Approximating Solid Objects by Ellipsoid-Tree

CONFERENCE PAPER · SEPTEMBER 2009

DOI: 10.1109/CADCG.2009.5246919 · Source: IEEE Xplore

5 AUTHORS, INCLUDING:

Shengjun Liu

Central South University

24 PUBLICATIONS **127** CITATIONS

SEE PROFILE

Kin-Chuen Hui

The Chinese University of Hong Kong

84 PUBLICATIONS **506** CITATIONS

SEE PROFILE

Xiaogang Jin

Zhejiang University

148 PUBLICATIONS **789** CITATIONS

SEE PROFILE

Approximating Solid Objects by Ellipsoid-Tree

Shengjun Liu*‡, Charlie C.L. Wang†, Kin-Chuen Hui†, Xiaogang Jin‡ and Hanli Zhao‡
*School of Mathematics and Computing Technology
Central South University, Changsha, P.R.China 410083
Email: shjliu.cg@gmail.com
†Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong, Hong Kong, P.R.China
Email: cwang@mae.cuhk.edu.hk
‡State Key Lab of CAD&CG, Zhejiang University, Hangzhou, P.R.China

Abstract

This paper presents an algorithm to approximate a solid model by a hierarchical set of bounding ellipsoids having optimal shape and volume approximation errors. The ellipsoid-tree is constructed in a top-down splitting framework. 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 and will be later subdivided into k ellipsoids for the next level in hierarchy. The difficulty for implementing this algorithm comes from how to evaluate the volume of an ellipsoid outside the given model effectively and efficiently (i.e., the outside-volume-error*). A new method — analytical computation based — is presented in this paper to compute the* outside-volume-error. *One application of ellipsoid-tree approximation has also been given at the end of the paper.*

1. Introduction

Object approximation using simple primitives is a very important issue in many applications of computer graphics, such as ray tracing, shadowing, collision handling and visibility. For different accuracy, those primitives construct a hierarchy — *Bounding Volume Hierarchies* (BVH). Usually, the bounding volume is a simple geometry primitive that bounds parts of the model. Many different primitives have been employed as the bounding volume, including Spheres [3], [4], [10], [17], [18], Axis Aligned Bounding Boxes (AABB) [26], Oriented Bounding Boxes (OBB) [9] and Spherical Shells [12], etc. To the best of our knowledge, our approach presented in [16] is the first piece of work using ellipsoids as primitives in a BVH approach. This paper presents a new analytical method here to evaluate the *outside-volume-error* and compares it with the voxel-based method in [16]. Furthermore, the application of ellipsoid-tree in shadow rendering has been given in this paper.

Main Results: A novel bounding volume hierarchy (BVH) — ellipsoid-tree has been presented in this paper to approximate both the surface and the volume of a solid model. Based on the study of the *sphere-outside-triangle volume* (SOTV), an analytical method to compute the ellipsoid-outside-triangle volume (EOTV) has been given, which is the most difficult part of the ellipsoid-tree approximation framework. Compared with a closely relevant BVH, sphere-tree, generated by the state-of-the-art approach (i.e., the adaptive medial axis approximation — AMAA [4]), the ellipsoid-tree bounds the volume of given models more tightly when using the same number of primitives.

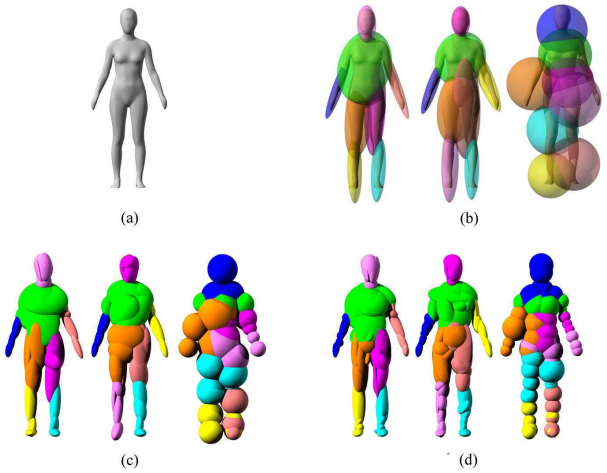

Figure 1. The ellipsoid-tree constructed by our analytical method (left), our voxel-based method (middle) and the sphere-tree generated by the Adaptive Medial Axis Approximation (AMAA) (right) for a human model: (a) the given mesh model of a human body, and the comparisons in (b) Level 1, (c) Level 2, and (d) Level 3.

