A Search Engine for 3D Models

Thomas Funkhouser, Patrick Min, Misha Kazhdan,
Joyce Chen, Alex Halderman, David Dobkin
Princeton University

David Jacobs
NEC Research Institute

Abstract

As the number of 3D models available on the Web grows, there is an increasing need for a search engine to help people find them. Unfortunately, traditional text-based search techniques are not always effective for 3D data. In this paper, we investigate new shape-based search methods. The key challenges are to develop query methods simple enough for novice users and matching algorithms robust enough to work for arbitrary polygonal models. We present a web-based search engine system that supports queries based on 3D sketches, 2D sketches, 3D models, and/or text keywords. For the shape-based queries, we have developed a new matching algorithm that uses spherical harmonics to compute discriminating similarity measures without requiring repair of model degeneracies or alignment of orientations. It provides 46–245% better performance than related shape matching methods during precision-recall experiments, and it is fast enough to return query results from a repository of 20,000 models in under a second. The net result is a growing interactive index of 3D models available on the Web (i.e., a Google for 3D models).

1 Introduction

Over the last few decades, computer science has made incredible progress in computer-aided retrieval and analysis of multimedia data. For example, suppose you want to obtain an image of a horse for a Powerpoint presentation. A decade ago, you could: 1) draw a picture, 2) go to a library and copy a picture, or 3) go to a farm and photograph a horse. Today, you can simply pick a suitable image from the millions available on the web. Although web search is commonplace for text, images, and audio, the information revolution for 3D data is still in its infancy.

However, three recent trends are combining to accelerate the proliferation of 3D models, leading to a time in the future when 3D models will be as ubiquitous as other multimedia data are today: (1) new scanners and interactive tools are making construction of detailed 3D models practical and cost effective, (2) inexpensive graphics hardware is becoming faster (at $3\times$ Moore's Law), causing an increasing demand for 3D models from a wide range of people, and (3) the web is facilitating distribution of 3D models.

These developments are changing the way we think about 3D data. For years, a primary challenge in computer graphics has been how to construct interesting 3D models. In the near future, the key question will shift from "how do we construct them?" to "how do we find them?". For example, consider a person who wants to build a 3D virtual world representing a city scene. He will need 3D models of cars, street lamps, stop signs, etc. Will he buy a 3D modeling tool and build them himself? Or, will he acquire them from a large repository of 3D models on the Web? We believe that research in retrieval, matching, recognition, and classification of 3D models will follow the same trends that can already be observed for text, images, audio, and other media.

An important question then is how people will search for 3D models. Of course, the simplest approach is to search for keywords in filenames, captions, or context. However, this approach can fail: (1) when objects are not annotated (e.g., "B19745.wrl"), (2) when objects are annotated with inspecific or derivative keywords (e.g., "yellow.wrl" or "sarah.wrl"), (3) when all related keywords are so common that the query result contains a flood of irrelevant matches (e.g., searching for "faces" – i.e., human not polygonal), (4) when relevant keywords are unknown to the user (e.g., objects with misspelled or foreign labels), or (5) when keywords of interest were not known at the time the object was annotated.

In these cases and others, we hypothesize that shape-based queries will be helpful for finding 3D objects. For instance, shape can combine with function to define classes of objects (e.g., *round* coffee tables). Shape can also be used to discriminate between similar objects (e.g., desk chairs versus lounge chairs). There are even instances where a class is defined entirely by its shape (e.g., things that roll). In these instances, "a picture is worth a thousand words."

Our work investigates methods for automatic shape-based retrieval of 3D models. The challenges are two-fold. First, we must develop computational representations of 3D shape (*shape descriptors*) for which indices can be built and similarity queries can be answered efficiently. In this paper, we describe novel methods for searching 3D databases using orientation invariant spherical harmonic descriptors. Second, we must find user interfaces with which untrained users can specify shape-based queries. In this paper, we investigate combinations of 3D sketching, 2D sketching, text, and interactive refinement based on shape similarity. We have integrated these methods into a search engine that provides a publicly available index of 3D models on the Web (Figure 1).

The paper is organized as follows. The following section contains a review of related work. Section 3 provides an overview of our system, while discussion of the main research issues appears in Sections 4-7, and implementation details are provided in Section 8. Section 9 presents experimental results of studies aimed at evaluating different query and matching methods. Finally, a brief summary and conclusion appears in Section 10, followed by a discussion of topics for future work in Section 11.

2 Related Work

Data retrieval and analysis have recently been a very active area of research [29, 51]. The most obvious examples are text search engines (e.g., Google [21]), which have become part of our daily lives. However, content-based retrieval and classification systems have also been developed for other multimedia data types, including audio [33], images [22], and video [90].

Retrieval of data based on shape has been studied in several fields, including computer vision, computational geometry, mechanical CAD, and molecular biology (see [2, 9, 14, 53, 66, 89] for

Figure 1: Screenshot of our search engine for 3D models. It allows a user to specify a query using any combination of keywords and sketches (left). Then, for each query, it returns a ranked set of thumbnail images representing the 16 best matching 3D models (right). The user may retrieve any of the 3D models by clicking on its thumbnail, and/or he may refine the search by editing the original input or by clicking on the "Find Similar Shape" link below any thumbnail.

surveys of recent methods). However, most prior work has focused on 2D data [32, 44, 60]. For instance, several content-based image retrieval systems allow a user to sketch a coarsely detailed picture and retrieve similar images based on color, texture, and shape similarities (e.g., [44]). Extending these systems to work for 3D surface models is non-trivial, as it requires finding a good user interface for specifying 3D queries and an effective algorithm for indexing 3D shapes. One problem for indexing 3D surfaces is boundary parameterization. Although the 1D boundary contours of 2D shapes have a natural arc length parameterization, 3D surfaces of arbitrary genus do not. As a result, common shape descriptors for 2D contours (e.g., [7, 8, 47, 52, 88, 94]) cannot be extended to 3D surfaces, and computationally efficient matching algorithms based on dynamic programming (e.g., [83, 86]) cannot be applied to 3D objects. Another problem is the higher dimensionality of 3D data, which makes registration, finding feature correspondences, and fitting model parameters more expensive. As a result, methods that match shapes using geometric hashing [49] or deformations [3, 45, 65, 85]) are more difficult in 3D.

Shape-based recognition of 3D objects is a core problem in computer vision. However, in vision, images or range scans of objects are usually obtained from specific viewpoints, in scenes with clutter and occlusion. Range images require partial surface matching [16, 23, 25, 87], and 2D images are further complicated by perspective distortions and lighting variations. Often these problems are addressed by methods that search for local correspondences between features (e.g., [35, 46, 50, 54]), which are expensive and do not readily lead to an indexable representation. Rather, we focus on 3D models of isolated objects (e.g., a bunny or a teapot) in 3D model files intended for computer graphics visualization or inclusion in a virtual world. While these models are mostly free of sensor noise and occlusions, they usually contain only unorganized sets of polygons ("polygon soups"), possibly with missing, wrongly-oriented, intersecting, disjoint, and/or overlapping polygons. The lack of a consistent solid and surface model makes them difficult for shape analysis. Meanwhile, fixing degenerate models is a difficult open problem [11, 36, 58].

For 3D object models, most shape analysis work has focused on registration, recognition, and

pairwise matching of surface meshes. For instance, representations for registering and matching 3D surfaces include Extended Gaussian Images [38], Spherical Attribute Images [27, 28], and Harmonic Shape Images [97]. Unfortunately, these previous methods usually assume that a topologically valid surface mesh is available for every object. Volumetric dissimilarity measures based on wavelets [34] or Earth Mover's Distance [73] rely upon a priori registration of objects' coordinate systems, which is difficult to achieve automatically and robustly. Other approaches are based on comparing high-level representations of shape, such as generalized cylinders [18], superquadrics [79], geons [93], shock graphs [78], medial axes [10], and skeletons [19, 20, 37, 81]. Methods to compute these representations are usually time-consuming and sensitive to small features. Also, most do not readily lead to a means for indexing a large database [77].

Finally, shapes have been indexed based on their statistical properties. The simplest approach represents objects with feature vectors [29] in a multidimensional space where the axes encode global geometric properties, such as circularity, eccentricity, or algebraic moments [68, 84]. Other methods have considered histograms of geometric statistics [1, 6, 15, 31, 62]. For instance, Ankerst et al. [4] proposed shape histograms decomposing shells and sectors around a model's centroid. Besl [15] used histograms of the crease angle for all edges in a 3D triangular mesh. Osada et al. [62] represented shapes with probability distributions of geometric properties computed for points randomly sampled on an object's surface. Often these statistical methods are not discriminating enough to make subtle distinctions between classes of shapes (e.g., living room chairs versus dining room chairs).

While several projects have been investigating 3D search engines concurrently with ours [64, 82], they are mainly focused on specific data types, such as mechanical CAD parts (e.g., [13, 17, 69, 41]), protein molecules (e.g., [5, 48]), or cultural artifacts [72, 76]. Others only support queries based on text and file attributes (e.g., [56]). To our knowledge, no previous system has: (1) indexed a large repository of computer graphics models collected from the Web, (2) supported 2D and 3D sketching interfaces for shape-based queries, or (3) studied interactions between text and shape in the search for 3D data. These topics are investigated in this paper.

