Continuous Collision Detection for Ellipsoids

Yi-King Choi, Jung-Woo Chang, Wenping Wang,
Myung-Soo Kim, and Gershon Elber, *Member*, *IEEE*

Abstract—We present an accurate and efficient algorithm for continuous collision detection between two moving ellipsoids under rational Euclidean or affine motion. We start with a highly optimized implementation of interference testing between two stationary ellipsoids based on an algebraic condition described in terms of the signs of roots of the characteristic equation of two ellipsoids. Then, we derive a time-dependent characteristic equation for two moving ellipsoids, which enables us to develop an efficient algorithm for computing the time intervals in which two moving ellipsoids collide. The effectiveness of our approach is demonstrated with practical examples.

Index Terms—Ellipsoid, rational motion, Euclidean motion, affine motion, continuous collision detection, characteristic equation, zero set.

1 Introduction

Motion design, analysis, and planning are important research topics that furnish a common scientific base to diverse engineering disciplines such as robotics, CAD/CAM, computer animation, and 3D computer games [1]. For the simulation of realistic dynamical motions, rigid objects should not penetrate each other; and when they collide, impulsive response needs to be handled properly. Real-time collision detection is also crucial to physics engines for 3D computer games and simulation of virtual environments [2].

Collision detection for general freeform moving objects is computationally very expensive. The use of bounding volumes reduces the computational cost significantly by first performing easy tests to simple geometric primitives such as spheres [3], axis-aligned bounding boxes [4], [5], oriented bounding boxes (OBBs) [6], and discrete oriented polytopes [7]. Due to its simplicity and superior capability of shape approximation, the ellipsoid is used as the bounding volume for robotic arms and convex polyhedra for collision detection [8], [9], [10], [11]. Bischoff and Kobbelt [12] use a set of overlapping ellipsoids for a compact, robust, and level-of-detail representation of 3D objects defined as polygonal meshes. Hyun et al. [13] show that sweeps of ellipsoids fit tightly to human arms and legs. Thus, ellipsoids have much potential as a bounding volume for 3D freeform objects.

Rimon and Boyd [9] present a numerical technique for computing the *quasi-distance*, called *margin*, between two separate ellipsoids. Sohn et al. [14] compute the distance between two ellipsoids using line geometry. Using the Lagrange conditions, Lennerz and Schömer [15] present an algebraic algorithm for computing the distance between two quadrics. Distance computation is a more difficult problem than collision detection since the latter can be solved as a subproblem of the former—a positive distance between two objects implies no collision between the two.

Ellipsoids are also used to represent the shapes of soil particles in geomechanics and the isopotential surface of a molecule in computational physics [16]. The overlap test for ellipsoids is of high interest in these fields [17], [18]. In the field of astronautics, ellipsoids are used to represent threat volumes of space objects to determine possible close approach events [19].

Previous solutions for overlap test are mainly based on numerical techniques; moreover, they are limited to the case of stationary ellipsoids. For ellipsoids moving with on-the-fly motions, collision detection exploiting interframe coherence using separating planes has been studied in [20]. To deal with moving ellipsoids with prespecified motions, one may perform a sequence of interference tests between two stationary ellipsoids along their respective motion paths at discrete time intervals. Although temporal coherence can be taken into account for speedup, errors often occur due to inadequate temporal sampling. Therefore, it is desirable to achieve fast *continuous collision detection* (CCD) of ellipsoids.

CCD is currently an active research direction. Redon et al. [21], [22], [23], Govindaraju et al. [24], and Kim et al. [25] consider CCD in various simulation environments, comprising of hundreds of thousands of polygons as obstacles and complex moving objects composed of articulated links. They develop efficient algorithms of interactive speed for CCD while employing effective computational tools for culling redundant geometry at various stages of computation. Redon et al. [21] use OBB as the basic bounding volume, whereas Redon et al. [22], [23] and Kim et al. [25] employ Line Swept Sphere (LSS). These methods take geometric approaches in culling redundant geometry. In particular, Redon et al. [22], [23] and Kim et al. [25] apply a GPU-based collision detection to the swept volumes of LSS primitives against the environment; moreover, Govindaraju et al. [24] present a GPU-based algorithm that can also deal

- *Y.-K. Choi and W. Wang are with the Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong. E-mail: {ykchoi, wenping}@cs.hku.hk.*
- *J.-W. Chang and M.-S. Kim are with the School of Computer Science and Engineering, Seoul National University, Seoul 151-742, South Korea. E-mail: jw98@3map.snu.ac.kr, mskim@cse.snu.ac.kr.*
- *G. Elber is with the Department of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel. E-mail: gershon@cs.technion.ac.il.*

Manuscript received 21 Dec. 2007; revised 5 Apr. 2008; accepted 29 Apr. 2008; published online 8 May 2008.
Recommended for acceptance by M.C. Lin.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2007-12-0187.
Digital Object Identifier no. 10.1109/TVCG.2008.80.

with deformable models. Zhang et al. [26], on the other hand, deal with the CCD of articulated models with the approach of conservative advancement that repeatedly moves objects by a computed time step while ensuring noncollision. Significant performance gain is achieved by using Taylor models to construct dynamic bounding volume hierarchies of the articulated models. However, real-time CCD of ellipsoids has not been addressed in the literature. In this paper, we use an algebraic formulation of the problem and propose an efficient numerical solution that achieves real-time performance.

Because swept volumes and distances are difficult to compute for ellipsoids, an algebraic approach seems more suitable for the CCD of ellipsoids. Research in surface-surface intersection of quadrics, which is closely related to the problem of collision detection of ellipsoids, also suggests that the algebraic treatment is a natural approach for ellipsoids—geometric approaches usually produce efficient intersection algorithms only for a limited class of simple quadrics, called *natural quadrics* (i.e., spheres, circular cylinders, and cones) [27], [28], while algebraic techniques are capable of handling general quadrics [29], [30], [31]. Indeed, our algebraic approach leads to an accurate solution to the CCD problem for moving ellipsoids under rational Euclidean or affine motions.

The CCD for moving ellipsoids in 3D space is far more complex than that for moving ellipses in 2D plane. An algebraic approach is used in solving the CCD problem for moving ellipses [32], where a univariate polynomial is formulated whose roots correspond to the time instants at which the ellipses are in internal or external touch. For moving ellipsoids, however, the same approach of relying on detecting the roots of the univariate polynomial is infeasible, since a root of such a univariate equation may not correspond to any contact between the two ellipsoids, as pointed out in [32].

Based on the algebraic condition of Wang et al. [33] for the separation of two stationary ellipsoids, we proposed in our preliminary study [34] a method that reduces the CCD problem for two moving ellipsoids to an analysis of the zero set of a bivariate polynomial equation, which has high degree in the time parameter t. The resulting algorithm proposed there cannot meet the real-time requirement as it takes seconds to perform a single CCD of moving ellipsoids.

In this paper, we use the same algebraic formulation in [34] but shall present a new efficient numerical method to solve the problem about three orders of magnitude faster than the previous method in [34], thus bringing ellipsoid-based CCD into the realm of computer graphics for real-time applications. This is achieved by exploring the special structure of the bivariate function under consideration and employing several novel and efficient search techniques. It is assumed throughout that the motions of moving ellipsoids, either Euclidean or affine, are expressible as rational functions of the time parameter t.

Our main contributions are given as follows:

- We present an efficient implementation of the algebraic method for detecting overlap between two stationary ellipsoids, which requires 107 additions/subtractions, 141 multiplications, and six divisions.
- We present an accurate and efficient algorithm for detecting the collision between two moving ellipsoids suitable for real-time applications. The proposed algebraic approach computes the contact time, contact point, as well as the time interval of collision.
- Our algorithm works not only for Euclidean motions but also for affine motions, which means that the moving ellipsoids may change their shapes under affine transformations. This facility can be a potential advantage over the traditional methods when adapting our method to collision detection for deformable moving objects, such as human or animal bodies.

Now, a few words on the practicality of our result are in order. According to the operation counts, our approach requires about 20 percent to 30 percent more arithmetic operations than the OBB overlap test [6] and even more operations than other tests such as spheres, AABBs, k-DOPs, and LSSs. Thus, the ellipsoidal CCD should be applied to special cases where ellipsoids provide tighter fit to freeform objects, possibly undergoing deformations that can locally be approximated by affine deformations. To this end, the recent trend in 3D modeling for the next generation GPU architecture [35], [36] is quite promising, where 3D shapes are directly represented using parametric surfaces to alleviate the bottleneck of bus bandwidth. As indicated by the Dupin indicatrix of a surface, convex parts of surfaces can be tightly fit with ellipsoids. Exact contact time and contact point of two ellipsoids would provide good initial solutions for further processing of the underlying parametric surfaces.

The rest of this paper is organized as follows: We first present an algorithm for detecting overlap between two stationary ellipsoids in Section 2, focusing on an efficient implementation with a minimized number of arithmetic operations. Then, we present the CCD algorithm for two moving ellipsoids in Section 3. We present some experimental results in Section 4, and conclude this paper in Section 5. To keep a comfortable flow of reading, detailed analysis and argument are given in the appendices.

2 Detecting Overlap between Stationary Ellipsoids

In this section, we present an efficient algorithm for detecting overlap between two stationary ellipsoids, which are assumed to be sampled instances of two moving ellipsoids at the same instant. This algorithm is based on the separation condition for two ellipsoids proved in [33]. The contribution here is an optimized algorithm with a minimal number of arithmetic operations; we conclude that 107 additions/subtractions, 141 multiplications, and six divisions are needed. This efficient implementation, while having practical values in its own right, will be invoked in the subsequent method for the CCD of moving ellipsoids.

An ellipsoid $\mathcal{A}$ is represented by a quadratic equation $X^TAX = 0$ in $\mathbb{E}^3$, where $X = (x, y, z, w)^T$ are the homogeneous coordinates of a point in 3D space. The symmetric matrix A is normalized so that the interior of $\mathcal{A}$ is given by $X^TAX < 0$; this amounts to assuming that the determinant $|A| < 0$.

Two ellipsoids are said to be *overlapping* if their interiors have nonempty intersection. They are said to be *separate* or *disjoint* if their boundary surfaces and interiors share no common points. Two ellipsoids that are not separate but share no common interior points are said to be *touching*.

For two ellipsoids $\mathcal{A} : X^T AX = 0$ and $\mathcal{B} : X^T BX = 0$ in $\mathbb{E}^3$, the quartic polynomial $\mathrm{f}(\lambda) = \det(\lambda A - B)$ is called the *characteristic polynomial* and $\mathrm{f}(\lambda) = 0$ is called the *characteristic equation* of $\mathcal{A}$ and $\mathcal{B}$. The polynomial $\mathrm{f}(\lambda)$ has degree 4, its leading term has a negative coefficient, and it always has two positive real roots. The following theorem [33] captures the relationship between the geometric configuration of two ellipsoids and the roots of their characteristic equation.

Theorem 1 (separation condition of two stationary ellipsoids). *Let $\mathcal{A}$ and $\mathcal{B}$ be two ellipsoids with characteristic equation $\mathrm{f}(\lambda) = 0$. Then,*

1. *$\mathcal{A}$ and $\mathcal{B}$ are separate if and only if $\mathrm{f}(\lambda) = 0$ has two distinct negative roots;*
2. *$\mathcal{A}$ and $\mathcal{B}$ touch each other externally if and only if $\mathrm{f}(\lambda) = 0$ has a negative double root.*

Remark 1. Note that the theorem in [33] assumes that the characteristic equation has the form of $\mathrm{f}(\lambda) = \det(\lambda A + B) = 0$, and therefore, the result there is stated in terms of positive roots. Our changes here make the presentation consistent with the classic literature in linear algebra.

