Ellipsoid-Tree Construction for Solid Objects

Shengjun Liu*
Zhejiang University

Charlie C.L. Wang†
The Chinese University of Hong Kong

Kin-Chuen Hui
The Chinese University of Hong Kong

Xiaogang Jin‡
Zhejiang University

Hanli Zhao
Zhejiang University

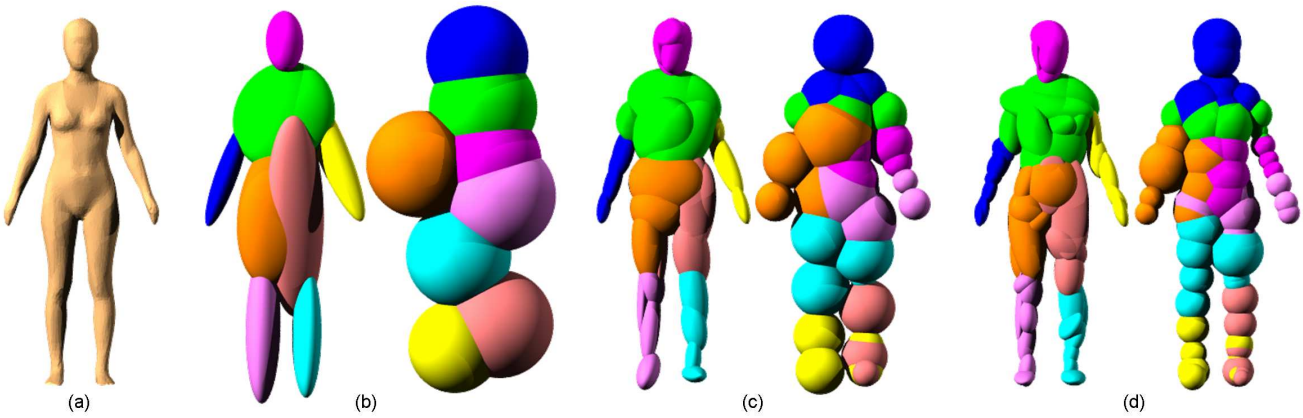

Figure 1: The ellipsoid-tree constructed by our method (left) vs. the sphere-tree generated by the Adaptive Medial Axis Approximation (AMAA) (right) for a human model: (a) the given mesh model of a human body, and the comparisons in (b) level 1, (c) level 2, and (d) level 3 – the ellipsoid-tree gives tighter volume bound and higher shape fidelity. The primitives in level 1 are displayed in different colors, and their children and children's children shown in the refined levels follow these colors.

Abstract

As ellipsoids have been employed in the collision handling of many applications in physical simulation and robotics systems, we present a novel algorithm for generating a bounding volume hierarchy (BVH) from a given model with ellipsoids as primitives. Our algorithm approximates the given model by a hierarchical set of optimized bounding ellipsoids. The ellipsoid-tree is constructed by a top-down splitting. Starting from the root of hierarchy, the volume occupied by a given model is divided into k sub-volumes where each is approximated by a volume bounding ellipsoid. Recursively, each sub-volume is then subdivided into ellipsoids for the next level in the hierarchy. The k ellipsoids at each hierarchy level for a sub-volume bounding is generated by a bottom-up algorithm – simply, the sub-volume is initially approximated by m spheres ($m \gg k$), which will be iteratively merged into k volume bounding ellipsoids and globally optimized to minimize the approximation error. Benefited from the anisotropic shape of primitives, the ellipsoid-tree constructed in our approach gives tighter volume bound and higher shape fidelity than another widely used BVH, sphere-tree.

CR Categories: I.3.5 [Computational Geometry and Object Modeling]: Curve, surface, solid, and object representations—Physically based modeling

Keywords: ellipsoid-tree, solid models, bounding volume hierarchy, volume approximation, shape approximation.

*The work is partially supported by the Hong Kong RGC/CERG grant CUHK/412405 and the CUHK project CUHK/2050341 when Shengjun Liu was a research assistant in The Chinese University of Hong Kong (CUHK).

†Corresponding Author. E-mail: cwang@mae.cuhk.edu.hk; Fax: (852) 2603 6002; Tel: (852) 2609 8052.

‡Xiaogang Jin is supported by the China 863 program (Grant No. 2006AA01Z314).

