EG UK Theory and Practice of Computer Graphics (2009)
Wen Tang, John Collomosse (Editors)

Object Interaction Using Tabulated Spheres Subsets

A. A. Navarro Newball[1,2,3], G. Wyvill[1] and B. McCane[1]

[1]Computer Science Department, University of Otago, New Zealand
[2]Pontificia Universidad Javeriana, Cali, Colombia [3]Department of Computing and Mathematics, Manchester Metropolitan University, UK

Abstract

Tabulated Sphere Subsets (TSSs) provide a fast way to approximate collision tests between objects whose motion is constrained. A TSS is a subset of a set of spheres that approximate the shape of two objects that might collide. The subset represents only those spheres that can collide under the constrained motion.
A TSS is created in three steps: 1) approximating the mesh with spheres; 2) searching the space of possible motion to find which spheres may collide: 3) extracting the spheres required for collision tests and building the table.
We applied TSSs to jaw motion and skin/muscle interaction in a model dog and measured the number of spheres generated and the number of calculations needed for collision tests. In these cases TSSs outperforms several standard techniques.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three Dimensional Graphics and Realism]: Animation, I.3.5 [Computational Geometry and Object Modelling]: Physically Based Modelling

1. Introduction

Object interaction often implies collision detection. Collision detection is important in computer graphics. In games, it needs to be fast. The most used strategies for collision detection in polygonal meshes are hierarchical bounding volumes and axis aligned bounding volumes [HEV*04]. It is suggested that methods based on boxes bound the geometry more tightly implying fewer hierarchical levels and faster performance [GLM96].

However, the intersection test between spheres is simple and an object can be approximated with spheres successfully [Hub96]. O'Rourke and Badler [OB79] present an algorithm to tightly approximate an object with spheres by anchoring big spheres to points on the object and shrinking the spheres until they just fit inside. Hubbard [Hub96] and Bradshaw and O'Sullivan [BO02] use automatically generated sphere hierarchies for an effective search structure which tightly approximates the mesh. They approximate the medial axis to guide spheres' placement (medial axes are more accurate than spatial subdivision [BO02]). The medial axis can be approximated using Voronoi diagrams [SFM05]. Construction of one sphere can be done in several ways. Welzl [Wel91] offers a permutation technique to find optimal spheres from a set of points. Hierarchies require the construction of the spheres surrounding a set of spheres for merging adjacent spheres [BO02]. Fischer and Gärtner [FG03] use rational arithmetic and Kumar et.al. [KMY03] use core sets to find minimal spheres from other spheres. In a sphere hierarchy, an initial bounding volume test is followed by rejection of non colliding objects or identification of regions of potential collisions using a sphere tree which approximates the geometry. Then, the actual geometry of the object is tested in the regions identified.

We introduce TSSs, a technique for constrained interacting bodies which has the advantages of: 1) reducing the number of tests and computations; 2) relying on a simple operation; 3) using a simple data structure; 4) providing flexibility. It has three phases: spheres generation; colliding spheres search; spheres reduction and tabulation.

2. Tabulated Spheres Subsets

TSSs are a tabulated selection of the colliding spheres from the spheres' approximation of two interacting meshes with constrained degrees of freedom (DOF). A TSS indicates which spheres from one mesh must be tested for collision with which spheres from another mesh. Figure 1a shows objects A and B with 1 DOF. Here, M_s is the spheres' approximation of a mesh. Figure 1b shows the subset M_{sc} of spheres which collide on motion. Figure 1c shows the smallest sub-

DOI: 10.2312/LocalChapterEvents/TPCG/TPCG09/057-060

set found, M_{sm} (several redundant colliding spheres were removed). Figure 1d shows the minimal subset M_{ss} (only non redundant colliding spheres). The typical tabulated sphere subset of the two objects is the table $TSS_{AB} = M^A_{sm} \times M^B_{sm}$. M_s can be volumetric (Figure 1a) or surface (Figure 1e). The steps for generating a TSS are described next.