The rest of the paper is organized as follows. After reviewing the related works in section 2, the error-metrics for ellipsoid approximation are introduced in section 3 together with their evaluation method. Section 4 presents the framework of ellipsoid-tree construction. The experimental results and one application in shadow rendering is given in section 5 and 6. Lastly, our paper ends with the conclusion section.

978-1-4244-3701-6/09/$25.00 ©2009 IEEE

2. Related Work

Being a simple bounding volume hierarchy, sphere-tree is widely employed in time-critical applications. A number of algorithms [3], [4], [10], [17], [18], [30] have been developed for the construction of sphere-tree. The authors in [3], [17] generated sphere-tree by using the octree data structure, where a sphere is placed at each non-vacant leaf node of the octree. The algorithm can be easily implemented, but the sphere-trees generated in this way often fit the given model quite poorly as the algorithm does not explicitly consider the tightness of fit. Quinlan [18] adopted top-down recursive splitting to construct sphere-trees. Using the method, Ruinkiewicz and Levoy [20] constructed sphere-trees for visibility culling and level-of-detail rendering. An alternative is to merge similar spheres in a bottom-up fashion as [19], [24], however splitting or merging in their methods is given greedily and thus the results are locally optimal. Methods based on the medial axis [1], [10] built a Voronoi diagram and centered spheres at its vertices. The Adaptive Medial Axis Approximation (AMAA) [4] extended this idea by using greedy optimization to merge or burst spheres. The approach is superior to other previous methods, therefore, their result will be conducted as a benchmark for evaluating our result. Recently, Wang et al. [30] presented a variational approximation method for approximating solid objects by spheres. Their method directly minimized the outside volume, thus it gave a tighter fitting.

As aforementioned, ellipsoids can give tighter bounding than spheres. Several approaches on collision detection between ellipsoids have been investigated in [6], [8], [27], [29]. However, these works are all based on the assumption that the objects have been successfully approximated by ellipsoids. A few modelling methods are available to approximate given objects by ellipsoid sets. In [2], Bischoff and Kobbelt proposed a method for approximating a 3D model by ellipsoids. Liu et al. in [14] recently extended the method of [2] for modelling 3D objects by implicit surface using ellipsoidal blobs. Simari et al. [23] adopted ellipsoids as the only type of proxies for approximating mesh surfaces using the Lloyd method with the error metric being a combination of Euclidean distance, angular distance and curvature distance. However, none of these approaches guaranteed the consequent coverage of the given model's volume, which is important for the downstream collision handling applications. Moreover, these methods cannot be directly extended to generate the BVH with ellipsoids. This yields the motivation of our work — to construct ellipsoid-tree for solid objects.

There are some approaches in literature about generating the bounding ellipsoid for a set of points (ref. [13], [22], [25], [28]). Simply applying their methods in the local region of a given model will generate a local minimal-volume ellipsoid; however, this local optimal ellipsoid may yield great volume (or shape) errors for the global approximation. Therefore, a global shape optimization method was developed in our previous paper [16], where the outside volume and the shape approximation error of the ellipsoid-tree to a given model are directly minimized.

3. Metrics for Ellipsoid Approximation

3.1. Metrics Definition

Before detailing the construction algorithm for ellipsoid-tree, we will first define the metrics for ellipsoid approximation.

Surface Approximation In [5], the approximation error between the surface X of a given model M which is a closed and oriented manifold and its approximation Y is defined as the distance

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

where $d(x,Y) = \inf_{y\in Y}\|x-y\|$, $\|\cdot\|$ is Euclidean distance, and $|\cdot|$ is surface area. However, we found that it is too complex to compute $d(x,Y)$ for an ellipsoid. Therefore, an alternative surface approximation error was used. We defined the surface approximation error between an ellipsoid o and its corresponding surface region X_o as

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

where $d_o(p_i)$ denotes the radial projection of p_i on o. Our $E_{sur}(o)$ actually simulates $L^\infty(X,Y)$.