1 Introduction

Collision handling is a very important issue in many applications of physics-based modelling and simulation, which is computationally expensive (especially in a large scale virtual environment). The start-of-the-art virtual reality approaches require the collision detection to be finished in a fixed time, so that many time-critical collision detection algorithms employ *Bounding Volume Hierarchies* (BVH) to effectively reduce the number of potentially colliding pairs by eliminating objects that are obviously too far away from each other. On the other hand, the hierarchy allows a time-critical algorithm to progressively refine the accuracy of its collision detection and stop as needed to maintain the real-time performance. Usually, the node of a BVH is a simple geometric primitive that bounds parts of the model. Many different primitives have been employed, including Spheres [Bradshaw and O'Sullivan 2004; Bradshaw and O'Sullivan 2002; Hubbard 1996; Hubbard 1995; Palmer and Grimsdale 1995; Quinlan 1994], Axis Aligned Bounding Boxes (AABB) [van den Bergen 1997], Oriented Bounding Boxes (OBB) [Gottschalk et al. 1996], Spherical Shells [Krishnan et al. 1998], Rectangle Swept Spheres [Larsen et al. 2000], etc. However, no research is found in literature using ellipsoids as primitives in a BVH approach although there are some approaches for the collision detection between ellipsoids (e.g., [Choi et al. 2006; Wang et al. 2004]).

This paper introduces two contributions: a novel BVH representation, ellipsoid-tree, and a global method that minimizes the tightness of volume bounding of the ellipsoid-tree for a given model. In our approach, the tightness of volume bounding is measured by a hybrid error that integrates both the shape and the solid approximations. Compared with a closely relevant BVH, sphere-tree, generated by the state-of-the-art approach (i.e., the adaptive medial axis approximation – AMAA [Bradshaw and O'Sullivan 2004]), the ellipsoid-tree bounds the volume of given models more tightly when using the same number of primitives.

2 Ellipsoid-Tree Construction Algorithm

The ellipsoid-tree is constructed by a top-down recursive splitting algorithm. Starting from the root of hierarchy, the volume occupied by a given model M is divided into k sub-volumes where each is approximated and fully covered by a volume bounding ellipsoid. k is named as the degree of the ellipsoid-tree, and can be freely specified by users. These k ellipsoids are the primitives in level 1. Recursively, each sub-volume is then subdivided into k ellipsoids for the next level in the hierarchy.

Suppose M_{o^i} denotes part of M whose volume is covered by an ellipsoid o^i, M_{o^i} should be fully occupied by the children ellipsoids o_j^{i+1} $(j = 1,...,k)$ of o^i, where i denotes the index of level in the hierarchy. The root node of an ellipsoid-tree is the smallest bounding ellipsoid of the whole model M, which can be computed by [Weltz 1991] once M has been sampled into points. When approximating the sub-volume of the given model M, not only the surface approximation error but also the volume approximation error (more specifically, the volume enclosed by ellipsoids but outside M) are considered. In order to efficiently and effectively measure these errors, M will be sampled into surface points and voxels. Note that not only voxels inside M but also voxels outside M are constructed as they will be adopted to evaluate the volume approximation error of ellipsoids. The center points of voxels inside M are considered as volume sampling points of M. Pseudo-code of the *Ellipsoid-Tree-Construction* algorithm is listed below.

Algorithm 1 *Ellipsoid-Tree-Construction*

1: Sample the surface of the given model M into a point set P_S;
2: Voxelize the space around M;
3: Compute the minimal bounding ellipsoid of P_S as the root of ellipsoid-tree;
4: **repeat**
5: **for** each ellipsoid o^i in the current level i of hierarchy **do**
6: k ellipsoids o_j^{i+1} $(j = 1,...,k)$ are constructed for approximating the corresponding part M_{o^i} of o^i in M;
7: Optimize the shape and position of these k ellipsoids so that its approximation to M_{o^i} is tighter;
8: **end for**
9: Move to the level $(i+1)$ of hierarchy;
10: **until** the approximation tolerance has been arrived;

In this algorithm, the most important steps are steps 6 and 7, which are the major contributions of our work. Although the ellipsoids will be optimized in step 7 (the details are given in section 4), a good initial value however can always speed up the convergence of optimization. Therefore, an error controlled merging algorithm will be introduced in section 3 to generate the initial ellipsoids by a bottom-up algorithm.

3 Error-Controlled Merging for Ellipsoids Construction

Before detailing the merging algorithm for generating k ellipsoids o_j^{i+1} $(j = 1,...,k)$ bounding the corresponding volume M_{o^i} of their parent ellipsoid o^i, we will present the metrics for ellipsoid approximation first.

3.1 Metrics for ellipsoid approximation

Surface Approximation In [Cohen-Steiner et al. 2004], the approximation error between an original surface X and its approximation Y is defined as the distance

$$L^p(X,Y) = \left(\frac{1}{|X|}\int\int_{x\in X} \|d(x,Y)\|^p dx\right)^{\frac{1}{p}}$$

where $d(x,Y) = \inf_{y\in Y}\|x-y\|$, $\|\cdot\|$ is Euclidean distance, and $|\cdot|$ is surface area. However, we find that it needs to solve a non-linear equation to determine $d(x,Y)$ for an ellipsoid, which is impractical. Therefore, we define the surface approximation error between an ellipsoid o and its corresponding surface region ∂M_o on M as

$$E_{sur}(o) = \max\{\|p_i - d_o(p_i)\|, \forall p_i \in P_o\} \quad (1)$$

where $d_o(p_i)$ denotes the radial projection of p_i on o, and $P_o \subset P_S$ is the set of surface sampling points on ∂M_o. Our $E_{sur}(o)$ simulates L^∞ error between ∂M_o and o.

