

Browsing Shape Collections

Pushkar Joshi*
U.C. Berkeley

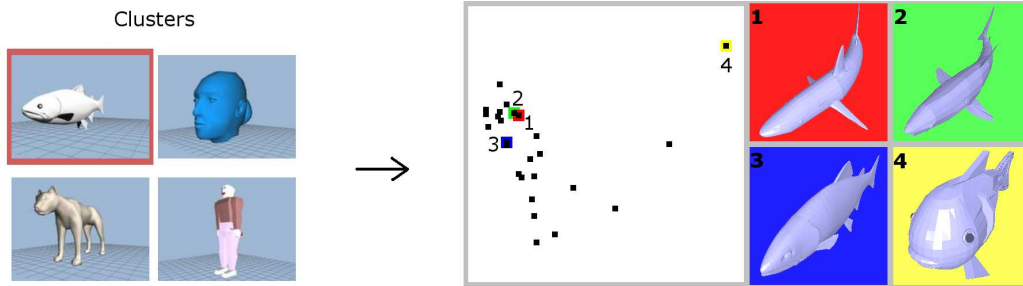


Figure 1: Left: *global* overview of the database with one icon per cluster; Right: *local* scatter plot of shapes in the selected cluster, with corresponding color-coded selected shapes shown on right in detail. Note that shape differences roughly match the relative distances on the 2D plot.

Abstract

Large collections of shapes are becoming freely available, and we will soon need tools that quickly give an overview of the contents of a shape database. We present a browsing tool that produces a hierarchical, cluster-based overview of a shape collection, and allows users to zoom in on individual clusters to get more detailed views. Our browsing interface is particularly effective for analyzing collections of specific types of shapes. We are also excited about the potential of this interface to quickly spot unintuitive outliers, which, in turn can be used to evaluate the effectiveness of a particular shape descriptor function.

1 Introduction

Tools for modeling and scanning 3D shapes are now easily accessible, and large collections of shapes are becoming more common. As a result, understanding the content of a given shape collection is becoming more important. Research in shape analysis has focussed on being able to treat a given shape collection like any other database and process queries on it. We now have ‘shape search engines’ (e.g. [Shilane et al. 2004]) that scan a stored shape collection and return shapes similar (according to some metric) to the input query shape.

Individual search-based methods are one way to access a database. Like other types of databases, it is important to also get an overview of the *entire* shape collection. That is, In addition to being able to answer questions like “which shape is the most similar to this one”, we will also need to answer

high-level questions like “what’s in this entire shape collection”. We need tools to quickly access a shape database, survey its content, zoom in on some shape clusters of particular interest, and get representative samples from within such clusters. Such a browsing interface can be used to immediately see what types of shapes (and how many of each type) are present in a given shape collection. As of now, we have yet to see such a tool.

We are in the process of constructing a visualization tool that gives a quick hierarchical overview of the contents of a database, shows the internal clustering of shapes, allows to zoom in on such clusters, provides a scatter plot of the contents of such clusters, and shows interesting outliers. The current implementation takes polygon meshes as input, produces a fixed-length shape description (a string of numbers) for each shape as a pre-process. At runtime, the program clusters the multi-dimensional shape descriptions to provide a hierarchical view, and produces a two-dimensional scatter plot of the shapes to be displayed. Although our system has a number of different components, we have deliberately kept each component modular. We plan to let the user experiment with different shape descriptors, clustering methods and dimensionality reduction methods.

Besides the obvious application of providing an overview of a shape collection, there are other technical uses of our browsing interface. A shape browser can immediately show trends within data and elaborate on which shapes are similar (according to the shape evaluation metric) and which shapes are the outliers in the collection. Such a tool can also be used to appraise a given shape descriptor. That is, unexpected clusters that do not match how humans perceive shape similarities can indicate loss of important information during shape description. A more promising use of a shape browser is for specific types of shapes. By fine-tuning some

*e-mail: ppj@eecs.berkeley.edu

parameters of the shape descriptor and adding some information about structure, a user may be able to explore the space spanned by the input shape collection.