Remark 2. Clearly, the leading coefficient and the constant term of $\mathrm{f}(\lambda)$ are $|A|$ and $|B|$. So, they are negative [33]. This implies that $\mathrm{f}(\lambda) = 0$ has two distinct negative roots if and only if $\mathrm{f}(\lambda_0) > 0$ for some $\lambda_0 < 0$. The latter condition on a sign test is more convenient, especially when we consider two moving ellipsoids.

2.1 Characteristic Polynomial

For efficient implementation, it is crucial to set up the characteristic equation using a minimal number of arithmetic operations. We now present an efficient algorithm for this computation.

An ellipsoid is said to be in canonical form if it is represented by a diagonal matrix

$$A = \begin{bmatrix} 1/a^2 & 0 & 0 & 0 \\ 0 & 1/b^2 & 0 & 0 \\ 0 & 0 & 1/c^2 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}. \tag{1}$$

Under an affine transformation M_A, this ellipsoid is transformed to one in a general form with coefficient matrix $(M_A^{-1})^T A M_A^{-1}$. Now, assume that we use two transforms M_A and M_B to obtain two ellipsoids $(M_A^{-1})^T A M_A^{-1}$ and $(M_B^{-1})^T B M_B^{-1}$, where A and B are diagonal matrices representing ellipsoids in canonical positions. Then, the characteristic polynomial of the two ellipsoids is $\mathrm{f}(\lambda) = \det(\lambda(M_A^{-1})^T A M_A^{-1} - (M_B^{-1})^T B M_B^{-1})$.

In the following, we first compute the coefficients of the quartic polynomial $\mathrm{f}(\lambda)$, and then the signs of the roots of the polynomial are computed to determine the relative configuration of the two ellipsoids. Given two ellipsoids represented as the images of their standard diagonal form [cf. (1)] under the transformations M_A and M_B, we may simultaneously transform them to A and $M_A^T (M_B^{-1})^T B M_B^{-1} M_A$, where A is a diagonal matrix as in (1), and $M_A^T (M_B^{-1})^T B M_B^{-1} M_A$ is treated as a general 4×4 matrix. The characteristic polynomial then takes the following form: $\mathrm{f}(\lambda) = \det(\lambda A - M_A^T (M_B^{-1})^T B M_B^{-1} M_A)$; obviously, the roots of the characteristic polynomial remain the same as before. The power form of $\mathrm{f}(\lambda)$ in λ can be obtained by expanding the determinant $\det(\lambda A - M_A^T (M_B^{-1})^T B M_B^{-1} M_A)$. Then, we can use its Sturm sequence to determine whether the two ellipsoids overlap, by Theorem 1.

2.2 Computational Cost

To count the number of negative real roots of $\mathrm{f}(\lambda)$, we will first compute the Sturm sequence of $\mathrm{f}(\lambda)$ and then check the sign flips of this sequence at 0 and $-\infty$. For the moment we assume that M_A and M_B are Euclidean transformations, since this is a case that is used most often in applications. To compute $M_A^T (M_B^{-1})^T B M_B^{-1} M_A$, we note that M_B is the composition of a rotation R_B followed by a translation V_B, so its inverse M_B^{-1} is equivalent to a rotation R_B^T followed by a translation $-R_B^T V_B$. Based on this observation, we can count the arithmetic operations as follows:

1. Computing M_B^{-1} requires nine additions/subtractions and nine multiplications.
2. $M_B^{-1} M_A$ requires 27 additions/subtractions and 36 multiplications.
3. $M_A^T (M_B^{-1})^T$ is the transpose of $M_B^{-1} M_A$ and so needs no arithmetic operation.
4. Since B is a diagonal matrix, $B M_B^{-1} M_A$ requires 12 multiplications.
5. Finally, $M_A^T (M_B^{-1})^T B M_B^{-1} M_A$ can be constructed using additional 21 additions/subtractions and 30 multiplications.

Thus, we need 57 additions/subtractions and 87 multiplications to obtain $M_A^T (M_B^{-1})^T B M_B^{-1} M_A$. Then, the characteristic polynomial can be computed with another 29 additions/subtractions and 39 multiplications using the algorithm presented in Appendix A. The derivative of a quartic polynomial can be computed using three multiplications. To divide a degree n polynomial by a degree $(n-1)$ polynomial, we need $2(n-1)$ additions/subtractions, $2(n-1)$ multiplications, and two divisions. Thus, we can compute the Sturm sequence using 12 additions/subtractions, 15 multiplications, and six divisions. To find the number of negative real roots, we need to examine the signs of the leading term and constant term of the polynomials in the Sturm sequence, for which eight additions/subtractions are needed to count the number of sign flips. In summary, we need a total of 107 additions/subtractions, 141 multiplications, and six divisions for collision detection between two stationary ellipsoids.

When the two stationary ellipsoids above are sampled from affine motions, it can be shown that we need a total of 125 additions/subtractions, 156 multiplications, and 18 divisions for detecting their collision. We skip the detailed counting here.

We have implemented the collision detection algorithm in C++ and run our tests on a desktop PC with an Intel Core 2 Duo E6600 2.40-GHz CPU (single-threaded) and a 2-Gbyte main memory. In the case of motion matrices with elements of rational degree 4, the matrices M_A and M_B are constructed

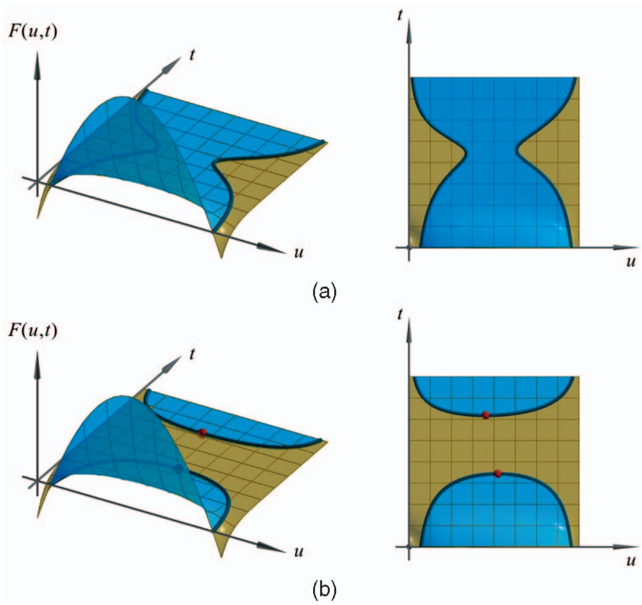


Fig. 1. (a) A CCD function for two collision-free ellipsoids. (b) A CCD function for two colliding ellipsoids. The blue and yellow regions are where $F(u,t) > 0$ and $F(u,t) < 0$, respectively. The zero set $F(u,t) = 0$ is given by the dark blue curve. The red points in (b) represent the moments when the ellipsoids are in external contact.

using about 100 additions/subtractions and 100 multiplications. Including this, the whole procedure of detecting overlap between two ellipsoids took less than 0.7 μs.

2.3 Contact Point of Two Touching Ellipsoids

Part 2 of Theorem 1 states that two ellipsoids have external contact if and only if their characteristic equation $\mathrm{f}(\lambda) = 0$ has a negative double root λ_0. It is also proved in [33, Lemma 5] that the contact point of two touching ellipsoids is given by the solution of $(\lambda_0 A - B)X = 0$, as summarized in the following theorem.

Theorem 2. *Suppose that two ellipsoids $X^T AX = 0$ and $X^T BX = 0$ touch externally, i.e., $\mathrm{f}(\lambda) = 0$ has a negative double root λ_0. Then, $\mathrm{rank}(\lambda_0 A - B) = 3$ and the homogeneous coordinates of the contact point X_0 are given by the unique nontrivial solution (up to a multiplicative constant) of $(\lambda_0 A - B)X = 0$.*

3 Continuous Collision Detection

In this section, we will present an efficient algorithm for CCD between two moving ellipsoids: $\mathcal{A}(t) : X^T A(t)X = 0$ and $\mathcal{B}(t) : X^T B(t)X = 0$. Here, the ellipsoids may move under affine deformations, including the commonly used Euclidean rigid motions as a special case. The formation of $A(t)$ and $B(t)$ in the case of rational motions are given in Appendix B.

3.1 CCD Equation for Moving Ellipsoids

We first introduce the CCD equation of the two moving ellipsoids $\mathcal{A}(t)$ and $\mathcal{B}(t)$. This CCD equation is simply the characteristic equation of $\mathcal{A}(t)$ and $\mathcal{B}(t)$, $\mathrm{f}(\lambda; t) = \det(\lambda A(t) - B(t)) = 0$, $t \in [0,1]$. We will call $\mathrm{f}(\lambda; t)$ the *CCD function*. The graph of the typical CCD function is shown in Fig. 1a for two collision-free moving ellipsoids and in Fig. 1b for two colliding moving ellipsoids. (Note that λ is replaced by a function of u as discussed below.)

Our CCD algorithm exploits some special features of the zero set of the CCD equation. Consider a fixed time $t_0 \in [0,1]$. If $\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ are separate, according to the discussions in Section 2, $\mathrm{f}(\lambda; t_0) = 0$ has two negative real roots, that is, the line $t = t_0$ has two intersection points with the zero set of $\mathrm{f}(\lambda; t)$ in the half-plane $\lambda < 0$. If $\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ overlap, $\mathrm{f}(\lambda; t_0) = 0$ has no negative real root, that is, the line $t = t_0$ has no intersection point with the zero set of $\mathrm{f}(\lambda; t)$ in the infinite strip $(-\infty, 0] \times [0,1]$. Finally, if $\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ are externally tangent, the line $t = t_0$ has a tangential intersection (i.e., a double intersection point) with the zero set of $\mathrm{f}(\lambda; t)$ in the half-plane $\lambda < 0$.

To facilitate numerical processing, we use the reparameterization $\lambda = \frac{u-1}{u}$ to map the variable $\lambda \in (-\infty, 0]$ to $u \in (0,1]$; therefore, the infinite strip $(\lambda, t) \in (-\infty, 0] \times [0,1]$ is mapped to the region $(u,t) \in (0,1] \times [0,1]$. This mapping preserves the structure of $\mathrm{f}(\lambda; t) = 0$ in the sense that the number of intersections between a horizontal line $t = t_0$ and the zero set of $\mathrm{f}(\lambda; t) = 0$ is the same as that between the line $t = t_0$ and the zero set of $\mathrm{f}(\lambda(u); t) = 0$. Clearly, the transformed characteristic equation $\mathrm{f}(\lambda(u); t) = 0$ has the same zero set as the equation

$$\hat{F}(u,t) \equiv \det((u-1)A(t) - uB(t)) = 0,$$
$$(u,t) \in (0,1] \times [0,1].$$

Recall that the elements of $A(t)$ and $B(t)$ are rational functions of t. Since we are only interested in the zero set of $\hat{F}(u,t)$, we use $F(u,t)$ to denote the bivariate polynomial after cleaning up the common denominator in $\hat{F}(u,t)$. Clearly, $F(u,t)$ and $\hat{F}(u,t)$ have the same zero set. Furthermore, to improve numerical robustness we represent $F(u,t)$ in the Bernstein form. From now on, we will also call $F(u,t) = 0$ the CCD equation.

