Methodology for Assessing Mesh-Based Contact Point Methods

KENNY ERLEBEN, University of Copenhagen

Computation of contact points is a critical sub-component of physics-based animation. The success and correctness of simulation results are very sensitive to the quality of the contact points. Hence, quality plays a critical role when comparing methods, and this is highly relevant for simulating objects with sharp edges. The importance of contact point quality is largely overlooked and lacks rigor and as such may become a bottleneck in moving the research field forward.

We establish a taxonomy of contact point generation methods and lay down an analysis of what normal contact quality implies. The analysis enables us to establish a novel methodology for assessing and studying quality for mesh-based shapes. The core idea is based on a test suite of three complex cases and a small portfolio of simple cases. We apply our methodology to eight local contact point generation methods and conclude that the selected local methods are unable to provide correct information in all cases. The immediate benefit of the proposed methodology is a foundation for others to evaluate and select the best local method for their specific application. In the longer perspective, the presented work suggests future research focusing on semi-local methods.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Physically based modeling*

General Terms: Methodology, Contact Points, Quality, Computer Simulation

Additional Key Words and Phrases: Contact point generation, contact normals, contact regions, computational contact mechanics

ACM Reference format:
Kenny Erleben. 2018. Methodology for Assessing Mesh-Based Contact Point Methods. *ACM Trans. Graph.* 37, 3, Article 39 (July 2018), 30 pages.
https://doi.org/10.1145/3096239

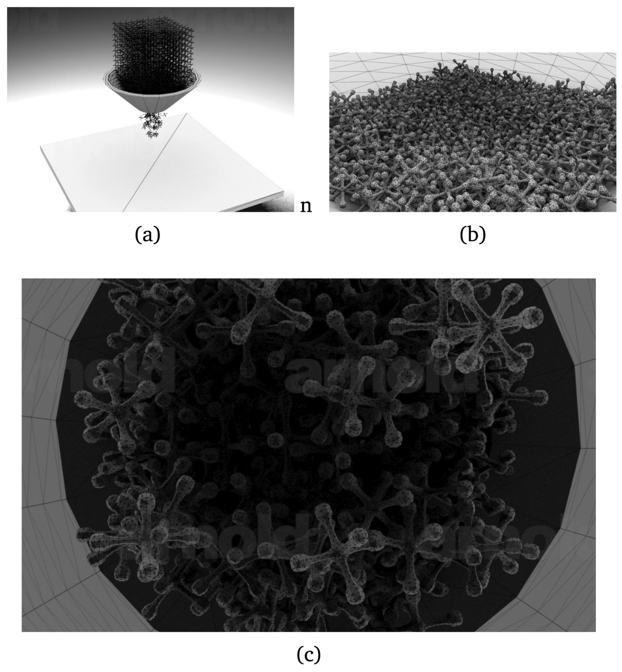


Fig. 1. Larger simulation example showing jamming of small objects in a funnel. From a distance (a) and close-up view (b) the results appear plausible. From the bottom view (c) objects are unexpectedly hanging onto other objects due to incorrect normal information.

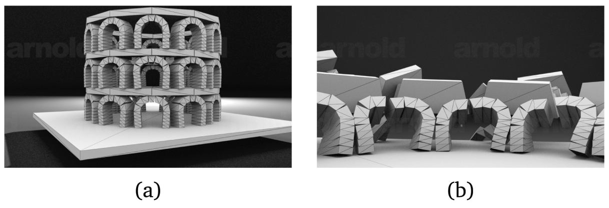
1 INTRODUCTION

The input for rigid and deformable body contact simulations are contact points. They describe the geometry of the interfaces where the physical bodies interact and are used to form kinematic constraints that govern the overall motion of the bodies. One common approach for polygonal models is to use vertex-face and edge-edge detection where contact normals are computed from cross-products (Bridson et al. 2002). This is sufficient for computing the geometry of the contact areas. It inherently deals with indeterminacy of normals, as cross-products give a unique choice at convex vertices and edges. However, incorrect normals are generated for degenerate vertex-edge and vertex-vertex cases. The degenerate normals lead to unexpected motion. The effects of incorrect normals manifest as unexpected sticking, unnatural collapse, or jittering of objects, as shown in Figures 1 and 2, and the supplementary video.