Sphere generation: Sphere generation can be done in many ways. We use a simple algorithm. For volumetric $M_s s$ we use a heuristic method to avoid inside / outside tests and random number generation. Radius growth is calculated along a line as in Figure 1f. Here, δt is found heuristically. In Figure 1g, we take the first point as a middle point from a triangle in the mesh (m_1) and grow the radius (r) towards a direction opposite to the normal (N). In Figure 1h, once we find the second point (m_2, middle point in other triangle), we obtain the middle point (m_L) between m_1 and m_2 and grow the sphere in a direction which goes from m_L to the centre of the old sphere (c). In Figure 1i, once we find the third point (m_3, middle point in other triangle), we obtain the middle point (m_T) of the triangle formed by m_1, m_2 and m_3 and grow the sphere in a direction which goes from m_T to the centre of the old sphere (c). The algorithm stops when a fourth point or a threshold radius is reached. For surface $M_s s$ we associate each vertex of the mesh with a sphere.

Colliding spheres search: this phase requires knowledge of the kinematics of the interacting objects to explore the space of motion and creates a table with all the collisions that occurred. In the case of the dog skull - mandible complex, kinematics can be implemented with two angles which consider opening / closing and lateral motion. The space is explored from many directions to diminish unexplored holes. In the case of the skin - Frontalis interaction, kinematics can be implemented contracting and expanding the muscle in all anatomically possible directions.

Spheres reduction and tabulation: after discarding the non colliding spheres, we create a collisions table (T_c). It indicates for each sphere in object A, which spheres in object B need to be tested. There are two cases: 1) spheres positions remain constant with respect to the meshes (rigid objects). 2) spheres positions are changed and interacting objects are deformed. For the latter, T_c is the TSS. For the former, if N_s spheres collided in object B, we create a $N_s \times N_s$ table (T_a). Then, we search in T_c how many times a sphere from object B is present and how many times every other sphere from object B is present at the same time. Results are counted and stored in T_a while T_c is searched . Once this is done, T_a contains all the information about the number of times that the spheres from object B have appeared. In T_a a sphere S_i is removed if the number of times it appears is the same as the number of times it appears together with some other sphere S_j (The number of times S_i appeared is stored in $T_a[S_i][S_i]$ and the number of times that S_i appeared together with S_j is stored in $T_a[S_j][S_i]$). Finally, we remove the redundant spheres from object A by searching in T_c for rows containing exactly the same spheres from object B. Results are stored in a TSS.

3. Results

TSSs were developed as a part of a virtual expressive dog head [NNWM08]. We tested them with two couples of interacting bodies: 1) the dog skull mandible complex; 2) the dog's facial skin and the Frontalis muscle. We generated an approximation with spheres for each body. Then, we covered the space of motion of the interacting bodies using dog's jaw kinematics (Couple 1) and muscle actuations (Couple 2). Finally, for each sphere, we identified with which ones it collides and created a collision table. We measured the number of spheres, tests and calculations done. We compared TSSs with sphere trees, angular regions and sphere - ellipsoid intersection.

For the dog skull and mandible, we generated volumetric spheres using the following parameters (refer to figures from 1f to 1i): $\delta t = 0.01$ for $0.01 \leq T_i \leq 0.101$ and a maximum radius of 0.02 (the use of a threshold value for the radii focused sphere generation on narrow regions). Figure 2a shows the initial set of spheres and Figure 2b the reduced subset. For the dog skin and the Frontalis we generated the surface spheres, one at each vertex. Figure 2c shows the initial set of spheres and Figure 2d the reduced subset. There was a significant decrease in the number of spheres for each case. The resulting TSSs made the collisions test very efficient requiring 22 tests per time step for jaw motion and 268 tests per time step for Frontalis skin interaction.

