-of-the-art QBE methods evaluated by using the same database but with *different* set of classes. (For reference, the *Light Field Descriptor* (LFD) by Chen et al. [2] has average R-precision=45% if evaluated by using all the 93 leaf node classes of the Base level PSB test set.) Such an apparently good performance of the QBT may be explained as follows. The proposed algorithm adds a tag by propagating TRR from multiple feature points sharing the tag.

This resembles to the case in which the MR is used in a relevance-feedback based 3D model retrieval system [11]. The system described in [11] produced a high retrieval performance after multiple 3D models are marked as relevant after a few relevance feedback iterations.

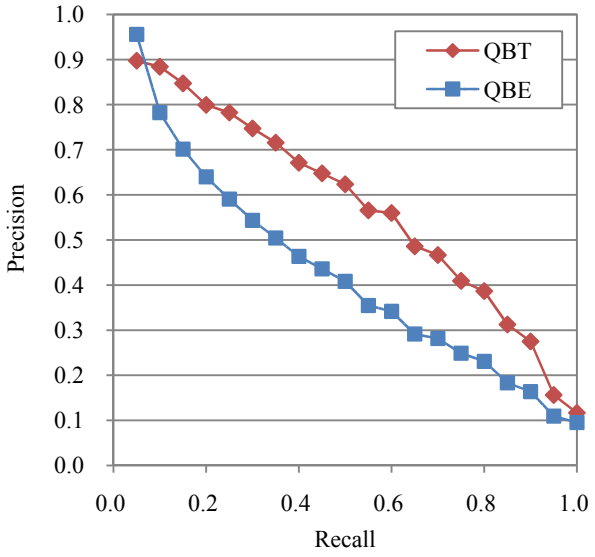

Fig. 4. Recall-precision plot for the QBE and QBT retrieval experiments. Overall, the QBT using the automatically added tags significantly outperformed the QBE using the shape feature.

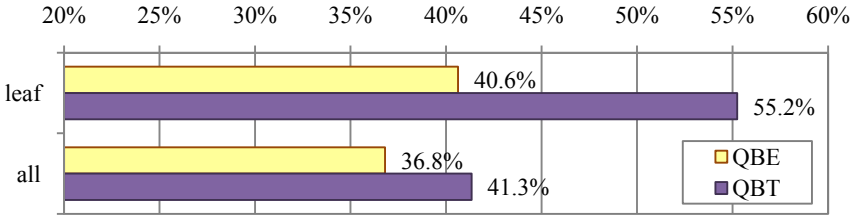

Fig. 5. The QBT based on the autotagging performed significantly better than the QBE for both the "leaf" case, which included only leaf classes in the Base level PSB, and the "all" case, which included both leaf and non-leaf classes in the Base level PSB.

3.3. Tagging Examples

Figure 6, Figure 7, and Figure 8 show examples of tags attached to models by using the proposed algorithm. Figure 6, Figure 7, and Figure 8 show, respectively, the successful, unsuccessful, and mixed cases. The tags in red are correct tags, while those in blue are incorrect tags. The size of the letters indicate the estimated quality of the tag; the larger the letter, the more confident the system is about the tag.

Successful examples in Figure 6 are associated with both correct and incorrect tags. However, the incorrect tags should matter little in practice as they have low confidence values. Unsuccessful examples, e.g., a balloon (m1345) mistakenly tagged as "body part" and "head", seem to suggest the need for a better shape feature as well as more training examples. For example, a multiresolution feature acquisition method similar to the one used in [10] might help.

4. Conclusion and Future Work

In this paper, we proposed and evaluated a shape-based algorithm for automatic annotation, or *autotagging*, of 3D models with text keywords by learning the tags from a corpus of tagged 3D models. The algorithm first extracts shape features from 3D models both with tags (training corpus) and without tags (autotagging target.) The algorithm then computes, for each tag, the relevance of each 3D model to the tag. To compute the relevance, the algorithm takes into account distribution of shape features in the ambient feature space with both local and global consistency by using the Manifold Ranking (MR) algorithm of Zhou, et al [17, 18]. The algorithm chooses those tags that are reasonably high in confidence based on the mutual distance of tagged models sharing tag.

