First publ. at: ACM International Conference on Multimedia Retrieval (ICMR2011), April 17-20, 2011, Trento, Italy

STELA: Sketch-Based 3D Model Retrieval using a Structure-Based Local Approach

Jose M. Saavedra
PRISMA Research Group
DCC, University of Chile
Av. Blanco Encalada 2120
Santiago, Chile
jsaavedr@dcc.uchile.cl

Benjamin Bustos
PRISMA Research Group
DCC, University of Chile
Av. Blanco Encalada 2120
Santiago, Chile
bebustos@dcc.uchile.cl

Maximilian Scherer
GRIS, TU Darmstadt
Fraunhoferstr, 5
64283, Darmstadt, Germany
maximilian.scherer@
gris.tu-darmstadt.de

Tobias Schreck
GRIS, TU Darmstadt
Fraunhoferstr, 5
64283, Darmstadt, Germany
tobias.schreck@
gris.tu-darmstadt.de

ABSTRACT

Since 3D models are becoming more popular, the need for effective methods capable of retrieving 3D models are becoming crucial. Current methods require an example 3D model as query. However, in many cases, such a query is not easy to get. An alternative is using a hand-draw sketch as query. We present a structure-based local approach (STELA) for retrieving 3D models using a rough sketch as query. It consists of four steps: get an abstract image, detect keyshapes, compute a local descriptor, and match local descriptors. We represent a 3D model by means of suggestive contours. Our proposal includes an additional step aiming at reducing the number of models that will be compared by our local approach. The proposed method is invariant to position, scale, and rotation changes as well. We evaluate our method using the first-tier precision and compare it with a current global approach (HELO). Our results show an increasing in precision for many classes of 3D models.

Categories and Subject Descriptors

I.4.9 [**Image Processing and Computer Vision**]: Applications—*3D Model Retrieval*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.3.8 [**Computer-Graphics**]: Miscellaneous

General Terms

Algorithms

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ICMR '11, April 17-20, Trento, Italy
Copyright ©2011 ACM 978-1-4503-0336-1/11/04 ...$10.00.

Keywords

Sketch-based multimedia retrieval, 3D model retrieval, local descriptors.

1. INTRODUCTION

In the last years there have been wide interest and progress on computer aided retrieval of multimedia data. The advances in this area have allowed users to look for a multimedia object in large repositories in a more efficient way . As advances in multimedia retrieval increase, new interesting applications come up. One of the current interesting applications is 3D model retrieval with impact extending from design to medical issues [4].

The simplest way for retrieving 3D models is by means of textual metadata describing the 3D objects. This requires 3D models to have reliable metadata. However, 3D models not always come with reliable human tags. Although, many authors have addressed the multimedia data annotation problem, this is still an open problem [23, 24]. Due to this fact, the ongoing research on multimedia retrieval relies on a content-based approach [8].

Although there is a lot of research on content-based image retrieval, the 3D model retrieval problem is still a young area. In fact, recently the typical question "How do we construct 3D models?"is shifting to "How do we find them?" [11].

In the context of content-based 3D model retrieval, several approaches to compute similarity between two 3D models have been proposed [21, 4]. Among these methods are *shape histogram*[1], *shape distribution*, [19], *moments*, [10], *light field* [6], *spherical harmonics* [11]. Following any of these approaches, users require a 3D model as an example for querying.

A 3D model example as query is not always available, requiring some kind of technical expertise to produce it. Even though some tools for making the 3D modeling task easy for any kind of users (e.g. Google Sketchup) are coming up, they are still hard to operate and produce detailed models on the fly. This fact clearly limits the 3D model retrieval usability.

An easy alternative for querying is simply drawing a 2D

Konstanzer Online-Publikations-System (KOPS)
URL: http://nbn-resolving.de/urn:nbn:de:bsz:352-152287

Figure 1: The Suggestive Contours from two different viewpoints of a cup.

stroke-based sketch lacking of color and texture. Although such a kind of sketch is composed of few lines, it is a coarse but detailed representation of what the user is looking for, which includes key features. This leads to *the sketch-based 3D model retrieval*.

In this work we are interested in rough sketches that novice users can draw easily. One important issue here is how to compare a 3D model with a 2D sketch. To this end, some strategies project and render the model from different viewpoints getting 2D representations. Then, they process these 2D representations as in the context of sketch-based image retrieval using, for instance, HELO [20] or HOG [7] descriptors.

An interesting technique for getting a 2D representation from a 3D model is *suggestive contours* [9]. This technique from non-photorealistic rendering resembles hand drawings of three dimensional objects very closely. Yoon et al. [25] showed that this technique performs better than the classical *contour* or *ridge and valley* techniques for retrieval tasks. Therefore, we use *suggestive contours* of the 3D models as 2D representations. An example of the suggestive contour of a 3D model is depicted in Figure 1.