Simulation results are very sensitive towards the quality of the contact points as demonstrated in Figure 3 and the supplementary video. The need for high-quality normals is known in Robotics (Shell and Drumwright 2009). To our knowledge, the definition of what good contact point quality means is still elusive and neither rigorously understood nor defined.

Many simulation papers use the concept of contact points but do not explicitly state how they are computed or defined. This is a concern in terms of external validity of experiments (Baraff 1991; Lin 1993; Baraff 1994; Gottschalk et al. 1996; Hubbard 1996; Baraff and Witkin 1998; Klosowski et al. 1998; van den Bergen 1998; Mirtich 2000; Redon et al. 2000, 2002; Bridson et al. 2002; Bradshaw and O'Sullivan 2002; James and Pai 2004; Kaufman et al. 2005, 2008; Govindaraju et al. 2005; Erleben 2007; Zhang et al. 2007; Tang et al. 2008, 2010, 2011, 2014; Harmon et al. 2008, 2009; Daviet et al. 2011; Tonge et al. 2012; Smith et al. 2012; Mazhar et al. 2015; He et al. 2015). We claim that contact point generation needs more rigor to help researchers better compare their simulators and

Author's address: K. Erleben, University of Copenhagen, Universitetsparken 1, DK-2100, Denmark; email: kenny@di.ku.dk.



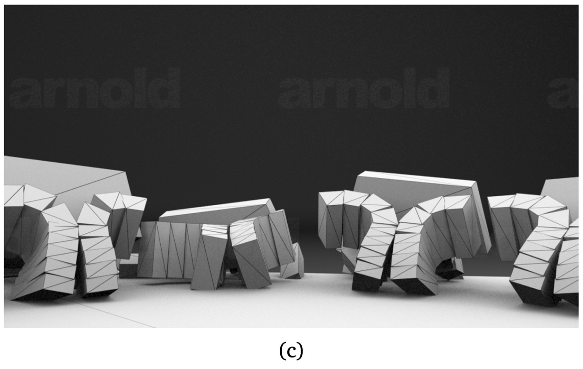


Fig. 2. Larger simulation example showing collapse of masonry structure due to poor design of the structure. Over time, incorrect normals from cliff edges cause pier stones to appear unnaturally stuck together.

solvers on even ground. Currently, a simulator can work quite well if provided good contact points, whereas the reverse is not true. This article will take a step towards a definition of contact quality.

In this work, we focus on accurate contact modeling and simulation of rigid bodies. This suggests our primary assumption that, as in the real world, large interpenetrations do not occur. This is important in digital prototyping, robotics, and training simulators as the predicted motion needs to be more accurate than the usual desired plausible motion quality known from entertainment oriented contexts.

The contributions of this work can be summarized as follows:

— A novel **taxonomy** for comparing and classifying contact point generation methods. The taxonomy uses three complementary ways to view and categorize methods. The first way uses a labeling of high-level traits; the second uses subgrouping based on geometric representation of shapes; and, last, the computational approach for normal generation is used as a descriptor.
— A careful analysis of failures in contact normal computation is conducted and leads to five generic **fundamental cases**.
— Derived from our fundamental case analysis, we formulate to our knowledge the first **test suite** for making qualitative and quantitative comparison studies of contact point generation methods. The scope of our test suite is focused on identifying a collection of necessary test cases based purely on the given geometry and instantaneous state that all simulators, regardless of their intrinsic choices, should be able to guarantee correctness of in terms of correctly computed contact points.
— We present **two new local contact point generation methods**: the most opposing surface method and the growth distance method, which we use to make **a comparison study** that spans all local methods.
— Finally, our analysis and results demonstrate that methods based on local information sometimes generate undesired normal directions and that methods that rely only on geometric and kinematic information will fail in cases where the dynamics are needed to choose the best normal direction.

