A Comparison of Text and Shape Matching for Retrieval of Online 3D Models

Patrick Min[1], Michael Kazhdan[2], and Thomas Funkhouser[2]

[1] Utrecht University, Padualaan 14, 3584 CH, Utrecht, the Netherlands
[2] Princeton University, 35 Olden St., Princeton, NJ 08544, United States

Abstract. Because of recent advances in graphics hard- and software, both the production and use of 3D models are increasing at a rapid pace. As a result, a large number of 3D models have become available on the web, and new research is being done on 3D model retrieval methods. Query and retrieval can be done solely based on associated text, as in image retrieval, for example (e.g. Google Image Search [1] and [2, 3]). Other research focuses on shape-based retrieval, based on methods that measure shape similarity between 3D models (e.g., [4]). The goal of our work is to take current text- and shape-based matching methods, see which ones perform best, and compare those. We compared four text matching methods and four shape matching methods, by running classification tests using a large database of 3D models downloaded from the web [5]. In addition, we investigated several methods to combine the results of text and shape matching. We found that shape matching outperforms text matching in all our experiments. The main reason is that publishers of online 3D models simply do not provide enough descriptive text of sufficient quality: 3D models generally appear in lists on web pages, annotated only with cryptic filenames or thumbnail images. Combining the results of text and shape matching further improved performance. The results of this paper provide added incentive to continue research in shape-based retrieval methods for 3D models, as well as retrieval based on other attributes.

1 Introduction

There has been a recent surge of interest in methods for retrieval of 3D models from large databases. Several 3D model search engines have become available within the last few years (e.g., [6–9]), and they cumulatively index tens of thousands of 3D polygonal surface models. Yet, still there have been few research studies investigating which types of query and matching methods are most effective for 3D data. Some 3D model search engines support only text queries [6], while others provide “content-based” queries based on shape [4]. But how do shape-based and text-based retrieval methods compare?

To investigate this question, we measured classification performance of the currently best-performing text-based and shape-based matching methods. We also evaluated several functions that combine text and shape matching scores. For the text matching, a 3D model is represented by a text document, created from several sources of text associated with the model, as well as synonyms and hypernyms (category descriptors) of the 3D model filename (added using WordNet, a lexical database [10]). For the shape matching, a 3D model is represented by a *shape descriptor*, computed from the polygons describing the model’s surface.

All classification tests were done using the “Princeton Shape Benchmark” (PSB) 3D model test database [5]. It contains 1814 3D models downloaded from the web, subdivided into a training set and a test set, containing 907 models each, manually classified into 90 and 92 comparable classes respectively. It is a subset of a larger database of about 33000 models downloaded from the web using an automatic crawler [11]. Retrieval results were evaluated using precision/recall curves [12].

We found that shape-based matching outperforms text-based matching in all our experiments. The main reason is that 3D models found on the Web are insufficiently annotated. They usually are presented in lists, annotated with at most a single name, which is often misspelled or a repeat of the filename. Of the available text sources, we found that the text inside the model file itself and the synonyms and hypernyms of the filename were the most discriminating. Additionally, we found that for combining the results of the shape and the text matching method a function returning the minimum of normalized scores showed the best performance.

The contribution of this paper is a comparison of text and shape matching methods for retrieval of online 3D models, and an evaluation of several different combination functions for combining text and shape matching scores. This paper shows that the relatively simple solution of using only associated text for retrieval of 3D models is not as effective as using their shape.

The rest of this paper is organized as follows. Text matching and our approach for maximizing text retrieval performance is described in the next section. Section 3 discusses shape matching and shows the performance of several recent shape matching methods. Text and shape matching are compared in Section 4 and combined in Section 5. Conclusions and suggestions for future work are in Section 6.

2 Text Matching

In this section we review related work on retrieval of non-textual data using associated text. Note that we do not discuss text retrieval itself. We refer the interested reader to [13–15]. We then describe the sources of text found with 3D models crawled from the web and investigate how well the text can be used to compute a similarity measure for the associated 3D models.