The other critical issue is how to represent appropriately rough sketches and 3D model projections for the retrieval task. We claim that different sketches and 3D model projections from the same object class must share structural and locality information. This information must be taken into account for making up a descriptor to boost the retrieval performance. Even though there have been proposed structure-based methods for 3D model retrieval [13, 22], they work only when the query is another 3D model. To our knowledge, the structural and locality information have not been exploited yet when the query is a rough sketch.

In this paper we propose STELA (STructurE-based Local Approach), a local approach for sketch based 3D model retrieval that takes into account *structural* and *locality* information not only of the query sketches but also of the 2D representations of a 3D model. Detecting the components which make up an object is an expensive task, so instead of detecting such components, our method relies on detecting *keyshapes*. A *keyshape* is a primitive shape from which a complex object is composed of. Moreover, structural information allows us to represent an object in a higher abstraction level with the capability of dealing with noise. This leads to an improvement on the retrieval task.

Furthermore, we combine our local approach with a global one to take advantage of the global similarity in order to reduce the number of necessary comparisons between a query sketch and the models of the training database. The global approach works as a filtering stage, this allows us not only to increase the efficiency of the proposed local method but also to improve the retrieval effectiveness reducing the number of false positives.

The results of our proposal show an increasing in precision with respect to current strategies applied for sktech-based 3D model retrieval. Particularly, our method achieves significant improvement over 3D models with a well defined structure as explained later in this paper.

The remaining part of this document is organized as follows. Section 2 briefly present the related work in the area of 3D model retrieval discussing the current work for the sketch-based approach. Section 3 describes in detail our local approach. Section 4 describes a global descriptor for the filtering step. Section 5 discusses the conducted experiments and analyses the achieved results. Finally, Section 6 presents some conclusions.

2. RELATED WORK

Although a lot of researchers have been working on content-based 3D model retrieval, the sketch-based approach, where the input is a rough line-based hand drawing, has not followed the same direction.

The 3D shape matching approach is possibly the most common approach for 3D model retrieval. A comprehensive study of shape-based 3D model retrieval is discussed in the work of Bustos et al. [4] and in the work of Tangelder et al. [21]. In those works *feature based methods* and *graph based methods* are discussed. Even though several of the discussed methods have a good performance on different kind of applications, they only work when the query is another 3D model, limiting their use in our case.

Since we use 2D representations from the 3D models to compare them against query sketches, an option is using the 2D shape matching approach. A survey on these techniques can be found in the work of Loncaric [16]. These techniques include *boundary-based representations*, *regions-based representations*, *boundary space domain* and *global space domain*. However, these techniques require to have a closed shape or a region represented in a binary format, which is far from our rough sketches. This fact makes the classical 2D shape matching impractical for our work.

Regarding the use of suggestive contour to get a 2D representation from the 3D model, we could use some techniques used in the context of sketch-based image retrieval (SBIR). The input sketch and the suggestive contour are formed just by edges, so the orientation of the edges could be used as a natural discriminative feature between different 2D objects. The *Histogram of Oriented Gradients* (HOG) [7] has shown being useful for detection and recognition tasks. Furthermore, a variant of this technique has been used in the work of Yoon et al. [25] to retrieve 3D models from sketches. More recently, Saavedra et al. [20] proposed a technique for computing an improved *histogram of edge local orientations* (HELO) dealing with noise and increasing the effectiveness in the context of SBIR. An important drawback of HOG and HELO descriptors is that they are global representations, getting easily confused when the sketches and the suggestive contours are depicting simple models as it happens in our case. Furthermore, they get confused easily if we have many views for the same 3D model. Figure 2 shows a sketch and a suggestive contour of a 3D model with their corresponding HELO descriptor. Both the descriptors look very

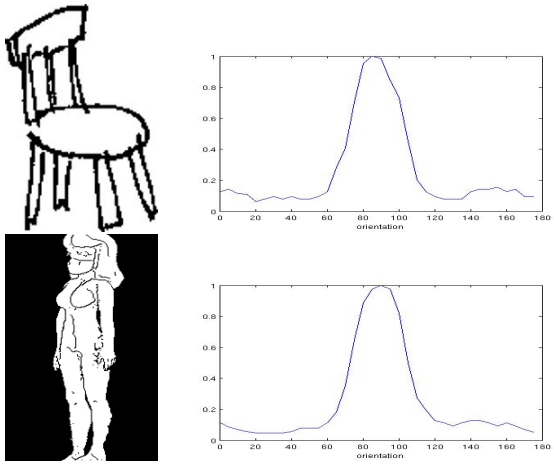

Figure 2: The behavior of a oriented-based global approach for comparing an sketch with a 3D model suggestive contour.

similar although the contours are not.