We evaluated the autotagging algorithm by using a 3D model retrieval scenario, assuming a good tagging result should produce a good retrieval performance by using the Query By Text (QBT) framework. Our experiment showed that the QBT retrieval that employed the automatically added tag significantly outperformed the Query By Example (QBE) retrieval that employed query 3D models and their shape features.

In the future, we definitely would like to employ more "realistic" corpus, e.g., larger size, presence of noise and error, etc., to evaluate the proposed algorithm. To this end, availability of a snapshot of the G3W will greatly benefit the research community. Assuming the availability of a large corpus, we need to improve the scalability of the MR algorithm so that it could handle a large number of features. We also would like to enhance the method, for example, by using a feature refined by using other learning based algorithms, or by using a combination of multiple shape features.

Acknowledgements

This research has been funded in part by the *Ministry of Education, Culture, Sports, Sciences, and Technology of Japan* (No. 18300068).

References

1. C. C. Aggarwal, A. Hinneburg, and D. A. Keim, On the Surprising Behavior of Distance Metrics in High Dimensional Space, *Proc. International Conf. On Database Theory*, 2001.
2. D-Y. Chen, X.-P. Tian, Y-T. Shen, M. Ouh-young, On Visual Similarity Based 3D Model Retrieval, *Computer Graphics Forum*, **22**(3), pp. 223-232, (2003).
3. C. Goldfeder, P. Allen, Autotagging to Improve Text Search for 3D Models, *Proc. 8th ACM/IEEE-CS Joint Conference on Digital Libraries* 2008 (*JCDL*'08), pp. 355-358, (2008).
4. Google 3D Warehouse, http://sketchup.google.com/3dwarehouse/
5. M. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, K. Ramani, Three Dimensional Shape Searching: State-of-the-art Review and Future Trends, *Computer Aided Design*, **5**(15), pp. 509-530, (2005).
6. G. Leifman, R. Meir, A. Tal, Semantic-oriented 3d shape retrieval using relevance feedback, *The Visual Computer*, **21**(8-10), pp. 865-875, (2005).
7. M. Novotni, G.-J. Park, R. Wessel, R. Klein, Evaluation of Kernel Based Methods for Relevance Feedback in 3D Shape Retrieval, *Proc. The Fourth International Workshop on Content-Based Multimedia Indexing* (*CBMI'05*) (2005).
8. Ohbuchi, et al. Modified SPRH Windows 32bit executable at http://www.kki.yamanashi.ac.jp/~ohbuchi/research/research_index.html
9. R. Ohbuchi, T. Takei, Shape-Similarity Comparison of 3D Shapes Using Alpha Shapes, *Proc. Pacific Graphics 2003*, pp. 293-302, (2003)
10. R. Ohbuchi, A. Yamamoto, J. Kobayashi, Learning semantic categories for 3D Model Retrieval, *Proc. ACM MIR 2007*, pp. 31-40 (2007).
11. R. Ohbuchi, Toshiya Shimizu, Ranking on semantic manifold for shape-based 3d model retrieval, *Proc. ACM MIR* 2008, pp. 411-418, (2008).
12. J. Pu, K. Lou, K. Ramani, A 2D Sketch-Based User Interface for 3D CAD Model Retrieval, *Computer Aided Design and Application*, **2**(6), pp.717-727 (2005).
13. P. Shilane, P. Min, M. Kazhdan, T. Funkhouser, The Princeton Shape Benchmark, *Proc. Shape Modeling International* (*SMI '04)*, pp. 167-178, (2004). http://shape.cs.princeton.edu/search.html
14. J. Tangelder, R. C. Veltkamp, A survey of content based 3D shape retrieval methods, *Multimedia Tools and Applications*, **39**(3), pp. 441-471, (2008).
15. E. Wahl, U. Hillenbrand, G. Hirzinger, Surflet-Pair-Relation Histograms: A Statistical 3D-Shape Representation for Rapid Classification, *Proc. 3DIM 2003*, pp. 474-481 (2003).
16. C. Zhang, T. Chen, An Active Learning Framework for Content-Based Information Retrieval, *IEEE Trans. Multimedia*, **4**(2), June (2002)
17. D. Zhou, O. Bousquet, T.N. Lal, J. Weston, B. Schölkopf, Learning with Local and Global Consistency, *Proc. NIPS 2003*, (2003).
18. D. Zhou, J. Weston, A Gretton, O. Busquet, B. Schölkopf, Ranking on Data Manifolds, *Proc. NIPS 2003*, (2003).