2.1 Related Work

There has been relatively little previous research on the problem of retrieving non-textual data using associated text. The web is an example of a large database for which such methods can be useful: (1) it contains many non-textual objects (e.g., images, sound files, applets) and (2) these objects are likely to be described on web pages using text. Examples of web search engines that take advantage of associated text are Google Image Search [1] (for images), FindSounds [16] (for sound files), and MeshNose [6] (for 3D models).

Probably the largest site for searching images using text keywords is Google’s image search. Unfortunately there are no publications available about the method they use. A related FAQ page suggests that heuristics are used to determine potentially relevant text related to an image, for example, the image filename, link text, and web page title. Each source is probably assigned a different weight, depending on its importance, similar to how the main Google search site assigns weights to text terms depending on whether they are in the title, headers, and so on.

Sable and Hatzivassiloglou investigated the effectiveness of using associated text for classifying images from online news articles as indoor or outdoor [17]. They found that limiting the associated text to just the first sentence of the image caption produced the best results. In other work, Sable *et al.* use Natural Language Processing (e.g., identifying subjects and verbs) to improve classification performance of captioned images into four classes [18]. Our problem is slightly harder since our source text is less well-defined (i.e., there is not an obvious "caption"), and the number of classes is much higher. To our knowledge, there has never been a study investigating the effectiveness of text indexing for 3D models.

2.2 Text Sources

In our study, we focus on the common "bag of words" approach for text matching: all text that is deemed relevant to a particular 3D model is collected in a "representative document," which is then processed and indexed for later matching. This document is created using several potentially relevant text sources. Because we are indexing 3D model files linked from a web page, we are able to extract text from both the model file itself as well as the web page (note that because we convert all models to the VRML 2.0 format, we only refer to text sources of this format). The following list describes the text sources we can use:

From the model file:

1. **model filename:** The filename usually is the name of the object type. The extension determines the filetype. For example, `alsation.wrl` could be the filename of a VRML file of an Alsation dog
2. **model filename without digits:** From the filename we create a second text source by replacing all digits with spaces. Very often filenames contain sequence numbers (for example, `chair2.wrl`) that are useless for text keyword matching
3. **model file contents:** A 3D model file often contains labels, metadata, filenames of included files, and comments. In VRML, it is possible to assign a label to a scene-graph node (a part of the model) and then re-use that node elsewhere in the file. For example, in a model of a chair, a leg can be defined once, assigned the identifier `LEG`, and then re-used three times to create the remaining legs. As such, these identifiers typically describe names of parts of the model. To describe metadata, a VRML 2.0 file may contain a `WorldInfo` node, which is used to store additional information about the model, such as a detailed description, the author name, etc. Filenames of included files can be names of other model files, textures, or user-defined nodes. Finally, a model file may contain descriptive comments. The model file comments were left out from our experiments because we found that many files contain commented-out geometry, which, when included, would add many irrelevant keywords

From the web page:

4. **link text:** This is the descriptive text of the hyperlink to the model file, i.e., the text between the `<a>` and `</a>` HTML tags. For example: `<a href="b747.wrl">a VRML model of a Boeing 747</a>`
5. **URL path:** These are the directory names of the full URL to the model file. If multiple models are organized in a directory structure, the directory names could be category names helpful for classification. For example, as in the URL `http://3d.com/objects/chairs/chair4.wrl`
6. **web page context (text near the link):** We define the context to be all plain text after the `</a>` tag until the next `<a href>` tag (or until the next HTML tag if there is none). This text could for example read "1992 Boeing 747-400 passenger plane, 210K, created by John Doe". Context found *before* the link text was found to be mostly irrelevant
7. **web page title:** The title of the web page containing the link to the 3D model. It often describes the category of models found on the page, for example, "Models of Airplanes"

Additional text source:

8. **Wordnet synonyms and hypernyms:** We create an additional eighth text source by adding synonyms and hypernyms (category descriptors) of the filename using WordNet, a lexical database [10] (if no synonyms or hypernyms can be found for the filename, the link text is tried instead). In related work, Rodriguez *et al.* use WordNet synonyms [19], and Scott and Matwin use synonyms and hypernyms [20] to improve classification performance. Recently, Benitez and Chang showed how WordNet can be used to disambiguate text in captions for content-based image retrieval [21]. Adding synonyms and hypernyms enables queries like "vehicle" to return objects like trucks and cars, or "television" to return a TV. WordNet returns synonyms and hypernyms in usage frequency order, so we can limit the synonyms and hypernyms used to only the most common ones.