TSSs take advantage of inherent motion constraints from interacting bodies and reduce the number of collision tests. TSSs can be compared with sphere trees; constrained angles of motion or; allowed motion paths. Sphere trees are better for general collision detection; however, we evaluated them in a constrained environment. Constrained angles are not appropriate for a number of cases. For example, using 2DOF kinematics with the model in Figure 2e can cause problems in the highlighted region because rotation along Y can produce teeth interpenetration. However, one solution can be to define one angular region of collision testing and one free of collisions. Finally, pre definition of allowed paths is limited and every possible path of motion should be stored.

TSSs vs. Sphere trees: in a TSS the number of collision tests T_{tss} is $\sum_n^{i=1} k_i \leq n \times m$ (n = number of spheres in object A; m = number of spheres in object B; k_i = number of spheres from B that need to be tested against a sphere i from A) and is constant at every time step. The number of collision tests in a sphere tree varies every time step. Sphere trees require a tight approximation at each level and enhance colliding sphere search by focusing tests only on the colliding branches. However, higher memory storage is required. Excellent sphere trees studies can be found in Bradshaw and O'Sullivan [BO04]. We analysed two cases. 1) A sphere tree is obtained from the low level mesh's spheres approximation. Even at Level 1 the number of spheres is high (e.g. Fig 1 from Bradshaw and O'Sullivan [BO04] or, Figure 2f). In a sphere tree, Level 0 encloses all spheres at Level 1. In a complex constrained pair, where meshes are very close to each other, these root spheres are likely to collide every time step. This enforces a full collision test at Level 1. Let N_1 and M_1 be the number of spheres at Level 1 for meshes A and B. It is likely that $N_1 \times M_1 \geq T_{tss}$ and the sphere tree will perform worse. In general, in a sphere tree the number of collision tests T_{st} is $\sum_{L}^{i=1} G_i$ (L = number of levels in the tree; G_i = number of spheres colliding at Level i). Whenever $T_{st} \geq T_{tss}$, TSSs will behave better. 2) A tight sphere tree is obtained using M_{sm} (figures 1c and 2h). Figure 2j shows the comparison of the performance of this tree with that from the TSS. Here, M_{sm}^A has 9 spheres, and M_{sm}^B has 22 spheres.

TSSs vs. Angular region of collision: in Figure 2i, using TSSs, the total number of collision tests is $T \times T_{tss}$ and; using angular regions of collisions, it is $(T-\tau)(n \times m)$. In general T_{tss} is much lower than $n \times m$. For example, in the dog's mandible skull, 0.111(n x m) = (0.111)(9x22) = 22 = T_{tss} using M_{sm}. In consequence, to achieve $(T-\tau)(n \times m) = 0.11 \times T(n \times m)$; we require that τ = 0.89, leaving 11% of time steps falling in the collision region. In practice we got about 22% of time steps falling in the collision region. Thus, TSSs behave better.

TSSs vs. Sphere - Ellipsoid intersection: we used bent ellipsoids for representing the muscles. For instance, an alternative method for skin / Frontalis interaction could be solving the characteristic polynomial described by Wang et.al. [WWK01] for at least each one of the 70 selected spheres (Figure 2d). Using TSSs we required 268 tests with 4 products each, for a total of 1072 product computations. With sphere - ellipsoid intersection (with 70 spheres) we require an unbending operation for the ellipsoid plus 2870 product / division computations (70 X 41). Figure 2g shows the deformation produced in the skin using TSSs.

4. Discussion and further work

TSSs are fast and appropriate for constrained object interaction. They rely on a simple test; only consider spheres and regions of the mesh which will collide and only test a sphere with the spheres with which it is most likely to collide. They use significantly fewer spheres than other approaches; and no hierarchy is required. TSSs are intended for approximate collision tests and do not deal with polygon to polygon tests. Exact polygon / polygon tests take much longer. TSSs perform better than sphere trees, angular regions and intersection calculations for the example applications we have shown here (constrained interaction). Even though in deformable bodies a few more spheres were required to produce more detail; TSSs are a good alternative for skin muscle interaction which uses fewer collision tests. We will apply TSSs to the rest of the dog's facial muscles and extend the concept to be able to deform the ears and the lips.