We claim that a local representation which takes into account not only the structural information but also the locality information of 2D representations is needed for tackling the sketch-based 3D model retrieval problem. Structural information tell us what components makes up a certain object, as the locality information allows us to set some relation between a reference component and those around it. For instance, an sketch representing a human should be composed of a head, two arms and two legs, this is the *structural information*. On the other hand we know that a head is located between the arms, moreover, the head is closer to the arms than to the legs. This is what we call *locality information* which takes into account spatial relationship between object components.

The well known SIFT [17] and Shape Context [3] are two of the most relevant local descriptors applied for 2D images. Even though these are local descriptors, they do not represent appropriately the structural information on the image as they rely merely on *keypoints*. In our case both a sketch and a 3D model are represented as edge maps. Since edge maps are commonly affected by noise, *keypoints* could fall into the noise regions leading into a detriment of the retrieving performance. Thus, we need another kind of approach that can be deal with noise representations.

Therefore, the main contribution of this work is to present a local approach for sketch based 3D model retrieval, taking advantage of the locality and structural information provided by the input sketches as well as by the suggestive contours representing the 3D models. Moreover, structural information allows us to represent an object in a higher abstraction level with the capability of dealing with noise. This leads to an improvement on the retrieval task. Additionally, we use a filtering step to select candidates with a global shape similar to the query sketch. This filtering step allows us not only the retrieval efficiency but also to improve the retrieval precision.

3. STELA

In this section we describe in detail our proposed local approach (STELA) for sketch based 3D model retrieval. It is necessary to be aware that this method requires 2D representations as input. The query sketch is already a 2D representation, but 3D models are not. To get 2D representations from 3D models we will use suggestive contours, as mentioned before. The main property of our approach is that it takes advantage of the structural information as well as of the locality information over the sketches and suggestive contours.

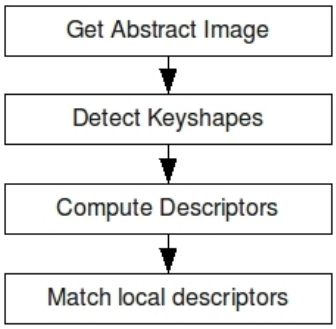

Figure 3: The pipeline of the proposed local approach.

For getting structural information, an object (a sketch or suggestive contour image) should be decomposed into simpler shapes. A first idea to this end is to decompose an object into shapes like squares, circles, ellipses, lines, triangles, etc. However, this is a non trivial task because our objects are formed only by edges and possibly these shapes are not well defined. An alternative would be to simply consider the most simple shapes such as straight lines, arcs, and circular shapes including ellipses. Bo Yu [26] proposed an interesting technique to detect these shapes, although his method needs strokes to be detected first.

For getting locality information, we need a local descriptor. A local approach requires to define local regions. Commonly, this is carried out by means of *keypoints* [18]. However, as indicated in the previous section, *keypoints* do not represent appropriately structural components. Instead of using *keypoints* we propose to use *keyshapes*, where *keyshapes* are defined as simple *2D primitive shapes* that form a complex 2D object. *keyshapes* are close related to the structural information. In this way, we represent the object using a higher lever information, allowing us to reduce the semantic gap in the retrieval task.

Our proposal is composed of the following steps: (1) *get an abstract image*, representing in a simpler way a sketch or suggestive contour image. (2) *detect keyshapes*, that should return a set of simple primitive shapes from an abstract image, (3) *compute local descriptors*, that computes a signature for each *keyshape*, and (4) *matching*, allowing us to set a correspondence mapping between local descriptors of a query sketch and local descriptors of a suggestive contour image representing a 3D model from a certain viewpoint. These four steps are shown in Figure 3. The next subsections describe each step in detail.

3.1 Abstract Image

To detect keyshapes easily, a preprocessing task is required. This task aims to get an abstract image from the query sketch and suggestive contours. The abstract image

allow us to reduce the effect of noise, keeping only relevant edges. To this end, we apply the *thinning* operator [12] over the query sketch, and the *canny* operator [5] over the suggestive contour. After this, we obtain edge map representations from both the query sketch and the suggestive contour.

Another important task here is to represent the edges as strokes. By achieving this, we will be capable of determining the shape of each stroke. However, detecting strokes in a sketch or suggestive contour image is a hard task. We approximate strokes using the *edgelink* operator [14] over an edge map representation. In this case, the *edgelink* operator returns a set of edge lists.

3.2 Detecting Keyshapes

As mentioned before, detecting simple shapes composing a more complex object allows us to get similarity between two objects taking into account structural information. In this work we only detect straight lines which form the object. Lines can be detecting easily and still keep enough discriminative information. We refer to these simple shapes as *keyshapes* following the same idea of *keypoints*.