A simulator can be quite complex and consist of a combination of many techniques to make it robust. It is not uncommon to add stabilization terms or provide error correction or even have strategies for specific recovery or classification techniques such as freezing and sleeping policies. Many such techniques can be used to combat bad contact information. Our goal is not to assess the quality of these post-facto methods, rather we wish to focus on the question of what contact quality means.

We make no claims that our test suite is sufficient in the sense that it covers all possible scenarios of mesh-based shapes in three dimensions (3D). We believe it to be infeasible to list all possible contact scenarios for 3D polygonal meshes, unlike for the 2D polygonal shapes where our analysis provides a closed set of cases. For instance, a generalized monkey saddle with n-dips letting $n \to \infty$ would generate a one-dimensional family of meshes with mixed curvatures, each member of the family giving rise to a possible unique test case scenario. We limit ourselves to provide necessary conditions for contact quality of mesh-based contact point generation and leave the question of a sufficient condition for future work.

Our test suite is designed to illuminate quality in normals generated by pairs of mesh elements. We believe this is not specific to rigid body simulation, although we only use rigid body simulation in this work to evaluate the quality of local contact point generation methods. We note that self-collision is computational challenging to detect but the actual contact point generation from local mesh features is the same for collisions and self-collisions.

2 CONTACT REPRESENTATIONS

A contact point models proximity information between two objects. We label the objects $\mathcal{A}$ and $\mathcal{B}$. Contact points are often used to model touching contact states as well as separation and penetrating states. The touching state is ideal for giving an intuitive description of the information associated with a contact point. Hence, we will use this state to introduce concepts. Conceptually a contact point provides three different kinds of information: a position, a normal, and a penetration (gap) measure. For continuous contact regions between two touching objects, there are infinitely many points of contact. Therefore, a contact point can in general be considered as a sample point of those infinitely many points of contact. For mesh-based methods the intersection points between the local mesh features of the two objects are often used as the contact points.

Position: In a touching state, a contact point has a reference to the two objects in contact and specifies the actual points of the two objects in contact. We describe the common touching point, $\mathbf{p} \in \mathbb{R}^3$, of the two surface points with respect to the two objects $\mathbf{p}_{\mathcal{A}}, \mathbf{p}_{\mathcal{B}} \in \mathbb{R}^3$. At the ideal touching state, one has $\mathbf{p} = \mathbf{p}_{\mathcal{A}} = \mathbf{p}_{\mathcal{B}}$. In case of separation or penetration, this equality breaks and one