Based on the preceding discussion and notation, we have the following theorems:

Theorem 3. *Any horizontal line $t = t_0 \in [0,1]$ has at most two intersections with the zero set of $F(u,t)$ in the region $(0,1] \times [0,1]$. In particular,*

1. *$\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ are separate if and only if the line $t = t_0$ intersects the zero set of $F(u,t)$ in two distinct points in $(0,1] \times [0,1]$;*
2. *the interiors of $\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ intersect if and only if the line $t = t_0$ does not intersect the zero set of $F(u,t)$ in $(0,1] \times [0,1]$;*
3. *$\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ are externally tangent if and only if the line $t = t_0$ has a double intersection point with the zero set of $F(u,t)$ in $(0,1] \times [0,1]$.*

The next theorem is fundamental to our CCD algorithm.

Theorem 4. *Let $\mathcal{A}(t)$ and $\mathcal{B}(t)$ be two moving ellipsoids in continuous motion in $t \in [0,1]$. Suppose that at $t = 0$, the ellipsoids $\mathcal{A}(0)$ and $\mathcal{B}(0)$ are separate. Then, $\mathcal{A}(t)$ and $\mathcal{B}(t)$ collide in $t \in [0,1]$ if and only if there exists a time t_0 in [0, 1] such that the line $t = t_0$ has a double intersection point (u_0, t_0) with the zero set of $F(u,t)$ in the region $(0,1] \times [0,1]$.*

Proof. Suppose that there exists a time t_0 in [0, 1] such that the line $t = t_0$ intersects the zero set of $F(u,t)$ at a double

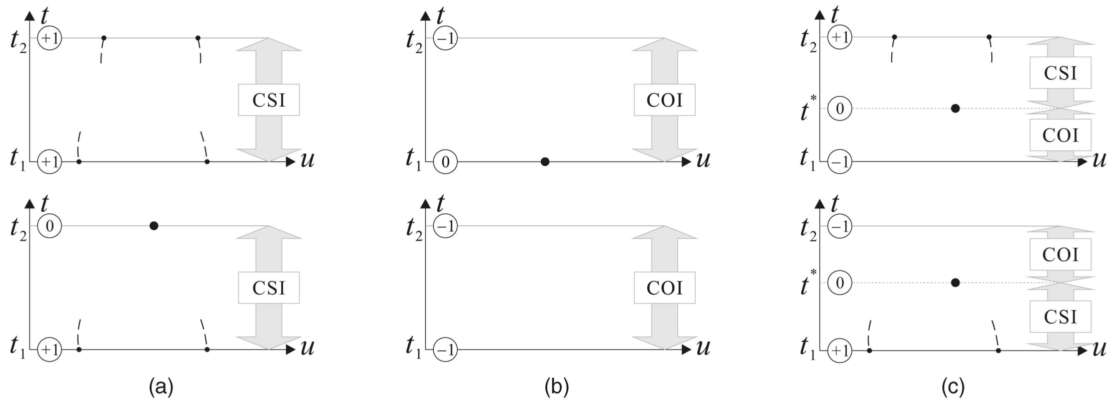


Fig. 2. Examples of intervals classified as (a) CSI, (b) COI, and (c) MIs with different collision statuses at t_1 and t_2 will be divided at a contact instant t^* and each of the two subintervals being classified as either a CSI or a COI. The circled numbers are the collision statuses at particular time instants ($+1$ for separation, 0 for external contact, and -1 for overlapping).

point in the region $(0,1] \times [0,1]$. Then, by Theorem 3, $\mathcal{A}(t_0)$ and $\mathcal{B}(t_0)$ touch each other externally. Therefore, $\mathcal{A}(t)$ and $\mathcal{B}(t)$ collide in $t \in [0,1]$.

Now, consider necessity. Suppose that $\mathcal{A}(t)$ and $\mathcal{B}(t)$ collide in $t \in [0,1]$. Then, either $\mathcal{A}(t)$ and $\mathcal{B}(t)$ touch each other externally at some time t_0 in [0, 1] or $\mathcal{A}(t)$ and $\mathcal{B}(t)$ overlap with each other at some time $t_1 \in [0,1]$. In the former case, we are done. In the latter case, since $\mathcal{A}(t)$ and $\mathcal{B}(t)$ are undergoing continuous motions and they are separate at $t = 0$, there exists time $t_0 \in [0, t_1]$ such that $\mathcal{A}(t)$ and $\mathcal{B}(t)$ touch each other externally at t_0. The proof is completed. □

Theorem 4 suggests how to detect whether two moving ellipsoids collide. First, we may check if $\mathcal{A}(0)$ and $\mathcal{B}(0)$ are separate, using the procedure in Section 2. If not, we are done; if yes, we need to check if there exists a time t_0 in [0, 1] such that the line $t = t_0$ has a double intersection point (u_0, t_0) with the zero set of $F(u,t)$ in $(0,1] \times [0,1]$. Clearly, such a point (u_0, t_0) is a solution of the equations $F(u,t) = F_u(u,t) = 0$, where $F_u(u,t)$ denotes $\partial F(u,t)/\partial u$. To find all the collision intervals, we note that whenever the collision status of two ellipsoids switches from separation to overlap (or vice versa), there must be a time instant at which the ellipsoids are in external contact; and hence, the key task of our collision detection algorithm now is to detect all real solutions of $F(u,t) = F_u(u,t) = 0$ in the region $(u,t) \in (0,1] \times [0,1]$.

3.2 Solving the CCD Equation

So far, we have given an algebraic formulation of the problem under consideration. Now, we shall present a numerical method based on this formulation. Given two moving ellipsoids over time [0, 1], if they are separate throughout a time interval $(t_0, t_1) \subseteq [0,1]$, then the interval (t_0, t_1) is called a *separation interval* (*SI*). An SI (t_0, t_1) is called a *maximal SI* if 1) the two ellipsoids contact each other at t_0 or $t_0 = 0$ and 2) the two ellipsoids contact each other at t_1 or $t_1 = 1$. If the ellipsoids overlap throughout the interval (t_0, t_1), then (t_0, t_1) is called an *overlapping interval* (*OI*). Similarly, we can define the *maximal OI*. An interval $(t_0, t_1) \subset [0,1]$ that is neither an SI nor an OI is called a *mixed interval* (*MI*). Our goal is to identify all the maximal SIs and maximal OIs.

By solving the CCD equation, we mean determining all *contact instants* at which the two ellipsoids are in external contact. Clearly, these instants define the endpoints of all the maximal SIs and maximal OIs. The contact instants correspond to the *critical points* in the zero set of the CCD equation—a solution (u^*, t^*) of $F(u,t) = 0$ is said to be a *critical point* if it further satisfies $F_u(u^*, t^*) = 0$. In this case, the contact instant is t^*.

Basic idea. The idea of our algorithm is to subdivide recursively the motion interval [0, 1] into a number of small intervals, which can be confirmed to be either SI or OI. Then, these intervals can be merged to form maximal SIs and maximal OIs.

During the process of our algorithm, for each interval (t_1, t_2) under consideration, we first determine the collision statuses of the two ellipsoids at the two endpoints of the interval. The interval (t_1, t_2) is temporarily labeled as a *candidate SI* (CSI) if the two ellipsoids are either separate or touching at t_1 and t_2 (Fig. 2a), since such an interval may be an SI in this case. Similarly, an interval (t_1, t_2) is temporarily labeled as a *candidate OI* (COI), if the two ellipsoids are either overlap or touching at t_1 and t_2 (Fig. 2b). Further processing is needed to confirm whether a CSI is an SI or a COI is an OI.

If the two moving ellipsoids have different collision statuses (either separate or collide) at t_1 and t_2 (Fig. 2c), then (t_1, t_2) is an MI. In this case, we will find a contact moment t^* in (t_1, t_2) and use it to subdivide (t_1, t_2) into two intervals (t_1, t^*) and (t^*, t_2). Evidently, one of the two intervals is a CSI and the other is a COI.

In the following, we are going to devise robust tests to determine definitely whether a CSI (or COI) is a separation (or overlapping) interval. If the collision status over the entire interval is confirmed, we are done and the interval is labeled as an SI or OI. Otherwise, the interval will be subdivided at some contact time t^* so that we will work on the resulting smaller intervals in a recursive manner, until the collision status of the ellipsoids in all subintervals can be confirmed.

To determine the collision status of two ellipsoids at a particular time t_0, we introduce the following *state function*:

$$\mathrm{State}(t_0) = \begin{cases} +1, & \text{if } \max_u F(u,t_0) > 0, \text{ i.e., separate,} \\ 0, & \text{if } \max_u F(u,t_0) = 0, \text{ i.e., touching,} \\ -1, & \text{if } \max_u F(u,t_0) < 0, \text{ i.e., overlap.} \end{cases}$$

Instead of using the efficient method in Section 2.2, this function makes use of the sign of $\max_u F(u, t_0)$ to check the collision status of two static ellipsoids, whose value can be found by solving the cubic equation $F_u(u,t) = 0$ and can be reused in other steps of the algorithm, e.g., for the computation of contact time as discussed below. Here, $\mathrm{State}(t_0) = 0$ if and only if (u_0, t_0) is a critical point for some $u_0 \in (0, 1]$.

We now describe our algorithm in details.

Initialization. We start by classifying the initial interval [0, 1] as a CSI or a COI, depending on the collision statuses at $t = 0$ and $t = 1$. If the collision statuses at $t = 0$ and $t = 1$ are different, we compute a contact instant t^* (corresponding to a critical point (u^*, t^*)), where the ellipsoids are in external contact, using the following operation:

- **ContactTime**: This is to determine a contact instant in an interval $[t_1, t_2]$, when the collision statuses of the ellipsoids at t_1 and t_2 are different. It is done by a binary search in t to find $t^* \in [t_1, t_2]$ such that $\mathrm{State}(t^*) = 0$. We then have ContactTime$(t_1, t_2) = t^*$. (In the binary search, we take into account the local maximum values $\max_u F(u, t_1)$ and $\max_u F(u, t_2)$. However, we omit the details here.)

We then subdivide [0, 1] into two smaller intervals $[0, t^*)$ and $(t^*, 1]$ and classify each as a CSI or a COI (as in Fig. 2c with $t_1 = 0$ and $t_2 = 1$).

Algorithm 1 Initialization

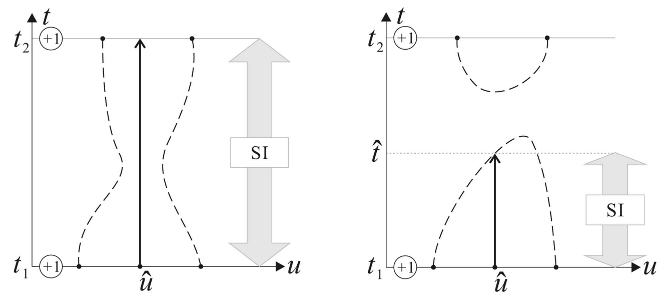
Input: $F(u, t)$ with $(u, t) \in (0, 1] \times [0, 1]$

if State(0) $= +1$ and State(1) $= +1$ **then**
  label $[0, 1]$ as CSI
else if State(0) $= -1$ and State(1) $= -1$ **then**
  label $[0, 1]$ as COI
else
  $t^* \leftarrow$ ContactTime$(0, 1)$
  report the contact time t^*
  if State(0) $= -1$ **then**
    label $[0, t^*)$ as COI and $(t^*, 1]$ as CSI
  else
    label $[0, t^*)$ as CSI and $(t^*, 1]$ as COI

Remark 3. For the sake of robustness, if $\mathrm{State}(0) = 0$, we replace State(0) by $\mathrm{State}(\epsilon)$, where $\epsilon > 0$ is a sufficiently small constant. Similarly, if $\mathrm{State}(1) = 0$, we replace State(1) by $\mathrm{State}(1 - \epsilon)$. Thus, we assume that State(0) and State(1) can never be 0.

