Contact Point Generation for Convex Polytopes in Interactive Rigid Body Dynamics

M. Silcowitz[1] S. Niebe[1] K. Erleben[1]

[1] Department of Computer Science, University of Copenhagen, Denmark.

Abstract

When computing contact forces in rigid body dynamics systems, most state-of-the-art solutions use iterative methods such as the projected Gauss–Seidel (PGS) method. Methods such as the PGS method are preferred for their robustness. However, the time-critical nature of interactive applications combined with the linear convergence rates of such methods, will often result in visual artifacts in the final simulation. With this paper, we address an issue which is of major impact on the animation quality, when using methods such as the PGS method. The issue is robust generation of contact points for convex polytopes. A novel contact point generation method is presented, which is based on growth distances and Gauss maps. We demonstrate improvements when using our method in the context of interactive rigid body simulation.

1. A Practical and Robust Method

In the following we will describe our approach to contact point generation. Taking a top-down approach, we first outline the method and then elaborate on the details in the following subsections. The method:

1. Compute a contact normal **n** using the GJK–GD algorithm described in Section 1.1.
2. Use the normal **n** to obtain the geometric features of the contact, using a Gauss map as explained in Section 2.
3. Obtain contact points and distance measures using the feature intersection approach outlined in Section 3.

The first step yields robust determination of the normal direction regards less of the contact state while the second step produces the best pair of geometric features that represent a contact manifold with the given normal direction. These steps are critical to the robustness. Step 3 is necessary to produce a deterministic generation of all contact points.

1.1. Growth Distances

An intuitive – and often used – measure for penetration depths and distances between objects is the minimum translational distance (MTD) [CC86], where the predominant algorithm is the expanding polytope algorithm (EPA) [vB01].

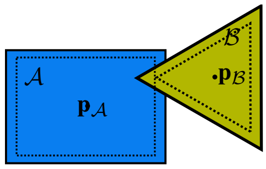

Figure 1: *The growth distance is defined as the amount that $\mathcal{A}$ and $\mathcal{B}$ must be contracted, in order to be in touching contact. $\mathbf{p}_A$ and $\mathbf{p}_B$ are the chosen contraction points for A and B respectively.*

However, the EPA requires explicit geometric representation, which is nontrivial to implement and has a large memory footprint. An alternative to MTD is growth distances (GD), which is topologically equivalent [OG94], such that

$$\alpha_1 d_{\text{MTD}} \le d_{\text{GD}} \le \alpha_2 d_{\text{MTD}} \tag{1}$$

for some $0 < \alpha_1 \le \alpha_2$. As we will show next, using growth distances to compute the growth vector, $\mathbf{p}_G$, and the contact normal, **n**, reduces to a simple ray casting.

We define the growth distance problem formally as

$$\mathbf{p}_G = \arg\min\{\alpha : \mathcal{A}_\alpha \cap \mathcal{B}_\alpha \neq \emptyset\} \tag{2}$$

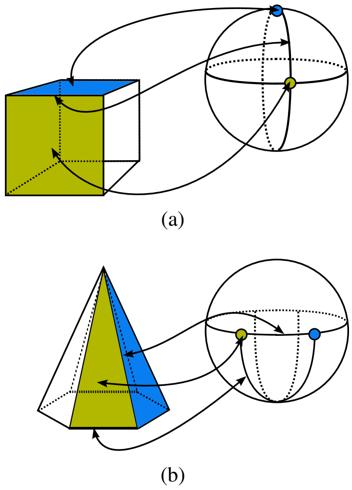

Figure 2: *Examples of Gauss maps. 2(a) The normal space of a box geometry is mapped onto a unit ball.*

where $\mathcal{A}_\alpha = \alpha(\mathcal{A} - \mathbf{p}_A) + \mathbf{p}_A$ is the contraction of $\mathcal{A}$ around contraction point $\mathbf{p}_A$. $\mathcal{B}_\alpha$ is defined analogously. The contraction points $\mathbf{p}_A$ and $\mathbf{p}_B$ have to be in the interior of $\mathcal{A}$ and $\mathcal{B}$ respectively. A good practical choice is the center of mass. Figure 1 illustrates the contraction needed for objects $\mathcal{A}$ and $\mathcal{B}$ to separate. Using these definitions, and rewriting (2) as a Minkowski sum, we get

