# Feature-Preserving 3D Thumbnail Creation via Mesh Decomposition and Approximation

Pei-Ying Chiang[1], May-chen Kuo[1], Teri M. Silva[2], Edgar Evangelista[2], Milton Rosenberg[2], and C.-C. Jay Kuo[1]

[1] Ming Hsieh Department of Electrical Engineering
University of Southern California, Los Angeles, CA 90089-2546
[2] institute for Creative Technologies, University of Southern California, Los Angeles, CA 90292-4019, USA
peiyingc@usc.edu
[⋆]

**Abstract.** We propose an innovative approach to feature-preserving 3D thumbnail creation in this research. The 3D thumbnail system aims to help the user efficiently browse 3D models in a large 3D database. The user can browse multiple 3D models with pre-generated 3D thumbnails, and view them from different angles interactively. The 3D thumbnail is a simplified version of the original model and requires much less memory and time to render. In the proposed system, we separate the framework into offline and online two processes for improving the run-time performance. Two types of descriptors of each 3D model are first generated offline and, then, the thumbnail result can be quickly rendered online using these pre-generated descriptors. Preliminary experimental results are given to demonstrate the proposed methodology.

## 1 Introduction

With the increasing amount of available multimedia digital assets, the digital asset management (DAM) tools become more and more valuable. People rely on these DAM tools to search for a specific digital asset in a large database. Although the technology of text-based searching is maturing, many existing digital assets are not well named and cannot be found by just a text-based query. Our research aims to improve the search efficiency for digital assets, especially for 3D objects, since 3D objects are playing an important role nowadays. The entertainment and engineering industries create more than millions of 3D objects in a decade. An effective DAM tool is needed to find a 3D object even it is not well named. Current DAM tools display the static 2D thumbnails on the search page for easier browsing. However, those 2D thumbnails need to be pre-captured

[⋆] The project or effort described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

manually and it is extremely time-consuming to capture thumbnails for a large scale 3D asset database. Some researchers attempted to develop systems that can take the best snapshot for a 3D object automatically by selecting the view angle that captures the most important features. This fixed selection rule does not work well for all objects and the result can just as easily capture the wrong features. In addition, while the 2D thumbnail might capture the best shot of a 3D object, there are still other features that cannot be seen from a fixed angle.

We propose an innovative approach to create a feature-preserving 3D thumbnail automatically in this work. The resultant 3D thumbnail system aims to help the user efficiently browse 3D models in a large 3D database. The user can browse multiple 3D models with pre-generated 3D thumbnails, and view the 3D thumbnails from different angles interactively. The 3D thumbnail is a simplified version of the original model and requires much less memory and time to render. It can also support devices without enough memory resources for 3D rendering. In our system, we separate the framework into offline and online two processes for improving the runtime performance. Two types of descriptors of each 3D model are generated offline so that the thumbnail can be quickly rendered online using the pre-generated descriptors. Preliminary experimental results are given to demonstrate the proposed methodology.

In the offline process, we first decompose a model into visually significant parts so that its individual parts can still be well preserved even when the model is extremely simplified. Then, the skeleton and the body measurements of all parts are extracted and saved as a parts descriptor, which describes the shape in a certain format. Third, we apply an iterative, error-driven approximation algorithm to find the best fitting primitives representing a simplified model. The approximation results are saved as a thumbnail descriptor for efficient online rendering. Our novel coarse-to-fine approximation is different from the prior art where a model is often approximated using the fine-to-coarse approach. In addition, an innovative deformable cylinder, called the d-cylinder, is developed for the primitive approximation. As a result, our thumbnail results are more discernable than other approximation methods using regular primitives.

In the online process, the 3D thumbnail is rendered according to the pre-generated thumbnail descriptors. Multiple thumbnails can be downloaded and displayed on Java applets based 3D thumbnail viewer within a few seconds. A remote user can also regenerate the 3D thumbnail if they prefer another level of detail by re-using the existing parts descriptor. This process can be done within a second.