2 Related Work

As mentioned above, to the best of our knowledge, there is no shape browser that provides an interactive overview of a given shape collection. There are, however, a number of papers that address the problem of shape *search*. Tangelder and Veltkamp [Tangelder and Veltkamp 2004] provide a useful survey of 3D search methods. The Princeton Search Engine [Funkhouser et al. 2003] is a popular search engine for 3D models; the input can be provided as a polygon mesh, a 2D projection image or a text description. There are also other web-based 3D search engines available: [Saltel] and [Chen and Ouhyoung 2002] being some of them.

Usually the most difficult aspect of constructing a 3D shape search engine is developing a shape descriptor function that produces a string of numbers from an input shape. Ideally, shape descriptions are compact, of fixed length, and invariant to all ‘non-shape’ features like sampling density, polygon mesh quality, parameterization, translation, scale and orientation. These requirements are fulfilled best by converting the input shape into a spherical function, and performing a spherical harmonic transform of that spherical function. Using spherical harmonics, it is trivial to get invariance to non-shape features, translation and scale (the latter two are normalized so the shapes’ centroids lie at the origin, and bounding spheres have unit radius).

Invariance to orientation, however is much more difficult. Spherical harmonics by themselves are not rotationally invariant: they contain phase information. Aligning 3D models to their principal axes using principal components analysis is very unstable and not robust enough in practice. [Kazhdan et al. 2003] combine contributions from coefficients of the same frequency to obtain a rotationally invariant descriptor of a spherical function. The rotation invariance is at the cost of some information loss, but the gain in robustness usually overrides the loss of information. [Belongie et al. 2000] introduced ‘shape contexts’, which is another rotationally invariant shape descriptor.

The spherical harmonic transform takes a sampling of a spherical function as input. There are several methods to construct spherical functions from an arbitrary polygon mesh. Spherical extents [Vranic and Saupe 2001], shape distributions [Osada et al. 2001] and Extended Gaussian images [Horn 1984] are some popular choices. We use spherical extents in this work, but will explore other, more sophisticated spherical functions in the future.

Researchers have developed browsing interfaces for other types of media. [Tzanetakis 2003] describes a browser for

music clips while [Kang and Schneiderman 2000] describe their interactive browser and querying interface for photographs. In this project, we wish to learn from the experiences of these other types of browser developers and apply those lessons to the development of the shape browser.

3 Our Approach

3.1 Overview

Our goal is to develop an interactive interface that shows the internal clustering of shapes, allows to select one cluster (zoom in), and provides a scatter plot of the contents of that cluster.

We use rotationally invariant spherical harmonics of spherical extent functions to produce a fixed-length shape signature for each shape. We use Principal Components Analysis (PCA) for projecting the multi-dimensional shape data down to two dimensions. Our system is flexible and is designed to easily incorporate various shape signature functions and dimensionality reduction algorithms.

3.2 Shape Descriptor Function

We are currently assuming a relatively general shape collection with a wide variety of shapes (we use a subset of the Princeton Shape Benchmark [Shilane et al. 2004] as sample data). The requirements on the quality of the input meshes are mild: simple ‘polygon soup’ — where the mesh is specified as a collection of disconnected faces — is acceptable. In our current implementation, we are assuming that the ‘shape’ of the object is entirely denoted by the volume enclosed by the surface of the object. We are *not* considering internal information; we plan to do so in future implementations.

Based on the positive results obtained with the Princeton search engine, we use the rotation invariant spherical harmonic transform of the ‘spherical extents’ as a shape evaluation function. We take an arbitrary polygon mesh as input and normalize its position (move centroid to the origin) and scale (unit radius of bounding sphere). Then, we shoot rays along the two angles that parameterize a sphere and record the distance of the farthest intersection: this gives us a sampling of the spherical function describing the shape. We perform a spherical harmonic transform of this spherical function (with bandlimit of 20) to obtain coefficients of the harmonic functions. We combine the contributions of the same frequency (from [Kazhdan et al. 2003]). In this way, we obtain a fixed-length numeric description for every shape. Fig. 2 illustrates this process.

