Precise Generalized Contact Point and Normal Determination for Rigid Body Simulation

Dylan A. Shell
Computer Science Department
University of Southern California
shell@usc.edu

Evan Drumwright
Computer Science Department
University of Memphis
edrmwrgh@memphis.edu

ABSTRACT

Modeling contact for rigid body simulation requires accurate determination of time of contact, contact points, and contact normals. Existing collision detection methods for rigid body simulation can be grouped into one of three categories: convexity-based discrete methods, *a posteriori* discrete methods, and continuous methods. Our proposed method combines the advantages of all three types: operating on arbitrary geometric representations, running in asymptotic linear time in the number of polyhedral features, having a parameterizable precision (a variation is guaranteed to miss no collisions), and avoiding simplex/simplex tests that are difficult to implement robustly. The algorithm is demonstrated on a pathological example involving both polyhedra and polygon soups.

1. INTRODUCTION

Rigid body simulation algorithms model contact by using a set of contact points and normals. The difficulty and importance of determining these contact data accurately is often overlooked. Approximations can lead to interpenetration, numerical and dynamic instability, and undesirable energy drain from the simulated system [3]. The latter effect may also result from polyhedral approximation of curved surfaces. This paper introduces a method for computing time of contact (TOC) and contact points and normals that is general with respect to the geometric representation: geometries may be non-convex, polyhedra, polygon soups, constructive solid geometry, implicit or parametric surfaces, etc.

Collision detection methods can be broadly grouped, with respect to rigid body simulation, into three categories:

Discrete convexity-based methods determine whether pairs of bodies intersect at a given instance of time, cf. [5, 1, 7]. Time of contact is found by using numerical root-finding methods. Non-convex geometries must be factored into convex pieces, potentially resulting in N^2 pieces [4].

Discrete *a posteriori* methods approximate contact information efficiently by using heuristics based on the interpenetrated configuration, e.g. [8, 6, 11].

Continuous collision detection methods test whether bodies contact over a given time interval, rather than at an instant of time [2].

SAC'09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.

Although invariably slower than discrete detection algorithms, competitive algorithms require small step-sizes [9], or convex polyhedra [12].

2. METHOD

We determine contact data and times over a time interval $[t_0, t_f]$ by computing the TOC of points sampled from the two bodies; we assume that the bodies are disjoint at t_0 and are intersecting at t_f. The times of impact of the sampled points are computed rapidly using the dynamic states of the bodies at t_0 and t_f; it is guaranteed that the first time of contact (and associated contact data) is found between a pair of bodies up to a given tolerance. Efficient TOC calculation is the primary contribution of the introduced approach.

Algorithm 1
DETERMINE-TOC($\mathcal{S}, t_0, t_f, \mathbf{R}_\mathcal{B}(t_0), \boldsymbol{v}_\mathcal{B}, \boldsymbol{\omega}_\mathcal{B}, \boldsymbol{p}(t_0), \boldsymbol{p}(t_f), \dot{\boldsymbol{p}}(t)$): given a stationary solid $\mathcal{S}$, a time interval $[t_0, t_f]$, and a point $\boldsymbol{p}(t)$ on a rigid body $\mathcal{B}$ with orientation $\mathbf{R}_\mathcal{B}(t_0)$ at t_0, linear and angular velocities $\boldsymbol{v}_\mathcal{B}$ and $\boldsymbol{\omega}_\mathcal{B}$ at time t_0, determines first time-of-impact t_χ of $\boldsymbol{p}(t)$ with $\mathcal{S}$, along with $\boldsymbol{p}(t_\chi)$ and the normal of $\mathcal{S}$ at $\boldsymbol{p}(t_\chi)$.