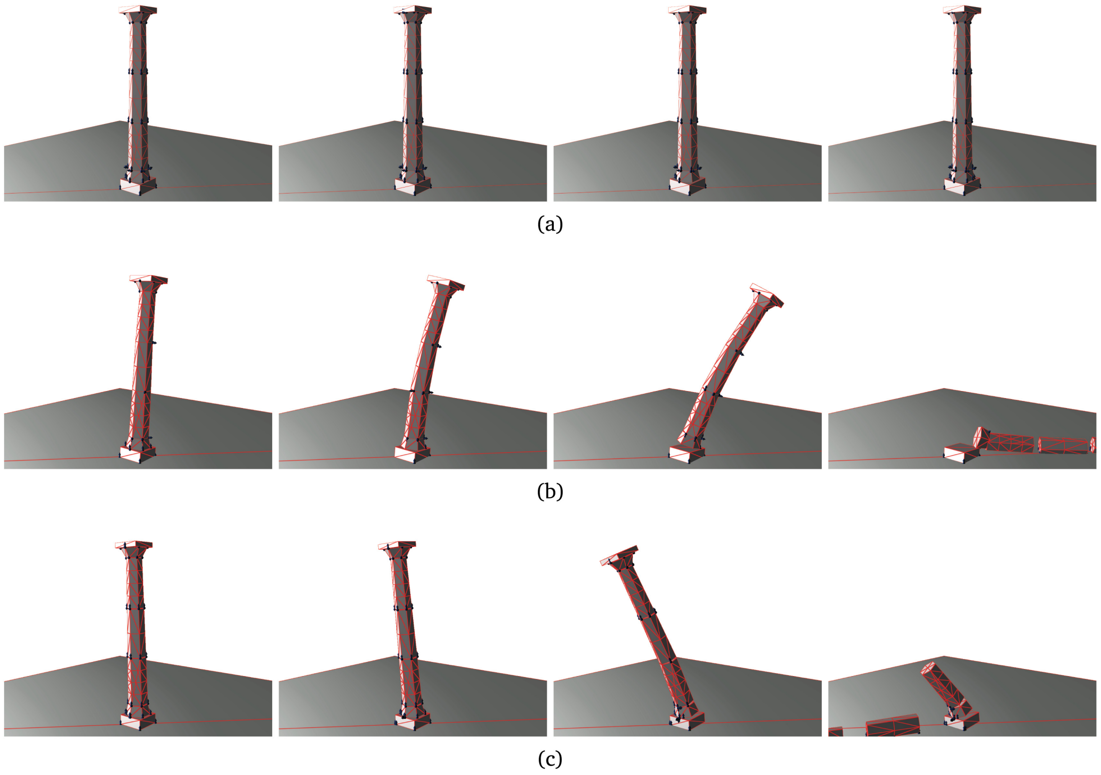


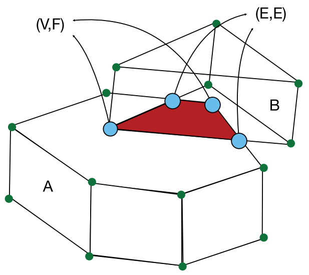
Fig. 3. Simulations are very sensitive to quality in contact point input data as shown here. Steps 200, 300, 400, and 500 are shown for (a) the **Opposing** method, (b) the **Vertex only** method, and (c) the **Closest points** method.

may define $\mathbf{p} = \frac{1}{2}(\mathbf{p}_{\mathcal{A}} + \mathbf{p}_{\mathcal{B}})$ or use some weighting of $\mathbf{p}_{\mathcal{A}}$ or $\mathbf{p}_{\mathcal{B}}$ based on volume size of the objects or similar. Positions are used as the point of action when applying contact forces. A consequence of using $\mathbf{p}_{\mathcal{A}}$ and $\mathbf{p}_{\mathcal{B}}$ in case of penetration is that ghost torques may be introduced if $\mathbf{p}_{\mathcal{A}} \neq \mathbf{p}_{\mathcal{B}}$.

Normal: In a touching state, the surfaces of two smooth objects will have unique parallel outward unit normals at any shared point on their respective surfaces. Let the two normals be $\mathbf{n}_{\mathcal{A}}, \mathbf{n}_{\mathcal{B}} \in \mathbb{S}^2$, and $\mathbf{n}_{\mathcal{A}} = -\mathbf{n}_{\mathcal{B}}$. Often only one normal $\mathbf{n}$ is associated with a contact point. Typically, implementations use a convention to use either $\mathbf{n} = \mathbf{n}_{\mathcal{A}}$ or $\mathbf{n} = \mathbf{n}_{\mathcal{B}}$ as the normal associated with a contact point. For cases of separation and penetration the concept of normal is perhaps less intuitive. In these cases, ideas such as using minimum distance vector or minimum translational distance may be used to define $\mathbf{n}$. To make matters worse, for non-smooth surfaces $\mathbf{n}_{\mathcal{A}}$ and $\mathbf{n}_{\mathcal{B}}$ becomes indeterminate, even if they are well defined one may not have $\mathbf{n}_{\mathcal{A}} = -\mathbf{n}_{\mathcal{B}}$. Normal information is used to apply normal forces in the correct direction.