The rest of this paper is organized as follows. Related previous work is briefly reviewed in Sec.2. The system framework and each individual process are described in Sec. 3. The evaluation and discussion of the achieved results are presented in Sec. 4. Finally, concluding remarks and future research directions are given in Sec. 5.

## 2 Review of Related Work

The proposed 3D thumbnail representation scheme is closely related to research on mesh simplification. Mesh simplification techniques have been developed more than a decade to support multi-resolution applications. Since a complex 3D model contains millions of polygons and requires a large amount of memory and time to process, it can significantly degrade the runtime performance. Many applications may require or can only support the low resolution model instead of the original complex model. For example, rendering time can be sped up if a low-level detail version of an object is used when it is far away from the camera. Previous work on mesh decomposition can be categorized into surface-based, voxel-based and hybrid approaches.

Garland *et al.* [1] developed a surface-based simplification algorithm for producing high quality approximations of polygonal models. Their algorithm used iterative contractions of vertex pairs to simplify meshes, and maintained minimum surface error approximations using quadric matrices. Cohen *et al.* [2] proposed an error-driven optimization algorithm for geometric approximation of surfaces. Their algorithm used a similar idea inspired by the Lloyd algorithm, which can reduce the distortion error through repeated clustering of faces into best-fitting regions. Their approach does not require parameterization or local estimations of differential quantities. While Cohen *et al.* [2] only considered planes for approximation, Wu *et al.* [3] extended their optimization technique by allowing different primitives to represent including spheres, cylinders, and more complex rolling-ball blend patches to represent the geometric proxy of a surface region. They segment a given mesh model into characteristic patches and provide a corresponding geometric proxy for each patch. These works used a similarly iterative approach, which would produce a fine-to-coarse approximation. The mesh would be simplified incrementally during their approximation process.

He *et al.* [4] proposed a voxel-based mesh simplification approach that used sampling and low-pass filtering to transform an object into a multi-resolution volume buffer. Then, the marching cubes algorithm [5] was used to construct a multi-resolution triangle-mesh surface hierarchy . Nooruddin*et al.* [6] adopted a hybrid approach that integrated the voxel-based and the surface-based mesh simplification approaches. They converted polygonal models to a volumetric representation, and then repaired and simplified a model with 3D morphological operators. Visually unimportant features, such as tubes and holes, can be eliminated with the morphological operators. The volumetric representation result was then converted back to polygons and a topology preserving polygon simplification technique was used to produce the final model.

However, most of previous mesh simplification work did not particularly consider preserving the significant parts of a model. For example, the limbs and the body can meld together when the model is extremely simplified. In our work, we avoid the elimination of these features by distinguishing object's important geometric components.

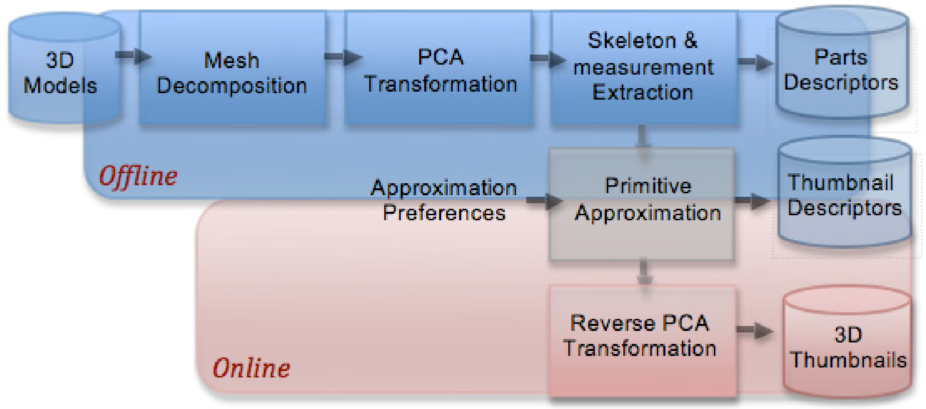

**Fig. 1.** The proposed system framework.

## 3 Proposed Feature-Preserving 3D Thumbnail Generation Algorithm

The proposed 3D thumbnail system aims to help the user browse multiple 3D models efficiently within a large 3D database. We create a feature-preserving 3D thumbnail by simplifying and approximating the original model with fitting primitives. The lower resolution thumbnail requires much less hardware resource to render and can be transmitted quickly. Additionally, to speed up the online process, we perform most of the processes offline and only leave the rendering process online. Fig. 1 shows the proposed system framework, where both the offline and online processes are illustrated.