Some parameters of this representation, such as the best spherical function and optimal harmonic bandwidth, still

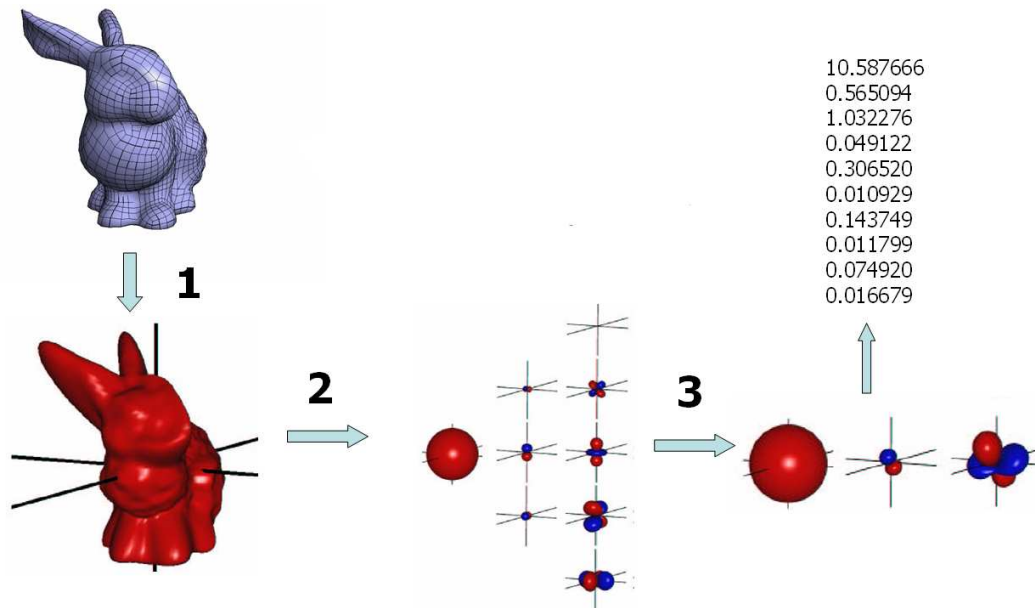


Figure 2: Pipeline of the shape descriptor function: **1.** Construct a spherical function from an arbitrary polygon mesh **2.** Decompose this input spherical function into spherical harmonic components **3.** Combine components of the same frequency to obtain a rotationally-invariant description of the shape (lower figures taken from [Kazhdan et al. 2003])

need to be determined and could be made available as options for the user. In our current system, this shape descriptor is invoked for all input shapes as a pre-process to ensure run-time performance of the browsing interface.

3.3 User Interface

On start-up, the system performs a global clustering of the entire data base using the high-dimensional shape descriptor. The database is partitioned into a small number of clusters (say from 4 to 20) and the icon of one representative shape from each cluster is displayed. Our current implementation requires the user to specify the number of clusters (this number can be changed interactively) and the system uses *k-means* clustering to segment the shape collection. The user refines the cluster view by selecting a single cluster; the visualization tool then recursively provides the same analysis and the corresponding display but for only the shapes within the selected cluster. Fig. 3 shows the intended user interface of the system.

Possible options for enhancing the current user interface include: allowing the user to draw a box and to explore a subset of clusters or shapes, automatically computing the optimum number of clusters (hierarchical clustering), and having multiple layers of clustered views of the shape collection.

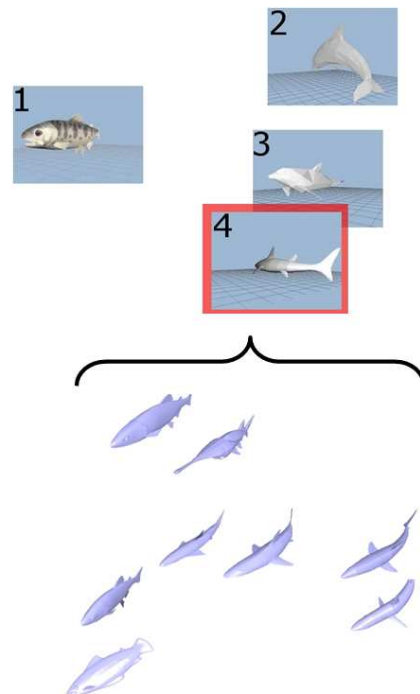


Figure 3: The intended user interface: a user sees icons of the representative clusters, and upon selection of a particular cluster, the user sees icons of the shapes within that cluster.