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

$$E_{vol}(o) = \int\int\int_{y\in o} \delta(y,M)dy \quad (2)$$

where for $y \in \Re^3$, $\delta(y,M) = 1$ is defined if y is outside the solid M and inside the ellipsoid o; otherwise, $\delta(y,M) = 0$ is given. Therefore, the approximation error measured in this way is also called solid approximation error.

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 integrated these two error terms into a hybrid one by taking a weighted sum

$$E(o) = \alpha(E_{sur}(o))^3 + \beta E_{vol}(o) \quad (3)$$

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 each of the individual error metrics. 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. Computation of Solid Approximation Error

For an ellipsoid, Eq.(2) can be reformulated as

$$E_{vol}(o) = V(T, o) \tag{4}$$

where T is the triangle set representing the object M, and V is the volume outside T but inside the ellipsoid o.

3.2.1. Voxel-based method. A simple method to compute the solid approximation error $V(T, o)$ is to evaluate its discrete form by

$$V(T, o) = N_{out}(M, o)V_{vox} \tag{5}$$

with N_{out} which denotes the number of voxels whose centers are inside o but outside M. V_{vox} is the volume of each voxel. The accuracy of this evaluation depends on the resolution of voxel sampling so we need very dense voxels to achieve a highly accurate computation. To avoid this, an analytical method was developed below to compute the volume by directly integrating it triangle by triangle.

3.2.2. Analytical method. If a triangle $t \in T$ is entirely or partly inside an ellipsoid o, then there is some volume between the ellipsoidal surface and the triangle. The volume is denoted by $V(t, o)$ and named *ellipsoid-outside-triangle volume* (EOTV). The solid approximation error $V(T, o)$ between the given triangular mesh and an ellipsoid is accumulated by adding or subtracting these EOTVs over all triangles $t \in T$ on the given solid model.

Because of the anisotropy of ellipsoids, it is difficult to compute the EOTV directly with an analytical method. Therefore, we will employ sphere as the media to compute EOTV. we deform both the ellipsoid o with radii (r_a, r_b, r_c) and the triangles in T to let o become a sphere with unit-radius. After computing the *sphere-outside-triangle volume* (SOTV), denoted by $V(T, s)$ the resultant outside volume is determined by scaling SOTV with $r_a r_b r_c$

$$V(t, o) = r_a r_b r_c V(t, s). \tag{6}$$

Computing SOTV In [30], an analytical result was presented for computing SOTV. However, only four possible configurations between a sphere s and a triangle t are listed in [30] as 3 vertices inside, 3 vertices outside, 2 vertices outside, and 1 vertex outside. However, from our investigation we found that there are eight configurations.

The eight configurations are listed in Fig.2, where cases (c), (f), (g) and (h) are missed in [30]. We illustrated the cases in 2D, and assumed that a circle C is the intersection between the sphere $s = (c, r)$ and the plane of the triangle $t \in T$. The relations between the sphere and the triangle will be determined by the configuration between the circle C and the triangle t. According to the number of the intersecting points, we classified the eight cases into four classes: 0-point class, 2-point class, 4-point class, and 6-point class.

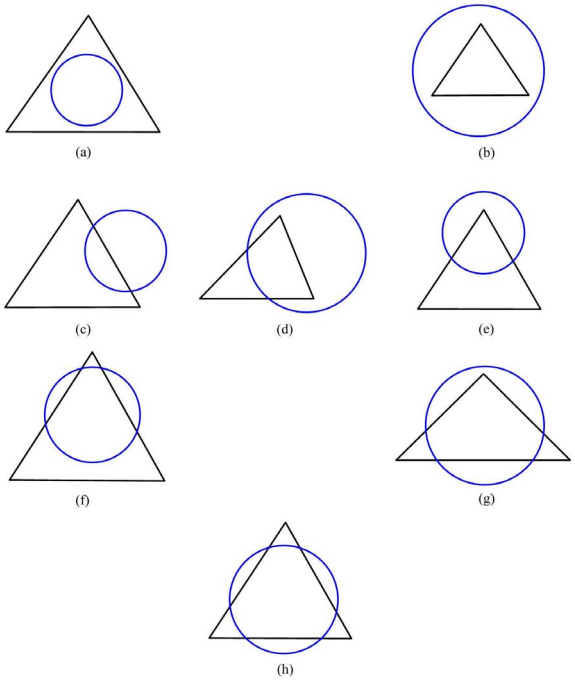

Figure 2. The relations between a sphere and a triangle. The first row is 0-point class, the second row is 2-point class, the third is 4-point class, and the fourth is 6-point class.