3 System Overview

The organization of our system is shown in Figure 2. Execution proceeds in four steps: crawling, indexing, querying, and matching. The first two steps are performed off-line, while the last two are done for each user query. The following text provides an overview of each step and highlights its main features:

1. **Crawling:** We build a database of 3D models by crawling the Web. 3D data still represents a very small percentage of the Web, and high quality models represent an equally small percentage of all 3D data. So, we have developed a focused crawler that incorporates a measure of 3D model "quality" into its page rank. Using this crawler, we have downloaded 17,834 VRML models from the Web. We augment this database with 2,873 commercial models provided by 3D vendors [43, 24].

2. **Indexing:** We compute indices to retrieve 3D models efficiently based on text and shape queries. In particular, we have developed a new 3D shape descriptor based on spherical

harmonics that is descriptive, concise, efficient to compute, robust to model degeneracies, and invariant to rotations.

3. **Querying:** We allow a user to search interactively for 3D models. Our system supports query methods based on text keywords, 2D sketching, 3D sketching, model matching, and iterative refinement. We find that methods based on both text and shape combine to produce better results than either one alone.

4. **Matching:** For each user query, our web server uses its index to return the sixteen 3D models that best match the query. Our method answers 3D shape queries in less than a quarter of a second for our repository; and, in practice, it scales sub-linearly with the number of indexed models.

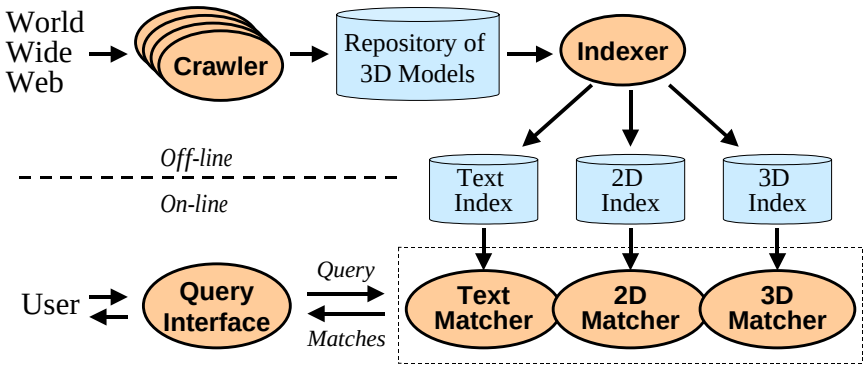


Figure 2: System organization.

The main research issue at the heart of this system is how to provide shape-based query interfaces and matching methods that enable easy and efficient retrieval of 3D models from a large repository. In the following two sections, we discuss these issues in detail for different query interfaces.

4 Shape Queries

The most straight-forward shape-based query interface is to provide the search engine with an existing 3D model and ask it to retrieve similar ones. Our search engine supports this strategy in two ways.

First, the user may type the name of a file to be uploaded from his computer (e.g., "`c:\dolphin.wrl`"), and then the system searches for 3D models with similar shapes. This method is useful for finding more objects of the same type (e.g., given one chair, find 100 others) or for finding more instances of a specific 3D object (e.g., checking for illegal copies of a proprietary model).

Second, the user may search for models with shapes like one returned in a previous search by clicking on the "Find Similar Shape" link under its image on a results page (blue text in Figure 1). This method is useful for iteratively refining searches to hone in on a specific class of objects.

The main challenge in supporting these 3D shape-based similarity queries is to find a computational representation of shape (a *shape descriptor*) for which an index can be built and geometric

matching can be performed efficiently. Generally speaking, the following properties are desirable for a shape descriptor. It should be: (1) quick to compute, (2) concise to store, (3) easy to index, (4) invariant under similarity transforms, (5) insensitive to noise and small extra features, (6) independent of 3D object representation, tessellation, or genus, (7) robust to arbitrary topological degeneracies, and (8) discriminating of shape differences at many scales.

Unfortunately, no existing shape descriptor has all these properties. Most high-level shape representations, such as generalized cylinders [18], superquadrics [79], geons [93], shock graphs [78], medial axes [10], and skeletons [19, 37, 81] require a consistent model of the object's boundary and interior, which is difficult to reconstruct for highly degenerate computer graphics models [11, 36, 58]. Other shape representations, such as Extended Gaussian Images [38], Spherical Attribute Images [27, 28], moments [68, 84], and wavelets [34], require a priori registration into a canonical coordinate system, which is difficult to achieve robustly. Finally, statistical shape descriptors, such as feature vectors [29] and shape distributions [62] are usually not discriminating enough to distinguish between similar classes of objects.

We propose a novel shape-descriptor based on spherical harmonics. The main idea is to decompose a 3D model into a collection of functions defined on concentric spheres and to use spherical harmonics to discard orientation information (phase) for each one. This yields a shape descriptor that is both orientation invariant and descriptive. While the original shape cannot be reconstructed from this representation, comparison of two descriptors provides a provable lower bound on the L_2 distance between them.

A significant advantage of our approach is that it can be indexed without registration of 3D models in a canonical coordinate system. While others have used spherical harmonics to obtain multiresolution representations of shape [75, 91], they require a priori registration with principal axes. In our experience, we find that principal axes are not good at aligning orientations of different models within the same class. Figure 3 demonstrates this problem for a collection of mugs. Despite the fact that the mugs have similar shapes, the derived principal axes are quite different. The main reason is that contributions to the second-order moments used for rotational alignment scale quadratically with distance from the center of mass, which causes small differences in the handles of the mugs to affect the principal axes significantly. The net result is poor alignments and poor match scores for algorithms that rely upon them. Our method takes advantage of phase elimination to avoid this problem.

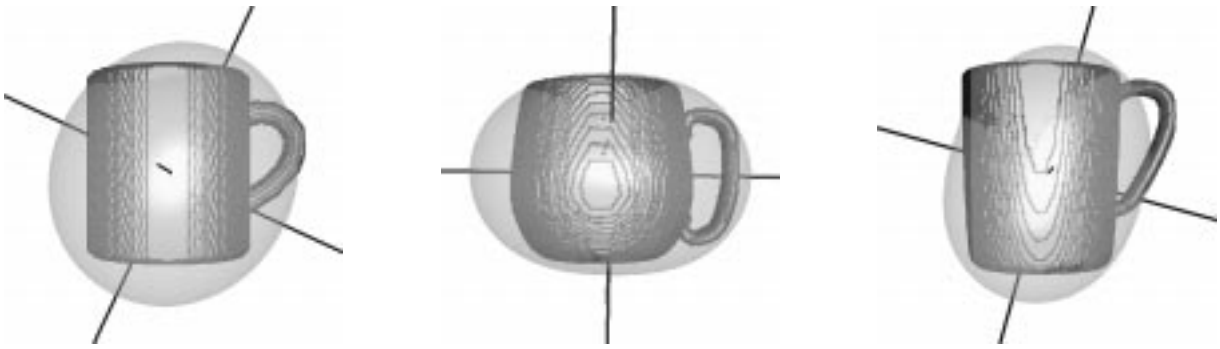


Figure 3: A collection of mugs drawn with their principal axes. Despite the similarity in the models, the principal axes orient the models in very different ways.

As compared to other rotation invariant shape signatures, we expect our spherical harmonics descriptor to be more discriminating of similar shapes. It is unique up to rotations of independent frequency components on concentric spheres and characterizes a shape at different resolutions. Other rotationally invariant descriptors discard significantly more information. For example, Ankerst [4] uses a histogram of the distances from each surface point to the object's center of mass as a 1D descriptor. This amounts to using the zero'th order spherical harmonic in each concentric shell. Our method encodes higher frequency information in a 2D descriptor, which provides more discriminating power.

The main steps for computing a spherical harmonics shape descriptor for a set of polygons are shown in Figure 4:

1. First, we rasterize the polygonal surfaces into a $2R \times 2R \times 2R$ voxel grid, assigning a cell a value of 1 if it is within one voxel of a polygonal surface, and assigning it a value of 0 otherwise.[1] The model is scaled and translated so that the center of mass lies at the point (R, R, R) and so that the bounding sphere of the model has radius equal to R.

2. We treat the voxel grid as a (binary) real-valued function defined on the set of points with length less than or equal to R and express the function in spherical coordinates:

$$f(r, \theta, \phi) = \text{Voxel}(r\sin(\theta)\cos(\phi) + R, r\cos(\theta) + R, r\sin(\theta)\sin(\phi) + R)$$

   where $r \in [0, R]$, $\theta \in [0, \pi]$, and $\phi \in [0, 2\pi]$. By restricting to the different radii we obtain a collection of spherical functions $\{f_0, f_1, \ldots, f_R\}$ with:

$$f_r(\theta, \phi) = f(r, \theta, \phi).$$

3. Using spherical harmonics, we express each function f_r as a sum of its different frequencies:

$$f_r(\theta, \phi) = \sum_m f_r^m(\theta, \phi)$$

   where

$$f_r^m(\theta, \phi) = \sum_{n=-m}^{m} a_{mn} \sqrt{\frac{(2m+1)}{4\pi} \frac{(m-|n|)!}{(m+|n|)!}} P_{mn}(\cos\theta) e^{in\phi}.$$

   (That is, the function f_r^m is the projection of the function f_r onto the m-th irreducible representation of the rotation group acting on the space of spherical functions.)

4. Noting that the different irreducible representations are fixed under rotation, and noting that rotations do not change the L_2 norm of functions, we observe that the value $\|f_r^m\|$ does not change if we rotate the function f_r. We define a rotation invariant signature for f_r as the collection of scalars $\{\|f_r^0\|, \|f_r^1\|, \ldots\}$.