Processing CSIs. For a CSI (t_1, t_2), we use the following operation, called *BézierShoot*, to either confirm that (t_1, t_2) is an SI or, if it is not, extract an SI, which is a subinterval of (t_1, t_2).

- **BézierShoot:** A *Bézier shoot* from t_1 to t_2, denoted as $\mathrm{BézierShoot}(t_1 \to t_2) = \hat{t}$, is to find an SI $(t_1, \hat{t}) \subseteq (t_1, t_2)$. It has two steps. In the first step, we find $\hat{u}$ such that $F(\hat{u}, t_1) = \max_u F(u, t_1)$. (As discussed in Remark 3, to ensure robustness, t_1 is replaced by $t_1 + \epsilon$ if t_1 is a contact instant.) If $F(\hat{u}, t_1) \leq 0$, we conclude that no SI can be thus extracted (and set $\hat{t} = t_1$). Otherwise, we use in the second step the Bézier clipping search [37] from t_1 to t_2 to find the first root of $F(\hat{u}, t) = 0$ (an equation in t with $\hat{u}$ being fixed), if there is one. This step either concludes that there is no real root of $F(\hat{u}, t) = 0$ in (t_1, t_2) (see Fig. 3a), which implies that (t_1, t_2) is an SI (and hence $\hat{t} = t_2$), or produces the smallest root $\hat{t}$ of $F(\hat{u}, t) = 0$ in (t_1, t_2) (see Fig. 3b), which gives an SI $(t_1, \hat{t}) \subset (t_1, t_2)$, since a Bézier shoot ensures that $F(\hat{u}, t) > 0$ for all $t \in (t_1, \hat{t})$. A Bézier shoot from t_2 to t_1, i.e., BézierShoot$(t_1 \leftarrow t_2)$, is defined similarly.



(a) (b)

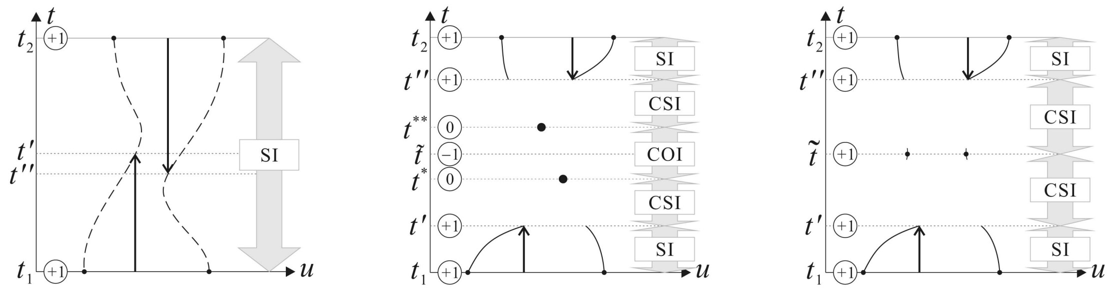
Fig. 3. A Bézier shoot operation. (a) $F(\hat{u}, t) = 0$ has no real root in $[t_1, t_2]$. (b) $\hat{t}$ is the smallest root of $F(\hat{u}, t) = 0$ in $[t_1, t_2]$.

Given a CSI (t_1, t_2), we perform two Bézier shoots from both ends of the interval to extract an SI from each end. This results in two possible cases: 1) the entire interval can be confirmed as an SI (Fig. 4a) or 2) two SIs (t_1, t') and (t'', t_2) are obtained, and depending on the collision status of the ellipsoids at $\tilde{t} = (t' + t'')/2$, the subdivided intervals from $[t', t'']$ are either labeled as CSIs or COIs (Figs. 4b and 4c) for further processing as in the following algorithm:

Algorithm 2 CSI handling

Input: A candidate separation interval (CSI) (t_1, t_2)

if interval width $t_2 - t_1$ is sufficiently small **then**
  report (t_1, t_2) as an OI (see Remark 4 below)
else
  $t' \leftarrow$ BézierShoot$(t_1 \to t_2)$
  $t'' \leftarrow$ BézierShoot$(t_1 \leftarrow t_2)$
  if $t' > t''$ **then**
    report (t_1, t_2) as an SI ▷▷▷ *Fig. 4(a)*
  else
    report (t_1, t') and (t'', t_2) as SIs
    $\tilde{t} \leftarrow (t' + t'')/2$
    if State$(\tilde{t}) = -1$ **then**
      $t^* \leftarrow$ ContactTime$(t', \tilde{t})$ ▷▷▷ *Fig. 4(b)*
      $t^{**} \leftarrow$ ContactTime$(\tilde{t}, t'')$
      report contact time instants t^* and t^{**}
      label $[t', t^*)$, $(t^{**}, t'']$ as CSIs and (t^*, t^{**}) as a COI
    else
      if State$(\tilde{t}) = 0$ **then**
        report contact time instant $\tilde{t}$
      label $[t', \tilde{t})$, $(\tilde{t}, t'']$ as CSIs ▷▷▷ *Fig. 4(c)*



(a) (b) (c)

Fig. 4. The handling of a CSI in the algorithm for solving a CCD.

Remark 4. In the case where the difference $t_2 - t_1$ is sufficiently small, as we cannot avoid some chance of having tiny loop(s) in the zero set of $F(u,t)$, therefore to be more conservative, we classify the interval (t_1, t_2) as an OI.

Processing COIs. For a COI (t_1, t_2), we aim to identify some OIs within a COI, so that the remaining subintervals can be further processed. Given a COI (t_1, t_2), if it contains any SI, then $F(u^*, t^*) > 0$ for some $t^* \in (t_1, t_2)$ and the zero set of $F(u,t) = 0$ contains some close loops in the strip $(u,t) \in (0,1] \times (t_1, t_2)$. Hence, a COI can be confirmed as an OI if it does not contain any loop, and this can be checked as follows:

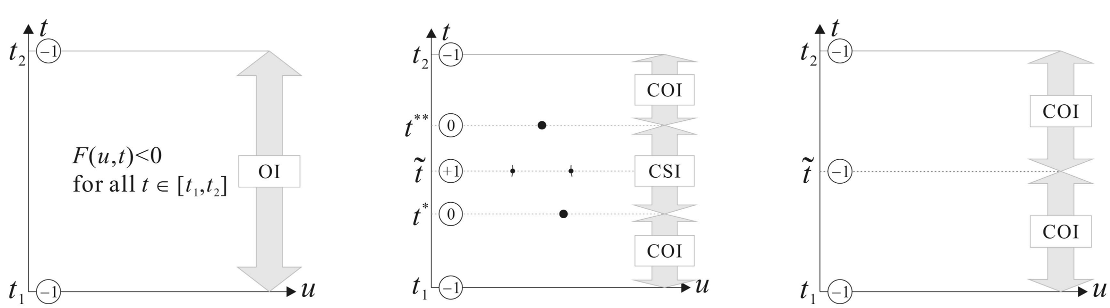
We first consider the coefficients of the Bernstein form of $F(u,t)$. Using the convex hull property of the Bernstein form [38], if all the coefficients are negative, the interval (t_1, t_2) is an OI since we must have $F(u,t) < 0$ in this interval (Fig. 5a). If the coefficients have different signs, we will check the existence of a loop in the zero set of $F(u,t) = 0$. The existence of a loop in (t_1, t_2) implies that the derivative $F_t(u,t)$ cannot be of the same sign for all $(u,t) \in (0,1] \times (t_1, t_2)$. For this, again using the convex hull property, we check whether the control coefficients of $F_t(u,t)$, expressed as a bivariate Bernstein function on $(0,1] \times (t_1, t_2)$, have the same sign. To make the test more effective, we further limit this check only to the subregion in which $F(u,t)$ can possibly be positive for $t \in (t_1, t_2)$; this subregion is the maximum extent of intersection of the convex hull of the control polyhedron of $F(u,t)$ and the ut-plane. If all these coefficients of the Bernstein form of $F_t(u,t)$ are of the same sign, then the zero set of $F(u,t)$ does not have a loop in the interval (t_1, t_2), implying that (t_1, t_2) is an OI; otherwise, if these coefficients have different signs, (t_1, t_2) is still a COI.

If a COI remains so after the above filtering using the sign checking on the Bernstein coefficients of $F(u,t)$ and $F_t(u,t)$, we further process this interval by checking the collision status of the two ellipsoids at $\tilde{t} = (t_1 + t_2)/2$. If the two ellipsoids are separate at $\tilde{t}$, the two MIs $(t_1, \tilde{t})$ and $(\tilde{t}, t_2)$ will be further processed (Fig. 5b). If the two ellipsoids are overlapping at $\tilde{t}$, we label the two subintervals $(t_1, \tilde{t})$ and $(\tilde{t}, t_2)$ as COIs (Fig. 5c) and process them using the above coefficient filtering operation recursively.

Algorithm 3 COI handling

Input: A candidate overlapping interval (COI) (t_1, t_2)

if interval width $t_2 - t_1$ is sufficiently small **then**
  report (t_1, t_2) as an OI
else
  if $F(u,t) = 0$ has no loop in (t_1, t_2) **then**
    report (t_1, t_2) as an OI ▷▷▷ *Fig. 5(a)*
  else
    $\tilde{t} \leftarrow (t_1 + t_2)/2$
    if State$(\tilde{t}) = +1$ **then**
      $t^* \leftarrow$ ContactTime$(t_1, \tilde{t})$ ▷▷▷ *Fig. 5(b)*
      $t^{**} \leftarrow$ ContactTime$(\tilde{t}, t_2)$
      report contact time instants t^* and t^{**}
      label (t_1, t^*), (t^{**}, t_2) as COIs and (t^*, t^{**}) as a CSI
    else
      if State$(\tilde{t}) = 0$ **then**
        report contact time instant $\tilde{t}$
      label $(t_1, \tilde{t})$, $(\tilde{t}, t_2)$ as COIs ▷▷▷ *Fig. 5(c)*



(a) (b) (c)

Fig. 5. The handling of a COI in the algorithm for solving a CCD.

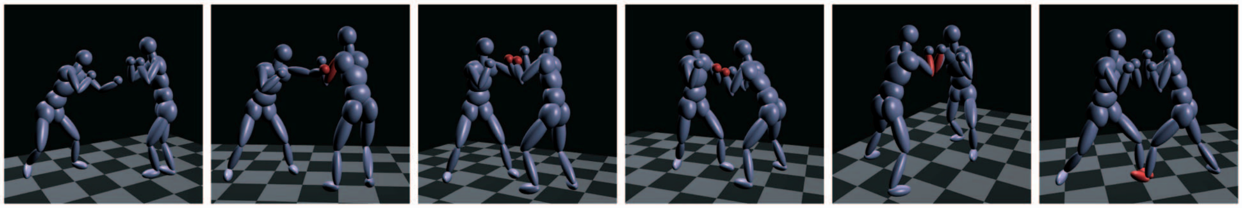


Fig. 6. Real-time CCD in a boxing game. Ellipsoids in collision are highlighted in red.

3.3 Finding the First Contact Time Only

Many real-time applications of collision detection require only the first contact time to be computed. Suppose that the two ellipsoids are separate at $t = 0$, i.e., $\mathrm{State}(0) = +1$. We then apply Bézier shoots recursively from $t = 0$, until we encounter the first contact time. We show that this process has quadratic convergence (Appendix C) and is efficient especially when the motion degree is low.