After considering the symmetry and removing the case that the intersecting points are on the vertices of the triangles (which can be classified into case (h)), there are two cases in 0-point class, three cases in 2-point class, two cases in 4-point class, and one case in 6-point class (see Fig.2).

After getting the intersection field generated by a sphere and a triangle, we decomposed it into some triangles and arches to compute the resultant SOTV. Therefore, the SOTV for cases (c)-(h) can be determined by adding SOTVs like case (b) and additional *swing* volumes. In summary, SOTV computed from decomposition can be expressed as

$$V(t, s) = \sum_{i=1}^{n} V_{b,i} + \sum_{j=1}^{m} V_{swing,j} \tag{7}$$

where n is the number of the decomposed triangles, m is the number of the arches, $V_{b,i}$ is the volume formed by the i-th triangle, and $V_{swing,j}$ is the j-th swing volume.

To be self-contained, the formulas for computing SOTV in case (a), (b) and the swing volume, which have been given in [30], are listed below.

1) For case (a), the volume is [30]
$$V_a = \pi h^2 (r - \tfrac{h}{3}),$$
where h is the height of the triangle t's plane above c and r is the sphere radius.

2) For case (b), the volume is given by [30]

$$V_b = V_{stri}(t,s) - V_{tet}(t,c) = \frac{1}{3}(r^3\Omega - Dh),$$

where $V_{stri}(t,s)$ is the volume bounded by the spherical triangle formed by projecting the vertices of triangle t onto the sphere s, and $V_{tet}(t,c)$ is the volume of the tetrahedron formed by the sphere center c and the three vertices of the triangle t. Ω is the solid angle of the triangle on the sphere and D is the area of triangle t.

3) Swing volume lies between two planes hinged between the two points p_0 and p_1 where the triangle edges exit the sphere. One of the planes is the triangle's, the other contains p_0, p_1 and the sphere center c. The angle between these two planes is denoted by φ, and then the volume V_s can be computed by [30]

$$V_{swing} = \int_{x_0}^{x_1} S(x)dx,$$

where $S(x) =$
$(\varphi - \arcsin(\frac{l_0}{\sqrt{1-x^2}}))r - \frac{1}{2}(\sqrt{r^2 - x^2 - l_0^2} - l_1)l_0,$
$l_0 = \sqrt{r^2 - (\frac{l}{2})^2}\sin(\varphi), l_1 = \sqrt{r^2 - (\frac{l}{2})^2}\cos(\varphi),$
$x_0 = \frac{-l}{2}, x_1 = \frac{l}{2}, l = \|p_1 - p_0\|.$

While gathering SOTV over all triangles inside or intersecting with the sphere s, we sign the volume according to two factors: whether the sphere center is inside the object or not, and whether the sphere center is behind the triangle's plane or not. The details were discussed in [30].

4. Ellipsoid-Tree Construction Framework

The ellipsoid-tree is constructed by a top-down splitting algorithm. Our basic idea was from [16] and was briefed as follows. Starting from the root of hierarchy, the volume occupied by a given model M is divided into k sub-volumes M_{o^i} where each is approximated and fully covered by a volume bounding ellipsoid o^i. 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 hierarchy.

Pseudo-code of the *Ellipsoid-Tree-Construction* algorithm is listed below.

In this algorithm, the most important steps are steps 6 and 7, which are the *Ellipsoids Merge Algorithm* and the *Variational Ellipsoids Optimization* respectively.

Details for these two techniques can be found in our previous publication [16].

5. Experimental Results

Three models — the human model in Fig.1, the horse in Fig.3 and the bunny in Fig.4, are tested based on both voxel-based solid approximation error evaluation and the analytical evaluation. Starting from the minimal volume bounding ellipsoid, four levels of the ellipsoid-tree are shown.