5. Combining these different signatures over the different radii, we obtain a two-dimensional rotation invariant *spherical harmonics descriptor* for the 3D model, with the value at index (r_0, m_0) corresponding to the length of the m_0-th frequency of the restriction of f to the sphere with radius r_0.

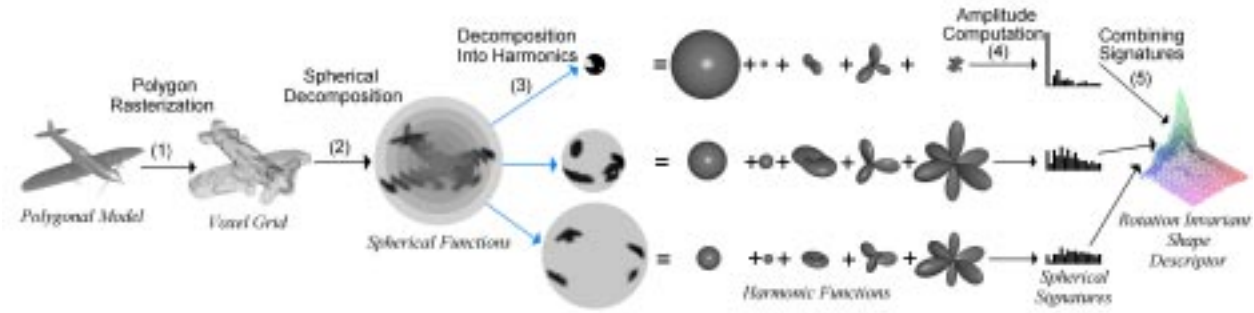


Figure 4: Computing our spherical harmonics shape descriptor.

To compare two spherical harmonics descriptors, we simply compute the Euclidean distance between them. Retrieving the K best matches for a 3D query model is equivalent to solving the K nearest-neighbors problem in a high-dimensional space. Although this problem is known to be hard in the worst case, we can build a search algorithm that works efficiently in practice by searching in multiple 1D spaces [42]. Our implementation works in two passes. In the first pass, we quickly compute a lower bound for the distance between the query model and all models in the database by finding the M-nearest neighbors on the projections of the space onto coordinate axes ($M >> K$). In the second pass we compute the true distance to the models, sorted by the lower bound distance. We stop when the distance to the current K-th nearest model is smaller than the smallest lower bound of the remaining models. When computing the true distance to a model, we use the most significant spherical harmonics first, allowing us to stop when the distance to that model is above our current threshold. In practice, a full comparison is required for a small subset of the database (experimental results are presented in Section 9).

5 Sketch Queries

Of course, shape similarity queries are only possible when the user already has a representative 3D model. In some cases, he will be able to find one by using a text search. However, in other cases, he will have to create it from scratch (at least to seed the search).

An interesting open question then is "what type of modeling tool should be used to create shapes for 3D retrieval queries?". This question is quite different than the one asked in traditional geometric modeling research. Rather than providing a tool with which a trained user can create models with exquisite detail and/or smoothness properties, our goal is to allow novice users to specify coarse 3D shapes quickly. In particular, the interface should be easy to learn for first time visitors to a website. Of course, this requirement rules out almost every 3D modeling tool available today – i.e., it would not be practical to require everybody who wants to use a 3D search engine to take a three week training course to learn the complicated menu structure of a commercial CAD tool. Instead, we have investigated two alternatives.

The first approach is to specify shape queries with a simple 3D sketching tool, such as Teddy [40]

[1]Note: we do not attempt to reconstruct and fill the volumetric interior of the object so as to work with arbitrary "polygon soups", a general and commonly found class of computer graphics models. Fixing degenerate models to form a consistent solid interior and manifold surface is a difficult open problem [11, 36, 58].

or Sketch [96]. To investigate this approach, we have developed a query interface in which the user creates a simple 3D model with Teddy [40], and then the system retrieves similar models using the matching algorithms described in the previous section (see Figure 5). Unfortunately, our early experiences suggest that even its simple gesture interface is still too hard for novice and casual users to learn quickly. During informal studies, we observed that most people do not readily understand "extrusions" and "cuts," and they have a difficult time getting used to rotating a 3D model to get the proper viewpoint for modeling operations. Moreover, only certain types of shapes can be created with Teddy (blobby objects with topological genus zero). We believe that making 3D tools even simpler would require further constraints on the types of shapes that could be produced. Thus, we were motivated to look for alternate sketching paradigms.

Our second approach is to draw 2D shapes with a pixel paint program and then have the system match the resulting image(s) to 2D projections of 3D objects (Figure 6). The main advantage of this approach is that the interface is easy to learn. All but the most novice computer users have used a 2D paint program before, and there are no complicated viewing or manipulation commands. Of course, the main disadvantage is that 2D images generally have less shape information than 3D models. We compensate for this factor somewhat by allowing the user to draw multiple 2D projections of an object in order to better define its shape.

Figure 5: 3D sketch query interface.

Figure 6: 2D sketch query interface.

The main challenge in implementing this approach is to develop algorithms that match 2D sketches to 3D objects. This problem is significantly different than classical ones in computer vision because the 2D input is hand-drawn rather than photographic and the interface is interactive. Thus, we must consider several new questions: How do people draw shapes? What viewpoints do they select? How should the interface guide or constrain the user's input? What algorithms are robust enough to recognize human-drawn sketches? These are big questions, with implications well beyond the scope of this paper. Unfortunately, the vast literature on how trained artists draw [70], how people use characteristic views [63], and how computers recognize photographic images [35] is not directly applicable in our case. Rather, we are interested in how untrained artists make quick sketches and how a computer can match them to 3D objects.

To investigate these questions, we first ran a pilot study in which 32 students from an introductory computer science class were instructed to "draw the shape of an <object>" for eight different objects. The students were only told what to draw, not how to draw it, and they had only 15 seconds for each object. What we found is that people tend to sketch objects with fragmented boundary

contours and few other lines, they are not very geometrically accurate, and they use a remarkably consistent set of view directions (see Figure 7). Interestingly, the most frequently chosen views were *not* characteristic views [63], but instead ones that were simpler to draw (front, side, and top views). These results give us clues about how to match sketches to 3D objects in our system.

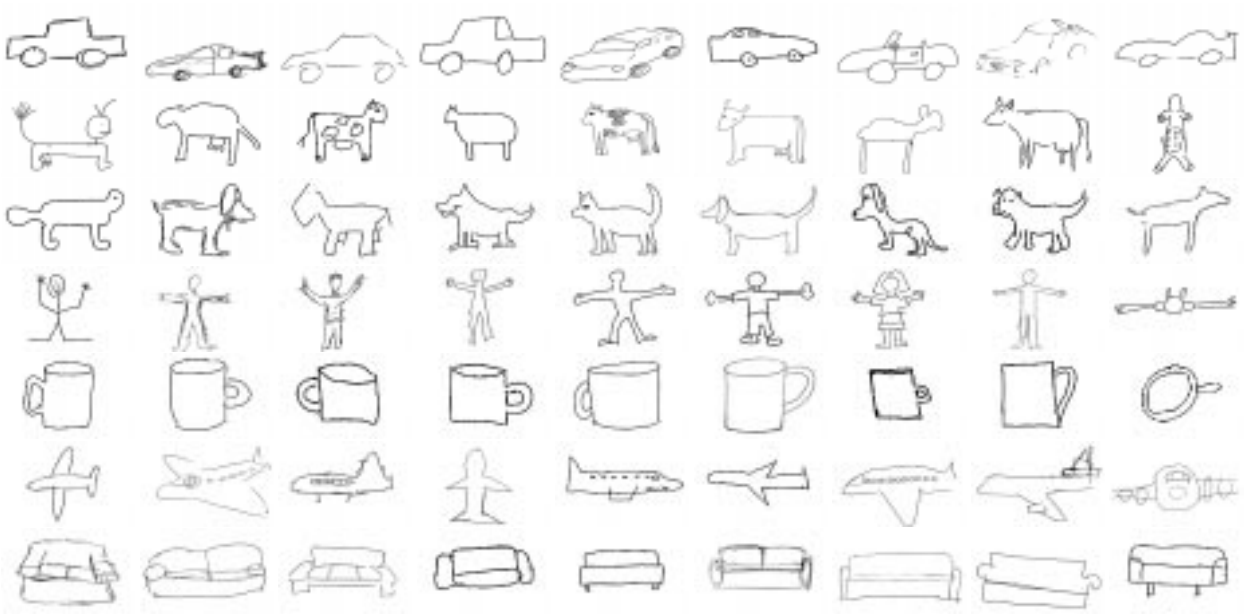


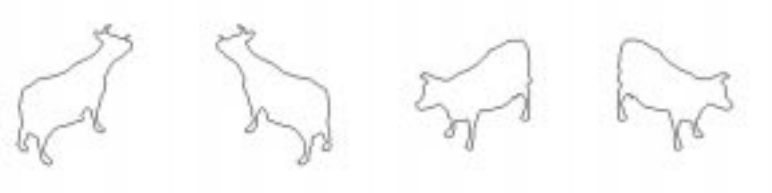
Figure 7: Sketches by people simply asked to "draw the shape of a" camaro car, cow, dog, human with arms out, mug, DC10 airplane, and sofa.