Following common practices from text retrieval, all collected text goes through a few processing steps. First, *stop words* are removed. These are common words that do not carry much discriminating information, such as "and," "or," and "my". We use the SMART system's stop list of 524 stop words [22], as well as stop words specific to our domain (e.g. "jpg," "www," "transform"). Next, the resulting text is *stemmed* (normalized by removing inflectional changes, for example "wheels" is changed to "wheel"), using the Porter stemming algorithm [23].

2.3 Text Matching Methods

Given a representative text document for each 3D model, we can define a textual similarity score for every pair of 3D models as the similarity of their representative text documents. To compute this score, we use a variety of text matching methods provided by `rainbow`, a program of the Bow toolkit, a freely available C library for statistical text analysis [24]. The tested methods were: three variations of TF/IDF [13], Kullback Leibler [25], K-nearest neighbors, and Naive Bayes [26]. A few methods supported by the `rainbow` program (e.g., Support Vector Machines) were also tested but failed to run to a finish.

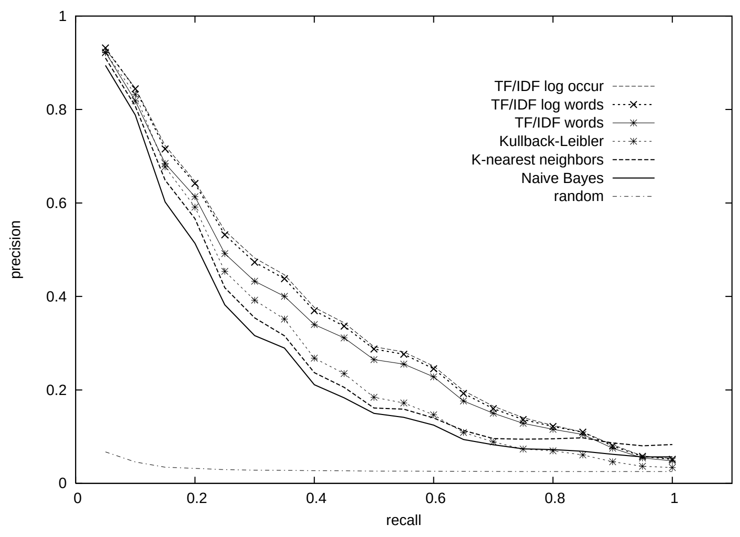

Fig. 1. Average precision/recall for four types of text matching methods, and random retrieval

Figure 1 shows the resulting precision/recall results obtained when the representative text document for each 3D model in the test set of the Princeton Shape Benchmark was matched against the representative text documents of all the other 3D models. The matches were ranked according to their text similarity scores, and precision-recall values were computed with respect to the base classification provided with the benchmark. From this graph, we see that the "TF/IDF log occur" method shows the best performance for our dataset, and thus we used this method for all subsequent tests.

To determine the most useful combinations of text sources, we ran a classification test using each combination of n out of the eight text sources for the representative text document, with $n \in \{1, ..., 8\}$ (so the total number of combinations tested was $\sum_{n=1}^{8} \binom{8}{n} = 255$). The performance of each combination was measured as the average precision over twenty recall values.

From these tests, we found that adding as many text sources as possible improves overall performance, in general. This may be explained by our observation that the addition of keywords helps classification performance if the keywords are relevant, but does not hurt performance if they are irrelevant, since they do not match many other models. We expect that as the database size increases, this property will no longer hold because irrelevant keywords would generate cross-class matches.