To this end, we use the abstract image, detected in the previous step, which is formed by edge lists. For each each *edge list* we could detect one or more straight lines composing the edge list following the next steps for each edge list:

1. Take the line L joining the end points of a edge list.
2. Finds the value m and position p of the maximum deviation of L.
3. if $m <= \mu$, add L to the set of detected lines.
4. If $m > \mu$, cut L into new lines L_1, L_2 and repeat the process for each line.

where, μ is a tolerance threshold. We set empirically $\mu = 3$.

The resulting set of lines contains lines of different size. It is worth pointing out that curve strokes will yield a set of very small lines. So, to get a set of lines representing appropriately straight strokes we need a threshold T_{short} to reject small lines. In addition, considering the possible discontinuities of strokes on the sketches, a process for merging nearby lines with similar slope is required. We use a threshold T_{near} to evaluate nearness between lines. Furthermore, lines with length above a threshold T_{large} should be split into smaller ones. We propose to use the following threshold values: $T_{short} = D * 0.05$, $T_{near} = 5$, $T_{large} = D * 0.5$, where D is the length of the abstract image diagonal. The final set of lines represent the set of *keyshapes*.

Finally, we regard the center of each line as the representative point of each *keyshape*. Figure 4 shows two suggestive contours with their corresponding abstract representations (second image) and the detected *keyshapes* (third image). In our proposal, each *keyshape* L is represented as a 5-tuple $[(x_1, y_1), (x_2, y_2), (x_c, y_c), s, \phi]$, where (x_1, y_1) is the start point, (x_2, y_2) is the end point, (x_c, y_c) is the representative point, s is the line length, and ϕ is the slope. Although the two first components representing L are enough to compute the remaining three components, we decided to keep the 5-tuple notation just for making our algorithm easily understood.

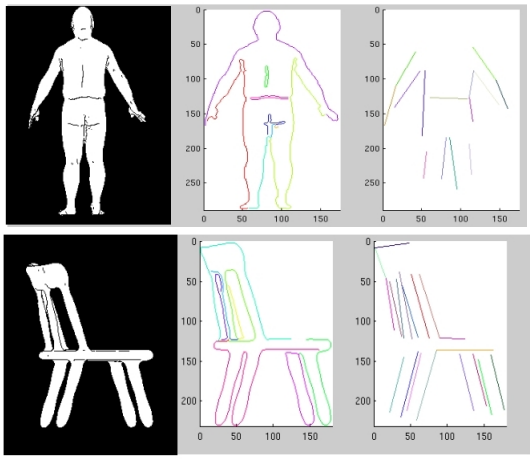

Figure 4: First column shows the suggestive contour of two 3D models, second column shows the corresponding abstract images, and third column shows the detected *keyshapes*.

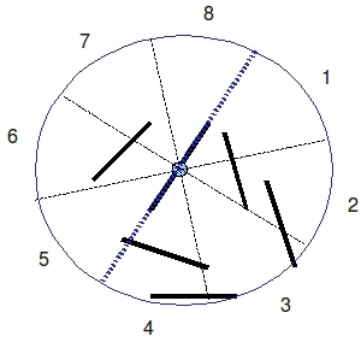

Figure 5: A synthetic representation of the partitioning to make up the proposed local descriptor.

3.3 The Local Descriptor

Different descriptors could be used in this step. For instance, an extension of Shape Context [3] to work over keyshapes instead of working over a point sampling is an alternative. However, this choice could yield a sparse descriptor considering that the number of keyshapes is much smaller than the number of sampled points.

Therefore, in our approach we use an *oriented angular 8-partitioning descriptor*. Figure 5 depicts a graphical representation of this descriptor.

Having a keyshape L as reference, this descriptor works as follows:

- Create a vector h, containing 8 cells. Initially, $h(i) = 0$, $i = 1 \ldots 8$.
- Let L be the reference keyshape represented as :

$$L = [(x_1, y_1), (x_2, y_2), (x_c, y_c), s, \phi']. \quad (1)$$

- Let $f_r : \mathbf{R}^2 \rightarrow \mathbf{R}^2$ be a rotation function around the point (x_1, y_1) with rotation angle $\beta = -\phi$. This func-

tion is defined as below:

$$\begin{aligned} f_r(x,y) &= (x_r, y_r), \text{where,} \quad (2)\\ x_r &= [(x-x_1)cos(\beta) - (y_1-y)sin(\beta)] + x_1\\ y_r &= y_1 - [(x-x_1)sin(\beta) - (y_1-y)cos(\beta)] \end{aligned}$$