Penetration Measure: The penetration is defined to be zero for touching, positive for separation, and negative for penetration. Conceptually it means the distance one would need to move along $\mathbf{n}$ to bring the two objects into a touching state. The measure is often used to add stabilization terms to counter drift errors. Hence, it need not be a distance measure but some monotone function that penalizes the error. For instance, as an alternative, volume overlap could be one such measure.

Mesh Feature Pairs: For mesh-based generation, the positions and normals are defined directly from mesh features such as vertices (V), edges (E), or faces (F) of the mesh surfaces. For instance, $\mathbf{p}_{\mathcal{A}}$ and $\mathbf{p}_{\mathcal{B}}$ can be computed as intersection points of mesh features or from closest points between mesh features. Similarly, normals can be computed from a face normal or a cross-product of two edge vectors. Hence, the types of mesh features are often associated with the contact point and used to classify the type of the contact point. For instance, one may write a contact as (V, F) or (E, E) to identify a vertex-face generated contact point or edge-edge generated contact point, respectively.



The illustration on the right shows the concept of feature pairs. The blue contact points are defined by intersection points of mesh vertices, edges and faces from the two objects $\mathcal{A}$ and $\mathcal{B}$. The drawing has two contact points of type (V, F) and two contact points of type (E, E). There is a total of six possible types. They are rarely all used as some of them can be considered sub-types of each other. For instance, a (F, F) type can be broken down into multiple (V, F) type contact points. Associating the mesh features with a contact point has the benefit of being able to track a contact point over time simply by using labels of features and objects. Feature labels are convenient to test for redundancy without having to use floating point comparisons.

3 A TAXONOMY OF CONTACT POINT METHODS

Fast, efficient, and robust computation of contact forces has been studied intensively in computer graphics, robotics, computational mechanics, and other related fields (Bender et al. 2014). This line of work rarely defines what makes up a good contact point, which is used as input for the presented improved models of contact or complex numerical methods. The field of collision detection covers a large body of work where the majority of the work has focused on data structures and algorithms for fast intersection testing or distance queries (Lin and Gottschalk 1998; Teschner et al. 2005). Unfortunately, research papers rarely cover the details of how contact points are generated from intersecting pairs of primitives such as a pair of triangles. We refer the reader to previously mentioned surveys for general work on rigid and deformable body simulations and collision detection.

In this article, we focus solely on the topic of how to compute high-quality contact points. We seek useful ways of describing the similarity between past works and find three different ways of classifying methods. The first way is taking a more high-level view using overall traits of the contact point computation. The following two ways are more concerned with low-level considerations such as how the shape is geometrically represented and how the normal computation is done.

3.1 The High-Level Traits

Our analysis will identify a classification matrix to discriminate between traits of methods. Table 1 summarizes the differences in traits we have identified.

We label contact point generation methods that share the trait that they only use local geometric information to generate contact points as *l*ocal methods. These can be methods that use a pair of intersecting triangles or tetrahedrons to produce contact points or from pairs of lower-dimensional geometric features such as vertices (V), edges (E), or triangle faces (F). Local methods are attractive as the pairwise generation is embarrassingly parallel, often does not require pre-computation and storage of extra information, and can be used by both deformable and rigid body simulations. In short, local methods are simple to implement, fast to use, and have low impact on memory footprint. In contrast, non-local methods use a partial subset (semi-local) or all of the surface features (global) of two intersecting objects to determine valid contact point information. A method that uses a neighborhood of triangles, geometric features, or a surface patch would fall into the category of semi-local methods. In contrast, if the whole shape representation of the object is used, then the method is classified as a global method. Global methods are capable of finding the minimum translational separation vector or generalizations hereof (van Bergen 2001; Kim et al. 2002). This offers at least a unique approach to dealing with objects suffering from very large penetrations but is computationally more expensive and may still cause discontinuous normal information when interior points move across the symmetry sets of the objects.