Looking more closely at how often each source occurs in the best combinations, we counted the number of times each source appears in the top 50 combinations (i.e., the 50 combinations out of 255 with the highest average precision). The results are shown as percentages in table 1. We see that the identifiers found inside the 3D model files themselves provided the most information for classification. The WordNet synonyms and hypernyms also turned out to be very useful, despite the fact that for 279 models (31%) no synonym or hypernym was found (model names for which WordNet did not return a synonym or hypernym included names (e.g., “justin”), abbreviated words (“satellt”), misspelled words (“porche”), and words in a different language (“oiseau”)).

Table 1. Percentage of all occurrences of each text source appearing in the best 50 combinations

source	percentage in top 50
model file	100
synonyms and hypernyms	100
link	62
filename without digits	58
filename	58
path	56
page title	54
page context	50

3 Shape Matching

In this section, we briefly review previous work on shape-based retrieval of 3D models. Then, we present results comparing several standard shape matching methods to determine which works best on the Princeton Shape Benchmark.

3.1 Related work

Retrieval of data based on shape has been studied in several fields, including computer vision, computational geometry, mechanical CAD, and molecular biology. For surveys of recent methods, see [27, 28]. For our purpose, we will only consider matching and retrieval of isolated 3D objects (so we do not consider recognition of objects in scenes, or partial matching, for example).

3D shape retrieval methods can be roughly subdivided into three categories: (1) methods that first attempt to derive a high-level description (e.g., a skeleton) and then match those, (2) methods that compute a feature vector based on local or global statistics, and (3) miscellaneous methods.

Examples of the first type are skeletons created by voxel thinning [29], and Reeb graphs [30]. However, these methods typically require the input model to be 2-manifold, and usually are sensitive to noise and small features. Unfortunately, many 3D models are created for visualization purposes only, and often contain only unorganized sets of polygons (“polygon soups”), possibly with missing, wrongly-oriented, intersecting, disjoint, and/or overlapping polygons, thus making them unsuitable for most methods that derive high-level descriptors.

Methods based on computing statistics of the 3D model are more suitable for our purpose, since they usually impose no strict requirements on the validity of the input model. Examples are shape histograms [31], feature vectors composed of global geometric properties such as circularity or eccentricity [32], and feature vectors (or *shape*

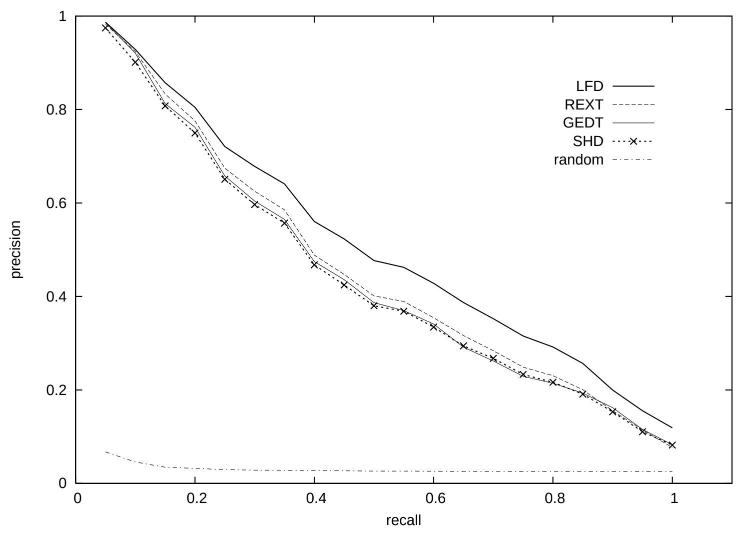

Fig. 2. Average precision/recall for four shape matching methods, and random retrieval. This figure is a partial reproduction of one in [5]

descriptors) created using frequency decompositions of spherical functions [33]. The resulting histograms or feature vectors are then usually compared by computing their L_2 distance.