- Let $(\hat{x}_c, \hat{y}_c) = f_r(x_c, y_c)$ be the normalized version of (x_c, y_c).
- For each keyshape $Q \neq L$ represented by $[(x'_1, y'_1), (x'_2, y'_2), (x'_c, y'_c), s', \phi']$.
  - Get $(\hat{x}'_c, \hat{y}'_c) = f_r(x'_c, y'_c)$.
  - $D_x = \hat{x}'_c - \hat{x}_c$ and $D_y = \hat{y}'_c - \hat{y}_c$
  - If $D_x > 0 \wedge D_y > 0 \wedge |D_x| > |D_y|$, $bin = 1$.
  - If $D_x > 0 \wedge D_y > 0 \wedge |D_x| <= |D_y|$, $bin = 2$.
  - If $D_x < 0 \wedge D_y > 0 \wedge |D_x| <= |D_y|$, $bin = 3$.
  - If $D_x < 0 \wedge D_y > 0 \wedge |D_x| > |D_y|$, $bin = 4$.
  - If $D_x < 0 \wedge D_y < 0 \wedge |D_x| > |D_y|$, $bin = 5$.
  - If $D_x < 0 \wedge D_y < 0 \wedge |D_x| <= |D_y|$, $bin = 6$.
  - If $D_x > 0 \wedge D_y < 0 \wedge |D_x| <= |D_y|$, $bin = 7$.
  - If $D_x > 0 \wedge D_y < 0 \wedge |D_x| > |D_y|$, $bin = 8$.
  - $h(bin) = h(bin) + s'/MAX_{LEN}$, where MAX_{LEN} is the length of the abstract image diagonal. This is a normalization parameter depending on the image size.
- Finally, $h(bin) = \dfrac{h(bin)}{\sum_{i=1}^{8} h(i)}$, $bin = 1 \ldots 8$.

3.4 Matching

In our approach, we treat an object, a query sketch or suggestive contour image, as a set of descriptors. This set captures the object shape.

Let $P = \{p_1, p_2, \ldots, p_m\}$ be the set of descriptors representing a query sketch, and $Q = \{q_1, q_2, \cdots q_n\}$ be the set of descriptors representing a suggestive contour obtained from a 3D model from a certain viewpoint. Here, $p_i, q_j \in \mathbf{R}^8$. Without loss of generality, we will suppose $n < m$. So, we need to find an assignment from Q to P. This is, for each q_j we need to look for the p_i that allow us to minimize an overall cost. We define the function $\pi : \{1, \ldots, n\} \rightarrow \{1, \ldots, m\}$ that map the j-th descriptor from Q with the i-th descriptor from P.

Furthermore, we define the cost T of the assignment using a certain mapping function π as follows:

$$T(\pi) = \sum_{i=1}^{n} C(q_i, p_{\pi(i)}) \quad (3)$$

where, $C(q,p)$ is the cost of matching a descriptor $q \in Q$ with $p \in P$. This cost function could be thought as the distance between p and q. In this way, the less similar the descriptors are, the more expensive the match become. As our proposal descriptor is, in fact, a probability distribution, we use the χ^2 test statistic:

$$C(q,p) = \frac{1}{2} \sum_{i=1}^{8} \frac{[q(i) - p(i)]^2}{q(i) + p(i)} \quad (4)$$

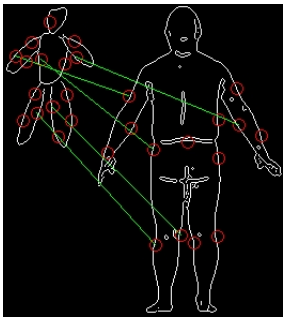

Figure 6: Matching between a sketch (top image) and a suggestive contour (bottom image).

Therefore, the problem of minimizing the overall cost is defined as:

$$\pi^\star = argmin T(\pi) \quad (5)$$

This problem may be regarded as an instance of the *Bipartite Graph Matching*. Different from the case of classical local methods for the image context, our number of descriptor per image is much lower. So, in our case we will resolve the assignment problem applying the Hungarian Method [15].

After the assignment stage, we need to look for a representative pose transformation between the matched descriptors. This will allow us to achieve a more consistent matching. To this end, we will use the stored information of the corresponding *keyshape* (see Eq. 1). We are only interested in finding the scale and position transformation. The position is represented by the center of the keyshape (x_c, y_c) as the scale is represented by the keyshape length s.

For estimating the pose transformation, we use the Hough Transform [2], where each candidate match must vote just for three parameters (scale, translation in x-axis and in y-axis). We keep the set of parameter with the highest vote. This set of transformation parameters characterizes the estimated pose. Only the matches which agree with the estimated pose are retained for the next process, the others are discarded.

Finally, the similarity between a sketch and a suggestive contour is computed as the average cost of the matched descriptors. The cost of the unmatched suggestive contour descriptors are set to 1. Figure 6 shows an example of this matching step.