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 are constructed for approximating M_{o^i};
7: Optimize the shape and position of these k ellipsoids so that the approximation to M_{o^i} is tighter;
8: **end for**
9: Move to the level $(i+1)$ of the hierarchy;
10: **until** the approximation tolerance has been arrived;

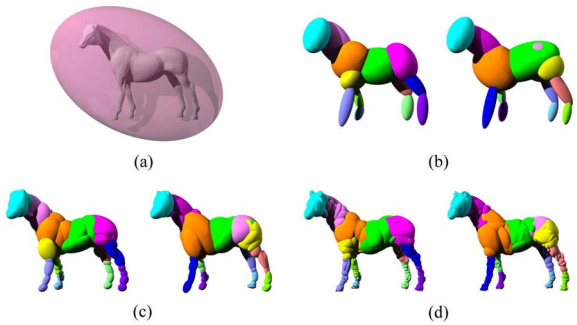

Figure 3. The ellipsoid-trees constructed by (left) the voxel-based and (right) the analytical EOTV based evaluation method for the horse model. Four levels are shown: (a) the given mesh model and the root node, (b) level 1, (c) level 2, (d) level 3. The ellipsoids in level 1 are displayed in different colors, and their children and children's children shown in the later levels follow these colors.

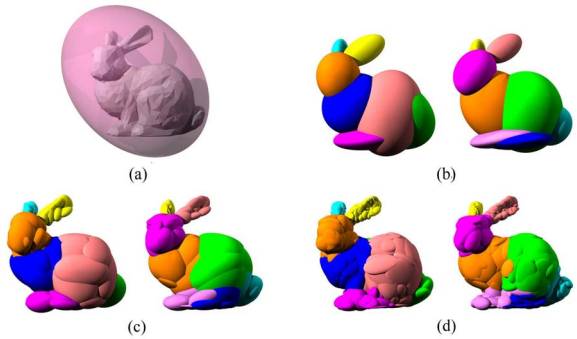

Figure 4. The ellipsoid-trees constructed by (left) the voxel-based and (right) the analytical EOTV based evaluation method for the bunny model — also four levels are shown: (a) the given mesh model and the root node, (b) level 1, (c) level 2, (d) level 3.

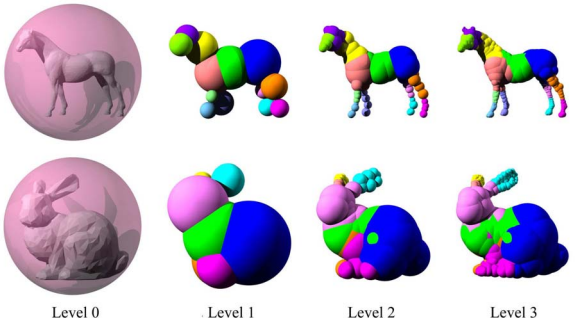

Figure 5. The sphere-tree for two models: the horse and the bunny — four levels are shown. The spheres in level 1 are displayed in different colors, and their children and children's children shown in the later levels follow these colors.

As the state-of-the-art method for sphere-tree construction, the adaptive medial axis approximation (AMAA) method [4] is chosen as the benchmark for our ellipsoid-tree construction method. The sphere-trees for the same three models have been given in Fig.1 and Fig.5. The sphere-trees are generated with the same number of primitives at each level as the ellipsoid-trees in Figs.1, 3 and 4. The sphere-trees are generated by the best strategy in [4] (i.e., the expand and select scheme). The ellipsoid-tree shows its strength for approximating the human and the horse rather than the bunny. The reason is that the shapes of previous two examples are more complex.

The computation cost of the proposed algorithm is relatively expensive. For computing an ellipsoid-tree with about 200 ellipsoids (e.g., the horse example), it takes about 500 seconds using the voxel-based method while 1,330 seconds using the analytic method. The expensive computation limits it to generate ellipsoid-tree with more than a few hundred ellipsoids.