Solid Approximation However, the error term $E_{sur}(o)$ only works for surface but not volume. Therefore, we introduce the following volume approximation error

$$E_{vol}(o) = \int\int\int_{x\in o}(1-\delta(x,M))dV$$

where for a point $x \in \Re^3$ in o, $\delta(x,M) = 1$ is defined if x is inside the solid M; otherwise, $\delta(x,M) = 0$ is given. The approximation error measured in this way is also called solid approximation error. We evaluate the solid approximation error by its discrete form

$$E_{vol}(o) = N_{out}(o,M)V_{vox} \quad (2)$$

with N_{out} denotes the number of voxels whose centers are inside o but outside M. V_{vox} is the volume of each voxel.

Hybrid Approximation $E_{sur}(o)$ controls the shape approximation of ellipsoids to the given model, and $E_{vol}(o)$ controls the volume bounding error – both are important to ellipsoid construction. Thus, in the procedure of ellipsoids generation, we integrate these two error terms into a hybrid one by taking a weighted sum

$$E_{hyb}(o) = \alpha(E_{sur}(o))^3 + \beta E_{vox}(o). \quad (3)$$

The cubic power on the term $E_{sur}(o)$ is adopted to unify the dimension of two terms. The values of α and β indicate the relative importance that we place on the errors. We can simply choose $\alpha = \beta = 0.5$ to show the same importance on the shape and the solid approximation, or choose $\alpha = 0.8$ and $\beta = 0.2$ to add more weight on the shape approximation error.

3.2 Ellipsoids merge algorithm

k children ellipsoids o_j^{i+1} $(j = 1,...,k)$ will be generated for bounding the volume $M_{o^i} \subset M$. As aforementioned, a bottom-up merging algorithm is adopted. Briefly, M_{o^i} is first divided into m uniform sub-regions where the sampling points falling in each sub-region

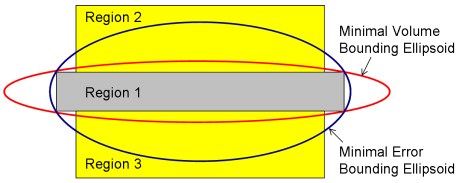

Figure 2: The minimal volume bounding ellipsoid (MVE) in general is not a minimal error bounding ellipsoid (MEE) in terms of the shape and solid approximations. To construct the bounding ellipsoid for the gray region, the MVE yields both the greater shape approximation error and the greater volume approximation error to the given model M consists of 3 regions. MEE gives a better approximation to M although its approximation to region 1 is not as good as MVE.

will be enclosed by a minimal volume bounding sphere (ref. [Weltz 1991]). Note that both the surface and the volume sampling points are considered. The neighboring relationship between ellipsoids (or spheres) can be very efficiently searched via the voxel structures. For two ellipsoids, they are defined as being linked if they have some sample points falling in neighboring voxels. Therefore, all the m primitives ρ_i can be presented in a graph $G = (V,E)$, where $E = \{\xi_1, ..., \xi_n\}$ is the collection of neighboring linking pairs and $V = \{\rho_1, ..., \rho_m\}$ is the set of primitives. After that, the ellipsoids are greedily merged into k final ellipsoids. Every time the pair of ellipsoids, whose merging introduces the least hybrid approximation error, will be chosen. Pseudo-code of the *Ellipsoids-Merge* algorithm is as follows.

Algorithm 2 *Ellipsoids-Merge*

1: M_{o^i} is uniformly divided into sub-regions;
2: Construct m spheres by computing the minimal volume bounding sphere for the sampling points in each sub-region;
3: Build the graph $G = (V,E)$;
4: Insert every pair $\xi = (\rho_\alpha, \rho_\beta)$ into a minimal heap H keyed by the error $E(\rho_{new})$ of the merged ellipsoid ρ_{new} from ρ_α and ρ_β;
5: **while** $|V| > k$ **do**
6: Remove the top node $\xi_{top} = (\rho_j, \rho_k)$ from H;
7: Merge the two ellipsoids ρ_j and ρ_k into a new one ρ_{new};
8: Replace ρ_j and ρ_k by ρ_{new} in V;
9: Update all relevant pairs (linking to ρ_j and ρ_k) in E into pairs linking to ρ_{new}, and adjust their positions in H;
10: **end while**

The algorithm starts from generating m minimal volume bounding spheres, where m is adaptively determined. Suppose that there are n sampling points (including both the surface and the volume ones) in M_{o^i}, the bounding space of M_{o^i} is uniformly subdivided into $(n/10)$ sub-regions. The minimal volume bounding sphere (by [Weltz 1991]) is constructed by the sampling points in each sub-region. If no sample point is found in a sub-region, no sphere needs to be generated for this sub-region.