In the offline process, we first use a mesh decomposition algorithm to separate a model into significant parts. Each part is then considered as an independent unit in the following procedure. Then, the Principal Component Analysis (PCA) transformation is adopted to normalize each part's orientation and each decomposed part is transformed from the world's space to its own principal component space. Third, we extract the skeleton and take body measurements for each part and saved them as a parts descriptor. Finally, we use the parts descriptor as the input to the primitive approximation algorithm and generate the thumbnail descriptor.

In the online process, multiple 3D thumbnails can be rendered quickly by applying a reverse PCA transformation to the pre-generated thumbnail descriptors. On the other hand, users can generate a lower/higher resolution thumbnail according to the existing parts descriptors and their preference. In the following subsections, we will describe each process within the system framework in detail.

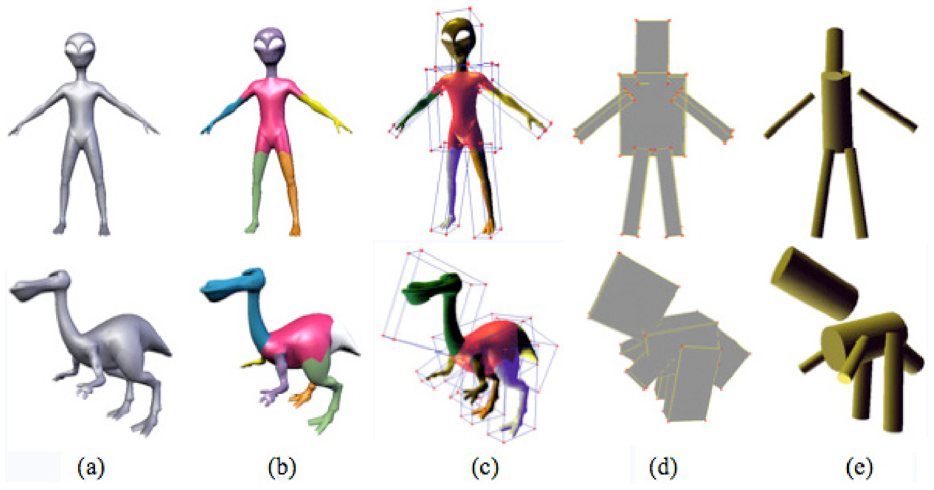

**Fig. 2.** (a) The original models, (b) mesh decomposition results from [7], (c) the minimum bounding boxes derived after PCA transformation (d) the simplest 3D thumbnails approximated with a single rectangle for each decomposed part, and (e) approximations with a single cylinder.

## 3.1 Mesh Decomposition

The reason to apply mesh decomposition is because we want to preserve the visually significant parts of each model. Thus, our 3D thumbnail will always present these parts by keeping, at least, the roughest shape of each decomposed part, even if the model has been extremely simplified. To conduct mesh decomposition, we extended the approach of Lin *et al.* [7], which can decompose a 3D model into one main body and several protruding parts. To give an example, two original models and their decomposition results are shown in Figs. 2 (a) and (b) using different colors. By identifying the significant parts, we can generate a rough 3D thumbnail that depicts each decomposed part with a single rectangle or cylinder as shown in Figs. 2(d) and (e).

## 3.2 Skeleton Extraction and PCA Transformation

After the mesh is fragmented, we want to extract each part's skeleton and body measurements so that its shape can be expressed in a simple format. This extracted data will be used to find the best fitting primitives in the next approximation process. Moreover, in order to preserve each decomposed part during simplification, we process each part individually till the approximation process is completed.