References

[BO02] BRADSHAW G., O'SULLIVAN C.: Sphere-tree construction using dynamic medial axis approximation. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2002), ACM, pp. 33–40.

[BO04] BRADSHAW G., O'SULLIVAN C.: Adaptive medial-axis approximation for sphere-tree construction. *ACM Trans. Graph. 23*, 1 (2004), 1–26.

[FG03] FISCHER K., GARTNER B.: The smallest enclosing ball of balls: combinatorial structure and algorithms. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry* (2003), ACM, pp. 292–301.

[GLM96] GOTTSCHALK S., LIN M. C., MANOCHA D.: Obbtree: a hierarchical structure for rapid interference detection. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), ACM, pp. 171–180.

[HEV*04] HADAP S., EBERLE D., VOLINO P., LIN M. C., REDON S., ERICSON C.: Collision detection and proximity queries. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Course Notes* (2004), ACM, p. 15.

[Hub96] HUBBARD P. M.: Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans. Graph. 15*, 3 (1996), 179–210.

[KMY03] KUMAR P., MITCHELL J. S. B., YILDIRIM E. A.: Approximate minimum enclosing balls in high dimensions using core-sets. *J. Exp. Algorithmics 8* (2003), 1.1.

[NNWM08] NAVARRO NEWBALL A. A., WYVILL G., MCCANE B.: Towards the creation of londra: a virtual expressive dog. In *New Zealand Computer Science Research Student Conference, NZCSRSC 2008 Proceedings* (2008), pp. 238–241. http://nzcsrsc08.canterbury.ac.nz/site/papers.

[OB79] O'ROURKE J., BADLER N. I.: Decomposition of three-dimensional objects into spheres. *IEEE Trans. Pattern Anal. Mach. Intell. PAMI-1* (1979), 295–305.

[SFM05] SUD A., FOSKEY M., MANOCHA D.: Homotopy-preserving medial axis simplification. In *SPM '05: Proceedings of the 2005 ACM symposium on Solid and physical modeling* (2005), ACM, pp. 39–50.

[Wel91] WELZL E.: Smallest enclosing disks (balls and ellipsoids). In *New Results and New Trends in Computer Science*, Maurer H., (Ed.), LNCS. Springer, 1991.

[WWK01] WANG W., WANG J., KIM M.-S.: An algebraic condition for the separation of two ellipsoids. *Computer Aided Geometric Design 18*, 6 (2001), 531–539.