3.4 Dimensionality Reduction

We present the visualization of the shape database as a 2D scatter plot, for both, the global, ‘cluster’ view and the individual shapes view. For the cluster view, the cluster representatives (or medoids) are used as samples. Our current implementation uses principal component analysis (PCA) to project the high-dimensional data onto the two principal axes of the shapes. While easy and fast, PCA projections can result in loss of significant information, leading to an incorrect visualization of the shapes subset. At the global overview of the whole database, the distance between the cluster medoids could be very large, and PCA may produce incorrect projections. However, the exact placement of the representative shapes in this global view is not very important, as we are simply trying to show the different clusters, not the relationships between them. More detailed views with individual shapes should show the relative constellations of shapes in the database, organized primarily by their similarity based on the chosen shape description. Our system produces a good projection for small sets of individual shapes: the relative distances in the projection match those in high-dimensional space. However, we do not always get consistent relative distances for large sets of individual shapes. In order to maintain these local relative differences within shapes, we are considering the use of more sophisticated manifold-based dimensionality reduction algorithms in future.

4 Results

Please refer to Fig. 4, 5, 6 for screenshots of our system.

Our system works well and successfully illustrates the intended concept. The visualizations are generated fast enough for the system to be fully interactive. The most expensive step is the pre-processing of the shapes to produce a list of shape signatures: the time required for this step is a linear function of the number of polygons. Small datasets of shapes that could be manually categorized into intuitive groups are processed correctly by the system — the clusters match what the user expects. However, the system is not yet sophisticated enough to *perfectly* browse large databases (greater than 200 shapes). While most of the clusters contain similar shapes and each cluster’s scatter plot displays proper shape differences, there are instances of unexpected outliers or of incorrect clustering. These are due to the limitations described below.

4.1 Drawbacks

Our system is built from several independent components, each with its own set of limitations. We will enumerate them here in detail and mention possible solutions.

Loss of information in shape descriptor function By far the component with the most potential for improvement is the shape descriptor function. There are two stages in which the system loses important shape information while forming the shape description:

- Loss of information due to spherical function: The choice of the spherical function greatly impacts the amount of information lost in the shape descriptor. For instance, the current implementation uses the Spherical Extent function, which completely ignores internal shape information. Additionally, we post-process the spherical harmonic output to make it rotationally invariant ([Kazhdan et al. 2003]); this step reduces the dimensionality of the system even further and results in further loss of information.

Information loss is unavoidable with any spherical function defined over an arbitrary polygon mesh. However, we are confident we can improve the current implementation by choosing a better spherical function: currently we are evaluating the use of the Shape Context function [Belongie et al. 2000] as a rotationally invariant shape descriptor.

- Loss of information due to lack of structural data: Considering purely the shape of an object ignores the often important morphological information of that object. For instance, a curled up armadillo and a soccer ball have similar shape, but completely different structure. We are looking into methods to incorporate some structural information into our shape descriptor. A perceptually driven shape metric (such as the number of salient features [Hoffman and Singh 1997]) seems like a promising approach to investigate.

Drawbacks of Clustering Methods Our current implementation uses a simple implementation of k-means clustering, which can be sensitive to local minima. That is, the initial position of the clusters can determine the final clustering. Instead of asking the user for number of clusters, we also need a method for the system to compute the optimal number of clusters. Therefore, we believe that hierarchical clustering methods (with simulated annealing to decrease the effect of local minima) need to be provided.

Drawbacks of Dimensionality Reduction Methods The current implementation uses principal components analysis (PCA), which is simple and fast. Since PCA does a linear projection, there is often important non-linear information that is lost on projection. The result is that objects incorrectly appear to be closer or farther from each other than their relative distance in high-dimensional space. Manifold-based projection methods (e.g. Locally Linear Embedding [Roweis and Saul 2000]) are known to better maintain relative differences, and would be interesting to try, at least for