We apply a PCA transformation to each part individually. The PCA process can normalize each part's orientation, *i.e.* its principal axes are aligned with the new coordinate axes in the transformed space. Thus, we can extract the

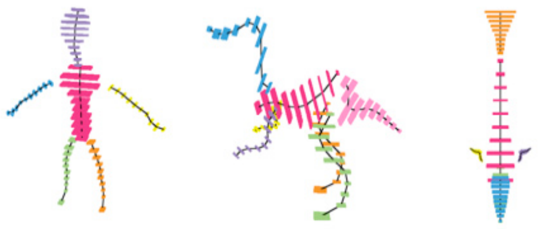

**Fig. 3.** The skeleton and body measurements are extracted after PCA transformation, where the black line represents each part's skeleton and each rectangular slice represents the estimated body measurements along the skeleton.

skeleton easily along its principal axis. For example, the bounding boxes show various orientations encapsulating the decomposed parts in Fig. 2(c). By applying a PCA transformation, each part is transformed to its temporary PCA transformed space, where the bounding box is centrally located and aligned to new coordinate axes, say ($\tilde{X}$, $\tilde{Y}$, $\tilde{Z}$). Moreover, the skeleton extraction process was based on sampling points from the original mesh's surface. (Note that, our mesh is constructed with multiple triangle faces. These triangles are assigned to different parts by the mesh decomposition process.) For each triangle face belonging to a decomposed part, we uniformly extract sample points within the surface of the triangle by interpolation and use them to estimate the skeleton and body measurements as detailed below.

We extract the skeleton by calculating the central line along its first principal axis, say $\tilde{X}$-axis, in the decomposed part's PCA transformed space. Imagine that the decomposed part is chopped into multiple slices along its $\tilde{X}$-axis. Assuming ($\tilde{x}_i$, $\tilde{y}_{x_i}$, $\tilde{z}_{x_i}$) is the center of sample points on the slice of $\tilde{X} = \tilde{x}_i$; $\tilde{x}_{min}$ and $\tilde{x}_{max}$ are the minimum and the maximum of $\tilde{X}$ among all sample points. We obtain the skeleton by connecting the adjacent center points ($\tilde{x}_i$, $\tilde{y}_{x_i}$, $\tilde{z}_{x_i}$) within the range of $\tilde{X} = [\tilde{x}_{min}, \tilde{x}_{max}]$ consecutively.

Finally, we take the body measurements by estimating the average distance from sample points to the center on each slice. For example, for all sample points whose $\tilde{X} = \tilde{x}_i$, we calculate their distance to central point ($\tilde{x}_i$, $\tilde{y}_{x_i}$, $\tilde{z}_{x_i}$) along $\tilde{Y}$-axis and $\tilde{Z}$-axis correspondingly. The average distances were then taken as the approximated body measurements for slice $\tilde{X} = \tilde{x}_i$, in the form of ($RW_{x_i}$, $RH_{x_i}$).

In Fig. 3, we show examples of the extracted skeleton and body measurements. Each black line represents the skeleton of each decomposed part. Each colorful rectangular slice represents a body measurement whose width and height are equal to $RW_{x_i}$ and $RH_{x_i}$, respectively, at $\tilde{X}= \tilde{x}_i$. Note that each part was reversed-PCA transformed to the original coordination system in this figure to display the temporary result. The extracted skeleton and body measurement are called the *parts descriptor* and stored in the database so that they can be re-used whenever the user wants to run the remaining processes.

### 3.3 Iterative Approximation and Primitive Selection

After decomposing a mesh into salient parts and performing skeleton extraction and PCA transformation, we will proceed to the last stage; namely, coarse-to-fine iterative approximation and selecting proper primitives. During the iterative process, the fitting primitives are applied according to the skeleton and body measurements that we extracted earlier. We first generate the roughest thumbnail composed of a minimum number of primitives and enhance the thumbnail representation by adding more primitives until the total bit budget is met.