The merge of two primitives to form a new ellipsoid is repeatedly computed in the algorithm *Ellipsoids-Merge*. Therefore, an effective and efficient method to compute the bounding ellipsoid is needed. Our first try is the minimal volume bounding ellipsoid computation method presented in [Weltz 1991]. However, this method has two drawbacks. Firstly, the computation of minimal volume bounding ellipsoid is not as stable as minimal volume bounding sphere. For example, if the points to be enclosed are less than 9, the algorithm fails. Secondly, there is an even more serious problem – minimal volume bounding ellipsoid in general is

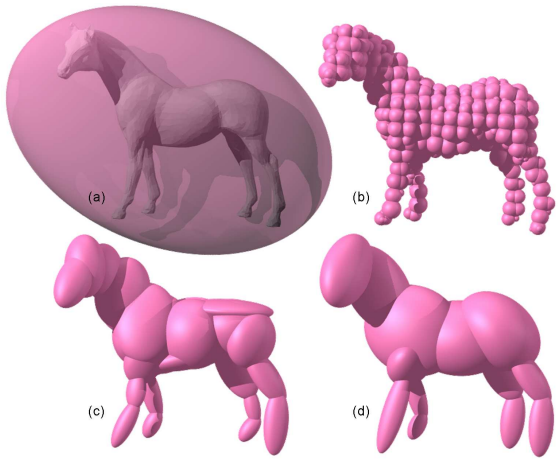

Figure 3: An example of applying our *Ellipsoids-Merge* algorithm on the horse model: (a) the model approximated by one minimal volume ellipsoid, (b) 497 initial spheres, (c) after being merged into 26 ellipsoids, and (d) final merging result with 13 ellipsoids.

not a minimal error bounding ellipsoid. Fig.2 gives an illustration for this. This drawback can only be solved through a global optimization (i.e., the process in section 5), which deforms the shape of ellipsoids. As the shape will be deformed later, we use a simple but stable method to construct volume bounding ellipsoids here, which is derived from [Lengyel 2004].

Another issue to be solved is about the sampling points classification. After generating the initial spheres, one sample point p may be inside several spheres. The point p is firstly classified into the point set of a sphere whose center is closest to p. In the later merging, all the sample points belonging to the two ellipsoids under merge will be classified into the point set of the newly constructed ellipsoid. Fig.3 shows an example result from our *Ellipsoids-Merge* algorithm.

4 Variational Ellipsoids Optimization

As the ellipsoids are greedily generated by the *Ellipsoids-Merge* algorithm, their approximation to the given model will be further improved globally in a variational optimization framework.

4.1 Variational optimization algorithm

Giving k ellipsoids, $\Omega = o_j^{i+1}$ $(j = 1, ..., k)$, for approximating M_{o^i}, a part of the given model, we are going to reduce both the shape and the solid errors on Ω. At the meanwhile, the volume of M_{o^i} should be fully enclosed by Ω. These requirements can be mathematically represented as blow

$$arg\min(\alpha(E_{sur}(\Omega))^3 + \beta E_{vox}(\Omega)) \quad s.t.\ E_{out}(\Omega) \equiv 0 \qquad (4)$$

where $(\alpha(E_{sur}(\Omega))^3 + \beta E_{vox}(\Omega))$ is the hybrid approximation error of Ω to M_{o^i} (i.e., Eq.(3)), and $E_{out}(\Omega) = M_{o^i} \cap \overline{\Omega}$ denotes the volume of M_{o^i} outside Ω. The constrained objective function will be minimized through a variational optimization algorithm similar to the iterative Lloyd clustering method [Lloyd 1982]. Basically, the variational optimization algorithm has two steps: 1) region segmentation and 2) proxy refitting. Here, the proxy type is ellipsoid. The algorithm pseudo-code is listed below.

Algorithm 3 *Variational-Ellipsoids-Optimization*

1: Start the iteration with k ellipsoids from *Ellipsoids-Merge*;
2: **repeat**
3: Segmenting all sampling points of M_{o^i} into k regions;
4: Each region will be approximated by an optimal ellipsoid;
5: **until** the determination condition has been satisfied;

Similar to other variational optimization approaches, a hybrid determination condition is conducted here – once no ellipsoid is changed or the iteration steps have been more than a user specified number (e.g., 20 is used in our implementation), the iteration stops.