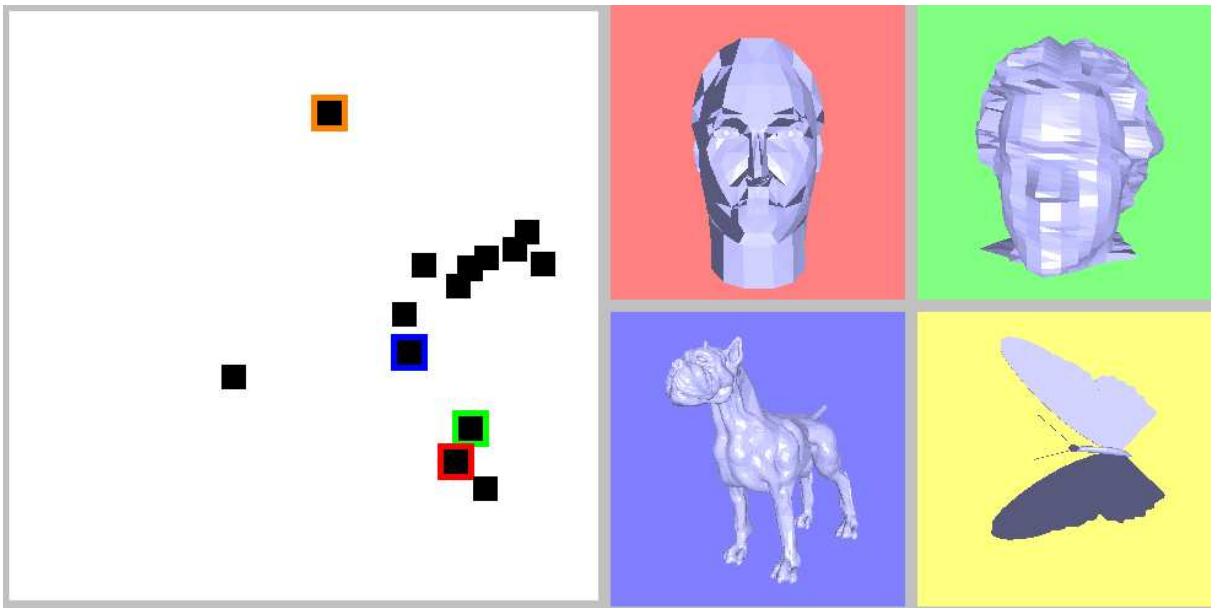


Figure 4: Current implementation of the global cluster-view of the shape collection; using k-means clustering, the system has partitioned the data and displayed the cluster representatives. Each black square represents a cluster, and upon selection of a cluster, the user is able to examine its representative shape on the right. The same display interface is used for displaying individual shapes within a cluster, where each black square represents a single shape.