Another trait to distinguish is exact versus approximate geometry. Exact geometric contact information means one uses an accurate geometric representation of two touching objects. Exact geometric representations are necessary when simulating objects with sharp edges. As an alternative, many works use approximate geometric representations (Hubbard 1996; James and Pai 2004; Baciu and Wong 2004; Allard et al. 2010; Civit-Flores and Susín 2015). Approximations can offer interesting tradeoffs, such as performance versus accuracy and control of the smoothness of the shapes, but inherently lack the ability to deal accurately with sharp edges.

Collision Detection comes in two variants: discrete and continuous (DCD and CCD). This classification extends naturally to contact point generation methods. Although the CCD group could be further classified into explicit continuous contact generation that tries to trace and locate the first time of contact and more recent work that considers the coupling of dynamics and contact point generation yielding a fully implicit continuous flavor to contact point generation (Williams et al. 2016).

3.2 Variation in Geometric Representations

The work on exact geometric local contact point generation methods is naturally grouped by geometric representations. We have identified five such groups: Surface-based Testing, Volume-based Testing, Approximate Methods, Implicit Fields, and Temporal Methods. Table 2 summarizes the characteristics.

3.2.1 Surface-based Testing. Early works investigate triangle meshes in detail. A contact point is identified by a pair of mesh features (E, E), (V, F), and (E, F) (Moore and Wilhelms 1988; Hahn 1988; Baraff 1989; Lin 1993). Their analysis shows that the (E, E) type of contact is necessary for correct modeling. They state that for most cases (V, F) testing is sufficient for animation (Moore and Wilhelms 1988). Further, the idea of clipping the E of an (E, F) contact type against the Voronoi planes of the polygon face F was introduced in this founding work. The gaming community later refers to this technique as clipping (Coumans 2010). The cause of ill-posed indeterminate contact has been established as surface normals not being well defined at edges and vertices (Hahn 1988; Baraff 1989). Many follow-up works have continued to use triangle element-wise testing (Baraff 1997; Baraff and Witkin 1998; Baraff et al. 2003; Govindaraju et al. 2005; Tang et al. 2008; Curtis et al. 2008).

Bounding volume hierarchies (BVH) are often used as efficient spatial acceleration data structures for general non-convex shapes. Ultimately, they result in pairwise triangle versus triangle intersection testing. Many variations exist over volume types (oriented bounding boxes, discrete oriented polytopes, axis aligned bounding boxes etc.) (Gottschalk et al. 1996; Hubbard 1996; Klosowski et al. 1998; van den Bergen 1998; Bradshaw and O'Sullivan 2002; James and Pai 2004). Many of the methods have been implemented in free available collision libraries from GAMMA at UNC and have been used for simulation (Cohen et al. 1995; Hudson et al. 1997; Kaufman et al. 2005; Zhang et al. 2007). Interfaces often return a list of pairs of overlapping triangles from which end-users must post-process data. The Lin-Canny and V-Clip algorithms are based on closest points computation between mesh features (Lin 1993;

Table 1. Summary of Different Traits of Contact Point Generation Methods

Category	Exact vs. Approximate Geometry	Discrete vs. Continuous
Local Methods	Both variants exist. Methods are fast, provide detailed contact points, and suffer from indeterminacy of normals and lack of knowledge of global shape.	Continuous extensions exist for many methods in this category and can deal with tunneling artifacts. Discrete versions have advantages for fixed timestepping methods.
Semi Local Methods	To our knowledge most work approximate many surface or volume mesh elements by smoother patches.	To our knowledge most work is discrete.
Global Methods	Exact geometry is often used, but results are often a single pair of extreme points describing a large contact area.	To our knowledge most work is discrete.