4.2 Region segmentation

The volume occupied by M_{o^i} has to be segmented into k regions so that each can be approximated by an optimal ellipsoid. Ideally, we wish the overlap between ellipsoids is as less as possible since more overlapped ellipsoids will slow down the collision detection. In our approach, the volume of M_{o^i} has been sampled into points, so we need to segment these points into k non-overlapped regions. Two possible methods have been investigated.

Method 1 For a point p_k in the sampling point set Γ of M_{o^i}, if p_k is only enclosed by a single ellipsoid o^i_j, p_k is classified into the j-th region; if p_k is inside of several ellipsoids, p_k belongs to the q-th region if the center of o^i_q is closest to p_k.

Method 2 For a point $p_k \in \Gamma$ of M_{o^i}, if p_k is only inside a single ellipsoid o^i_j, p_k is classified into the j-th region; if p_k is inside several ellipsoids, p_k belongs to the q-th region if the ellipsoid o^i_q shows the smallest hybrid approximation error, $E_{hyb}(o^i_q)$ (i.e., Eq.(3)).

The first segmentation method is distance-based, and the second is error-based. For the distance-based segmentation, it essentially approximates the Voronoi diagram in the overlapped region of ellipsoids. Although a Voronoi diagram segments space into convex-hull, however this property is only limited in the overlapped region. In general, this distance-based method still produces concave regions which waste a lot of space in ellipsoid approximation (e.g., the bottom green region in Fig.4(b) is very concave). Our second segmentation method is error-based. It is generated from the idea that: we want to reduce the volume of the ellipsoid whose approximation error is greater so that it can be optimized more flexibly in the later ellipsoid refitting process. The examples shown in Fig.4 and Fig.5 illustrate the difference of these two segmentation methods. It is clear to see in Fig.4(c) that the segmented region by the distance-based method for the most right ellipsoid prevents the further optimization of its shape. When employing the second segmentation method, the region for the most right ellipsoids (which shows the greatest error in the initial ellipsoid approximation) is reduced (see Fig.5(b)). Therefore, the approximation on it can be significantly improved after the ellipsoid refitting.

4.3 Ellipsoid refitting

Giving a set of points Γ_j which have been classified to the ellipsoid o_j, we need to compute the optimal shape of o_j by Γ_j. In Γ_j, some of the points are surface sampling points Γ^S_j and others are the volume sampling points Γ^V_j (i.e., $\Gamma_j = \Gamma^S_j \cup \Gamma^V_j$). As analyzed in section 3.1, a good approximation of o_j to Γ_j should minimize the shape approximation error between o_j and Γ^S_j in terms of E_{sur} (i.e., Eq.(1)) and the solid approximation error between o_j and M defined by E_{vox} (i.e., Eq.(2)). Meanwhile, the consecutive occupation constraint E_{out} discussed in section 4.1 should also be considered. Therefore, the ellipsoid refitting is formulated as a constrained optimization problem, where the solution minimize the following objective function which is derived by the penalty function method.

$$J(o_j) = \alpha(E_{sur}(o_j))^3 + \beta E_{vox}(o_j) + \gamma E_{out}(o_j) \qquad (5)$$

where the value of $E_{out}(...)$ is evaluated by the number of point in Γ_j but outside o_j multiplying the volume of a voxel. As a penalty term, an extremely large value should be chosen for the coefficient γ weighting $E_{out}(...)$. We choose 10^3 for all our examples, and the bounding boxes of all models are scaled into unit size.

A good initial value is important for variational optimization. Therefore, once getting the segmented point set, Γ_j, we conduct the method presented in section 4.2 to construct a new bounding ellipsoid o^{new}_j for Γ_j. o^{new}_j or the previous ellipsoid o_j, which gives less error on $E_{hyb}(...)$ (i.e., Eq.(3)), is chosen as the initial ellipsoid to be optimized. The objective function in Eq.(5) is minimized by using the *Downhill Simplex Method*. The progressive results from this variational ellipsoid optimization on a bunny model are given in Fig.6.

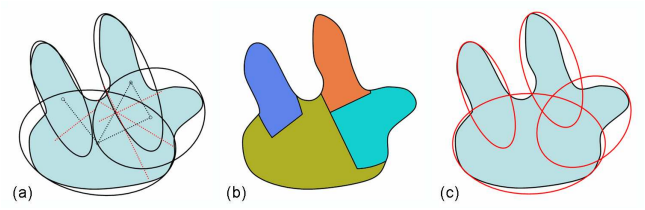

Figure 4: The distance-based segmentation method: (a) initial bounding ellipsoid set, (b) the region segmentation result, and (c) the ellipsoid set after refitting.