For a set of sketches entered by a user, we match them to projected images of 3D models rendered from different viewpoints and return the best matches (as in [59] and others). During a preprocessing phase, we render thumbnail images with the boundary contours of each 3D object as seen from 13 orthographic view directions. As shown in Figure 8, we take viewpoints at the center of the three faces, the four top corners, and the middle of six edges of a cube (tilt views).[2] Then, for each query with m sketches, we compute the match score for any 3D object as the minimal sum of m (out of $13m$) pairwise sketch-to-thumbnail dissimilary scores, subject to the constraint that no thumbnail can be matched to more than one sketch. This sampling ensures that any sketched view is within 22.5° of a sampled view. Moreover, it also takes advantage of the fact that some 3D models will be aligned with Cartesian axes, in which case our sampled views perfectly match the views preferred by users. We enhance this effect even further by labeling three sketch windows "Side View," "Front View," and "Top View" in our system.

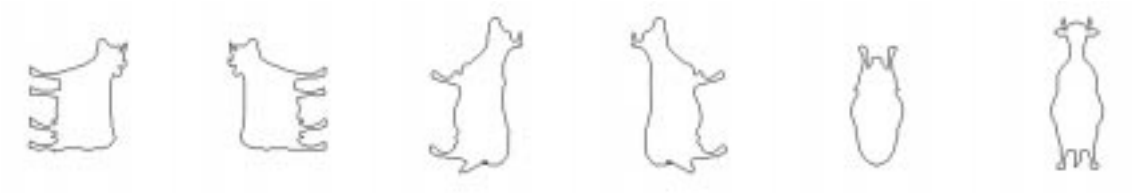
Matching hand drawn sketches to projected silhouettes of 3D models poses another problem. Although we prompt users with example sketches containing clean boundary contours, user input is often made up of fragmented sketch marks. Thus, we cannot use efficient contour matching algorithms (e.g., [7, 8, 88]). Instead, we compare sketches and rendered views with an image matching method. To handle deformations and geometric inaccuracies, we first apply the distance

[2]Our matching method is invariant to rotations and reflections, so views rotated around the view direction or from the opposite side of the object are not needed.

(a) 3 Side views



(b) 4 Corner views



(c) 6 Tilt views

Figure 8: 2D boundary contours rendered from 13 views of each object.

transform to both the sketch and rendered image. This helps make our method robust to small variations in the positions of lines, as in Chamfer matching [12] and Hausdorff matching [39]. It also provides an indexable distance measure.

For cases where 3D models are arbitrarily oriented, the image matching method must be robust to reflections and rotations in the image plane. To address this issue, we use a 2D analog of the spherical harmonics method described in the previous section. Figure 9 demonstrates the details of our process: (1) Compute the distance transform of the boundary contour. (2) Obtain a collection of circular functions by restricting to different radii. (3) Expand each circular function as a sum of trigonometric functions. (4) Using the fact that rotations do not change the amplitude within a frequency, define the signature of each circular function as a list of the amplitudes of its constituent trigonometrics. (5) Finally, combine these different signatures to obtain a 2D signature for the boundary contour. We index these descriptors using the same nearest neighbor search method described in Section 4. This method is inspired by Zhan's work on Fourier Descriptors [95], which provides a rotation invariant signature for boundary curves, obtained by computing the Fourier series and storing only the amplitude of each frequency component.

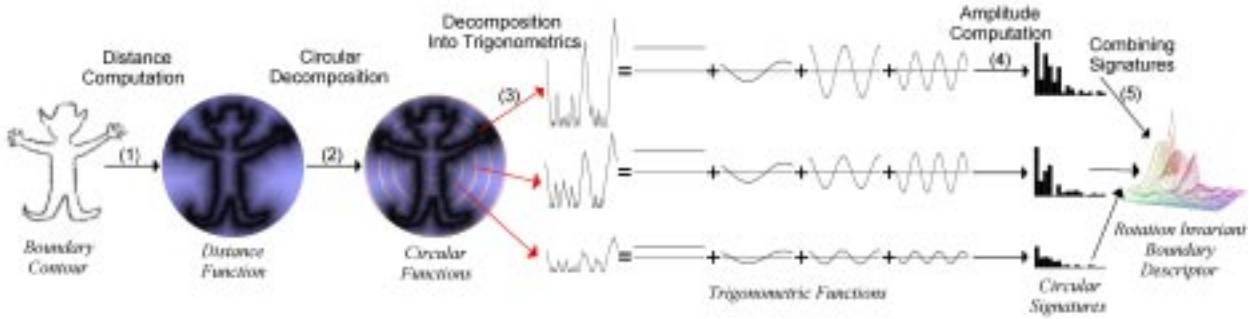


Figure 9: Computing our shape descriptor for boundary contours.

6 Text Queries

Our system also supports searching for 3D models by matching keywords in their textual descriptions. To support this feature, we construct a representative document for each 3D model. The text in that document includes the model filename, the anchor and nearby text parsed from its referring Web page, and ASCII labels parsed from inside the model file. For instance, we include part names (e.g., "DEF" nodes in VRML), texture file names, and informational fields (e.g., the "WorldInfo" node in VRML).[3]

Each document is preprocessed by removing common words (*stop words*) that don't carry much discriminating information, such as "and", "or", "my", etc. We use the SMART system's stop list of 524 common words as well as words specific to our domain (e.g. "jpg", "www", "transform", etc.) [74]. Next, the text is stemmed (normalized by removing inflectional changes) using the Porter stemmer [67]. Finally, synonyms of the filename (without the extension) are added using WordNet [57].

In order to match documents to user-specified keywords or to other documents, we use the *TF-IDF/Rocchio* method [71], a popular weighting and classification scheme for text documents. This method assigns a similarity score based on a term's frequency in the document and its inverse frequency over all documents. We use the Bow toolkit [55] in our implementation.

7 Multimodal Queries

Since text and shape queries can provide orthogonal notions of similarity corresponding to function and form, our search engine allows them to be combined.

We support this feature in two ways. First, text keywords *and* 2D/3D sketches may be entered in a single multimodal query. Second, text and shape information entered in successive queries can be combined so that a user can refine search terms adaptively. For instance, if a user entered text keywords in a first query, and then clicked a "Find Similar Shape" link, the text and 3D shape would combine to form a second query.

These types of multimodal queries are often helpful to focus a search on a specific subclass of objects (Figure 10). For example, a query with only keywords can retrieve a class of objects (e.g.,

[3]We found that including comments is counter-productive, as models often contain commented-out geometry, which floods the documents with indiscriminating keywords.

tables), but it is often hard to hone in on a specific subclass with text alone (e.g., round tables with a single pedestal). Similarly, a query with only a sketch can retrieve objects with a particular shape, but it may include objects with different functions (e.g., both tables and chairs). Multimodal input can combine ways of describing objects to form more specific queries (Figure 10(c)).

(a) Text query (b) 2D sketch query (c) Multimodal query

Figure 10: Multimodal queries are often effective at finding specific types of objects.

In order to find the K top matches for multimodal queries, we find the best M match scores for each mode separately ($M >> K$), mean-normalize them (so the mean is 0 and the variance is 1) to avoid over-weighting any query interface, and then return the K models with the highest average normalized scores. Currently, we choose $K = 16$ and $M = 128$. This method strikes a balance between returning the intersection of match results returned by different modes (which is useful for finding specific objects) and returning their union (which is useful for finding all objects within a class). Later, we plan to allow users to control how search terms are combined (e.g., "OR" and "AND" qualifiers).

8 Implementation

We have implemented our 3D search engine in C/C++, Java, and Perl. The main components are shown in Figure 11. All components run under Red Hat Linux 7.1, except for the web server (Solaris 8) and the 3D model conversion (Irix 6.5). This section describes the flow of data through the system and its implementation details.

The crawler is written in Perl, and runs on three 933 MHz Pentium III machines, each with 1 GB memory. Each crawler process is multithreaded and downloads files using up to 50 simultaneous connections. It searches for AutoCAD, LightWave, PLY, 3D Studio, VRML, and Wavefront files, possibly contained within pkzip, gzip, or lharc compressed archives. We initially seed it with URLs returned by Google and other search engines for queries such as "3D AND (models OR meshes)". Each page retrieved is scored as follows. For 3D models, the score is an estimate of its "quality" (currently we use the logarithm of the triangle count). For HTML pages, the score is a count of keywords contained within the title and body text that suggest its relationship to 3D model files (e.g., "3D model" "Wavefront object" etc.). Each unvisited URL is assigned a priority that is a weighted sum of: 1) a distance-weighted average of the scores of documents linking to it, 2) a distance-weighted average of model scores for models nearby in the link graph, and 3) a site score