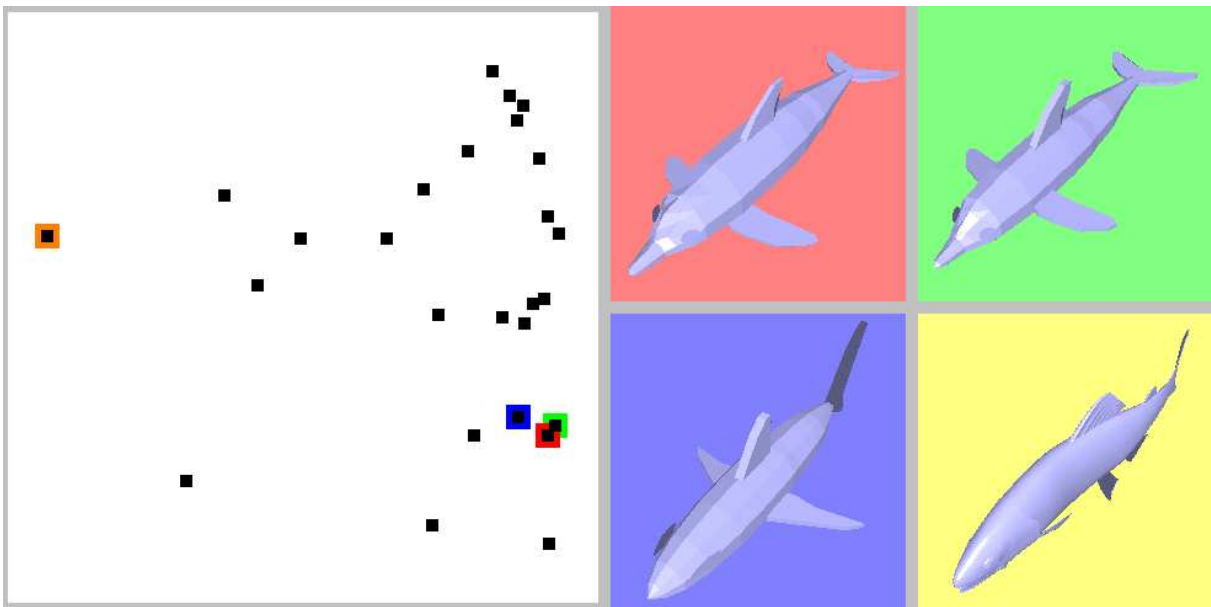


Figure 5: Individual shapes view of a single cluster — good example: the upper two shapes are very similar to each other, the lower left is slightly different (notice the tail and snout) and lower right is much more different. The relative shape differences are maintained in the visualization. Our system performs well for such clusters of similar shapes.

scatter plots with a large number of samples. The expectation is that these better projection methods will work better for large number of samples in a scatter plot.

5 Discussion

We think this project was instructive about the effectiveness of two-dimensional scatter plots. Both the visualizations (the global cluster view and the individual shapes view of

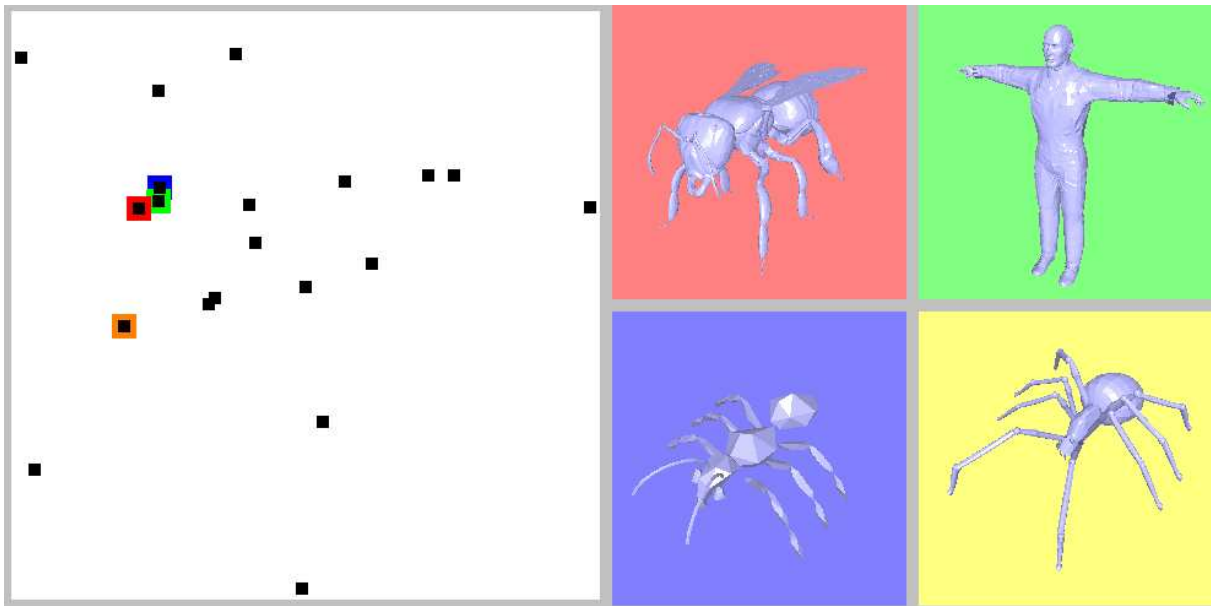


Figure 6: Individual shapes view of a single cluster — bad example: the upper two shapes are not similar to each other, yet are displayed as being close. On the other hand, the lower right shape is incorrectly too far away, despite being close in shape to the lower left and upper left shape. This example demonstrates some drawbacks of our system, especially for clusters that contain a wide range of shapes.

each cluster) were implemented as scatter plots. The users were able to quickly predict similarities and differences in the shapes by simply looking at the scatter plots (having yet to see the actual shapes). It was very easy to spot obvious clusters and outliers. Upon seeing a scatter plot, the users first clicked on pairs of shapes that were close according to our distance metric, and also clicked on shapes that were far removed from the rest of the shapes. This in turn led to a very fast evaluation of the shape metric: if two nearby shapes were drastically different, we were told we had problems with the shape metric (similarly for a shape that was presented in the visualization as being farther than it was actually perceived by the users). Overall, we think the audience increased their appreciation of simple two-dimensional scatter plots and the ability to predict relationships between data samples.