Some alternative approaches use 2D views (2D projections of a 3D model), justified by the heuristic that if two 3D shapes are similar, they should look similar from many different directions. Examples are the "prototypical views" of Cyr and Kimia [34], and the "Light Field Descriptor" of Chen *et al.* [35].

3.2 Shape Matching Methods

In our experiments, we considered four shape matching methods: (1) the *Light Field Descriptor* (LFD) [35], (2) the *Radialized Spherical Extent Function* (REXT) [36], (3) the *Gaussian Euclidian Distance Transform* (GEDT) [33], and (4) the *Spherical Harmonics Descriptor* (SHD) [33]. These four methods have been shown to be state-of-the-art in a recent paper [5].

We ran an experiment in which these four methods were used to compute a similarity score for every pair of 3D models in the test set of the Princeton Shape Benchmark. The similarity scores were used to rank the matches for each model and compute an average precision-recall curve for each matching method with respect to the benchmark's base classification. Results are shown in Figure 2 (see [5] for details). From these curves, we find that the Light Field Descriptor provides the best retrieval performance in this test, and thus we use it in all subsequent experiments.

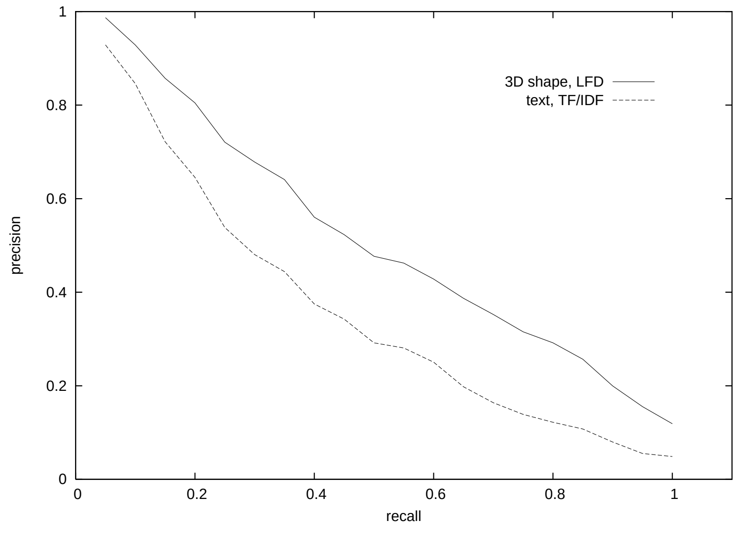

Fig. 3. Average precision/recall for text and 3D shape matching

4 Comparing Text Matching to Shape Matching

Next, we compare the classification performance of the best text matching method to the best shape matching method. Figure 3 shows the resulting average precision/recall plot. The shape matching method significantly outperforms text matching: average precision is 44% higher.

The main cause of the relatively poor performance of the text matching method is the low quality of text annotation of online 3D models. Upon examination of the actual text associated with each model, we found that many filenames were meaningless (e.g., “avstest” for a face), misspelled (e.g., “ferrar” for a ferrari), too specific (e.g., “camaro” for a car), or not specific enough (e.g., “model” for a car). Some models were annotated in a language other than English (e.g., “oiseau” for a bird). By running a spell checker on the filenames with the digits removed, we found that 36% of all model filenames were not English words.

Also, for several sources there is simply no useful text available. For example, many link texts were either a repeat of the filename, or contained no text at all: for 446 models in the training set (51%) no link text could be found (usually a thumbnail image is used instead). Furthermore, 4 (0.4%) models had no path information (i.e., no directories in their URL), 193 (21%) web page titles were missing, for 279 (31%) filenames or link texts no synonym could be found, for 692 (76%) models no web page context was found, and for 153 (17%) models no text inside the model file was found.

Even commercial 3D model databases are not necessarily well annotated. Of three commercial databases available to us (provided by CacheForce, De Espona, and Viewpoint, containing approximately 2000, 1000, and 1000 models respectively), only one was consistently well annotated.