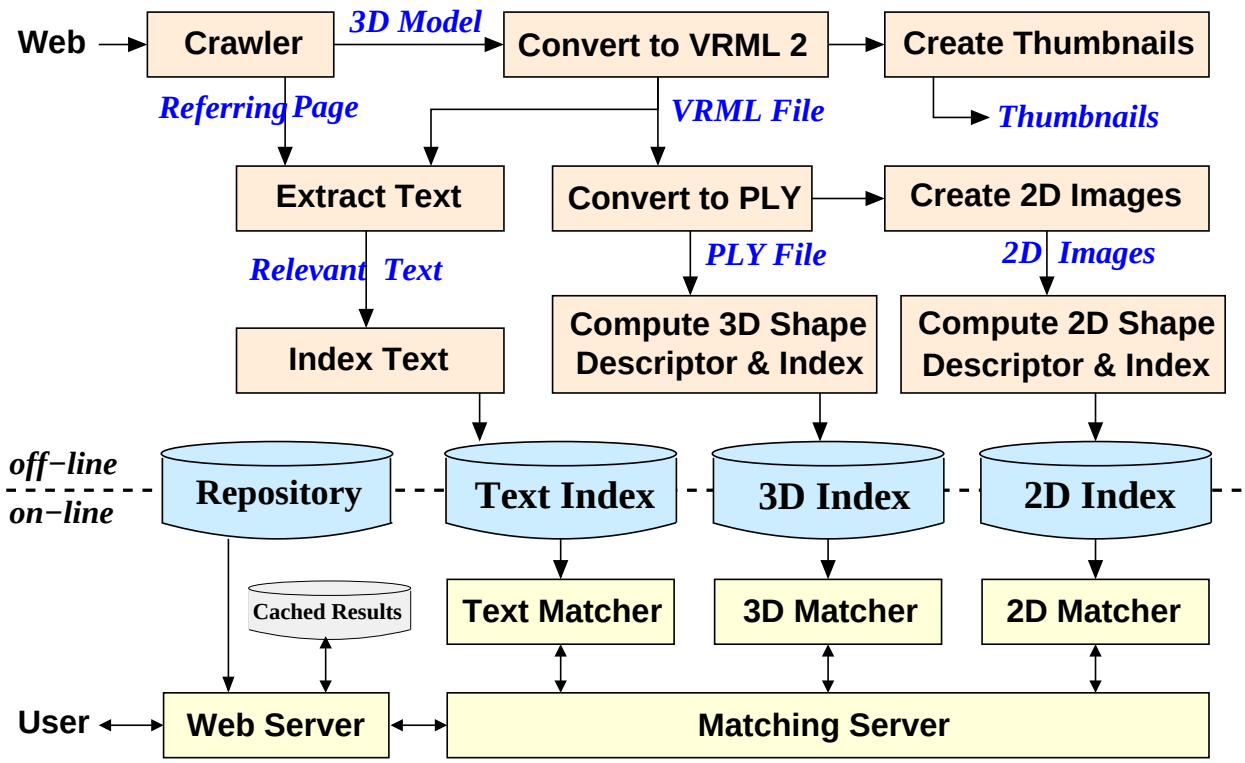


Figure 11: Flow of data through the search engine. The data shown in blue text is stored in the repository.

that reflects the proportion of documents retrieved from the site that are models. We maintain a hash table of visited URLs to avoid visiting pages more than once.

Every downloaded 3D model goes through several preprocessing steps. We convert first to the VRML 2 format (generally using PolyTrans [61]) and then to the Ply format in order to simplify parsing in later steps. Then, we extract text, create thumbnail and 2D contour images, and compute shape signatures. Once in a while, the recently downloaded models are added to the repository and all indices are updated. To compute the 3D shape descriptor for a model, we rasterize its polygonal surfaces into a 64x64x64 voxel grid, which is then decomposed into 32 concentric spheres. We compute the amplitudes of the first 16 harmonics for each sphere using SpharmonicKit [80]. The net result is a 16x32 shape descriptor. The shape analysis process takes 1.6 seconds per model with 3500 polygons on average (it is dominated by the time to rasterize polygons into the voxel grid). To compute the 2D shape descriptor for each thumbnail image, we downsample to 64x64 pixels and apply the same procedure. The 2D computation takes only 0.4 seconds per image.

For each query, the web server communicates via TCP to a matching server (running on a Dell Precision 530 PC with two 1.5GHz Pentium III processors and 1 GB of memory). There, a Perl job control script forks a separate process for each incoming query. Text queries are stemmed and passed directly to the Bow toolkit classifier `rainbow` [55]. For 2D sketches and uploaded 3D model files a shape signature is computed and compared against an in-memory index by a separate shape matching process. All match results (model ids, scores, statistics) are returned to the web server, which constructs a web page with results and returns it to the user. Match results are cached to enable fast browsing of multiple results pages.

9 Experimental Results

In this section, we report data collected during a series of experiments with our 3D search engine. The goals of these experiments are: (1) to evaluate how well our shape matching methods work, (2) to test whether shape can combine with text to provide more effective search tools, and (3) to characterize the experiences of people using our web site.

9.1 Shape Matching Results

In our first experiment, we aim to test how well our new 3D spherical harmonics matching algorithm finds similar objects. In order to investigate this question in a controlled manner, independent of user input, we ran a series of experiments in which we matched each model in a database with all others and analyzed how well the computed ranks correlate with a human's classification of the models.

While the purpose of the experiment is mainly to evaluate our matching algorithm, the results are indicative of how well our search engine works when a user provides his own 3D model and asks our system to find similar ones, or when a user clicks on the "Find Similar Shape" link under the image of an object returned by a previous query.

For this experiment, we used a test database with 1890 models of "household" and "miscellaneous" objects provided by Viewpoint [24]. The models contain between 120 and 120,392 triangles, with a median of 1,536 triangles per object (mean and standard deviation are 3,504 and 6,656, respectively). Every model came annotated with at least a few descriptive keywords (e.g., "chair, folding"). Objects were clustered into 85 classes based on functional similarities, largely following the groupings provided by Viewpoint. Examples from ten representative classes are shown in Figure 12. The smallest class had 5 models, the largest had 153 models, and 610 models did not fit into any meaningful class.

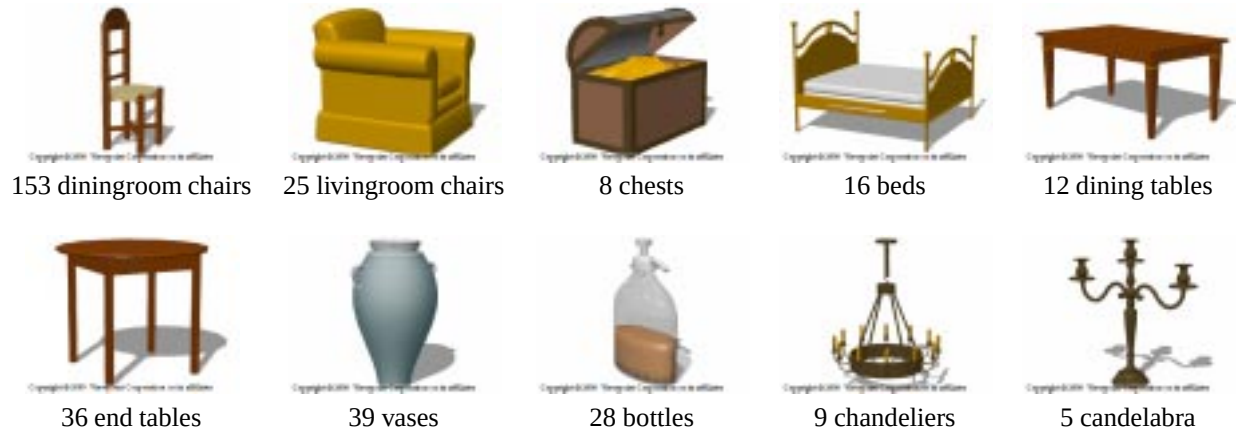


Figure 12: Samples from ten representative classes from the Viewpoint "household" and "miscellaneous" database (images courtesy of Viewpoint [24]).

We chose this Viewpoint database because it provides a representative repository of models with uniform quality and it is difficult for shape-based classification. In particular, several distinct

classes contain objects with very similar shapes. For example, there are five separate classes of chairs (153 dining room chairs, 10 desk chairs, 5 director's chairs, 25 living room chairs, and 6 lounge chairs, respectively). Meanwhile, there are objects spanning a wide variety of shapes (e.g., 8 forks, 5 cannons, 6 hearts, 17 plates of food, etc.). Thus, the database stresses the discrimination power of our shape matching algorithms while testing them under a variety of conditions.

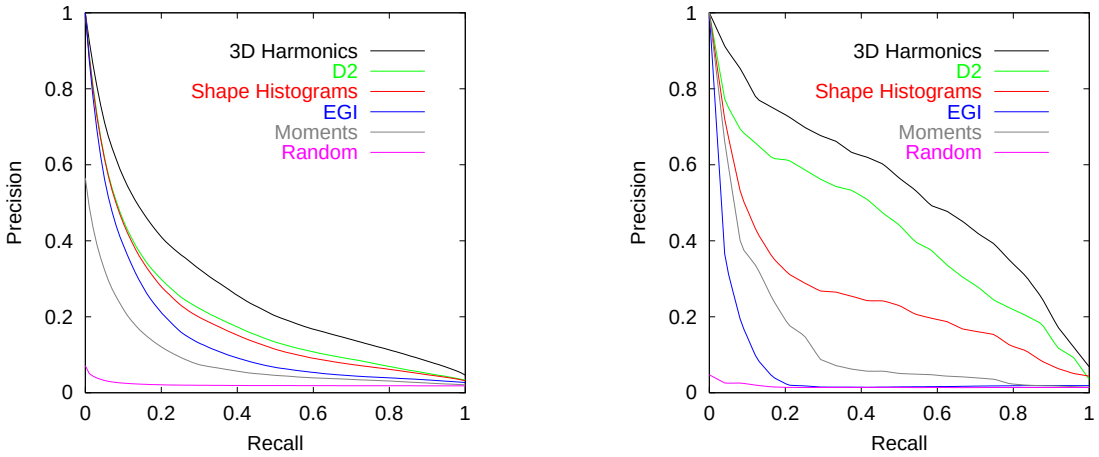
For the purpose of comparison to related approaches, we implemented five competing shape matching algorithms:

- **Random:** this method ranks all models in random order. It provides a baseline for evaluation of the other methods.
- **Moments:** this method characterizes the moments of inertia for points (x, y, z) on the surface S of an object ($m_{pqr} = \int_S x^p y^q z^r \, dx \, dy \, dz$). The first two moments (center of mass and principal axes) were used to register the models in a common coordinate system, and then the moments up to a sixth order were compared using a component-by-component L_2 difference (up to sixth order moments were chosen because they produce the best results for the test database). Our implementation follows the description in [30].
- **Extended Gaussian Images (EGI):** this method characterizes a 3D model in terms of its distribution of surface normal vectors [38]. We aligned the EGI for each model based on its principal axes, and we compared two aligned EGIs by computing their L_2 difference.
- **Shape Histograms:** this method characterizes the area of intersection with a collection of concentric spheres. The distribution of areas is normalized so that the overall volume is 1 and two distributions are compared by computing their L_2 difference (as in [4]).
- **D2 Shape Distributions (D2):** this method represents the shape of a 3D model by the distribution of Euclidean distances between pairs of points on its surface. The distribution for every model is normalized for scale by dividing by its mean, and two distributions are compared by computing their L_1 difference (as in [62]).