We also found this project to demonstrate the usefulness of view selection. Almost every shape, upon selection by the users, was rotated until the user understood its intricacies. Hopefully the users realized that it is difficult to convey all the information of a 3D shape using a static 2D image: it is important to be able to access the 3D information. At the same time, if one were to develop a robust view selection technology, the users will immediately see our browsing interface as a useful application of view selection.

Finally, we think that 3D shapes are fun to play with. One could potentially spend a long time simply clicking on the scatter plots and inspecting the individual shapes. It was entertaining to observe various shapes of the same type (espe-

cially models of animals) and to spot the slight differences between extremely similar shapes. Hopefully the users had a similar experience, and this browsing interface makes a favorable case for more interactive 3D content in mainstream media.

6 Future Work

6.1 General Shape Collection

We do not expect to develop a *fully* accurate shape browser for arbitrary shapes — information loss in some stage of the system is inevitable, which will lead to at least a few incorrect results. However, there are three promising research directions that can help us improve the current design:

Evaluating Shape Descriptors Our browsing interface can be used as a tool to evaluate the effectiveness of various shape descriptor functions. Unexpected outliers (similar shapes that incorrectly appear far apart in the visualization) can indicate a loss of significant information, thereby indicating the particular shape descriptor as less effective. We believe a browsing interface is a faster and easier method to evaluate a shape descriptor than individual search based methods, where one needs to carefully select the input query shape.

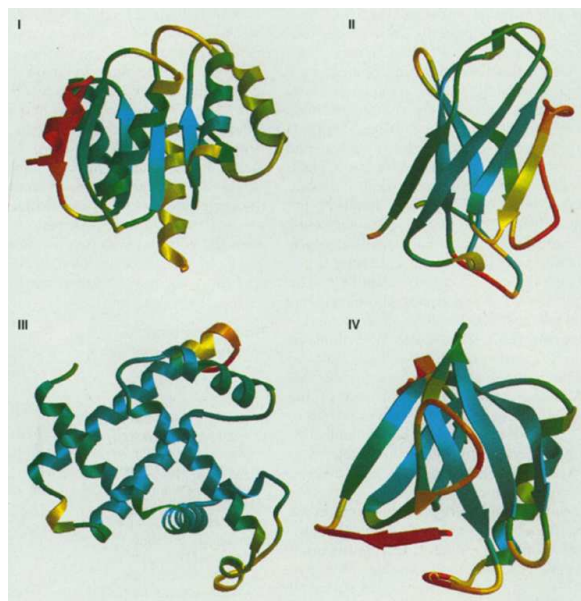


Figure 7: Visualization of protein molecules
(from [Holm and Sander 1996])

Improving Shape Distance Metric from User Input

Our system uses a distance metric to calculate shape differences. We could modify this distance metric to match what a user specifies as similar and different shapes. That is, we can learn a better shape distance metric from user input (e.g. see [Xing et al. 2003]). The user interface will be simple: the user will drag a shape to be closer to the shape(s) that she thinks are similar, and drag a shape away from shape(s) that she thinks are different. The system will update its distance metric based on this user input and the same, improved metric can now be re-applied to data sets.

View Selection We currently have no special processing of presenting a selected shape in a particular orientation; we use a default orientation (global axes aligned view) and allow the user to rotate the object with the arcball interface. While this works well for small sets of data, for larger sets, it might be useful to automatically find the best view (e.g. by using [Vazquez et al. 2003]) of the shape to present to the user. This will minimize the amount of user interaction required to understand the shape and will result in faster browsing.

6.2 Specific Shape Collection

We believe that the true power of a shape browser will be apparent while browsing specific types of shapes. Knowing that shapes are of a given type allows us to exploit additional (often structural) information about the shape. For instance, we are exploring the use of a shape browser for exploring a collection of protein molecules. Current research in pro-

tein shape analysis focusses on finding a distance between *two* query shapes, but as yet, there is no research to describe a shape space containing all the protein shapes. The development of this shape space requires a robust shape distance metric. There have been significant advances in the representation of protein shape metric (see [Holm and Sander 1996]), but it is still an area of active research.