$$\mathbf{p}_G = \arg\min\left\{\alpha : \underbrace{\left(1 - \frac{1}{\alpha}\right)}_{t}(\mathbf{p}_A - \mathbf{p}_B) \in \mathcal{A} - \mathcal{B}\right\} \qquad (3)$$

where the scaling factor $0 < \alpha \leq 1$ and $t \in (-\infty, 0]$. Clearly, when $t = 0$ then $\alpha = 1$ and both $\mathcal{A}$ and $\mathcal{B}$ are unscaled. Equation (3) describes the problem of finding a point along the line $t(\mathbf{p}_A - \mathbf{p}_B)$ that intersects $\mathcal{A} - \mathcal{B}$ and minimizes t. To solve this problem using a GJK based ray casting, the ray must begin outside of $\mathcal{A} - \mathcal{B}$, that is $t_0(\mathbf{p}_A - \mathbf{p}_B) \notin \mathcal{A} - \mathcal{B}$. This can easily be accomplished by using a support mapping of $\mathcal{A} - \mathcal{B}$. During ray cast iterations, we ensure that the ray does not move into contact with $\mathcal{A} - \mathcal{B}$, terminating when the ray enters the collision envelope.

2. Gauss Maps

A Gauss map is simply a mapping between surface normals and object features. Since all points on a facet have the same surface normal, the facet is represented as a single point on the Gauss map. See Figure 2.

Using a Gauss map allows us to expand a contact normal into a region and thereby in later steps allowing us to find all the contact points of the contact region.

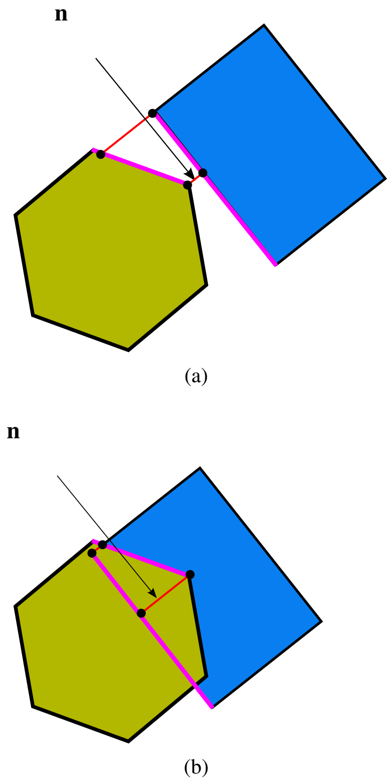

Figure 3: *By way of the contact normal* **n**, *the corresponding faces (highlighted in pink) are found using the Gauss map. The faces are intersected along the contact normal.*

3. Feature Intersection in the Contact Plane

After obtaining one geometric feature from each Gauss mapping, we find the contact points by projecting each feature onto the contact plane, defined by the contact normal, and take their intersection in the plane. Each vertex of this intersection will have two natural witness points in world space, one on each original feature. Such a pair of witness points is only displaced along the contact normal. The distance measure d_i is the signed length of this displacement, and each intersection vertex is a contact point. See Figure 3(a). d_i is an accurate measure of the distance between geometries at the contact point, which enables robust error correction.

References

[CC86] Cameron S. A., Culley R.: Determining the minimum translational distance between two convex polyhedra. In *Proceedings of International Conference on Robotics and Automation* (1986), pp. 591–596.

[OG94] Ong C. J., Gilbert E. G.: Robot path planning with penetration growth distance. In *ICRA* (1994), pp. 2146–2152.

[vB01] van Bergen G.: Proximity queries and penetration depth computation on 3d game objects. In *Proceedings of Game Developers Conference (GDC)* (2001).