Figure 13(a) shows retrieval results obtained with our spherical harmonics shape matching algorithm as compared to the other methods. Each curve plots precision versus recall averaged over all classified models in the database. The plot axes can be interpreted as follows. For each target model in class C and any number K of top matches, "recall" represents the ratio of models in class C returned within the top K matches, while "Precision" indicates the ratio of the top K matches that are members of class C. A perfect retrieval result would produce a horizontal line along the top of the plot, indicating that all the models within the target object's class are returned as the top hits. Otherwise, plots that appear shifted up and to the right generally indicate superior retrieval results.

Note that for every recall value, spherical harmonics (black curve) gives better precision than the competing methods. On average, the precision values are 46% higher than D2, 60% higher than Shape Histograms, 126% higher than EGIs, and 245% higher than moments. The reasons are two-fold. First, matching based on moments and EGIs relies upon principal components to align models into a canonical coordinate system, and thus those methods tend to do poorly for classes of objects where the principal axes are not consistent. In contrast, our spherical harmonics descriptor is rotationally invariant, and thus it is less sensitive to such deviations within a class.

Second, the other shape descriptors blend shape information from different parts of an object, and thus they seem to have trouble discriminating fine details of objects. For instance, one can



(a) All classes (b) Living room chairs

Figure 13: Plots of precision versus recall of our spherical harmonics descriptor versus other shape matching methods.

view Shape Histograms as an implementation of our spherical harmonics method where only the zero-th order frequency is used. Our method describes objects up to rotations of multiple independent frequency components, and thus it achieves a nice combination of rotational invariance *and* discriminating power. As an example, Figure 13(b) shows retrieval results averaged over 25 queries with living room chairs. Although there are hundreds of other types of chairs and sofas in the database (e.g., 153 dining room chairs), our method is largely able to discriminate the different types and achieve high precision even in this difficult case.

Our spherical harmonics method also can be indexed effectively. Figure 14 shows the average time (in seconds) required to find the 16 closest matches in databases of increasing size. Note that the search time grows sublinearly, and the total search time for a database of 17,500 models is less than 0.25 seconds.

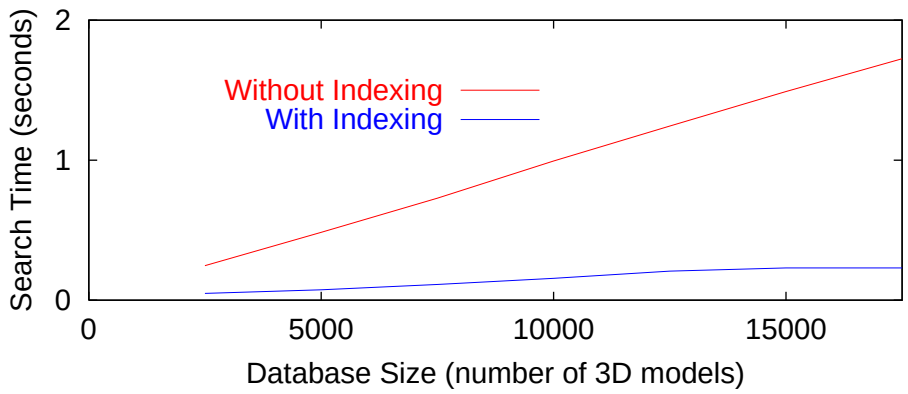


Figure 14: Search times (in seconds) for spherical harmonics with/without indexing.

9.2 Sketch Interface Results

In our second experiment, we investigated how well our system produces matches for queries entered by humans. Our hypothesis is that shapes are useful in conjunction with text for finding specific objects. To test this hypothesis, we ran an experiment where we compared the ease of input and descriptive power of text and 2D sketches provided by untrained users.

The subjects in this experiment were 43 students in an introductory computer science class (not for computer science majors). Each subject was given a pen and sheet of paper and told that their task was to write text and draw sketches that could be used by a search engine to retrieve "target objects" from a database of household objects. After the process was demonstrated once by the professor, the subjects performed the test for five target objects from the Viewpoint database (described in the previous section). For each test, the target object was shown rotating around on a projection screen at the front of a classroom. After fifteen seconds (three rotations), it disappeared, and the subjects were asked to write up to five text keywords and to draw three 2D sketches from front, side, and top views that distinguish it from other household objects. They were given two minutes for each target object, and no feedback was given after each object. After the experiment was completed, the students were asked to rate text and sketch queries based on "how easy" they were to construct and "how descriptive" they were for specifying the target objects (Table 1).

Query Interface	How Easy	How Discriminating
Text Keywords	8.0	6.2
2D Sketches	5.1	6.4

Table 1: Average student ratings of text and 2D sketch query interfaces on a scale from 1 to 10 (10 is best).

Later, we scanned their sketches and logged their keywords so that we could enter them as input to our search engine (example sketches for a chair and an elf are shown in Figure 15). Table 2 lists results achieved with queries using: 1) only their text keywords, 2) only their 2D sketches, and 3) both text keywords and 2D sketches combined in a multimodal query. For each query type, the table lists the median ranks of the target object and the percentages of the queries where the target object appeared among the top 16 matches. The latter statistic reflects how often the target would appear on the first page in our search engine.

Target Object Name	Median Rank (out of 1890)			% in Top 16		
	Only Text	Only Sketch	Both Combined	Only Text	Only Sketch	Both Combined
Chair	216	17	28	0.0%	46.2%	25.6%
Elf	10	12	2	89.7%	53.8%	97.4%
Table	100	571	252	5.1%	5.1%	10.3%
Cannon	7	40	2	82.1%	33.3%	89.7%
Bunkbed	3	64	2	89.7%	20.5%	89.7%

Table 2: Comparison of retrieval results with queries comprising only text, only 2D sketches, and both combined.

Figure 15: Sketches drawn by students to retrieve a specific chair (top three rows) and an elf (bottom three rows) during an in-class experiment.

The results in Table 2 suggest that text and shape can be complementary in the information they provide a search engine. For example, for the chair, text keywords were not discriminating enough to differentiate it from the hundreds of other chairs and related furniture in the database, and yet very simple sketches were able to describe it fairly precisely. On the other hand, simple text keywords picked out the five cannons and four bunk beds, while the 2D sketches were not as discriminating. Generally speaking, the text was effective at identifying classes of objects, while the sketches were helpful at selecting the best matches from within a class. The net result was that the combination of sketches and text usually produced a better match than either one alone.

9.3 Interactive Search Results

In our third experiment, we investigated how shape combines with text in interactive searches. While the results of the previous section suggest shape helps in a single query, a different question is whether it is still useful when users are allowed to iterate.

To study this question, we created two visually identical test versions of our web site, both comprising: (1) a box for typing text keywords, (2) buttons for viewing the next and previous page of match results, and (3) buttons labeled "Find Similar Object" under the thumbnails returned from previous queries. The only difference between the two web sites was in the way that searches could be iteratively refined. For one web site, clicking the "Find Similar Object" button retrieved models with the most similar *shapes* (as in Section 4). For the other, it retrieved models with the most similar *text documents* (as in Section 6). Our sketching interfaces were disabled during these experiments in order to isolate the effect of shape similarity to a single query modality.

We conducted an on-line experiment with 18 students from another introductory computer science class. Each student was asked to visit a URL, which was redirected to one of our two test web sites at random. He was led through a short tutorial that described how to use the search engine and then presented with the task of finding fifteen target objects (the same five listed in the previous section plus ten others selected randomly).

For each search, the target object was shown rotating on the web page for fifteen seconds (three rotations), after which it disappeared. Then, the student entered text keywords to initiate a search, and then iterated by either re-entering text, paging down/up, or finding similar objects until the target object was visible among the 16 matches on the results page.

Statistics logged during the students' sessions appear in Table 3. Columns 2 and 3 list the average time (in seconds) and number of iterations required for the student to find each target object. Column 4 indicates the percentage of students that found the target on the initial query. The last column indicates the percentage that had found it by the tenth iteration, at which time they were instructed to give up.

Similarity Measure	Search Time	Number of Iterations	Found on 1st Query	Found after $\leq$10 Iterations
Text	48	2.8	60%	77%
3D Shape	40	2.4	54%	89%

Table 3: Results of study with different iteration methods.