$t_\chi \leftarrow \infty$; push(Q, $\{t_0, t_f, \boldsymbol{p}(t_0), \boldsymbol{p}(t_f), \mathbf{R}_\mathcal{B}(t_0)\}$)
while Q not empty **do**
  $\{t_a, t_b, \boldsymbol{p}(t_a), \boldsymbol{p}(t_b), \mathbf{R}_\mathcal{B}(t_a)\} \leftarrow$ pop(Q)
  $n_\alpha \leftarrow \frac{\boldsymbol{v}_\mathcal{B}}{||\boldsymbol{v}_\mathcal{B}||}$
  form orthonormal basis $\{n_\alpha, n_\beta, n_\gamma\}$
  determine ν_a^b according to Equation 2
  $e_\alpha \leftarrow \|\boldsymbol{v}_\mathcal{B}\| + 2\|\boldsymbol{u}\|$
  $e_{\{\beta,\gamma\}}^{(\{1,2\})} \leftarrow \min(\nu_a^b/\sqrt{2}, \|\boldsymbol{u}\| + (-1)^{\{1,2\}} u^\mathsf{T} \mathbf{R}_\mathcal{B}(t_0)^\mathsf{T} \boldsymbol{n}_{\{\beta,\gamma\}})$
  $\mathbf{c} \leftarrow \frac{1}{2}[\boldsymbol{v}_\mathcal{B}(t_b - t_a) + \boldsymbol{n}_\beta(e_\beta^{(1)} - e_\beta^{(2)}) + \boldsymbol{n}_\gamma(e_\gamma^{(1)} - e_\gamma^{(2)})]$
  $\mathcal{O} \leftarrow OBB(\mathbf{c}, \{n_\alpha, e_\alpha\}, \{n_\beta, e_\beta\}, \{n_\gamma, e_\gamma\})$
  if $\mathcal{O}$ intersects $\mathcal{S}$ **then**
    if $\nu_a^b > \epsilon$ **then**
      $t_i \leftarrow \frac{1}{2}(t_a + t_b)$
      $\mathbf{R}_\mathcal{B}(t_i) \leftarrow \mathbf{R}_\mathcal{B}(t_a) + \text{integrate } \dot{\mathbf{R}}_\mathcal{B}(\mathrm{t}) \text{ over } [\mathrm{t_a}, \mathrm{t_i}]$
      $\dot{\boldsymbol{p}}(t_a) \leftarrow \boldsymbol{x}_\mathcal{B} + \tilde{\boldsymbol{\omega}}_\mathcal{B} \mathbf{R}_\mathcal{B}(t_a) \boldsymbol{u}$
      $\boldsymbol{p}(t_i) \leftarrow \boldsymbol{p}(t_a) + \text{integrate } \dot{\mathrm{p}}(\mathrm{t}) \text{ over } [\mathrm{t_a}, \mathrm{t_i}]$
      push(Q, $\{t_a, t_i, \boldsymbol{p}(t_a), \boldsymbol{p}(t_i), \mathbf{R}_\mathcal{B}(t_0)\}$)
      push(Q, $\{t_i, t_b, \boldsymbol{p}(t_i), \boldsymbol{p}(t_b), \mathbf{R}_\mathcal{B}(t_i)\}$)
    else
      for all intersections p_x between $\mathcal{O}$ and $\mathcal{S}$ **do**
        $t_\alpha \leftarrow t_a + \frac{\|p_x - p_b\|}{\|p_a - p_b\|}$
        if $t_\alpha < t_\chi$ **then**
          $t_\chi \leftarrow t_\alpha$
          $\boldsymbol{q} \leftarrow \boldsymbol{p}(t_\chi)$
          $\hat{\boldsymbol{\eta}} \leftarrow \hat{\boldsymbol{n}}(\mathcal{S}^\mathbf{q})$ {Get normal to $\mathcal{S}$ at $\boldsymbol{p}(t_\chi)$}
return $\{t_\chi, \boldsymbol{q}, \hat{\boldsymbol{\eta}}\}$

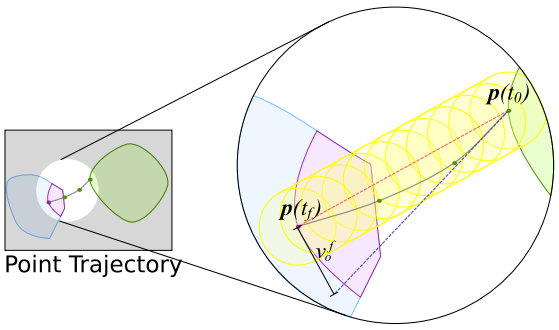

Figure 1: Depiction of the bounds (shaded in yellow) of a trajectory of a point on a rigid body; the true trajectory is drawn in black, and the linearization of the trajectory by connecting end-points is drawn in red, the linearization by expansion around $\boldsymbol{p}(t_0)$ is drawn in blue.