6. Application — Shadow Rendering

In rendering, if the resolution of the given object is very high, it is time consuming to render the original mesh directly. The sphere set and ellipsoid set can be used as a proxy for rendering the original mesh or its shadow. We can use spheres or ellipsoids to approximate the mesh and then to speed its rendering. in Fig.6, we compare results using ellipsoid sets and sphere sets to cast shadows from a face light, where a ground plane is used to "catch" shadows underneath the objects generated by a face light. In Fig.6, the top row shows the original view, and the bottom row shows the ground plane images. Fig.6(a) is the original mesh and its shadow. Fig.6(b) and Fig.6(c) are shadowed from the ellipsoid sets generated by our algorithm in which the EOTV is computed using the analytical method and the voxel-based method respectively. Fig.6(d) shows the sphere set generated by AMAA and its shadow. Among the tests shown

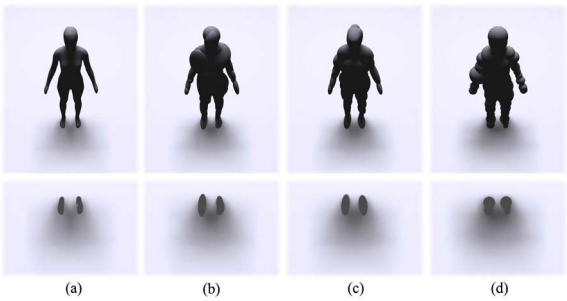

Figure 6. Shadow Rendering for Human model, (a) original mesh, (b) analytical method, $n = 32$, (c) voxel-based method, $n = 32$, (d) AMAA, $n = 64$, where n is the number of primitives.

in Fig.6, shadows generated by ellipsoid sets are much closer to the true shadow generated directly from the triangular mesh than the sphere sets from AMAA while using less primitives. These figures were generated by ray tracing using an available free software POVRay.

7. Conclusion and Future Work

In this paper, we develop an ellipsoid-tree construction method which can create a bounding volume hierarchy (BVH) from a given model with ellipsoids as primitives. In the framework, an analytical method in solid error computation is proposed. We tested the methods on several models. From the results, benefited from the anisotropic shape of primitives, it is obvious that the ellipsoid-tree constructed in our approach gives higher shape fidelity than the sphere-tree. The approximation errors on ellipsoid-trees generated by the voxel-based method and the analytical method are more or less similar for lower levels in the hierarchy. And although the analytical method are time-consuming in refining levels of the hierarchy, it can provide accurate results and better visual effects.

It is well known that how this novel hierarchy can be used in the collision handling of rigid objects, but it is still under research that how we can generate a bounded deformation tree (similar to [11]) using ellipsoid-tree. This will be one of our future work. The numerical test for the application of ellipsoid-tree in collision detection will also be done as future work.

Acknowledgment

The work is supported by the Hong Kong RGC/CERG grant CUHK/412405 , the CUHK DAG project CUHK/2050341, the Open Project Program of the State Key Lab of CAD&CG (Grant No.A0805), Zhejiang University, the National Natural Science Foundation of China (Grants No.60833007) and the Key Technology R&D Program (Grant No.2007BAH11B03).

References

[1] Amenta N, Choi S, and Kolluri RK. The power crust. In: Proceedings of the sixth ACM symposium on Solid modeling and applications, pp. 249-266, ACM Press, New York, NY, USA, 2001.

[2] Bischoff S and Kobbelt L. Ellipsoid decomposition of 3D-models. In:Proceedings of the 1st Symposium on 3D data processing, visualization and transmission, pp. 480-488, IEEE Press, 2002.

[3] Bradshaw G and O'Sullivan C. Sphere-tree construction using dynamic medial axis approximation. In: Proceedings of the 2002 ACM SIGGRAPH / Eurographics symposium on Computer animation, pp.33-40, ACM Press, New York, NY, USA, 2002.

[4] Bradshaw G and O'Sullivan C. Adaptive medialaxis approximation for sphere-tree construction. ACM Transactions on Graphics 2004; 23(1):1-26.

[5] Cohen-Steiner D, Alliez P, and Desbrun M. Variational shape approximation. ACM Transactions on Graphics 2004; 23(3):905–914.

[6] Choi Y-K, Chang J-W, Wang W, Kim M-S, and Elber G. Real-Time Continuous Collision Detection for Moving Ellipsoids under Affine Deformation. Technical Report TR-2006-02, Department of Computer Science, The University of Hong Kong, HongKong, 2006.