In all text matching experiments, the representative document created for each 3D model was used as a query. However, because the size and quality of text annotation varies a lot from model to model, one may argue that this text is not representative of actual user queries. Users of a retrieval system are more likely to enter a few descriptive keywords or class names. To investigate classification performance given this kind of user queries, we ran an additional classification experiment in which the full category names of the Princeton Shape Benchmark were used as simulated user queries. Some obvious keywords were added manually (e.g., “blimp” and “zeppelin” for the “dirigible hot air balloon” category, or “house” for the “home” category). The average precision achieved when using these query keywords was 11% higher than when using the representative documents, but still 30% lower than the best shape matching method.

5 Combining Text and Shape Matching

In our final tests, we investigate how to combine the best text and shape matching methods to provide better retrieval results than can be achieved with either method alone.

5.1 Related Work

A considerable amount of research has been presented on the problem of how to best combine multiple classifiers [37]. Most work in this area has been done in content-based image retrieval. For example, Srihari presents a system (“Piction”) that identifies faces in annotated news photographs using a face detection algorithm and Natural Language Processing of the captions [38]. Smith and Chang describe a system for the retrieval of arbitrary online images [3]. Relevant text for each image is extracted from the URL and the `alt` parameter of the HTML `<img>` tag, for example. However, searches based on low-level image features or on text can not be combined. This combination *has* been investigated in later retrieval systems. La Cascia *et al.* combine text and image features into a single feature vector, to improve search performance [39]. Text is extracted from the referring web page, with different weights assigned depending on the HTML tag that enclosed it (e.g., text in a `<title>` is more important than text in an `<h4>` (small header) tag). Paek *et al.* present a method that combines a text- and image-based classifier for the classification of captioned images into two classes (“indoor” and “outdoor”) [40], which improved classification accuracy to 86.2%, from 83.3% when using the text alone.

5.2 Multiclassifiers

In previous work we suggested that the results of text and shape matching can be combined to improve classification performance [4], and proposed a combination function that simply averaged mean-normalized matching scores. However, no evaluation was done to see which combination function works best. Here we consider the simple case of combining the scores of two classifiers, using a static combination function. We experimented with four types of functions: (1) linear weighted average, (2) minimum, (3) (minimum) rank, and (4) using confidence limits. The first two were also tested on mean-normalized scores.

Table 2. Average precision achieved when combining the matching scores of the text and shape matching methods using various static combiners, and the percentage improvement over shape matching alone

method	average precision	% improvement over shape alone
shape (LFD)	0.507	-
minimum, normalized scores	0.536	5.8
confidence limits, normalized scores	0.533	5.1
weighted average, normalized scores	0.523	3.2
confidence limits	0.520	2.6
weighted average	0.519	2.4
minimum	0.508	0.2
minimum rank	0.495	-2.3
weighted average rank	0.487	-4

1. **linear weighted average:** If s_1 and s_2 are the matching scores of a pair of models, then the combined score is $w \cdot s_1 + (1 - w) \cdot s_2$, with w the weight setting. We computed average precision/recall for $w \in \{0, 0.05, 0.1, ..., 1.0\}$, and picked the value of w which resulted in the highest overall precision. The optimal weight setting for the training set was ($0.1 \cdot text + 0.9 \cdot shape$)
2. **minimum:** the lowest matching score (signifying highest similarity) is returned
3. **rank:** The matching scores are ordered, and the resulting rank of each query becomes its new matching score. We can then apply one of the first two functions (linear weighted average or minimum)
4. **confidence limits:** The “confidence limits” method is based on the idea that if a similarity score of a single classifier is sufficiently close to zero, then that classifier can be trusted completely. The output of other classifiers is then ignored. Sable uses a variant of this method when combining a text- and image-based classifier [2]: feature vectors from both are classified using a Support Vector Machine, and a confidence level is assigned to the classification, depending on the distance of the vector from the dividing hyperplane (the decision boundary). If the confidence level of the image-based classifier is high enough, then the text-based classifier is ignored. If not, then the text-based classifier is used and the image-based classifier is ignored.
   We used the training set to determine optimial limit settings of 0.09 and 0.22 for shape and text matching respectively (and -2.45 and -1.5 for mean-normalized scores). If both scores were above their limit, we reverted to the linear weighted average (other alternatives yielded worse results)