We feel that an interactive browser of protein shapes will provide further insights into the protein shape space and stimulate a formal definition of the space. We are considering adding structural information (in the form of α -helices and β -strands) to our spherical function-based shape descriptor. This will allow us to design a specialized shape distance metric for protein molecules. We will also try to improve the distance metric by using the protein matching information (similar shapes and different shapes) generated over past years by several researchers (e.g. SCOP [Murzin et al. 1995]). The ultimate hope is that a browsing interface for proteins will help the development of a protein shape space and therefore a better protein shape metric, which is a significant and current research problem.

Acknowledgements We would like to thank the organizers and all students of the UC Berkeley CS-294 (Spring 2006 - Visualization) class for useful feedback on this project. We are also grateful to the Princeton Shape Analysis group for making the Princeton Shape Benchmark mesh collection freely available for download. Finally, we would like to thank the developers of S2Kit ([Healy et al. 2003]) and FFTW ([Frigo and Johnson 2005]) which are publicly available and open-source programs for spherical harmonic transform and discrete Fourier transform respectively.

7 Conclusion

We believe that a shape browser is useful for analyzing large shape collections. Of particular note is the potential of this tool for analyzing collections of *specific types* of shapes and for appraising various shape evaluation functions.

References

- BELONGIE, S., MALIK, J., AND PUZICHA, J. 2000. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*.
- CHEN, D., AND OUHYOUNG, M. 2002. A 3d object retrieval system based on multi-resolution reeb graph. In *Computer Graphics Workshop*.

- FRIGO, M., AND JOHNSON, S. 2005. The design and implementation of fftw3. In *IEEE Special Issue on Program Generation, Optimization, and Platform Adaptation*.
- FUNKHOUSER, T., MIN, P., KAZHDAN, M., CHEN, J., HALDERMAN, A., DOBKIN, D., AND JACOBS, D. 2003. A search engine for 3d models. *ACM Trans. Graph.* 22, 1.
- HEALY, D., ROCKMORE, D., KOSTELEK, P., AND MOORE, S. 2003. Ffts for the 2-sphere - improvements and variations. *Journal of Fourier Analysis and Applications*.
- HOFFMAN, D., AND SINGH, M. 1997. Saliency of visual parts. *Cognition* 63.
- HOLM, L., AND SANDER, C. 1996. Mapping the protein universe. *Science*.
- HORN, B. 1984. Extended gaussian images. *PIEEE*.
- KANG, H., AND SCHNEIDERMAN, B. 2000. Visualization methods for personal photo collections: Browsing and searching in the photofinder. In *IEEE International Conference on Multimedia and Expo (ICME2000)*.
- KAZHDAN, M., FUNKHOUSER, T., AND RUSINKIEWICZ, S. 2003. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Eurographics Symposium on Geometry Processing*.
- MURZIN, A., BRENNER, S., HUBBARD, T., AND CHOTHIA, C. 1995. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*
- OSADA, R., FUNKHOUSER, T., CHAZELLE, B., AND DOBKIN, D. 2001. Matching 3d models with shape distributions. In *Shape Matching International*.
- ROWEIS, S., AND SAUL, L. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*.
- SALTEL, E. Personal web page: <http://www-c.inria.fr/eric.saltel/download/download.php>. *INRIA*.
- SHILANE, P., MIN, P., KAZHDAN, M., AND FUNKHOUSER, T. 2004. The princeton shape benchmark. In *Shape Modeling International*.
- TANGELDER, J. W. H., AND VELTKAMP, R. C. 2004. A survey of content based 3d shape retrieval methods. In *SMI*.
- TZANETAKIS, G. 2003. Musescape: an interactive content-aware music browser. In *Int. Conf. on Digital Audio Effects (DAFX)*.
- VAZQUEZ, P., FEIXAS, M., SBERT, M., AND HEIDRICH, W. 2003. Automatic view selection using viewpoint entropy and its applications to image-based modelling. *Computer Graphics Forum*.
- VRANIC, D., AND SAUPE, D. 2001. 3d model retrieval with spherical harmonics and moments. In *DAGM*.
- XING, E., NG, A., JORDAN, M., AND RUSSELL, S. 2003. Distance metric learning, with application to clustering with side-information. In *NIPS*.