[7] Gonzalez-Ochoa C, McCammon S, and Peters J. Computing moments of objects enclosed by piecewise polynomial surfaces. ACM Transactions on Graphics 1998; 17(3):143-157.

[8] Choi Y-K, Wang W, Liu Y, and Kim M-S. Continuous collision detection for elliptic disks. IEEE Transactions on Robotics 2006; 22(2):213-224.

[9] Gottschalk S, Lin MC, and Manocha D. OBBTree: a hierarchical structure for rapid interference detection. In: Proceedings of SIGGRAPH '96, pp.171–180, ACM Press, New York, NY, USA, 1996.

[10] Hubbard PM. Collision detection for interactive graphics applications. Brown University, 1995.

[11] James DL and Pai DK. BD-tree: output-sensitive collision detection for reduced deformable models. ACM Transactions on Graphics 2004; 23(3):393-398.

[12] Krishnan S, Pattekar A, Lin MC, and Manocha D. 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, pp. 177–190, A. K. Peters, Ltd., Natick, MA, USA, 1998.

[13] Kumar P and Yildirim EA. Minimum volume enclosing ellipsoids and core sets. Journal of Optimization Theory and Applications 2005; 126(1):1-21.

[14] Liu S, Jin X, Wang CCL, and Hui K-C. Ellipsoidal-Blob Approximation of 3D Models and Its Applications. Computers & Graphics 2007; 31: 243-251.

[15] Lloyd S. Least squares quantization in PCM. IEEE Transactions on Information Theory 1982; 28(2):129-137.

[16] Liu S, Wang CCL, Hui K-C, Jin X, and Zhao H, Ellipsoid-tree construction for solid objects, In: Proceedings of ACM Solid and Physical Modeling Symposium 2007, pp.303-308, Beijing, China, June 4-6, 2007.

[17] Palmer I and Grimsdale R. Collision detection for animation using sphere-trees. Computer Graphics Forum 1995; 14(2):105-116.

[18] Quinlan S. Efficient distance computation between non-convex object. In: roceedings of IEEE International Conference on Robotics and Automation, pp.3324-3329, IEEE Press, 1994.

[19] Ranjan V and Fournier A. Union of spheres (UoS) model for volumetric data. In: Proceedings of the eleventh annual symposium on Computational geometry, pp. 402–403, ACM Press, New York, NY, USA, 1995.

[20] Rusinkiewicz S and Levoy M. QSplat: a multiresolution point rendering system for large meshes. In: Proceedings of SIGGRAPH 2000, pp. 343-352, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 2000.

[21] Samet H. The Design and Analysis of Spatial Data Structures. Addison-Wesley, 1989.

[22] Sun P and Freund RM. Computation of minimum volume covering ellipsoids. Operations Research 2004; 52:690-706.

[23] Simari PD, Singh K. Extraction and remeshing of ellipsoidal representations from mesh data. In: Proceedings of the 2005 conference on Graphics interface, pp.161-168, Canadian Human-Computer Communications Society, Waterloo, Ontario, Canada, 2005.

[24] Tam RC and Fournier A. Image interpolation using unions of spheres. The Visual Computer 1998; 14(8/9): 401-414.

[25] Todd M and Yildirim E. On Khachiyan's Algorithm for the Computation of Minimum Volume Enclosing Ellipsoids. Technical Report TR-1435, School of Operations Research and Industrial Engineering, Cornell University, 2005.

[26] van den Bergen G. Efficient collision detection of complex deformable models using AABB trees. Journal of Graphical Tools 1997; 2(4):1-13.

[27] Wang W, Choi Y-K, Chan B, Kim M-S, and Wang J. Efficient collision detection for moving ellipsoids using separating planes. Computing 2004; 72(1-2): 235-246.

[28] Weltz E. 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, pp.359-370, 1991.

[29] Wang W,Wang J, and Kim M. An algebraic condition for the separation of two ellipsoids. Computer Aided Geometry Design 2001; 18(6):531-539, 2001.

[30] Wang R, Zhou K, Snyder J, Liu X, Bao H, Peng Q, and Guo B. Variational sphere set approximation for solid objects. The Visual Computer 2006; 22(9): 612-621.