4 Experimental Results

We have tested our method in two applications to demonstrate its robustness and effectiveness. The first one features a human character animation in which two virtual human characters bounded by ellipsoids move in a sequence of frames. We determine the first contact instant of the characters in between every two consecutive frames. The motion of each ellipsoid is obtained by interpolating its orientations and positions at two consecutive frames. Both rigid and affine motion interpolations are tested and the performances in both cases are evaluated. In the second experiment, we perform collision detection between a robotic arm moving with prespecified rigid motion and a stationary obstacle. CCD is applied among the bounding ellipsoids of the links of the robotic arm and the obstacle, and all collision time intervals are reported.

4.1 Test in Human Character Animation

To test the efficiency of our method, we use two virtual boxers performing action in close proximity of each other, as shown in Fig. 6. The first contact instant in each time interval $[t_i, t_{i+1}]$ is to be determined, where the t_i are the time instants of each animation frame. Each character is bounded tightly by 20 ellipsoids, enclosing different body parts such as heads, limbs, and so forth. The motions of the two boxers are driven by motion capture data, together with a simple control mechanism. Between every two consecutive frames, the collision detection algorithm is applied to 400 pairs of ellipsoids, formed by picking one ellipsoid from each of the characters. We do not consider self-collision here, which can easily be dealt with by taking into account the pairwise CCD of nonadjacent ellipsoids of the same character.

Two fast and simple culling techniques are first used to quickly eliminate unlikely colliding pairs of ellipsoids. For each pair of moving ellipsoids, we first test whether their bounding spheres collide. The bounding spheres assume linear translation between the end positions of the ellipsoids. The moving spheres are guaranteed to bound the ellipsoids with the interpolating motions. To test whether the bounding spheres collide, we formulate a simple squared distance function $d(t) = |\mathbf{c}_1(t) - \mathbf{c}_2(t)|^2 - (r_1 + r_2)^2 < 0,\ t \in [0, 1]$, of two spheres, where $\mathbf{c}_1(t)$ and $\mathbf{c}_2(t)$ are the sphere centers and r_1 and r_2 are the sphere radii. Then, two moving spheres are collision-free in [0, 1] if and only if $d(0) > 0$ and $d(t)$ has no real roots in [0, 1]. The bounding spheres test is very efficient—it takes only 1.5 μs per test and can filter out a large number of trivially noncollision cases, i.e., when the ellipsoids are far apart.

If the bounding spheres collide, we then apply a separating plane method to further eliminate the remaining easy cases of noncolliding ellipsoids. We compute a plane that separates the two ellipsoids [20] at the beginning of the time frame and then test whether the two moving ellipsoids are continuously separated by the plane during the whole frame period. We assume that the separating plane is under the same motion as one of the ellipsoid, say $\mathcal{B}(t)$, so that it is always separate from $\mathcal{B}(t)$ in [0, 1]. The collision test is now between $\mathcal{A}(t)$ and the moving plane $\mathcal{P}(t)$, which are then transformed so that $\mathcal{A}(t)$ becomes the unit sphere at the origin and $\mathcal{P}(t)$ becomes $\mathcal{P}'(t)$; $\mathcal{A}(t)$ and $\mathcal{P}(t)$ collide if and only if the distance from the origin to $\mathcal{P}'(t)$ is less than 1, which can also be determined algebraically as in the bounding sphere test. The separating plane test involves also a static collision detection of the ellipsoids at $t = 0$ and hence can identify trivial collision cases where the ellipsoids overlap at $t = 0$.

A total of 1,000 frames are processed for the boxing sequence. A continuous rigid motion is used for interpolation between every two consecutive frames; the center positions of the ellipsoids are linearly interpolated and the orientations are interpolated by a linear quaternion curve, producing a rotation matrix of rational degree 2. As a result, 400,000 pairs of moving ellipsoids were tested, out of which 93.8 percent of the pairs were filtered out by the sphere test, and 34.1 percent and 62.6 percent of the remaining pairs were determined as colliding or collision-free, respectively, at $t = 0$ using separating planes as witnesses. For the remaining 780 pairs (0.195 percent), we applied the algorithm from Section 3.3 that computes the first contact point in CCD. Of these, 742 were collision-free and 38 were in collision. Since only the first contact time is needed, we also maintain an upper bound, $\bar{t}$, on the contact time, which is the minimum of all the first contact time that have been computed so far. Subsequent CCD is only determined within the interval $[0, \bar{t}]$. Including all the above procedures and the generation of interpolating motions $M_A(t)$ and $M_B(t)$ for 40 ellipsoids, the average time for collision detection for each frame took 1.33 ms, in which 400 pairs of moving ellipsoids were handled. A total of 40 motion matrices were generated in 195 μs. The formulation of the bivariate function $F(u, t)$ takes considerable computation.

TABLE 1
Average CPU Time Taken for CCD of Two Virtual Human Characters in a Boxing Animation

	$\tau_{\rm frm}$ (ms)	$\tau_{\rm mot}$ (ms)	$\tau_{\rm e}$ (μs)	$\tau_{\rm f}$ (μs)	$\tau_{\rm c}$ (μs)	$\tau_{\rm w}$ (μs)
Rigid motion, setup 1*	1.33	0.20				
over all 400K pairs			2.8	2.5	17.8	73.9
over 780 close pairs**			55.2	54.7	65.4	73.9
Rigid motion, setup 2*	1.90	0.20				
over all 400K pairs			4.2	4.0	13.8	100.1
Affine motion, setup 2*	1.19	0.13				
over all 400K pairs			2.6	2.5	6.6	39.4

$\tau_{\rm frm}$ represents the average time per frame; $\tau_{\rm mot}$ for constructing the interpolating motion; $\tau_{\rm e}$, $\tau_{\rm f}$, and $\tau_{\rm c}$ represent the average time for pairwise CCD of all ellipsoids, collision-free ellipsoids, and colliding ellipsoids, respectively; and $\tau_{\rm w}$ is the worst-case running time for one CCD among all 400,000 pairwise CCDs.

**Setup 1—with sphere and separating plane tests, CCD over $[0, \bar{t}]$. Setup 2—with sphere test only, CCD over [0, 1].*

***where both sphere and separating plane tests fail to declare separation.*

However, this is needed only when the ellipsoids are in close proximity, when both the sphere test and separating plane test fail to declare separation. The first row of Table 1 summarizes the average and the worst-case running time for all pairwise CCD tests. The performance for the close proximity cases is also presented.

Using a rigid motion of ration degree 2 as motion interpolant, the degree of the CCD equation $F(u,t)$ is 28 in t. In Appendix D, we describe an affine motion interpolation, which approximates the relative motion between two moving ellipsoids and results in a CCD equation of degree 6 in t. In order to compare properly the performance of our CCD method with the two different motions, the separating plane test that depends on the interpolating motion is not used and all CCD computations are carried out in the time interval [0, 1], i.e., the upper bound $\bar{t}$ of the first contact time is not maintained, since $\bar{t}$ varies with different motions. The performance of our CCD method with the two motion interpolations is shown in the second and third rows of Table 1. The average time per frame has a significant 37.6 percent speedup using the proposed affine motion interpolation, due to the more efficient motion construction and a CCD computation of a much lower degree. In our experiment, both motion interpolations gave the same collision result of whether a pair of ellipsoids collide or not. Not accounting those pairs with first contact at $t = 0$, the differences between the first contact time of the ellipsoids with affine motion interpolation and that with rigid motion interpolation have an average, standard deviation, and maximum of 0.008, 0.03, and 0.49, respectively. We notice that the differences in the order of the maximum value occurs only in extreme cases; neglecting the maximum value gives an average, standard deviation, and maximum of 0.006, 0.01, and 0.14, respectively. When using affine motion interpolation to achieve low degree polynomial computation and therefore a more efficient collision detection, significant deviation from the rigid motion may occur due to the affine approximation that varies the sizes of the ellipsoids.

4.2 Test in Robotic Arm Movements

In our second experiment, a CRS F3 robotic arm collides with an I-shaped obstacle. The robotic arm assumes a predefined rigid motion and is tightly bounded by 10 ellipsoids (0-9) and the obstacle by three ellipsoids (U, V, W) (Fig. 7). We perform 30 pairwise collision tests using our algorithm to find all the collision time intervals between the robotic arm and the obstacle. The motion of the robotic arm is designed in such a way that the three joints of the arm rotate with rational motions of degree 2, and hence, the fingers move with rational motions of degree 6. The total time for processing all 30 pairs of ellipsoids is 43.8 ms. Note that the time needed for collision detection in general depends on the motion degree as well as the complexity of the zero set of the CCD equation. The degree of $F(u,t)$ in t, the time taken for obtaining $F(u,t)$, and that for solving the CCD for each pair of ellipsoids are summarized in Table 2.

4.3 Two Further Examples

We present two more examples to test the accuracy of our method and its efficiency in the case of general affine motions.

Example 1. Consider the two moving ellipsoids $\mathcal{A}(t)$: $\frac{x^2}{4}+\frac{y^2}{16}+\frac{z^2}{4}=1$ and $\mathcal{B}(t): x^2+\frac{y^2}{9}+\frac{z^2}{16}=1$ under rigid motions with the following degree-2 rotations $(R_A(t), R_B(t))$ and degree-3 translations $(T_A(t), T_B(t))$:

$$R_A(t) = \frac{1}{E_A(t)}\begin{pmatrix} -(8t^2-8t+1) & -2(2t-1) & 2(2t-1) \\ 2(2t-1) & 1 & 2(2t-1)^2 \\ -2(2t-1) & 2(2t-1)^2 & 1 \end{pmatrix},$$

$$R_B(t) = \frac{1}{E_B(t)}\begin{pmatrix} \sqrt{2}(t-1)(3t-1) & 2t(2t-1) & \sqrt{2}(t-1)^2 \\ \sqrt{2}(2t-1) & -2t(t-1) & \sqrt{2}(2t-1)^2 \\ -\sqrt{2}t(3t-2) & 2(2t-1)(t-1) & \sqrt{2}t^2 \end{pmatrix},$$

where $E_A(t) = 8t^2-8t+3$, $E_B(t) = -2(3t^2-3t+1)$, and

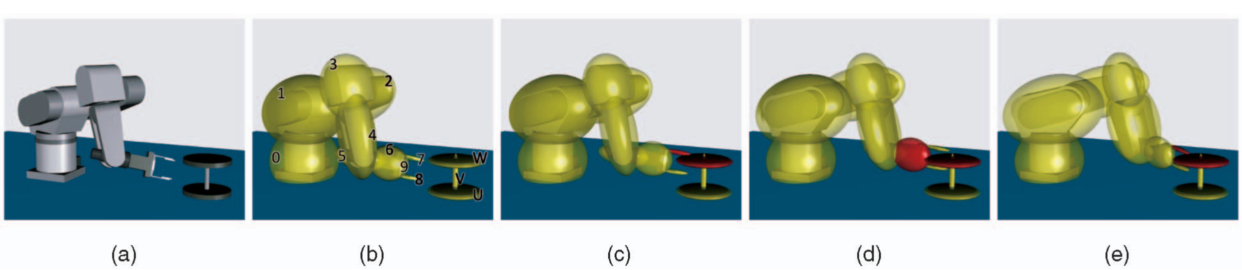


Fig. 7. (a) An F3 robotic arm and an I-shaped obstacle. (b) The bounding ellipsoids. (c), (d), and (e) The robotic arm in motion with $t = 0.104, 0.311$, and 0.778, respectively, and the colliding ellipsoids are shown in red.