The bit budget is determined by available hardware resource or user's preference. For example, if the memory on a computer can afford 3000 primitives rendered simultaneously, and the user prefers to view 10 objects at a single web page, then the budget for each thumbnail is 300 primitives.

Moreover, to approximate an object with regular primitives may produce a relatively stiff result as shown in Figs. 2(d)(e), since the shape of the primitive is not flexible. We propose a deformable primitive ”d-cylinder” which can produce more pleasant 3D thumbnails.

**Customized D-cylinder** The deformable d-cylinder is composed of an upper ellipse, a lower ellipse, and a body composed of multiple quadrangles dividing the ellipses uniformly. The major and the minor radii of the two ellipses can be adjusted individually to fit the approximated object. The number of quadrangles can decide the smoothness of the d-cylinder curve.

The data structure of this d-cylinder contains four types of elements that can be adjusted to fit the shape: (1) $n_{seg}$: the number of divisions of the body. There are $n_{seg}$ points that divide each ellipse and form $n_{seg}$ quadrangles of the body. The greater $n_{seg}$ is assigned, the smoother the d-cylinder curve will be. However, more memory is required to render this d-cylinder. (2) $Radius_{upper}$ $(\alpha_1, \beta_1)$ and $Radius_{lower}$ $(\alpha_2, \beta_2)$: the major and the minor radii of the two ellipses, which are set to fit body measurements of the 3D model. (3) $Center_{upper}(x_1, y_1, z_1)$ and $Center_{lower}(x_2, y_2, z_2)$: the center of each ellipse. The two values are set to the skeleton points of the 3D model.

**Primitive Approximation** We propose a coarse-to-fine iterative approximation method, which approximates the original shape with a minimum number of primitives first and then adds more primitives iteratively to produce a finer result until the budget is met. In this process, the skeleton and body measurements are used as the input for shape approximation with deformable d-cylinders. The main idea is illustrated in Fig. 4. First, we assign one approximating d-cylinder to each decomposed part and the shape of this d-cylinder is decided according to the part's skeleton points and body measurements. Second, we examine the distortion between the original shape and the approximating d-cylinder and choose the area that has the highest distortion as the split point. Third, we divide the decomposed part into two regions and assign a new d-cylinder to each region.

In the iterative approximation process, we estimate the distortion error between the original shape and its approximating d-cylinder along each slice. The

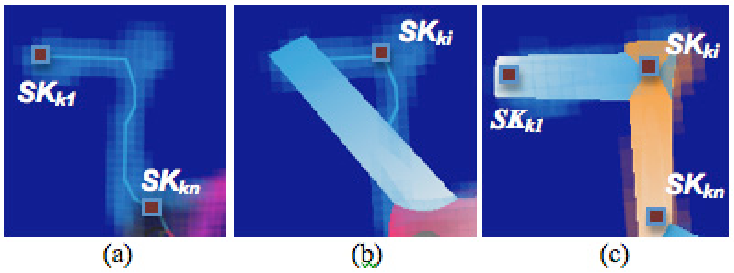

**Fig. 4.** The coarse-to-fine approximation process: (a) the decomposed part $P_k$ of the skeleton within the range of $[SK_{k_1}, SK_{k_n}]$, (b) the decomposed part $P_k$ is approximated by a single d-cylinder and the slice with the max distortion error is found at $SK_{k_i}$, and (c) the old d-cylinder is divided at $SK_{k_i}$ and replaced by two new d-cylinders.

slice which has the maximum distortion error derived from Eq. (4), is chosen to assign new approximating d-cylinders. Assuming the slice at $SK_{k_i}$ has the maximum error as shown in Fig. 4(b), we mark $SK_{k_i}$ as a split point where the current region is going to be split. The old d-cylinder assigned to fit the region of $[SK_{k_1}, SK_{k_n}]$ will then be replaced by two new d-cylinders assigned to the new regions of $[SK_{k_1}, SK_{k_i}]$ and $[SK_{k_i}, SK_{k_n}]$ respectively, as shown in Fig. 4(c). The new d-cylinders have their centers of ellipses located at $SK_{k_1}$ and $SK_{k_i}$, respectively, and the radii of ellipses are equal to the body measurements at $SK_{k_1}$ and $SK_{k_i}$, respectively. Similarly, the other d-cylinder has its centers located at $SK_{k_i}$ and $SK_{k_n}$ and the two radii are equal to the body measurements associated to $SK_{k_i}$ and $SK_{k_n}$. The worst approximating area is replaced at each iterative process, and the iterative process is repeated until the budget is reached, or the total distortion error is sufficiently small.