3.5 Invariance issues

Our local approach is robust under positions, scale, and rotation changes. The translation invariance is directly derived as our descriptor extract local information. We achieve scale invariance normalizing the length of the keyshape by the MAX_{LEN}, a parameter depending on the image size. Finally, we get rotation invariance making the keyshape be coincident with the x-axis of the partitioning system as shown in Figure 5.

4. AN ADDITIONAL FILTERING STEP

Taking into account that local approach is commonly expensive in time, we add a filtering step, where a reduced number of suggestive contour images are selected. In addition, this filtering step allows us to get an improvement in precision, reducing the number of false positives that could

arise from the fact that we have 14 suggestive contour images for each 3D model. That means, having many viewpoints for each 3D model could make the method get confused during the retrieval process.

Each chosen suggestive contour represents a different 3D model. Our filter considers the global shape of the objects, therefore only 3D model with global shape similar to that of the query are kept. To this end, we use HELO [20] as the global descriptor with a slight variation.

In this approach, instead of applying HELO just over the whole image, we apply it over three kind of zones, leading to three types of HELO:

1. HELO : This is applied over the whole image, using a 36-size descriptor.

2. HELO_V: This is applied over four equal-sized vertical regions, juxtaposing each descriptor to make up the global one. In this case, since the HELO_V is applied over smaller regions, we use 18-size descriptors for each region, generating a 72-size global descriptor.

3. HELO_H: This is applied over four equal-sized horizontal regions, juxtaposing each descriptor to make up the global one. The size of this descriptor is the same as that of the HELO_V.

Each one of the three mentioned descriptors are evaluated separately using the χ^2 distance, producing 3 different distances, $d_{HELO}, d_{HELO_V}, d_{HELO_H}$. The final distance D is computed as follows:

$$D = w_1 * d_{HELO} + w_2 * d_{HELO_V} + w_3 * d_{HELO_H} \qquad (6)$$

where w_i are appropriate weights such that $\sum_{i=1}^{3} w_i = 1$. Empirically, we set $w_1 = 0.2$, $w_2 = 0.4$, $w_3 = 0.4$. We call this global approach vHELO.

4.1 Handling viewpoint changes

In our approach, we project each 3D model from 14 different viewpoints [25] getting the corresponding 14 suggestive contours. The global dissimilarity between a query sketch and a 3D model is computed as the minimum distance between the input and the 14 corresponding suggestive contours.

After selecting the candidates using the global filter, we keep only one suggestive contour for each selected 3D model. The local approach explained in the previous section will give the final rank. A graphical representation of how our proposal works is illustrated in Figure 7.

5. EXPERIMENTAL EVALUATION

5.1 Dataset Description

For our experiments, we used the benchmark used by Yoon et al. [25]. This benchmark has been developed using several 3D mesh models from the Princeton Shape Benchmark[1], from where 260 models belonging to 13 different classes were selected. These classes are: *ant*, *bear*, *bird*, *chair*, *cup*, *fish*, *glasses*, *hand*, *human*, *octopus*, *plane*, *table*, *tool*.

[1]http://segeval.cs.princeton.edu

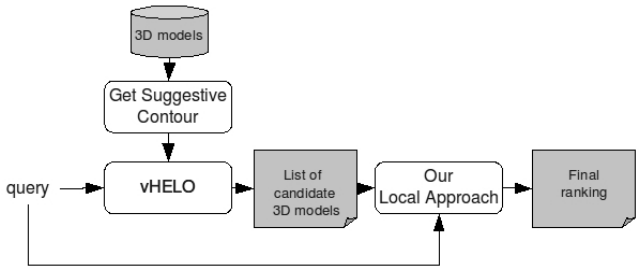

Figure 7: Combining a global approach with a local one for 3D model retrieval.

Using the 3D models, $260 \times 14 = 3640$ suggestive contours from different viewpoints are rendered, which we use for our experiments as training examples. Additionally, the benchmark provides 250 user hand-drawn sketches, which are used as input for the retrieval task evaluation. It is worth mentioning that the sketches are, in fact, rough sketches drawn by users in a free way. No constraint are imposed to the users for drawing such sketches.

Figure 8 shows examples of the 3D model used as training data and Figure 9 shows several sketches corresponding to four different classes.

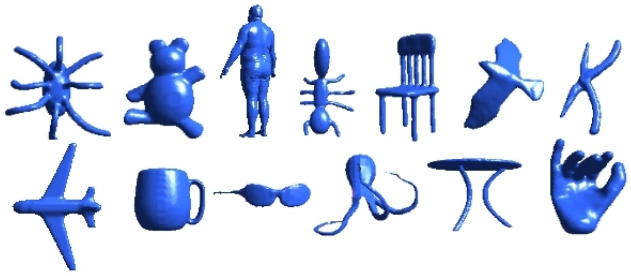

Figure 8: Examples of 3D models used as training data.