TABLE 2
Average CPU Time Taken for CCD of a Robotic Arm and an Obstacle

	Degree		Time (ms)			Collision
	Motion	$F(u,t)$ in t	Obtain $F(u,t)$	Solve CCD	Total	intervals in $[0,1]$
U-0	0	0	0.045	0.077	0.122	-
U-1	2	32	0.272	0.125	0.397	-
U-2	2	32	0.272	0.130	0.402	-
U-3	2	32	0.272	0.126	0.398	-
U-4	4	64	0.813	0.838	1.651	-
U-5	4	64	0.812	0.256	1.068	-
U-6	6	96	1.612	1.642	3.253	-
U-7	6	96	1.676	0.338	2.014	-
U-8	6	96	1.680	0.356	2.036	-
U-9	6	96	1.684	1.498	3.181	-
V-0	0	0	0.045	0.076	0.122	-
V-1	2	32	0.272	0.126	0.397	-
V-2	2	32	0.272	0.131	0.402	-
V-3	2	32	0.272	0.125	0.397	-
V-4	4	64	0.813	0.860	1.673	-
V-5	4	64	0.812	0.881	1.693	-
V-6	6	96	1.612	1.591	3.203	-
V-7	6	96	1.676	0.052	1.729	-
V-8	6	96	1.680	1.505	3.186	-
V-9	6	96	1.683	1.545	3.228	-
W-0	0	0	0.045	0.077	0.122	-
W-1	2	32	0.272	0.127	0.399	-
W-2	2	32	0.272	0.130	0.402	-
W-3	2	32	0.272	0.126	0.398	-
W-4	4	64	0.812	0.859	1.671	-
W-5	4	64	0.813	0.827	1.641	-
W-6	6	96	1.612	0.075	1.687	[0.311,0.677]
W-7	6	96	1.677	0.085	1.762	[0.104,0.323], [0.778,0.943]
W-8	6	96	1.680	1.506	3.186	-
W-9	6	96	1.684	0.261	1.945	[0.451,0.538]

$$
\begin{aligned}
T_A(t) = (&-8t^3+24t^2-6t-2, -24t^3+24t^2+6t-6,\\
&-32t^3+48t^2-12t-2)^T,\\
T_B(t) = (&(72-24\sqrt{2})t^3+(-156+72\sqrt{2})t^2+(114-72\sqrt{2})t\\
&-27+24\sqrt{2}, 12t-6, (88-24\sqrt{2})t^3\\
&+(-168+72\sqrt{2})t^2+(114-72\sqrt{2})t-26+24\sqrt{2})^T.
\end{aligned}
$$

These motions are designed so that the ellipsoids have their first contact at $t_0 = 1/2$. The degree of $F(u,t)$ in t is 34, and our algorithm reports contact at $t = 0.5$ with an error in the order of 10^{-8}. The whole computation took 0.7 ms and extracted two OIs.

Example 2. In Fig. 8, two ellipsoids are in motions of degree 4 with rather large affine deformations. Here, the degree of $F(u,t)$ in t is 48 and it took 2.7 ms to compute all the four OIs using the algorithm presented in Section 4. Detection of the first contact time takes 0.6 ms.

5 Conclusions

We have presented an efficient and accurate algorithm for CCD between two moving ellipsoids under rational motions. Significant speedup was realized by developing an efficient scheme to quickly compute the critical points of the zero set of the bivariate CCD equation, which correspond to the contact time instants of two ellipsoids, and determine whether the ellipsoids are overlapping or separate within a time interval. Our experiments showed that real-time CCD of ellipsoids can be achieved for time-critical applications.

We believe that there are many other interesting properties of our algebraic condition, which should lead to more efficient geometric algorithms for dealing with ellipsoids and affine deformations. The robotic arm example also shows that, because of the composite motions of the joints, the degree of the CCD equation in t can easily raise well beyond 100, which cannot be dealt with reasonably using numerical methods. Therefore, the approximation of high degree motions or nonrational motions (which is not handled currently by our numerical scheme for solving the CCD equation) by low-degree rational motions is worth pursuing in this regard.

Appendix A

Coefficients of the Characteristic Polynomial

We present an efficient algorithm for computing the five coefficients of the characteristic polynomial $\mathrm{f}(\lambda)$ of degree 4. Let $M_A^T(M_B^{-1})^T B M_B^{-1} M_A = [b_{ij}]_{4\times4}$. Then, the characteristic polynomial is given in the following simple form:

$$
\mathrm{f}(\lambda) = \det\left(\lambda A - M_A^T\left(M_B^{-1}\right)^T B M_B^{-1} M_A\right)
= \begin{vmatrix}
\lambda/a^2 - b_{11} & -b_{12} & -b_{13} & -b_{14}\\
-b_{21} & \lambda/b^2 - b_{22} & -b_{23} & -b_{24}\\
-b_{31} & -b_{32} & \lambda/c^2 - b_{33} & -b_{34}\\
-b_{41} & -b_{42} & -b_{43} & -\lambda - b_{44}
\end{vmatrix}.
$$

By expanding this determinant, the five coefficients can be constructed as follows:

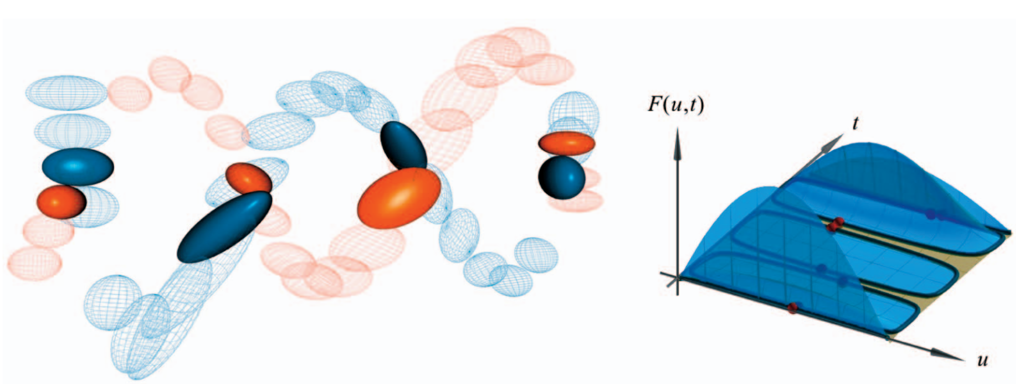


Fig. 8. Two moving ellipsoids under degree-4 dependent motion with affine deformations, the CCD equation $F(u,t) = 0$ is of degree 48 in t and four OIs are detected.

- The fourth-degree term (T4): $-\frac{1}{a^2b^2c^2}$.
- The third-degree term (T3): $\frac{b_{11}}{b^2c^2}+\frac{b_{22}}{a^2c^2}+\frac{b_{33}}{a^2b^2}-\frac{b_{44}}{a^2b^2c^2}$.
- The second-degree term (T2):

$$\frac{b_{33}b_{44}-b_{34}b_{43}}{a^2b^2}+\frac{b_{11}b_{44}-b_{14}b_{41}}{b^2c^2}+\frac{b_{22}b_{44}-b_{24}b_{42}}{a^2c^2}+\frac{b_{23}b_{32}-b_{22}b_{33}}{a^2}+\frac{b_{13}b_{31}-b_{11}b_{33}}{b^2}+\frac{b_{12}b_{21}-b_{11}b_{22}}{c^2}.$$

- The first-degree term (T1):

$$\begin{aligned}&\frac{-b_{22}b_{33}b_{44}+b_{22}b_{34}b_{43}+b_{33}b_{42}b_{24}}{a^2}\\&+\frac{b_{44}b_{32}b_{23}-b_{32}b_{24}b_{43}-b_{42}b_{23}b_{34}}{a^2}\\&+\frac{-b_{11}b_{33}b_{44}+b_{11}b_{34}b_{43}+b_{33}b_{14}b_{41}}{b^2}\\&+\frac{b_{44}b_{13}b_{31}-b_{31}b_{14}b_{43}-b_{41}b_{13}b_{34}}{b^2}\\&+\frac{-b_{11}b_{22}b_{44}+b_{11}b_{24}b_{42}+b_{22}b_{14}b_{41}}{c^2}\\&+\frac{b_{44}b_{12}b_{21}-b_{21}b_{14}b_{42}-b_{41}b_{12}b_{24}}{c^2}\\&+b_{11}b_{22}b_{33}-b_{11}b_{23}b_{32}-b_{22}b_{13}b_{31}-b_{33}b_{12}b_{21}\\&+b_{21}b_{13}b_{32}+b_{31}b_{12}b_{23}.\end{aligned}$$

- The constant term (T0):

$$\begin{aligned}&b_{11}b_{22}b_{33}b_{44}-b_{11}b_{22}b_{34}b_{43}-b_{11}b_{33}b_{24}b_{42}-b_{11}b_{44}b_{23}b_{32}\\&-b_{22}b_{33}b_{14}b_{41}-b_{22}b_{44}b_{13}b_{31}-b_{33}b_{44}b_{12}b_{21}+b_{11}b_{32}b_{24}b_{43}\\&+b_{11}b_{23}b_{34}b_{42}+b_{22}b_{13}b_{34}b_{41}+b_{22}b_{31}b_{14}b_{43}+b_{33}b_{12}b_{24}b_{41}\\&+b_{33}b_{21}b_{14}b_{42}+b_{44}b_{12}b_{23}b_{31}+b_{44}b_{21}b_{13}b_{32}+b_{12}b_{21}b_{34}b_{43}\\&+b_{13}b_{31}b_{24}b_{42}+b_{14}b_{41}b_{23}b_{32}-b_{21}b_{14}b_{43}b_{32}-b_{21}b_{13}b_{34}b_{42}\\&-b_{31}b_{12}b_{24}b_{43}-b_{31}b_{14}b_{42}b_{23}-b_{41}b_{12}b_{23}b_{34}-b_{41}b_{13}b_{32}b_{24}.\end{aligned}$$

If M_A and M_B are rigid transformations, the constant term is equal to $\det(-B)$ and the following function efficiently computes the coefficients $f(\lambda)$ using 29 additions/subtractions and 39 multiplications.

```
Generate-Characteristic-Polynomial
 /* Variable definition
 ea, eb, ec are the diagonal members of matrix A
 ab = ea * eb, ac = ea * ec, bc = eb * ec,
 abc = ea * eb * ec
 bij is a member of the matrix M_A^T (M_B^-1)^T B M_B^-1 M_A
 */
 begin
 b12s = b12 * b12; b13s = b13 * b13;
 b14s = b14 * b14; b23s = b23 * b23;
 b24s = b24 * b24; b34s = b34 * b34;
 b2233 = b22 * b33;

 termA = b11 * bc + b22 * ac + b33 * ab;
 termB = (b2233 - b23s) * ea + (b11 * b33 - b13s) * eb
   +(b11 * b22 - b12s) * ec;

 T4 = -abc;
 T3 = termA - b44 * abc;
 T2 = termA * b44 - termB - b34s * ab - b14s * bc
   -b24s * ac;
 tmp1 = termB * b44;
 tmp2 = b11 * (b2233 + eb * b34s + ec * b24s - b23s);
 tmp3 = b22 * (ea * b34s + ec * b14s - b13s);
 tmp4 = b33 * (ea * b24s + eb * b14s - b12s);
 tmp5 = b34 * (ea * b23 * b24 + eb * b13 * b14)
   +b12 * (ec * b14 * b24 - b13 * b23);
 tmp5+= tmp5; // multiply by 2

 T1 = -tmp1 + tmp2 + tmp3 + tmp4 - tmp5;
 T0 = constant; // constant value det[-B]
 end;
```