In general, points on a rigid body follow a curved path: $\boldsymbol{p}(t) = \boldsymbol{x}_{\mathcal{B}}(t) + \mathbf{R}_{\mathcal{B}}(t)\boldsymbol{u}$ where $\boldsymbol{x}_{\mathcal{B}}(t)$ is the position of the body's center-of-mass, $\mathbf{R}_{\mathcal{B}}(t)$ is the rotation matrix of $\mathcal{B}$, and $\boldsymbol{u}$ is the vector from the center-of-mass to point $\boldsymbol{p}(t)$ in body $\mathcal{B}$'s frame. The time derivative of this equation is given below:

$$\dot{\boldsymbol{p}}(t) = \dot{\boldsymbol{x}}_{\mathcal{B}}(t) + \tilde{\boldsymbol{\omega}}_{\mathcal{B}}(t)\mathbf{R}_{\mathcal{B}}(t)\boldsymbol{u} \tag{1}$$

where $\dot{\boldsymbol{x}}_{\mathcal{B}}(t)$ is the velocity of $\mathcal{B}$'s center-of-mass and $\tilde{\boldsymbol{\omega}}_{\mathcal{B}}(t)$ is the antisymmetric matrix formed using the angular velocity of the body in the world frame. Both the direction and magnitude of $\dot{\boldsymbol{p}}(t)$ change over $[t_0, t_f]$, in general, even though the linear and angular velocities remain constant over this interval[1].

Thus, we define ν_0^f, given $\varphi \triangleq \|\boldsymbol{\omega}_{\mathcal{B}}(t_0)\|\,(t_f - t_0)$:

$$\nu_0^f = \begin{cases} \sqrt{2}(1-\cos(\varphi))\,\|\dot{\boldsymbol{\omega}}_{\mathcal{B}}(t_0) \times \boldsymbol{u}\| & \text{if } \varphi < \pi \\ 2\sqrt{2}\,\|\dot{\boldsymbol{\omega}}_{\mathcal{B}}(t_0) \times \boldsymbol{u}\| & \text{if } \varphi \geq \pi. \end{cases} \tag{2}$$

This equation permits us to evaluate the quality of the approximation of $\boldsymbol{p}(t)$ over $[t_0, t_f]$ by a line segment; if the approximation is good, then ν_0^f will be small. If the approximation is not sufficiently close (as defined by a parameter, ϵ), then the interval is bisected until it is within the desired tolerance. Equation 2 also enables us to bound the values that $\boldsymbol{p}(t)$ can take over $[t_0, t_f]$. As seen in Figure 1, $\boldsymbol{p}(t)$ can deviate from the line segment $\overline{\boldsymbol{p}(t_0)\boldsymbol{p}(t_f)}$ by at most ν_0^f in any direction over $[t_0, t_f]$.

The preceding intersection of a point trajectory with a stationary body is directly extended to consider intersection of trajectory with a moving body through an appropriate change in coordinate system.

3. PERFORMANCE & COMPLEXITY

The number of iterations necessary to reduce the curved trajectory into line-segments is a function of $\dot{\boldsymbol{x}}_{\mathcal{B}}, \dot{\boldsymbol{x}}_{\mathcal{S}}, \dot{\mathbf{R}}_{\mathcal{B}}, \dot{\mathbf{R}}_{\mathcal{S}}$ and the precision parameter ϵ. In the limit as $\epsilon \to 0$ the algorithm resolves the query exactly. With a finite ϵ it is *not* the case that the method finds the earliest time at which the solid passes within ϵ of the trajectory, but instead the method finds the time at which intersects the volume of a piecewise linear curve of boxes with ϵ sides, and is guaranteed to contain the original trajectory.