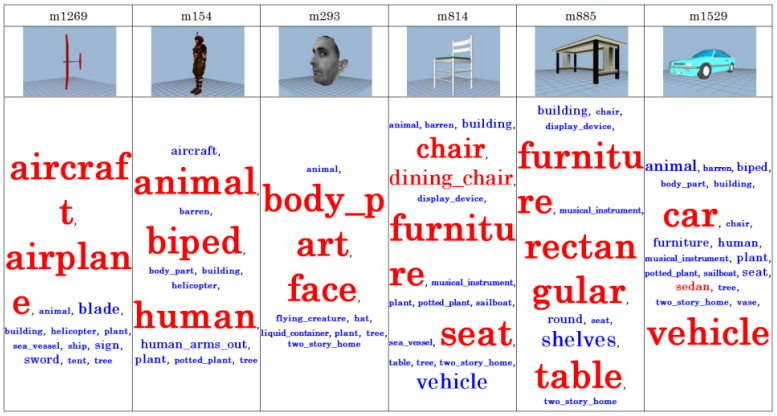

Fig. 6. Examples of tags (successful cases.)

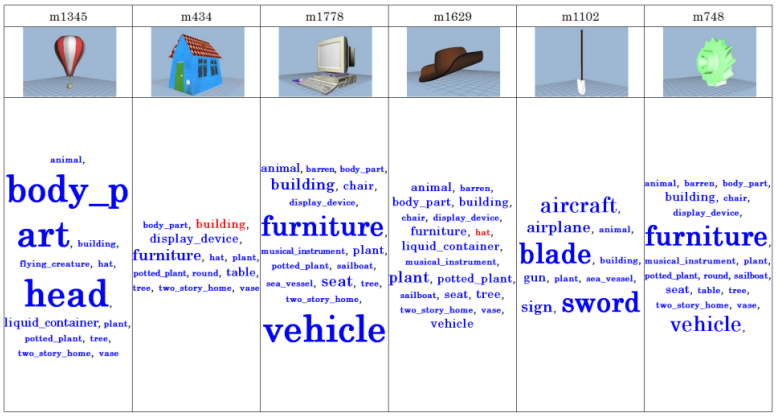

Fig. 7. Examples of tags (unsuccessful cases.)

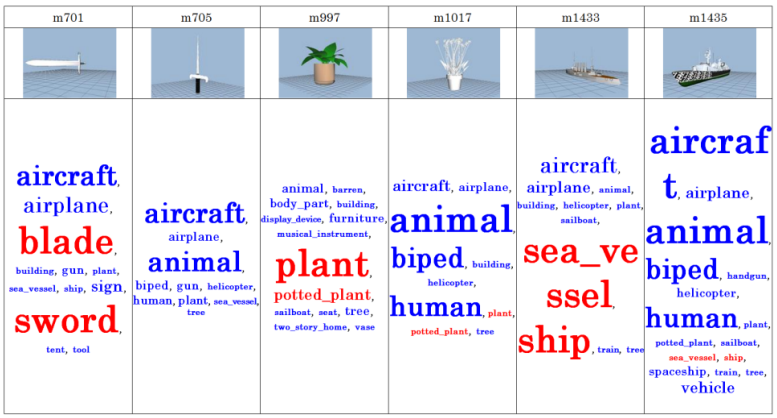

Fig. 8. Examples of tags (mixed cases.)