To estimate the distortion error for each slice between an original shape and an approximating primitive, we consider three factors: (1) the surface distortion (2) the location distortion and (3) the volumetric distortion.

The surface distortion, $\psi_1$, measures the surface distance between the original mesh and the approximating d-cylinder at each slice. Since the center and the radii of both ends of the d-cylinder is known, The value of the center and the radii of each slice within this d-cylinder can be derived via interpolation. Thus, the surface distortion $\psi_1$ at slice $L_{k_i}$ can be defined as

$$\psi_1(L_{k_i}) = \frac{\sum_{\forall v \in L_{k_i}} d(v, C_{k_t})}{\mid \forall v \in L_{k_i} \mid}, \tag{1}$$

where $v$ is the sampling point extracted from the original mesh, $C_{k_t}$ is the approximating d-cylinder that covers slice $L_{k_i}$, and $d(v, C_{k_t})$ is the distance from sampling point $v$ to the surface of $C_{k_t}$ at this slice.

The location distortion, $\psi_2$, measures the distance between the center of the original mesh's slice and the center of d-cylinder's slice. It is defined as

$$\psi_2(L_{k_i}) = \frac{\mid SK_{k_i} - Center_{k_i}(C_{k_t}) \mid}{max_{\forall P_k}\{maxCenterDistance(P_k)\}}, \tag{2}$$

where $SK_{k_i}$ is the skeleton point of slice $L_{k_i}$, $Center_{k_i}(C_{k_t})$ is the center of $C_{k_t}$ at this slice, and $maxCenterDistance(P_k)$ is the maximum distance between any pair of this decomposed part $P_k$'s skeleton points projected on a $P_k$'s slice, which constrains $\psi_2$ within [0,1].

The volumetric distortion, $\psi_3$, measures the non-overlapping region between the slice of the original mesh and that of the approximating d-cylinder. It is defined mathematically as

$$\psi_3(L_{k_i}) = \frac{(origSliceSize + cSliceSize) - 2 \times overlap}{2 \times maxSliceSize}, \tag{3}$$

where *origSliceSize* and *cSliceSize* are the slice sizes of the original mesh and the approximating d-cylinder $C_{k_t}$, which can be estimated from the radii of this slice, respectively, *overlap* is the the overlapping size between the original mesh and $C_{k_t}$, and *maxSliceSize* is the maximum slice size among all parts and constrains $\psi_3$ within [0,1].

To summarize, let $C_{k_t}$ be the d-cylinder approximating one of the divisions of part $P_k$ and slice $L_{k_i}$ within the approximating region of $C_{k_t}$. The distortion error for slice $L_{k_i}$ can be estimated as

$$E(L_{k_i}) = w_1 \times \psi_1(L_{k_i}) + w_2 \times \psi_2(L_{k_i}) + w_3 \times \psi_3(L_{k_i}), \tag{4}$$

where $w_1$, $w_2$, and $w_3$ are weighting parameters with $w_1 + w_2 + w_3 = 1$. They are used to adjust the contribution of each distortion factor. The default values are set to 1/3, *i.e.* equal weight. However, we can adjust the weight whenever we would like to give more weight to a particular factor. For example, a larger $w_2$ value can better capture the detail of high curvature features than a smaller $w_2$ value.