Appendix B

Three-Dimensional Rational Euclidean and Affine Motions

A rational Euclidean motion in $\mathbb{E}^3$ is given by

$$M(t)=\begin{pmatrix}R(t) & V(t)\\ \mathbf{0^T} & 1\end{pmatrix}, \tag{2}$$

where $V(t)\in\mathbb{E}^3$, $R(t)$ is a 3×3 orthogonal matrix, and t can be considered as a parameter of time. The motion is a composition of a rotation $R(t)$ acting upon a point in $\mathbb{E}^3$, followed by a translation $V(t)$. All rational Euclidean motions can be represented in (2) with

$$V(t)=\left(\frac{v_0}{v_3},\frac{v_1}{v_3},\frac{v_2}{v_3}\right)^T,\quad\text{and}$$

$$R(t)=\frac{1}{E}\begin{pmatrix}e_0^2+e_1^2-e_2^2-e_3^2 & 2e_1e_2-2e_0e_3 & 2e_0e_2+2e_1e_3\\ 2e_0e_3+2e_1e_2 & e_0^2-e_1^2+e_2^2-e_3^2 & 2e_2e_3-2e_0e_1\\ -2e_0e_2+2e_1e_3 & 2e_0e_1+2e_2e_3 & e_0^2-e_1^2-e_2^2+e_3^2\end{pmatrix},$$

where $E=e_0^2+e_1^2+e_2^2+e_3^2$, and $v_0,\ldots,v_3,e_0,\ldots,e_3$ are polynomials in t [39]. The Euler parameters e_0, e_1, e_2, and e_3 describe a rotation about a vector in $\mathbb{E}^3$ and are called the *normalized Euler parameters* when $E=1$. Readers are referred to [40] for a survey on rational motion design and [39], [41], [42] for interpolating a set of positions in $\mathbb{E}^3$ using piecewise B-spline motions.

When the entries of $V(t)$ and $R(t)$ are rational polynomials of maximal degree k, we called $M(t)$ a *rational motion of degree* k. An ellipsoid $\mathcal{A}(t)$ moving under a rational motion $M(t)$ is represented as $X^TA(t)X=0$, where $A(t)=(M^{-1}(t))^T AM^{-1}(t)$. Assume that the maximal degree of the entries in $R(t)$ and $V(t)$ are k_R and k_V, respectively. Then,

$$A(t)=\begin{pmatrix}P(t)_{<2k_R>} & U(t)_{<2k_R+k_V>}\\ U(t)^T_{<2k_R+k_V>} & s(t)_{<2(k_R+k_V)>}\end{pmatrix}$$

for some 3×3 matrix $P(t)$, three-vector $U(t)$, and scalar function $s(t)$. Here, the bracketed subscript represents the maximal degree of the entries of the associated entity.

For a rational affine motion in $\mathbb{E}^3$, the motion matrix $M(t)$ is formed by replacing $R(t)$ in (2) by a 3×3

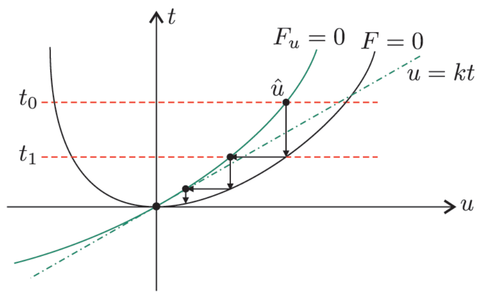


Fig. 9. Quadratic convergence of recursive Bézier shoot.

nonsingular matrix $L(t)$. The motion is then a composition of a linear transformation $L(t)$ acting upon a point in $\mathbb{E}^3$, followed by a translation $V(t)$. Assume that the maximal degree of the entries in $L(t)$ and $V(t)$ are k_L and k_V, respectively. Here,

$$A(t) = \begin{pmatrix} P(t)_{<6k_L>} & U(t)_{<6k_L+k_V>} \\ U(t)^T_{<6k_L+k_V>} & s(t)_{<6k_L+2k_V>} \end{pmatrix}$$

for some 3×3 matrix $P(t)$, three-vector $U(t)$, and scalar function $s(t)$.

Appendix C

Quadratic Convergence of Recursive Bézier Shoot

We now show that recursive Bézier shoot in search of a contact time, i.e., a regular solution (u^*, t^*) of $F(u,t) = F_u(u,t) = 0$, has quadratic convergence. Without loss of generality, we may assume that (u^*, t^*) is located at the origin (0, 0), with $F(u,t) = 0$ and $F_u(u,t) = 0$ as shown in Fig. 9. Then, by the regularity assumption and Implicit Function Theorem, the solution of $F(u,t) = 0$ can be represented locally at (0, 0) by Taylor expansion $t = \alpha u^2 + o(u^2)$, and the solution of $F_u(u,t) = 0$ by $u = kt + o(t)$. Now, consider a Bézier shoot from t_0. The solution of $F_u(u,t_0) = 0$ is $\hat{u} = kt_0 + o(t_0)$. So, the first root of $F(\hat{u}, t) = 0$ is

$$t_1 = \alpha\hat{u}^2 + o(\hat{u}^2) = \alpha[kt_0 + o(t_0)]^2 + o\big(t_0^2\big) = \alpha k^2 t_0^2 + o\big(t_0^2\big).$$

It follows that $t_1/t_0^2 = \alpha k^2 + o(1)$, i.e., recursive Bézier shoot has quadratic convergence. However, if (u^*, t^*) is a singular solution representing tangential contact of the two ellipsoids, then the convergence is in general linear.

Appendix D

An Affine Motion Interpolant

Assume that an ellipsoid $\mathcal{A}(t)$ is under a motion $M_A(t)$, and another ellipsoid $\mathcal{B}(t)$ is similarly under a motion $M_B(t)$, for $t \in [t_0, t_1]$. The two ellipsoids $\mathcal{A}(t)$ and $\mathcal{B}(t)$ collide if and only if the standard ellipsoid $\mathcal{A}$ collides with the moving ellipsoid $\hat{\mathcal{B}}(t)$ under a relative motion $\hat{M}_B(t) = M_A^{-1}(t) M_B(t)$, for $t \in [t_0, t_1]$. The ellipsoid $\hat{\mathcal{B}}(t_0)$ with its center $\mathbf{c}_0$ is represented as

$$\hat{\mathcal{B}}(t_0) : (\mathbf{x} - \mathbf{c}_0)^T B_0 (\mathbf{x} - \mathbf{c}_0) = 1,$$

where B_0 is a symmetric positive definite matrix, and $\mathbf{x} = (x,y,z)^T$ is a point on $\hat{\mathcal{B}}(t_0)$. Similarly, the ellipsoid $\hat{\mathcal{B}}(t_1)$ with its center $\mathbf{c}_1$ is represented as

$$\hat{\mathcal{B}}(t_1) : (\mathbf{x} - \mathbf{c}_1)^T B_1 (\mathbf{x} - \mathbf{c}_1) = 1,$$

where B_1 is a symmetric positive definite matrix. Now, consider an interpolation $\hat{\mathcal{B}}(t)$ between the two ellipsoids

$$\hat{\mathcal{B}}(t) : (\mathbf{x} - \mathbf{c}(t))^T B(t) (\mathbf{x} - \mathbf{c}(t)) = 1, \tag{3}$$

where

$$\mathbf{c}(t) = \frac{t_1 - t}{t_1 - t_0}\mathbf{c}_0 + \frac{t - t_0}{t_1 - t_0}\mathbf{c}_1,$$
$$B(t) = \frac{t_1 - t}{t_1 - t_0} B_0 + \frac{t - t_0}{t_1 - t_0} B_1.$$

Note that $B(t)$ is symmetric positive definite if B_0 and B_1 are both symmetric and positive definite. Thus, $\hat{\mathcal{B}}(t)$ represents a moving ellipsoid under an affine motion, for $t_0 \le t \le t_1$. By expanding the above representation of the ellipsoid $\hat{\mathcal{B}}(t)$, we obtain

$$\hat{\mathcal{B}}(t) : \mathbf{x}^T B(t)\mathbf{x} - 2\mathbf{x}^T B(t)\mathbf{c}(t) + \mathbf{c}(t)^T B(t)\mathbf{c}(t) = 1.$$

Using the homogeneous coordinates $X = (x,y,z,w)^T$, we represent the ellipsoid $\hat{\mathcal{B}}(t)$ as $X^T \hat{B}(t) X = 0$, where

$$\hat{B}(t) = \begin{pmatrix} B(t) & -B(t)\mathbf{c}(t) \\ -\mathbf{c}(t)^T B(t)^T & \mathbf{c}(t)^T B(t)\mathbf{c}(t) - 1 \end{pmatrix}. \tag{4}$$

We can then easily show that the characteristic polynomial $\mathrm{f}(\lambda, t) = \det(\lambda A - \hat{B}(t))$ has degree 6 in t.

Acknowledgments

The authors would like to thank anonymous reviewers for their invaluable comments. Prof. Jehee Lee at Seoul National University provided the motion capture data for the boxing animation. The work of Wenping Wang was partially supported by the National Key Basic Research Project of China (2004CB318000), the Research Grant Council of Hong Kong (HKU7178/06E), and the Innovative and Technology Fund of Hong Kong (ITS/090/06). This work was also supported in part by KICOS through the Korean-Israeli Binational Research Grant (K20717000006) provided by MEST in 2007.

References

[1] D.H. Eberly, *3D Game Engine Design.* Academic Press, 2001.
[2] D.H. Eberly, *Game Physics.* Morgan Kaufmann, 2003.
[3] P.M. Hubbard, "Approximating Polyhedra with Spheres for Time-Critical Collision Detection," *ACM Trans. Graphics,* vol. 15, no. 3, pp. 179-210, 1996.
[4] M. Held, J. Klosowski, and J. Mitchell, "Evaluation of Collision Detection Methods for Virtual Reality Fly-Throughs," *Proc. Seventh Canadian Conf. Computational Geometry (CCCG '95),* pp. 205-210, 1995.
[5] G. van den Bergen, "Efficient Collision Detection of Complex Deformable Models Using AABB Trees," *J. Graphics Tools,* vol. 2, no. 4, pp. 1-13, 1997.
[6] S. Gottschalk, M.C. Lin, and D. Manocha, "OBBTree: A Hierarchical Structure for Rapid Interference Detection," *Proc. ACM SIGGRAPH '96,* pp. 171-180, 1996.

[7] J. Klosowski, M. Held, J. Mitchell, H. Sowizral, and K. Zikan, "Efficient Collision Detection Using Bounding Volume Hierarchies of k-Dops," *IEEE Trans. Visualization and Computer Graphics,* vol. 4, no. 1, pp. 21-36, Jan.-Mar. 1998.

[8] M. Ju, J. Liu, S. Shiang, Y. Chien, K. Hwang, and W. Lee, "A Novel Collision Detection Method Based on Enclosed Ellipsoid," *Proc. IEEE Conf. Robotics and Automation (ICRA '01),* pp. 21-26, 2001.

[9] E. Rimon and S. Boyd, "Obstacle Collision Detection Using Best Ellipsoid Fit," *J. Intelligent and Robotic Systems,* vol. 18, pp. 105-126, 1997.

[10] S. Shiang, J. Liu, and Y. Chien, "Estimate of Minimum Distance between Convex Polyhedra Based on Enclosed Ellipsoids," *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS '00),* pp. 739-744, 2000.

[11] C. Wu, "On the Representation and Collision Detection of Robots," *J. Intelligent and Robotic Systems,* vol. 16, pp. 151-168, 1996.