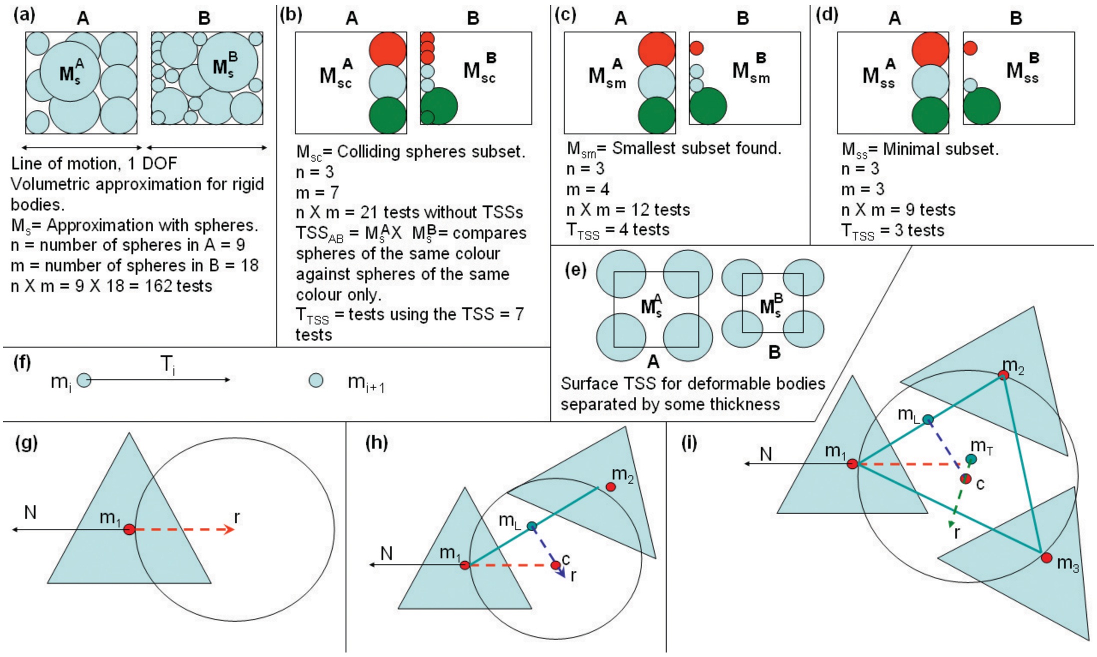

Figure 1: *a)b)c)d) n is the number of spheres in A, m is the number of spheres in B. Note that TSSs perform T_{TSS} tests which is lower than m X n. Also, $M_{ss} \subset M_{sm} \subset M_{sc} \subset M_s$. a) A and B approximated with volumetric spheres. b) The non colliding spheres are removed. c) The typical subset of spheres found (a few redundancies remain). d) Minimal subset with no redundancies. e) A and B approximated with surface spheres. f) The growth of a line between points m_i and m_{i+i} can be obtained from the parametric equation of a line in 3D space $m_i + T_i \times (m_{i+1} - m_i)$ by varying T_i by some δt. g) Growing radius (searching for the second point). h) Growing radius (searching for the third point). i) Growing radius (searching for the fourth point).*

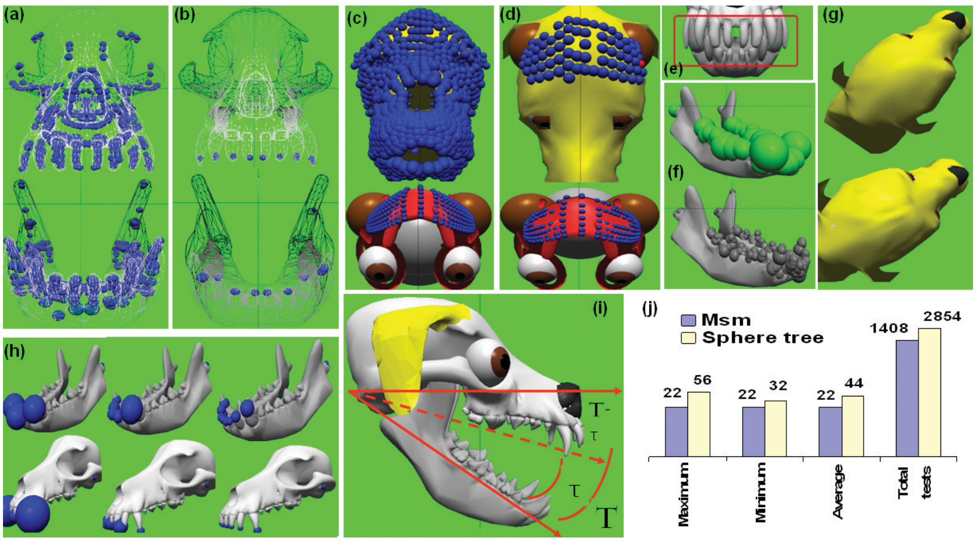

Figure 2: *a) Mandible: 812 spheres, skull: 1158 Spheres. b) Mandible: 22 spheres, skull: 9 spheres. c) Skin: 1426 spheres, Frontalis: 280 spheres. d) Skin: 70 spheres, Frontalis: 190 spheres. e) Problematic region. f) Two level of the sphere tree. g) Deformation of skin using TSSs. h) Tight sphere tree. i) The collision region is (T-τ), assume that the skull has n spheres and the mandible m. j) Performance of a sphere tree versus that of TSSs.*