Table 2 shows the resulting average precision values achieved for each combiner, and the percentage improvement over using shape matching alone (computed using the test set). We achieve an additionol 5.8% improvement in average precision, using the function that returns the minimum of normalized scores. These results confirm that the text and shape representations of a 3D model are sufficiently independent, such that when they are combined, they become more discriminating. There may well be other representations (e.g. appearance-based) that capture a very different aspect of a 3D model, and as such could increase performance even further.

6 Conclusions and Future Work

This paper evaluates text and shape matching methods for retrieval of online 3D models, as well as their combination. Classification tests were done using the Princeton Shape Benchmark, a large benchmark database of 3D models downloaded from the web.

For text matching, we found that a variant of TF/IDF showed the best classification performance. Text found inside a 3D model file itself and synonyms and hypernyms (category descriptors) of the model filename were most useful for classification.

The currently best shape matching method (based on *Light Field Descriptors* [35]) significantly outperformed the best text matching method, yielding 44% higher average precision. The main reason is that the quality of text annotation of online 3D models is relatively poor, limiting the maximum achievable classification performance with a text-based method.

We investigated several simple multiclassifiers, and found that a function returning the minimum of normalized matching scores produced an additional performance improvement of about 6%. Studying other more sophisticated multiclassifiers is an interesting area for future work, considering there are many other attributes of 3D models (e.g., color, texture, structure) for which additional classifiers can be designed.

The main contribution of this paper is that it demonstrates the advantage of using shape-based matching methods over text-based methods for retrieval of 3D models. This should encourage designers of future 3D model retrieval systems to incorporate query methods based on shape, and other attributes that do not depend on annotation provided by humans, as they hold much potential for improving retrieval results.

Acknowledgements

Patrick Min was supported in part by the AIM@SHAPE Network of Excellence grant 506766 by the European Commission. The National Science Foundation provided partial funding for this project under grants CCR-0093343 and IIS-0121446.

References

1. Google: Image search. (`http://www.google.com/images`)
2. Sable, C.: Robust Statistical Techniques for the Categorization of Images Using Associated Text. PhD thesis, Columbia University (2003)
3. Smith, J.R., Chang, S.F.: Searching for images and videos on the world-wide web. Technical Report 459-96-25, Columbia University (1996)
4. Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, D., Jacobs, D.: A search engine for 3D models. ACM Transactions on Graphics **22** (2003)
5. Shilane, P., Kazhdan, M., Min, P., Funkhouser, T.: The Princeton Shape Benchmark. In: Proc. Shape Modeling International, Genoa, Italy (2004)
6. MeshNose: 3D objects search engine. (`http://www.deepfx.com/meshnose`)
7. Suzuki, M.T.: A web-based retrieval system for 3D polygonal models. In: Proc. IFSA/NAFIPS, Vancouver, Canada (2001) 2271–2276
8. Chen, D.Y., Ouhyoung, M.: A 3D object retrieval system based on multi-resolution Reeb graph. In: Proc. Computer Graphics Workshop, Taiwan (2002) 16–20
9. Min, P.: A 3D Model Search Engine. PhD thesis, Princeton University (2004)
10. Miller, G.A.: WordNet: A lexical database for English. CACM **38** (1995) 39–41