**Thumbnail Descriptor and Online Rendering** At the end of the offline procedure, the result of the primitive approximation process is saved as the *thumbnail descriptor* which can speed up the online rendering. The thumbnail descriptor contains items for each individual part such as its inverse matrix, its inverse translation vector, the number of the d-cylinder applied to approximate a part, the center $SK_i$ and the radii ($RW_{SK_i}$, $RH_{SK_i}$) of slices where a d-cylinder has been assigned. The average size of the thumbnail descriptor is 12KB for a thumbnail composed of 50 d-cylinders using the plain text file format, which means that a remote user can download the descriptors quickly. Moreover, if the user prefers to view the thumbnail with another level of detail, they can re-use the parts descriptors and re-generate the thumbnail from the approximation process.

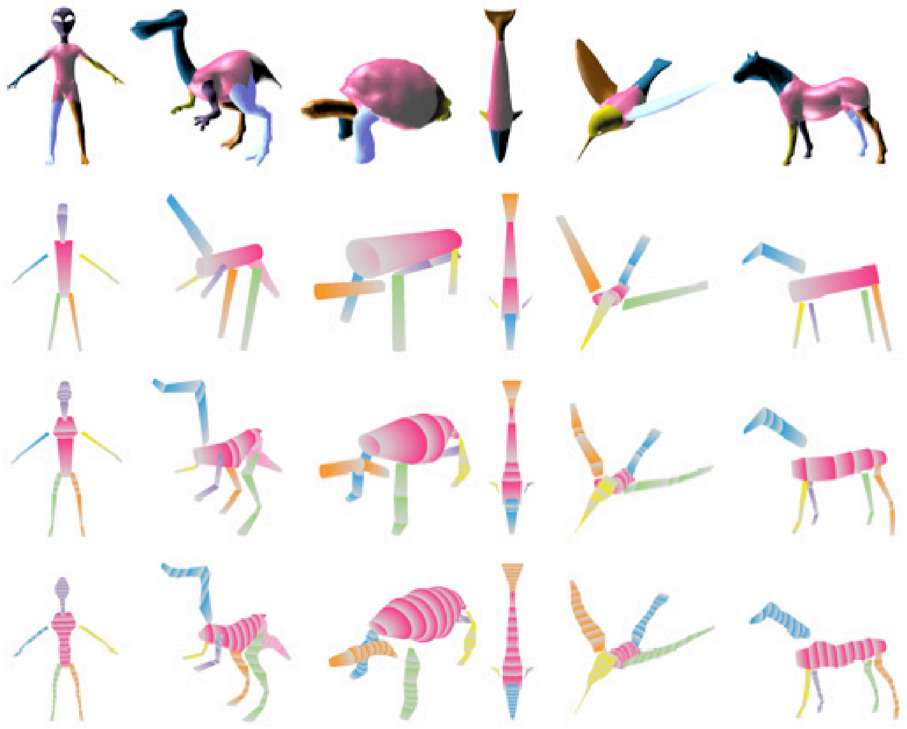

**Fig. 5.** The approximating thumbnail results.

We leave only the rendering part in the online process to improve the browsing performance. When rendering the 3D thumbnail, all approximating d-cylinders are reverse transformed back to the world coordinates from the PCA transformed space. The pre-generated thumbnail descriptors contain all the information needed to render a thumbnail.

## 4 Experimental Results

The proposed algorithm was applied to different 3D objects in the experiments. All 3D objects were converted into the same format (.obj) and normalized into the same distribution range. The sample thumbnail results are shown in Fig. 5. From the top to the bottom, the first row of pictures shows that the original mesh is composed of a main part and several protruding parts displayed in different colors. The second, third, and fourth rows show our 3D thumbnail results composed of 7, 20, 50 d-cylinders respectively. When a thumbnail model is extremely simplified, such as the examples shown in the second row of Fig. 5, each part is still represented by at least one primitive, so that the significant components can be preserved. A small number of primitives does preserve the lower-level details, and require fewer resource to render.