Given two non-convex polyhedra and Q with N and M features, respectively, each with a zero level-set constructed offline, intersection of vertices of each polyhedron inside the other exhibits time complexity $O(kNM + kMN)$ where k is the number of iterations used to resolve the curved trajectory into linear components.

[1]This assertion is untrue for the underlying continuous system but is true when numerically integrating the rigid body equations of motion.

Variation sans bounding-boxes

If intersection queries between the solid and an oriented bounding box cannot be computed easily, then an alternative method using a piecewise linear approximation to $\boldsymbol{p}(t)$ can be employed. This method requires that line segments between points $\boldsymbol{p}(t_a)$ and $\boldsymbol{p}(t_b)$ be bisected recursively until $\nu_i^j < \epsilon$ for any consecutive i, j. Each line segment is then tested for intersection with the surface. This alternate approach is equally correct, but generally requires many more intersection queries.

This variation has the advantage that the first TOC t_α between a line segment and an implicit surface can be found rapidly using methods for ray tracing level sets; for example, the method of Singh and Narayanan [10] achieves interactive frame rates using GPUs. Thus allowing constant time checks, and allowing the algorithm complexity $O(kN + kM)$, i.e., linear in the polyhedral features.

The role of the ϵ has a different interpretation despite the fact that in the limit of small ϵ the two algorithms are equivalent. A solid intersecting the original curve with depth less than ϵ can be missed by this variation whereas it would be detected by Algorithm 1.

Further speedups

The collision detection process can be spedup considerably by quickly eliminating pairs of bodies from consideration that have no possibility of intersecting (i.e., *broad phase collision detection*. During a collision check between a pair of bodies, each bounding box is extended; only if these enlarged OBs intersect are the algorithms presented previously (i.e., *narrow phase collision detection*) applied.

Finally, as contact points are found, we reduce the interval that Algorithm 1 is called in order to determine additional contact points more quickly. This technique, which was also used in [9], speeds contact determination considerably.

4. REFERENCES

[1] S. Cameron. Enhancing GJK: Computing minimum and penetration distances between convex polyhedra. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, Albuquerque, NM, USA, April 1997.

[2] J. Canny. Collision detection for moving polyhedra. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(2), 1986.

[3] N. Chakraborty, S. Berard, S. Akella, and J. Trinkle. An implicit time-stepping method for multibody systems with intermittent contact. In *Proc. of Robotics: Science and Systems*, 2007.

[4] B. Chazelle. Convex partitions of polyhedra: A lower bound and worst-case optimal algorithm. *SIAM J. Comput.*, 13:488–507, 1984.

[5] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE J. of Robotics and Automation*, 4(2):193–203, April 1988.

[6] E. Guendelman, R. Bridson, and R. Fedkiw. Nonconvex rigid bodies with stacking. *ACM Trans. on Graphics*, 22(3):871–878, 2003.

[7] B. Mirtich. V-Clip: fast and robust polyhedral collision detection. *ACM Trans. on Graphics*, 17(3):177–208, 1998.

[8] M. Moore and J. Wilhelms. Collision detection and response for computer animation. In *Proc. of Intl. Conf. on Computer Graphics and Interactive Techniques*, pages 289–298, 1988.

[9] S. Redon, A. Kheddar, and S. Coquillart. Fast continuous collision detection between rigid bodies. In *Proc. of Eurographics (Computer Graphics Forum)*, 2002.

[10] J. M. Singh and P. Narayanan. Real-time ray tracing of implicit surfaces on the gpu. Technical Report IIIT/TR/2007/72, Intl. Inst. of Information Technology, July 2007.

[11] K. Yamane and Y. Nakamura. Stable penalty-based model of frictional contacts. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Orlando, FL, USA, May 2006.

[12] X. Zhang, M. Lee, and Y. J. Kim. Interactive continuous collision detection for non-convex polyhedra. In *The Visual Computer (Proc. of Pacific Graphics)*, 2006.