[12] S. Bischoff and L. Kobbelt, "Ellipsoid Decomposition of 3D-Models," *Proc. First Int'l Symp. 3D Data Processing, Visualization and Transmission (3DPVT '02),* pp. 480-488, 2002.

[13] D.-E. Hyun, S.-H. Yoon, M.-S. Kim, and B. Jüttler, "Modeling and Deformation of Arms and Legs Based on Ellipsoidal Sweeping," *Proc. 11th Pacific Conf. Computer Graphics and Applications (Pacific Graphics '03),* pp. 204-212, 2003.

[14] K.-A. Sohn, B. Jüttler, M.-S. Kim, and W. Wang, "Computing Distances between Surfaces Using Line Geometry," *Proc. 10th Pacific Conf. Computer Graphics and Applications (Pacific Graphics '02),* pp. 236-245, 2002.

[15] C. Lennerz and E. Schömer, "Efficient Distance Computation for Quadratic Curves and Surfaces," *Proc. Geometric Modeling and Processing, Theory and Applications (GMP '02),* pp. 60-69, 2002.

[16] A. Donev, S. Torquato, and F.H. Stillinger, "Neighbor List Collision-Driven Molecular Dynamics Simulation for Nonspherical Hard Particles II: Applications to Ellipses and Ellipsoids," *J. Computational Physics,* vol. 202, pp. 765-793, 2005.

[17] X. Lin and T. Ng, "Contact Detection Algorithms for Three-Dimensional Ellipsoids in Discrete Element Modeling," *Int'l J. Numerical and Analytical Methods in Geomechanics,* vol. 19, pp. 653-659, 1995.

[18] J. Perram, J. Rasmussen, E. Prastaard, and J. Lebowitz, "Ellipsoids Contact Potential: Theory and Relation to Overlap Potentials," *Physics Rev. E,* vol. 54, no. 6, pp. 6565-6572, 1996.

[19] V. Coppola and J. Woodburn, "Determination of Close Approaches Based on Ellipsoidal Threat Volumes," *Proc. AAS/AIAA Space Flight Mechanics Meeting,* pp. 1013-1023, Feb. 1999.

[20] W. Wang, Y.-K. Choi, B. Chan, M.-S. Kim, and J. Wang, "Efficient Collision Detection for Moving Ellipsoids Using Separating Planes," *Computing,* vol. 72, nos. 1-2, pp. 235-246, 2004.

[21] S. Redon, A. Kheddar, and S. Coquillart, "Fast Continuous Collision Detection between Rigid Bodies," *Computer Graphics Forum,* vol. 21, no. 3, pp. 279-288, 2002.

[22] S. Redon, Y.J. Kim, M.C. Lin, D. Manocha, and J. Templeman, "Interactive and Continuous Collision Detection for Avatars in Virtual Environments," *Proc. IEEE Virtual Reality Conf. (VR '04),* pp. 117-124, 2004.

[23] S. Redon, M.C. Lin, D. Manocha, and Y.J. Kim, "Fast Continuous Collision Detection for Articulated Models," *J. Computing and Information Science in Eng.,* vol. 5, no. 2, pp. 126-137, 2005.

[24] N.K. Govindaraju, I. Kabul, M.C. Lin, and D. Manocha, "Fast Continuous Collision Detection among Deformable Models Using Graphics Processors," *Computers and Graphics,* vol. 31, no. 1, pp. 5-14, 2007.

[25] Y.J. Kim, S. Redon, M.C. Lin, D. Manocha, and J. Templeman, "Interactive Continuous Collision Detection Using Swept Volume for Avatars," *Presence,* vol. 16, no. 2, pp. 206-223, 2007.

[26] X. Zhang, S. Redon, M. Lee, and Y.J. Kim, "Continuous Collision Detection for Articulated Models Using Taylor Models and Temporal Culling," *ACM Trans. Graphics,* vol. 26, no. 3, p. 15, 2007.

[27] K.-J. Kim, M.-S. Kim, and K. Oh, "Torus/Sphere Intersection Based on a Configuration Space Approach," *Graphical Models and Image Processing,* vol. 60, no. 1, pp. 77-92, 1998.

[28] J.R. Miller and R.N. Goldman, "Geometric Algorithms for Detecting and Calculating All Conic Sections in the Intersection of Any Two Natural Quadric Surfaces," *CVGIP: Graphical Model and Image Processing,* vol. 57, no. 1, pp. 55-66, 1995.

[29] C. Tu, W. Wang, and J. Wang, "Classifying the Nonsingular Intersection Curve of Two Quadric Surfaces," *Proc. Geometric Modeling and Processing, Theory and Applications (GMP '02),* pp. 23-32, 2002.

[30] W. Wang, R.N. Goldman, and C. Tu, "Enhancing Levin's Method for Computing Quadric-Surface Intersections," *Computer Aided Geometric Design,* vol. 20, no. 7, pp. 401-422, 2003.

[31] W. Wang, B. Joe, and R.N. Goldman, "Computing Quadric Surface Intersections Based on an Analysis of Plane Cubic Curves," *Graphical Models,* vol. 64, no. 6, pp. 335-367, 2002.

[32] Y.-K. Choi, W. Wang, Y. Liu, and M.-S. Kim, "Continuous Collision Detection for Two Moving Elliptic Disks," *IEEE Trans. Robotics,* vol. 22, no. 2, pp. 213-224, 2006.

[33] W. Wang, J. Wang, and M.-S. Kim, "An Algebraic Condition for the Separation of Two Ellipsoids," *Computer Aided Geometric Design,* vol. 18, no. 6, pp. 531-539, 2001.

[34] Y.-K. Choi, W. Wang, and M.-S. Kim, "Exact Collision Detection of Two Moving Ellipsoids under Rational Motions," *Proc. IEEE Conf. Robotics and Automation (ICRA '03),* pp. 349-354, 2003.

[35] D. Blythe, "The Direct3D 10 System," *ACM Trans. Graphics,* vol. 25, no. 3, pp. 724-734, 2006.

[36] M. Lee, "Next-Generation Graphics Programming on Xbox 360," *Proc. Game Developers Conf. (GDC '06),* http://download.microsoft.com/download/d/3/0/d30d58cd-87a2-41d5-bb53-baf560aa2373/Next_Generation_Graphics_Programming_on_Xbox_360.ppt, Mar. 2006.

[37] T. Nishita, T.W. Sederberg, and M. Kakimoto, "Ray Tracing Trimmed Rational Surface Patches," *Proc. ACM SIGGRAPH '90,* pp. 337-345, 1990.

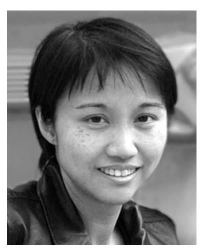
[38] G. Farin, *Curves and Surfaces in Computer Aided Geometric Design,* third ed. Academic Press, 1993.

[39] B. Jüttler and M.G. Wagner, "Kinematics and Animation," *Handbook of Computer Aided Geometric Design,* M.K.G. Farin, J. Hoschek, eds., Elsevier, pp. 723-748, 2002.

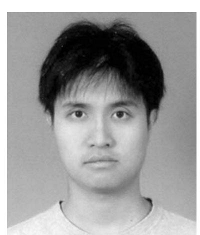
[40] O. Röschel, "Rational Motion Design—A Survey," *Computer-Aided Design,* vol. 30, no. 3, pp. 169-178, 1998.

[41] T. Horsch and B. Jüttler, "Cartesian Spline Interpolation for Industrial Robots," *Computer-Aided Design,* vol. 30, no. 3, pp. 217-224, 1998.

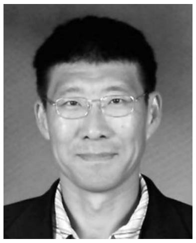
[42] B. Jüttler and M.G. Wagner, "Computer-Aided Design with Spatial Rational B-Spline Motions," *ASME J. Mechanical Design,* vol. 118, no. 2, pp. 193-201, 1996.



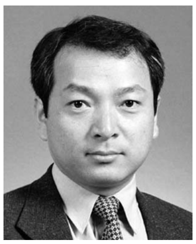
Yi-King Choi received the BSc and MPhil degrees in computer science from the University of Hong Kong, Hong Kong, in 1996 and 2000, respectively. She is a PhD student in the Computer Graphics Group headed by Professor Wenping Wang at the University of Hong Kong. Her research interests include computer graphics, geometric computing, and medical visualization.



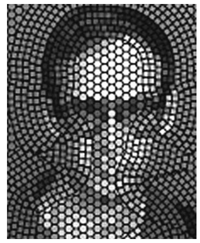
Jung-Woo Chang received the BS degree in computer science and engineering in 2002 from Seoul National University, Seoul, where he is currently working toward the PhD degree. His research interests include computer graphics and geometric modeling.



Wenping Wang received the BSc and MEng degrees in computer science from Shandong University in 1983 and 1986, respectively, and the PhD degree in computer science from the University of Alberta in 1992. He is an associate professor of computer science at the University of Hong Kong (HKU). His research covers computer graphics, geometric computing, and visualization. He has published more than 100 papers in these fields. He is an associate editor of the Springer journal *Computer Aided Geometric Design* and has been a program chair of several international conferences, including Geometric Modeling and Processing (GMP 2000), Pacific Graphics (PG 2000 and PG 2003), ACM Symposium on Virtual Reality Software and Technology (VRST 2001), and ACM Symposium on Physical and Solid Modeling (SPM 2006). He received the Teaching Excellence Award of the Department of Computer Science, HKU in 2006 and the HKU Research Output Prize in 2007.



Myung-Soo Kim received the BS and MS degrees from Seoul National University, Seoul, in 1980 and 1982, respectively, and the MS degree in applied mathematics and the MS and PhD degrees in computer science from Purdue University, West Lafayette, Indiana, in 1985, 1987, and 1988, respectively. He is currently a professor of the School of Computer Science and Engineering, Seoul National University. He is also the CIO and the Director of University Computer Center, Seoul National University. His research interests are in computer graphics and geometric modeling. From 1988 to 1998, he was with the Department of Computer Science, Pohang University of Science and Technology (POSTECH), Pohang, Korea. He serves on the editorial boards of the journals *Computer-Aided Design*, *Computer Aided Geometric Design*, and the *International Journal of Shape Modeling*. He also edited several special issues of journals including *Computer-Aided Design*, *Graphical Models*, the *Journal of Visualization and Computer Animation*, *The Visual Computer*, and the *International Journal of Shape Modeling*. Recently, together with G. Farin and J. Hoschek, he edited the *Handbook of Computer Aided Geometric Design* (Amsterdam: North-Holland, 2002).



Gershon Elber received the BSc degree in computer engineering and the MSc degree in computer science from the Technion, Haifa, Israel, in 1986 and 1987, respectively, and the PhD degree in computer science from the University of Utah, Salt Lake City, in 1992. He is a professor in the Computer Science Department, Technion. His research interests span computer-aided geometric designs and computer graphics. He has published more than 100 papers in international conference proceedings and journals and is one of the authors of a book entitled "Geometric Modeling with Splines—An Introduction." He serves on the editorial board of the *Computer Aided Design*, *Computer Graphics Forum*, *The Visual Computer*, and the *International Journal of Computational Geometry and Applications* and has served in many conference program committees including Solid Modeling, Shape Modeling, Geometric Modeling and Processing, Pacific Graphics, Computer Graphics International, and ACM SIGGRAPH. He was one of the paper chairs of Solid Modeling 2003 and Solid Modeling 2004. He is a member of the ACM and the IEEE.