We also developed an online 3D thumbnail viewer implemented with the Java 3D applet. The viewer can be embeded into a web browser easily and allows the user to browse multiple 3D thumbnails from different angles interactively. Currently, the viewer is set to display 12 thumbnails at the same page. All 12 thumbnails can be displayed within 5 seconds. In our experiment, the browser can display up to 22 thumbnails without running out of memory when each thumbnail is composed of 40 d-cylinders. The lower the level of detail of a thumbnail, the more thumbnails can be displayed at the same time, and vice versa.

Our experiment was running on a desktop computer with an Intel Core 2 Duo 2.53 GHz CPU, and 4G RAM. The average processing time for each process is listed in Table 1. Each process can be done within a second, and the total processing time from the extraction process to the thumbnail rendering is around 1 second. The generated thumbnail descriptors composed of 7, 20, 50 d-cylinders were about 10KB, 10KB, 12KB, respectively, for models whose size were within the range of 160 - 870 KB. The file size of the thumbnail descriptor is determined by the number of primitives rather than the size of the original model. Thus, a thumbnail descriptor can be much smaller than its original model. In addition, the result of the skeleton and body measurement extraction is always the same, and will not be affected by the number of primitives. Note that the mesh decomposition processing time was not included in this table, since it was based on Lin's work [7] and will be replaced by another method in the future. As reported in their work, the mesh decomposition time for a model, such as a dinopet model in Fig. 2 that contains 2039 vertices and 3999 faces, was 2.3 seconds.

| **Number of primitives assigned** | Processing time (sec.) | | | **Thumbnail descriptors size** |
|---|---|---|---|---|
| | Skeleton & measurements Extraction | Primitive approximation | Thumbnail Rendering | |
| 7 | 0.51 | 0.072 | 0.36 | 10KB |
| 20 | 0.51 | 0.142 | 0.43 | 10KB |
| 50 | 0.52 | 0.238 | 0.37 | 12KB |

**Table 1.** System performance

## 5 Conclusion and Future Work

In this work, we proposed a novel feature-preserving 3D thumbnail system for efficiently browsing multiple 3D objects in a large database. The significant components of the original model can be well preserved in the 3D thumbnails even the model is extremely simplified, and it requires much less hardware resources to render. Since the data size of the thumbnail descriptor is much less than that of the original mesh, it can be downloaded quickly. Additionally, the online thumbnail viewer can display multiple 3D thumbnails within a few seconds so that the remote user can browse them interactively and efficiently in a large database.

The limitation of the proposed system is that when a 3D model has a complex topology and cannot be well decomposed by Lin's algorithm [7]. In the future, we will focus on improving the mesh decomposition method. We are developing a new volumetric based decomposition method to address this issue. The skeleton and body measurement extraction process will also be adjusted so as to capture more features that have not been considered in this work. Furthermore, more types of primitives will be evaluated for primitive approximation and the textures of a model will also be considered. Finally, a mesh simplification benchmark will be developed for evaluating the quality of results of different mesh simplification approaches.

## References

1. Garland, M., Heckbert, P.S.: Surface simplification using quadric error metrics. In: SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM (1997) 209–216
2. Cohen-Steiner, D., Alliez, P., Desbrun, M.: Variational shape approximation. In: SIGGRAPH '04:Proceedings of the 31th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM (2004) 905–914
3. Wu, J., Kobbelt, L.: Structure recovery via hybrid variational surface approximation. Computer Graphics Forum **24**(3) (sep 2005) 277–284(8)
4. He, T., Hong, L., Kaufman, A., Varshney, A., Wang, S.: Voxel based object simplification. In: VIS '95: Proceedings of the 6th conference on Visualization '95, Washington, DC, USA, IEEE Computer Society (1995) 296
5. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM (1987) 163–169
6. Nooruddin, F.S., Turk, G.: Simplification and repair of polygonal models using volumetric techniques. IEEE Transactions on Visualization and Computer Graphics **9**(2) (2003) 191–205
7. Lin, H.Y.S., Liao, H.Y.M., Lin, J.C.: Visual salience-guided mesh decomposition. IEEE Transactions on Multimedia **9** (2007) 45–57