These results suggest that refining searches based on 3D shape similarity is useful in conjunction with text for finding specific objects. Using the web site equipped only with text matching, the students were able to find the target object within ten iterations only 77% of the time. In contrast, when the students were able to iterate by finding similar shapes, they found the target object more often (89% versus 77%), in fewer iterations (2.4 versus 2.8), and in less time (40 sec. versus 48 sec.). Moreover, we conjecture that students using shape-based iteration learned that they still could find objects quickly if they entered less descriptive keywords in their initial queries (iteration accounted for 35% of the objects found – i.e., 89% - 54%). Although this experiment was not a controlled study (the students performed the tests over the Internet using any computer on campus), the results are consistent with our expectation that shape can help discriminate specific objects more effectively than text alone.

9.4 Search Engine Results

Our 3D search engine has been publicly available on the Web since early November, 2001. It currently indexes 20,707 models. In this section, we report experiences about how people use the site.

Table 4 lists statistics gathered during one recent week of usage. During that period, the site processed 4,522 queries entered from 1,346 unique hosts (not counting local queries) in 55 different countries, and served 1,029 models to end users. The first row shows the numbers of queries for each type. After search results have been displayed, a user can (1) request more information about a downloaded model, (2) go to its referring page, or (3) download the actual model. The remaining rows in Table 4 shows the percentage of searches for which these events happened at least once.

While it is difficult to make conclusions from these statistics, we observe that people are willing to make shape-based queries. It will be interesting to see how the usage patterns change as our site grows.

	Text Only	Sketch Only	Text & Sketch	Similar Shape	Upload Model
Queries	3122	187	332	826	7
Get more info	32 %	29 %	29 %	36 %	14 %
Visit ref page	8 %	7 %	6 %	9 %	0
Download model	14 %	8 %	11 %	18 %	0

Table 4: Statistics gathered from one week's usage of our on-line search engine.

10 Conclusion

In summary, this paper investigates issues in building a search engine for 3D models. The main research contributions are: (1) new query interfaces that integrate text, 2D sketches, 3D sketches, and 3D models, (2) a new shape descriptor based on spherical harmonics that is both discriminating and robust, and (3) results of experiments suggesting shape is useful in conjunction with text during search for 3D models. Finally, we provide a large repository of 3D models ... and a way to find the interesting ones.

11 Future Work

This paper has just scratched the surface of research on shape-based retrieval and analysis for computer graphics. The following are just a few of the many topics that deserve further investigation:

- **New query interfaces:** it will be interesting to consider other methods for specifying shape-based queries. For instance, the following constraint-based description might be used to retrieve 3D models of a chair: "give me objects consisting of a box-shaped seat with four equal length and nearly cylindrical legs attached to the bottom side of each corner and a box-shaped back above the seat with width matching that of the seat, etc." This approach captures parameterized classes of objects with a compact description [26, 92].

- **New matching and indexing algorithms:** follow-up work should consider other types of shape matching problems. For instance, we currently compare whole objects, but it would be interesting to match partial objects as well. They could be used to find a car within a city scene or to find a Mercedes by looking for its hood ornament. Other matching algorithms might consider attributes of 3D models, including color, texture, structure, and animations.

- **New modeling tools:** future 3D modeling systems should consider integrating shape-based matching and retrieval methods into interactive sketching tools. For instance, consider a 3D model synthesis paradigm in which a user draws a rough sketch of a desired 3D model and the system "fills in the details" semi-automatically by suggesting matching detailed parts retrieved from a large database. In such a paradigm, the user could retain much of the creative control over model synthesis, while the system performs most of the tedious tasks required for providing model detail.

- **New applications:** it would be interesting to see whether the shape-based query and indexing methods described in this paper can be used for other applications, such as in mechanical CAD, medicine, and molecular biology.

In the near future, we expect that shape-based retrieval and analysis of 3D models will become a very important research area in computer graphics. This paper makes a small step in that direction.

Acknowledgements

We would like to thank Bernard Chazelle, Adam Finkelstein, Emil Praun, and Szymon Rusinkiewicz, who provided useful comments and insights during the writing of the paper. We also appreciate the cooperation of Brian Kernighan and the students of COS109 and COS111, who were willing to try out our shape-based search engine. Viewpoint and Jose Maria De Espona donated commercial databases of polygonal models for experiments. The National Science Foundation provided partial funding for this project under grants CCR-0093343, 11S-0121446, CCR-99-88173, and DGE-9972930. The Army Research Organization provided partial funding under grant DAAD19-99-1-0205. Thomas Funkhouser is partially supported by an Alfred P. Sloan Fellowship.

References

[1] F.J. Aherne, N.A. Thacker, and P.I. Rockett. Optimal pairwise geometric histograms. In *Proc. BMVC*, pages 480–490, Essex, UK, 1997.

[2] Helmut Alt and Leonidas J. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation: A survey. Technical Report B 96-11, EVL-1996-142, Institute of Computer Science, Freie Universität Berlin, 1996.

[3] Y. Amit, U. Grenander, and M. Piccioni. Structural image restoration through deformable templates. *J. Am. Statistical Assn.*, 86(440):376–387, 1991.

[4] Mihael Ankerst, Gabi Kastenmüller, Hans-Peter Kriegel, and Thomas Seidl. 3D shape histograms for similarity search and classification in spatial databases. In *Proc. SSD*, 1999.

[5] Mihael Ankerst, Gabi Kastenmüller, Hans-Peter Kriegel, and Thomas Seidl. Nearest neighbor classification in 3D protein databases. In *Proc. ISMB*, 1999.

[6] A.P.Ashbrook, N.A.Thacker, P.I.Rockett, and C.I.Brown. Robust recognition of scaled shapes using pairwise geometric histograms. In *Proc. BMVC*, pages 503–512, Birmingham, UK, July 1995.

[7] Klaus Arbter, Wesley E. Snyder, Hans Burkhardt, and Gerd Hirzinger. Application of affine-invariant fourier descriptors to recognition of 3-D objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):640–647, July 1990.

[8] Esther M. Arkin, L. Paul Chew, Daniel P. Huttenlocher, Klara Kedem, and Joseph S.B. Mitchell. An efficiently computable metric for comparing polygonal shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):209–216, March 1991.

[9] F. Arman and J.K. Aggarwal. Model-based object recognition in dense-range images - a review. *ACM Computing Surveys*, 25(1):5–43, 1993.

[10] Eric Bardinet, Sara Fernández Vidal, Sergio Damas Arroyo, Grégoire Malandain, and Nicolas Pérez de la Blanca Capilla. Structural object matching. Technical Report DECSAI-000303, Dept. of Computer Science and AI, University of Granada, Spain, February 2000.

[11] G. Barequet and S. Kumar. Repairing CAD models. *IEEE Visualization '97*, pages 363–370, 1997.

[12] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf. Parametric correspondence and chamfer matching: two new techniques for image matching. *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 659–663, 1977.

[13] Stefan Berchtold and Hans-Peter Kriegel. S3: Similarity search in CAD database systems. In Joan Peckham, editor, *Proc. SIGMOD*, pages 564–567. ACM, May 1997.

[14] Paul J. Besl and Ramesh C. Jain. Three-dimensional object recognition. *Computing Surveys*, 17(1):75–145, March 1985.

[15] P.J. Besl. Triangles as a primary representation. *Object Recognition in Computer Vision*, LNCS 994:191–206, 1995.

[16] P.J. Besl and N.D. McKay. A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[17] B. Bhanu. CAD-based robot vision. In *Computer*, volume 20, 1987.

[18] T.O. Binford. Visual perception by computer. *IEEE Conference on Systems Science and Cybernetics*, 1971.

[19] Jules Bloomenthal and Chek Lim. Skeletal methods of shape manipulation. In *Proc. Shape Modeling and Applications*, pages 44–47. IEEE, 1999.

[20] Harry Blum. A transformation for extracting new descriptors of shape. In Weiant Wathen-Dunn, editor, *Proc. Models for the Perception of Speech and Visual Form*, pages 362–380, Cambridge, MA, November 1967. MIT Press.

[21] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[22] Vittorio Castelli and Lawrence Bergman. *Image Databases: Search and Retrieval of Digital Imagery*. John Wiley & Sons, 2001.

[23] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.

[24] Viewpoint Corporation. http://www.viewpoint.com, 2001.

[25] B. Curless and M. Levoy. A volumetric method for building complex models from range images. *Computer Graphics (SIGGRAPH 96)*, August 1996.

[26] Paul Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proc. SIGGRAPH 1996*, pages 11–20. ACM, 1996.

[27] H. Delingette, M. Hebert, and K. Ikeuchi. Shape representation and image segmentation using deformable surfaces. *Image and vision computing*, 10(3):132–144, April 1992.

[28] H. Delingette, M. Hebert, and K. Ikeuchi. A spherical representation for the recognition of curved objects. In *Proc. ICCV*, pages 103–112, 1993.

[29] R.O. Duda, P.E. Hart, and David Stork. *Pattern Classification, Second Edition*. John Wiley & Sons, New York, 2001.

[30] Michael Elad, Ayellet Tal, and Sigal Ar. Content based retrieval of VRML objects - an iterative and interactive approach. In *EG Multimedia*, pages 97–108, September 2001.

[31] A.C. Evans, N.A. Thacker, and J.E.W. Mayhew. Pairwise representation of shape. In *Proc. 11th ICPR*, volume 1, pages 133–136, The Hague, the Netherlands, 1992.