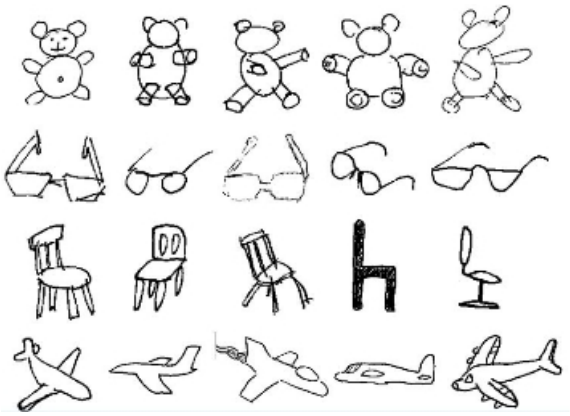

Figure 9: Examples of sketches used as queries.

5.2 Result Analysis

In this section, we show the results of the retrieved 3D models from various test sketches. Two examples of the retrieval task are shown in Figure 10. In these examples we

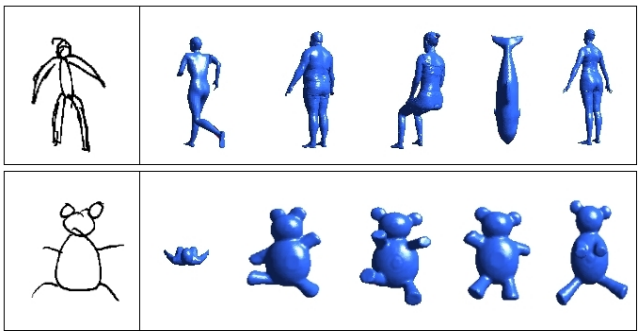

Figure 10: Examples of the 3D model retrieval using the proposed local approach. The first columns shows a sketch query, the other five images correspond to the first five retrieved models.

depict only the first five retrieved models. We can note that our proposal retrieves only one false positive among the first five retrieved objects, the remaining four objects correspond to objects belonging to the same class of the query. Additionally, it is important to note that our method retrieves relevant objects, even though they undergo different kind of deformation.

To assess the performance of our method we use the *first-tier precision*. In this case, each class contains 20 different models, so any input sketch, belonging to one of the 13 classes, must retrieve 20 models. This number corresponds to the number of relevant models for each query. Using the first-tier precision, we take as evaluation measure the number of the retrieved relevant models divided by the number of the overall retrieved models, after retrieving the first 20 models.

To our knowledge, there are no published local-based techniques for 3D model retrieval using a rough sketch as query, so we compare our method against the performance of using a global approach. We use HELO, a current method for sketch-based image retrieval, as the global approach as described in [20].

In Table 1, we show the results of our method (STELA) which use the vHELO as filtering method. The database is composed of 3640 suggestive contour images which correspond to 260 different 3D models. Using the filtering step, we select only 100 3D models. For each selected 3D model we keep the suggestive contour image with the most similar global shape respect to the query shape. The result are presented independently for each class. We increase the precision for 8 of 13 classes, achieving significant improvement for classes that have a well defined structural information such as *bear*, *chair*, *human*, *octopus*, and *plane*. For instance, the gained precision for *bears* is 57% better than that gained by the global method. In addition, we should note that the precision of our method for the other 5 classes keeps similar to that achieved by the global approach. A graphical comparison is depicted in Figure 11.

The reason for our better achieved performance is the higher level approach we are using. Instead of relying on *keypoints*, that in the case of line-based sketches these points could be part of noise, we rely on straight lines regarded as *keyshapes*. We keep lines of certain range of lengths, discarding those that have a length below a threshold. The retained

Table 1: First-tier precision for each class.

Class	HELO	STELA
Ant	0.147	0.126
Bear	0.210	0.338
Bird	0.107	0.110
Chair	0.088	0.121
Cup	0.138	0.142
Fish	0.162	0.152
Glasses	0.029	0.079
Hand	0.333	0.319
Human	0.255	0.321
Octopus	0.108	0.150
Plane	0.021	0.117
Table	0.135	0.120
Tool	0.079	0.045

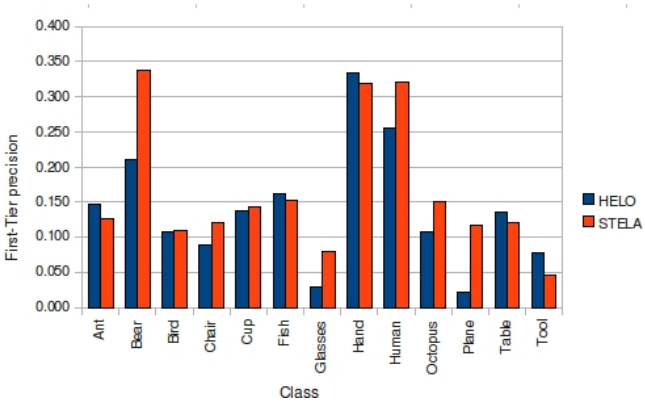

Figure 11: The first-tier precision for each class. We compare our proposal (STELA) with a global descriptor (HELO).

lines represent the structure of an object, being robust under noise.