11. Min, P., Halderman, A., Kazhdan, M., Funkhouser, T.: Early experiences with a 3D model search engine. In: Proc. Web3D Symposium, St. Malo, France, ACM (2003) 7–18
12. van Rijsbergen, C.J.: Information Retrieval. Butterworth (1979)
13. Salton, G.: Automatic text processing: the transformation, analysis, and retrieval of information by computer. Addison-Wesley, Reading, Massachusetts (1988)
14. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley (1999)
15. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34** (2002) 1–47
16. FindSounds: Sound file search engine. (`http://www.findsounds.com`)
17. Sable, C.L., Hatzivassiloglou, V.: Text-based approaches for the categorization of images. In: Proc. Research and Advanced Technologies for Digital Libraries, Paris (1999)
18. Sable, C., McKeown, K., Church, K.W.: NLP found helpful (at least for one text categorization task). In: Proc. EMNLP, Philadelphia, PA (2002)
19. de Buenaga Rodríguez, M., Gómez-Hidalgo, J.M., Díaz-Agudo, B.: Using WordNet to complement training information in text categorization. In: Proc. RANLP, Stanford (1997)
20. Scott, S., Matwin, S.: Text classification using WordNet hypernyms. In: Proc. Workshop Usage of WordNet in Natural Language Processing Systems. (1998) 45–52
21. Benitez, A.B., Chang, S.F.: Semantic knowledge construction from annotated image collections. In: Proc. Int. Conf. on Multimedia and Expo ICME, Lausanne, Switzerland (2002)
22. Salton, G.: The SMART retrieval system. Prentice-Hall, Englewood Cliffs, NJ (1971)
23. Porter, M.: An algorithm for suffix stripping. Program **14** (1980) 130–137
24. McCallum, A.: Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/˜mccallum/bow` (1996)
25. Kullback, S., Leibler, R.: On information and sufficiency. Ann. Math. Stat. **22** (1951) 79–86
26. Mitchell, T.: Machine Learning. McGraw-Hill (1997)
27. Loncaric, S.: A survey of shape analysis techniques. Pattern Recognition **31** (1998)
28. Tangelder, J.W., Veltkamp, R.C.: A survey of content based 3D shape retrieval methods. In: Proc. Shape Modeling International, Genoa, Italy (2004)
29. Sundar, H., Silver, D., Gagvani, N., Dickinson, S.: Skeleton based shape matching and retrieval. In: Proc. SMI, Seoul, Korea (2003)
30. Hilaga, M., Shinagawa, Y., Kohmura, T., Kunii, T.L.: Topology matching for fully automatic similarity estimation of 3D shapes. In: Proc. SIGGRAPH. (2001) 203–212
31. A.P.Ashbrook, N.A.Thacker, P.I.Rockett, C.I.Brown: Robust recognition of scaled shapes using pairwise geometric histograms. In: Proc. BMVC, Birmingham, UK (1995) 503–512
32. Taubin, G., Cooper, D.: Object recognition based on moment (or algebraic) invariants. In: Geometric Invariance in Computer Vision. MIT Press (1992)
33. Kazhdan, M., Funkhouser, T., Rusinkiewicz, S.: Rotation invariant spherical harmonic representation of 3D shape descriptors. In: Proc. SGP, ACM (2003) 156–165
34. Cyr, C.M., Kimia, B.B.: 3D object recognition using shape similarity-based aspect graph. In: Proc. ICCV, IEEE (2001)
35. Chen, D.Y., Ouhyoung, M., Tian, X.P., Shen, Y.T., Ouhyoung, M.: On visual similarity based 3D model retrieval. In: Proc. Eurographics, Granada, Spain (2003)
36. Vranic, D.V.: An improvement of rotation invariant 3D shape descriptor based on functions on concentric spheres. In: Proc. ICIP. Volume 3. (2003) 757–760
37. Jain, A.K., Duin, R.P., Mao, J.: Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 4–37
38. Srihari, R.K.: Automatic indexing and content-based retrieval of captioned images. IEEE Computer **28** (1995) 49–56
39. Sclaroff, S., Cascia, M.L., Sethi, S.: Unifying textual and visual cues for content-based image retrieval on the world wide web. CVIU **75** (1999) 86–98
40. Paek, S., et al.: Integration of visual and text-based approaches for the content labeling and classification of photographs. In: ACM Workshop on Multim. Indexing and Retr. (1999)