[32] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, and Q. Huang. Query by image and video content: the QBIC system. *IEEE Computer*, 28(9):23–32, 1995.

[33] J. Foote. An overview of audio information retrieval. *ACM Multimedia Systems*, (7):2–10, 1999.

[34] James Gain and James Scott. Fast polygon mesh querying by example. *SIGGRAPH Technical Sketches*, page 241, 1999.

[35] W.E.L. Grimson. *Object recognition by computer: the role of geometric constraints*. MIT Press, Cambridge, MA, 1990.

[36] A. Gueziec, G. Taubin, F. Lazarus, and W. Horn. Converting sets of polygons to manifold surfaces by cutting and stitching. *IEEE Visualization '98*, pages 383–390, 1998.

[37] Masaki Hilaga, Yoshihisa Shinagawa, Taku Kohmura, and Tosiyasu L. Kunii. Topology matching for fully automatic similarity estimation of 3D shapes. *Computer Graphics (SIGGRAPH 2001)*, pages 203–212, August 2001.

[38] B.K.P. Horn. Extended gaussian images. *Proc. of the IEEE*, 72(12):1671–1686, December 1984.

[39] D. Huttenlocher, D. Klanderman, and A. Rucklige. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[40] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3D freeform design. In *Proc. SIGGRAPH 1999*, pages 409–416, Los Angeles, CA, August 1999. ACM.

[41] K. Ikeuchi and P.J. Flynn. Recent progress in CAD-based vision. *Comput. Vis. Image Understanding*, 61(3), 1995.

[42] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. of 30th STOC*, 1998.

[43] De Espona Infografica. 3D model collection. http://www.deespona.com.

[44] Charles E. Jacobs, Adam Finkelstein, and David H. Salesin. Fast multiresolution image querying. *Proceedings of SIGGRAPH 95*, pages 277–286, August 1995.

[45] A. Jain, Y. Zhong, and S. Lakshmanan. Object matching using deformable templates. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(3):267–278, 1996.

[46] Andrew E. Johnson and Martial Hebert. Using spin-images for efficient multiple model recognition in cluttered 3-D scenes. *IEEE PAMI*, 21(5):433–449, 1999.

[47] R. Kashyap and R. Chellappa. Stochastic models for closed boundary analysis: Representation and reconstruction. *IEEE Transactions on Information Theory*, 27:627–637, 1981.

[48] Gabi Kastenmüller, Hans-Peter Kriegel, and Thomas Seidl. Similarity search in 3D protein databases. In *Proc. GCB*, 1998.

[49] Y. Lamdam and H.J. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *Proc. ICCV*, December 1988.

[50] Y. Lamdan, J. Schwartz, and H. Wolfson. Affine invariant model-based object recognition. *IEEE Trans. on Robotics and Automation*, 6:578–589, 1990.

[51] Michael Lesk. *Practical Digital Libraries*. Morgan Kaufmann Publishers, San Francisco, CA, 1997.

[52] Y. Lin, J. Dou, and H. Wang. Contour shape description based on an arch height function. *Pattern Recognition*, 25:17–23, 1992.

[53] Sven Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, 1998.

[54] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, 1985.

[55] A.K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/˜ mccallum/bow, 1996.

[56] MeshNose. http://www.deepfx.com/meshnose, 2001.

[57] George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[58] T.M. Murali and Thomas Funkhouser. Consistent solid and boundary representations from arbitrary polygonal data. *Computer Graphics (1997 SIGGRAPH Symposium on Interactive 3D Graphics)*, pages 155–162, March 1997.

[59] H. Murase and S. Nayar. Visual learning and recognition of 3D objects from appearance, 1995.

[60] V. Ogle and M. Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, 1995.

[61] Okino. Polytrans. http://www.okino.com/conv/conv.htm, 2001.

[62] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Matching 3D models with shape distributions. *Shape Modeling International*, May 2001.

[63] S. Palmer, E. Rosch, and P. Chase. Canonical perspective and the perception of objects. *Attention and Performance IX*, pages 135–151, 1981.

[64] E. Paquet and M. Rioux. Nefertiti: A tool for 3-D shape databases management. *SAE Transactions: Journal of Aerospace*, 108:387–393, 2000.

[65] A. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(7):715–729, 1991.

[66] Arthur R. Pope. Model-based object recognition: A survey of recent research. Technical Report TR-94-04, University of British Columbia, January 1994.

[67] M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[68] Richard J. Prokop and Anthony P. Reeves. A survey of moment-based techniques for unoccluded object representation and recognition. *CVGIP: Graphical Models and Image Processsing*, 54(5):438–460, 1992.

[69] William Regli. National design repository. Geometric and Intelligent Computing Laboratory, Drexel University, http://repos.mcs.drexel.edu, 2001.

[70] Rudy De Reyna. *How to Draw What You See*. September 1996.

[71] J. Rocchio. *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter Relevance Feedback in Information Retrieval, pages 313–323. Prentice-Hall, 1971.

[72] Jeremy Rowe, Anshuman Razdan, Dan Collins, and S. Pachanathan. A 3D digital library system: Capture, analysis, query, and display. *4th International Conference on Digital Libraries (ICADL)*, December 2001.

[73] Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image databases. In *Proc. 6th ICCV*, pages 59–66, Bombay, India, January 1998.

[74] G. Salton. *The SMART retrieval system*. Prentice-Hall, Englewood Cliffs, NJ, 1971.

[75] D. Saupe and D. V. Vrani. 3D model retrieval with spherical harmonics and moments. *DAGM 2001*, pages 392–397, September 2001.

[76] Utsav Schurmans, Anshuman Razdan, Arleyn Simon, Peter McCartney, Mary Marzke, David Van Alfen, Gram Jones, Jeremy Rowe, Gerald Farin, Daniel Collins, Mary Zhu, Dezhi Liu, and Myungsoo Bae. Advances in geometric modeling and feature extraction on pots, rocks and bones for representation and query via the internet. *Computer Applications in Archaeology (CAA)*, April 2001.

[77] Ali Shokoufandeh, Sven J. Dickinson, Kaleem Siddiqi, and Steven W. Zucker. Indexing using a spectral encoding of topological structure. In *Proc. Computer Vision and Pattern Recognition*, volume 2, pages 491–497. IEEE, 1999.

[78] Kaleem Siddiqi, Ali Shokoufandeh, Sven J. Dickinson, and Steven W. Zucker. Shock graphs and shape matching. *Int. Journal of Computer Vision*, 35(1):13–31, 1999.

[79] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):131– 147, February 1990.

[80] SpharmonicKit 2.5. Fast spherical transforms: Spharmonickit. http://www.cs.dartmouth.edu/˜ geelong/sphere/, 1998.

[81] Duane W. Storti, George Turkiyyah, Mark A. Ganter, Chek T. Lim, and Derek M. Stal. Skeleton-based modeling operations on solids. In *Proc. Solid Modeling*, pages 141–154. ACM, 1997.

[82] Motofumi T. Suzuki. A web-based retrieval system for 3D polygonal models. *Joint 9th IFSA World Congress and 20th NAFIPS International Conference (IFSA/NAFIPS2001)*, pages 2271–2276, July 2001.

[83] C. Tappert. Cursive script recognition by elastic matching. *IBM Journal of Res. Develop.*, 26(6):765–771, 1982.

[84] G. Taubin and D.B. Cooper. *Geometric Invariance in Computer Vision*, chapter Object recognition based on moment (of algebraic) invariants. MIT Press, 1992.

[85] D. Terzopoulos and D. Metaxas. Dynamic 3D models with local and global deformations: Deformable superquadrics. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(7):703–714, 1991.

[86] W. Tsai and S. Yu. Attributive string matching with merging for shape recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7:453–462, 1985.

[87] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. *Computer Graphics (SIGGRAPH 94)*, pages 311–318, 1994.

[88] C. Uras and A. Verri. On the recognition of the alphabet of the sign language through size functions. In *Proc. IAPR*, pages 334–338, Jerusalem, Israel, 1994.

[89] Remco C. Veltkamp. Shape matching: Similarity measures and algorithms. In *Shape Modelling International*, pages 188–197, May 2001.

[90] Remco C. Veltkamp, Hans Burkhardt, and Hans-Peter Kriegel. *State-of-the-Art in Content-Based Image and Video Retrieval*. Kluwer Academic Publishers, 2001.

[91] D. V. Vranic, D. Saupe, and J. Richter. Tools for 3D-object retrieval: Karhunen-Loeve transform and spherical harmonics. *IEEE 2001 Workshop Multimedia Signal Processing*, pages 293–298, October 2001.

[92] Andrew Witkin, Kurt Fleischer, and Alan Barr. Energy constraints on parameterized models. In *Proc. SIGGRAPH 1987*, pages 225–232. ACM, 1987.

[93] K. Wu and M.D. Levine. Recovering parametric geons from multiview range data. In *Proc. CVPR*, pages 159–166, June 1994.

[94] I. Young, J. Walker, and J. Bowie. An analysis technique for biological shape. *Computer Graphics and Image Processing*, 25:357–370, 1974.

[95] C.T. Zahn and R.Z. Roskies. Fourier descriptors for plane closed curves. *IEEE Trans. Computers*, 21:269–281, 1972.

[96] Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. Sketch: An interface for sketching 3D scenes. *Computer Graphics (SIGGRAPH 96)*, pages 163–170, August 1996.

[97] Dongmei Zhang and Martial Hebert. Harmonic maps and their applications in surface matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '99)*, 1999.