6. CONCLUSIONS

In this work we have presented a novel local approach for 3D model retrieval having a rough sketch as query. Our approach takes advantage of the structural and locality information, as well as of the global similarity to increase the retrieval precision.

Our approach outperforms the state of the art, achieving significant improvement for the many classes of 3D models. It is important to note that our test database is composed of actual rough sketches, turning the retrieval task really a big challenge.

For keyshape detection we are only using straight lines. Therefore our ongoing work is to consider other primitive shapes like arcs and circular forms as keyhapes. In addition, we are already working in extending our proposal for partial matching.

7. ACKNOWLEDGMENTS

This research was supported by CONICYT-Chile, Fondecyt-Chile (Project 1110111), VAA of the University of Chile, and the Fraunhofer IGD Gris, TU Darmstadt

8. REFERENCES

[1] M. Ankerst, G. Kastenmüller, H.-P. Kriegel, and T. Seidl. 3D shape histograms for similarity search and classification in spatial databases. In *Proc. of the 6th International Symposium on Advances in Spatial Databases*, pages 207–226, 1999.

[2] D. H. Ballard. Readings in computer vision: issues, problems, principles, and paradigms. chapter Generalizing the hough transform to detect arbitrary shapes, pages 714–725. 1987.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:509–522, April 2002.

[4] B. Bustos, D. Keim, D. Saupe, and T. Schreck. Content-based 3D object retrieval. *IEEE Computer Graphics and Applications*, 27(4):22–27, 2007.

[5] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[6] D.-Y. Chen, X.-P. Tian, Y. te Shen, and M. Ouhyoung. On visual similarity based 3D model retrieval. In *Proc. in Eurographics, Computer Graphics Forum*, pages 223–232, 2003.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, pages 886–893, 2005.

[8] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Survey*, 40(2):1–60, April 2008.

[9] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. *ACM Transaction og Graphics*, 22:848–855, July 2003.

[10] M. Elad, A. Tal, and S. Ar. Content based retrieval of vrml objects: an iterative and interactive approach. In *Proc. of the sixth Eurographics workshop on Multimedia 2001*, pages 107–118, 2002.

[11] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A search engine for 3D models. *ACM Transactions on Graphics*, 22(1):83–105, 2003.

[12] Z. Guo and R. W. Hall. Parallel thinning with two-subiteration algorithms. *Commun. ACM*, 32(3):359–373, 1989.

[13] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii. Topology matching for fully automatic similarity estimation of 3D shapes. In *Proc. of the 28th annual conference on Computer graphics and interactive techniques*, pages 203–212, 2001.

[14] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[15] H. W. Kuhn. The hungarian method for the assignment problem. In *50 Years of Integer Programming 1958-2008*, pages 29–47. 2010.

[16] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31:983–1001, 1998.

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, November 2004.

[18] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[19] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3D models with shape distributions. In *Proc. of the International Conference on Shape Modeling & Applications*, pages 154–, 2001.

[20] J. Saavedra and B. Bustos. An improved histogram of edge local orientations for sketch-based image retrieval. In *Pattern Recognition*, volume 6376 of *Lec. Notes in Computer Science*, pages 432–441. 2010.

[21] J. W. Tangelder and R. C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia Tools Applications*, 39:441–471, September 2008.

[22] J. Tierny, J.-P. Vandeborre, and M. Daoudi. Partial 3d shape retrieval by reeb pattern unfolding. *Computer Graphics Forum*, 28:41–55, 2009.

[23] C. Wang, D. Blei, and F.-F. Li. Simultaneous image classification and annotation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1903 –1910, 2009.

[24] X.-J. Wang, L. Zhang, M. Liu, Y. Li, and W.-Y. Ma. Arista - image search to annotation on billions of web photos. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2987 –2994, 2010.

[25] S. M. Yoon, M. Scherer, T. Schreck, and A. Kuijper. Sketch-based 3D model retrieval using diffusion tensor fields of suggestive contours. In *Proc. of the international conference on Multimedia*, pages 193–200, 2010.

[26] B. Yu. Recognition of freehand sketches using mean shift. In *Proc. in IUI 2003, (Miami FL)*, pages 204–210, 2003.