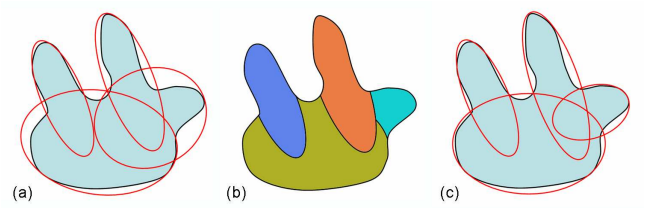

Figure 5: The error-based segmentation method: (a) initial bounding ellipsoid set, (b) the region segmentation result, and (c) the ellipsoid set after refitting.

5 Experimental Results

We have implemented the approach presented in this paper and tested it on several models. As mentioned at the beginning of this paper, when approximating a given model M by an ellipsoid-tree, the approximation errors in terms of shape and solid need to be optimized at each level of the hierarchy. Without loss of generality, at one particular level i of the ellipsoid-tree, M is approximated by the union of n ellipsoids – denoted by $Y^i = o^i_1 \cup ... \cup o^i_n$, The shape approximation error between Y^i and M is defined as the Hausdorff distance between the surface ∂Y^i of Y^i and the surface ∂M of M

$$L^\infty(\partial Y^i, \partial M) = \max\{\sup_{x\in\partial M}\inf_{y\in\partial Y^i}\|x-y\|, \sup_{y\in\partial Y^i}\inf_{x\in\partial M}\|x-y\|\}. \quad (6)$$

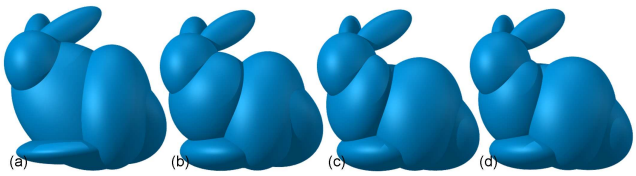

Figure 6: Progressive results for the variational ellipsoid optimization: (a) the initial ellipsoid set generated by *Ellipsoids-Merge*, (b) the temporary ellipsoids set after one optimization step in the *Variational-Ellipsoids-Optimization* algorithm, (b) the result after 3 steps, and (c) the final result (after 6 steps).

The solid approximation is defined as

$$V^{out}(Y^i, M) = \int\int\int_{x \in Y^i} \delta(x, \overline{M}) dV \tag{7}$$

with $\overline{M}$ representing the space outside M and

$$\delta(x, \Omega) = \begin{cases} 1 & \forall x \in \Omega \\ 0 & \text{otherwise} \end{cases} .$$

To evaluate $L^\infty(\partial Y^i, \partial M)$, we sample ∂M and ∂Y^i into points and compute L^∞ between these two point sets. Note that every sampling point for ∂Y^i should be a point on the surface of an ellipsoid but not inside other ellipsoids in Y^i. $V^{out}(Y^i, M)$ is evaluated by the *Monte Carlo* method – randomly sampling the bounding space Υ of Y^i into m points, if there are n points outside M but inside an ellipsoid of Y^i, we have

$$V^{out}(Y^i, M) \simeq \frac{n}{m} V_\Upsilon \tag{8}$$

where V_Υ is the volume of Υ.

Three models – the human model in Fig.1, the horse model in Fig.7, and the bunny model in Fig.7, are tested. Starting from the minimal volume bounding ellipsoid, four levels of the ellipsoid-tree are shown. The shape and the solid approximation errors in Eq.(9) and (10) have been evaluated at each level of the hierarchy. The statistics for the errors have been shown in Fig.9. It is easy to find that with the refinement of the hierarchy, both L^∞ and V^{out} are significantly decreasing.

As the state-of-the-art method for sphere-tree construction, the adaptive medial axis approximation (AMAA) method [Bradshaw and O'Sullivan 2004] is chosen as the benchmark for our ellipsoid-tree construction method. The sphere-trees for the same three models are given in Fig.1 and Fig.8. The sphere-trees are generated with the same number of primitives at each level as the ellipsoid-trees in Fig.1 and Fig.7. The best strategy in [Bradshaw and O'Sullivan 2004] (i.e., the expand and select scheme) is employed to construct the sphere-trees. The approximation errors L^∞ and V^{out} of the sphere-trees are also shown in Fig.9. From the statistics in Fig.9, it is not difficult to find that both the shape approximation error and the solid approximation error of an ellipsoid-tree are much smaller than a sphere-tree when using the same number of primitives. We also observe that the error differences between ellipsoid-tree and sphere-tree are decreasing during the refinement of the hierarchy. This is because that the subdivided volumes tend to be isotropic during the refinement. Furthermore, the ellipsoid-tree shows its strength for approximating the human and the horse rather than the bunny model. This is because that the shape of previous two examples are more complex.

The computation cost of the current implementation is relative expensive. For computing an ellipsoid-tree with about 260 ellipsoids (e.g., the horse example), it takes about 500 seconds on a PC with

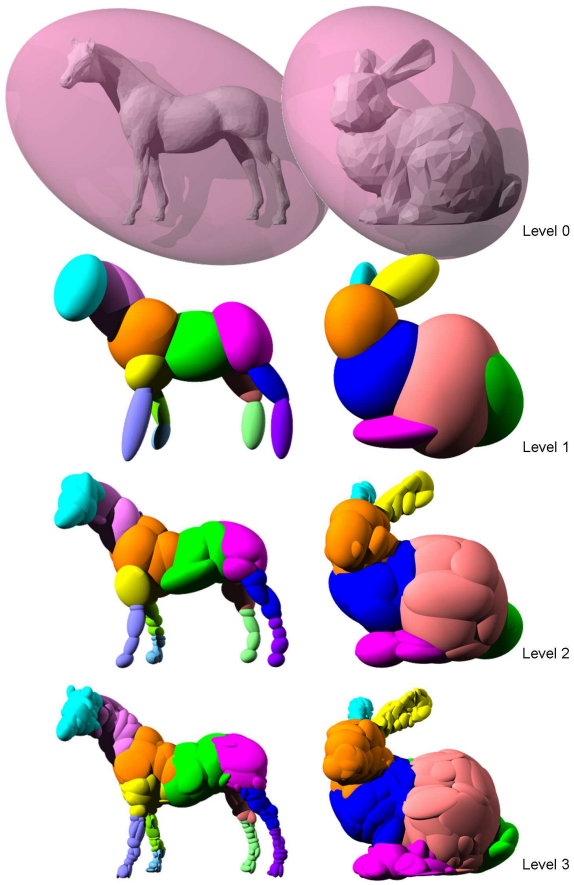

Figure 7: The ellipsoid-tree for two models: the horse and the bunny — four levels are shown. The ellipsoids in level 1 are displayed in different colors, and their children and children's children shown in the later levels follow these colors. The horse example is computed with $\alpha = 0.8$ and $\beta = 0.2$, and the bunny example is with $\alpha = 0.85$ and $\beta = 0.15$.

standard configuration. The expensive computation limits it to generate ellipsoid-tree with more than a few hundred ellipsoids. Fortunately, the approximation of objects with ellipsoids in such a limited number of primitives is sufficient for many real-time applications.

References

BRADSHAW, G., AND O'SULLIVAN, C. 2002. Sphere-tree construction using dynamic medial axis approximation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM Press, New York, NY, USA, 33–40.

BRADSHAW, G., AND O'SULLIVAN, C. 2004. Adaptive medial-axis approximation for sphere-tree construction. *ACM Transactions on Graphics 23*, 1, 1–26.

CHOI, Y.-K., WANG, W., LIU, Y., AND KIM, M.-S. 2006. Continuous collision detection for elliptic disks. *IEEE Transactions on Robotics 22*, 2, 213–224.

COHEN-STEINER, D., ALLIEZ, P., AND DESBRUN, M. 2004. Variational shape approximation. *ACM Transactions on Graphics 23*, 3, 905–914.

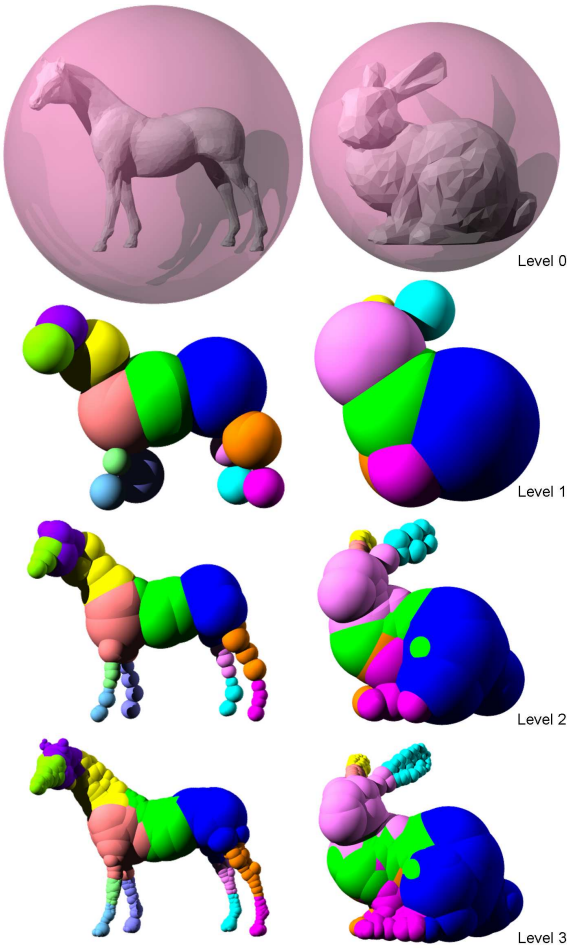

Figure 8: The sphere-tree for two models: the horse and the bunny – four levels are shown. The spheres in level 1 are displayed in different colors, and their children and children's children shown in the later levels follow these colors.

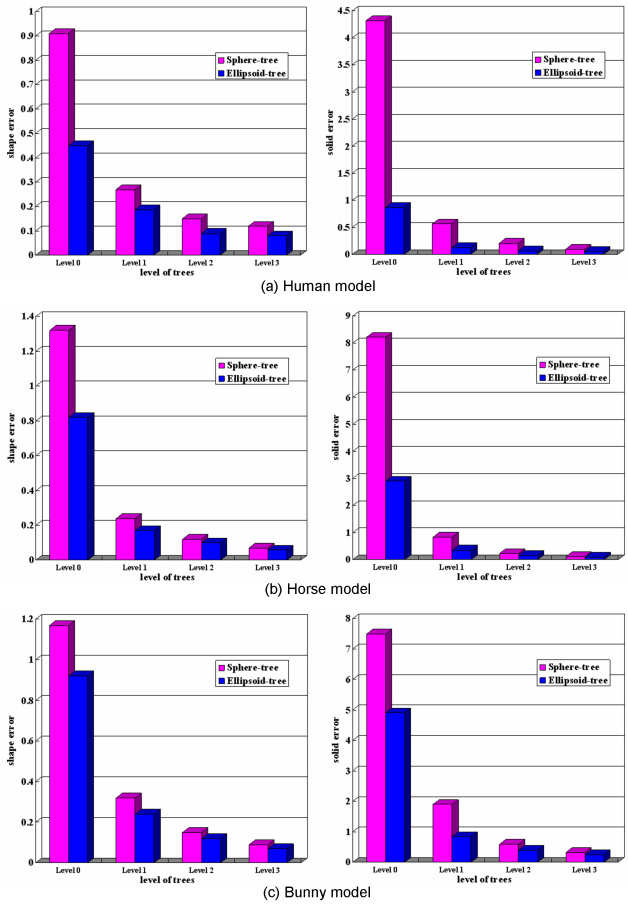

Figure 9: The statistics of L^{∞} (left) and V^{out} (right) in three models: (a) the human body, (b) the horse and (c) the bunny.

GOTTSCHALK, S., LIN, M. C., AND MANOCHA, D. 1996. OBBTree: a hierarchical structure for rapid interference detection. In *Proceedings of SIGGRAPH '96*, ACM Press, New York, NY, USA, 171–180.

HUBBARD, P. M. 1995. *Collision detection for interactive graphics applications*. PhD thesis, Brown University.

HUBBARD, P. M. 1996. Approximating polyhedra with spheres for time-critical collision detection. *ACM Transactions on Graphics 15*, 3, 179–210.

KRISHNAN, S., PATTEKAR, A., LIN, M. C., AND MANOCHA, D. 1998. Spherical shell: a higher order bounding volume for fast proximity queries. In *Proceedings of the Third Workshop on the Algorithmic Foundations of Robotics on Robotics: the Algorithmic Perspective*, A. K. Peters, Ltd., Natick, MA, USA, 177–190.

LARSEN, E., GOTTSCHALK, S., LIN, M. C., AND MANOCHA, D. 2000. Fast distance queries with rectangular swept sphere volumes. In *Proceedings of IEEE International Conference on Robotics and Automation*, IEEE Press, 3719–3726.

LENGYEL, E. 2004. *Mathematics for 3D Game Programming and Computer Graphics (2nd ed.)*. Hingham, Mass.: Charles River Media.

LLOYD, S. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory 28*, 2, 129–137.

PALMER, I., AND GRIMSDALE, R. 1995. Collision detection for animation using sphere-trees. *Computer Graphics Forum 14*, 2, 105–116.

QUINLAN, S. 1994. Efficient distance computation between nonconvex object. In *Proceedings of IEEE International Conference on Robotics and Automation*, IEEE Press, 3324–3329.

VAN DEN BERGEN, G. 1997. Efficient collision detection of complex deformable models using AABB trees. *Journal of Graphical Tools 2*, 4, 1–13.

WANG, W., CHOI, Y.-K., CHAN, B., KIM, M.-S., AND WANG, J. 2004. Efficient collision detection for moving ellipsoids using separating planes. *Computing 72*, 1-2, 235–246.

WELTZ, E. 1991. Smallest enclosing disks (balls and ellipsoids). In *New Results and New Trends in Computer Science*, Springer-Verlag, H. Maurer, Ed., vol. 555 of Lecture Notes in Compututer Science, 359–370.