Table 2. Summary of Different Geometry Representations Used for Local Methods

Geometric Representation	Benefits	Drawbacks
Surface-based Testing	Efficient, and many public implementations exist.	May suffer from redundancy, ill-defined normals, and tunneling.
Volume-based Testing	Efficient, robust inside/outside determination.	Suffers from same problems as surface-based testing, may generate more tests.
Approximations	Efficient, often well-defined (smooth) normals.	Imprecise representation of true contact area, and suffers from tunneling.
Implicit Fields	Efficient, robust inside/outside determination, easy to define gap functions, and often well-defined smooth normals.	Large memory consumption, non-trivial updates for deformations, smoothed surfaces, discontinuous gap functions, local normals, and suffers from tunneling.
Temporal Methods	Sweeps or bounds motion to prevent tunneling, can take larger timesteps, surface- and volume-based testing and approximations are trivially extended to temporal methods.	Can suffer from infinite collapse, performance-wise expensive testing, numerical precision and sliding motion are challenging.

Mirtich 1998b). Other distance-based settings can estimate penetration from the locally maximal distance of overlapping feature pairs (Sud et al. 2006). Other work assumes sufficient smoothness to define contact planes for mesh nodes (Jourdan et al. 1998).

3.2.2 Volume-based Testing. Usage of volumetric elements, also termed shape primitives, has been and still is the main approach in interactive physics simulators (Erleben et al. 2005; Coumans 2010). Here, specialized test methods have been devised and optimized by large communities. The methods are fast and often well tested to provide robustness. The gaming community approach to non-convex shapes is to decompose the shapes into smaller convex pieces. The extreme would be to consider a general non-convex rigid body as a decomposition into tetrahedra or volumetric hierarchical convex decomposition (HACD) as done in Bullet (Coumans 2010). In the Mazhar et al. (2015) primitive, spherical shapes appear to be used. Cylindrical primitives are used for hair simulation in Daviet et al. (2011). Primitive shapes can be used as an approximation technique when their union does not coincide exactly with the object they represent. Various sphere tree representations can be seen as examples of this (Hubbard 1996; Bradshaw and O'Sullivan 2002; James and Pai 2004). Contact point generation for tetrahedral meshes for deformable objects is described in Heidelberger et al. (2004). Here a multi-stage method is used to overcome two common artifacts, incorrect normals due to choosing normal direction based on closest surface point and discontinuities in penetration depths. The method relies on having penetrations and will not be applicable otherwise. In Spillmann and Teschner (2005) the approach is extended to deal with side effects of coarse meshes. Polyhedral representations are found for pairwise overlapping tetrahedra in Parker and O'Brien (2009). The centroids of the polyhedra are used as the action points for the resolving forces.

3.2.3 Approximate Methods. Sphere-trees of triangle meshes generated from medial axes are a well-established approach for time-critical simulation (Hubbard 1996; Bradshaw and O'Sullivan 2002; James and Pai 2004). Spheres built from geometry images were used in Beneš and Villanueva (2005). In all this work, sphere primitives are used to generate approximate contact points when the collision query is interrupted. Hysteresis of contact tracking was presented in Mirtich (1998a); contact caching and perturbation are similar concepts known from the gaming community (Coumans 2010). The idea is to generate the full contact information through temporal sampling. Moreau and Jean used elastic polygonal blocks for simulating buildings. The blocks are decomposed into triangles in 2D or eight-node hexahedra in 3D. Midpoints on edges or faces are used as candidates for contact between blocks (Jean 1999).

Contact volumes for triangle meshes offer interesting aspects of balancing computational resources and exactness (Allard et al. 2010; Faure et al. 2008). However, scenarios with objects tilting over a sharp ridge may be inaccurately modeled. Multi-level representations are known too such as the work by Civit-Flores and Susín (2015). This work embeds a triangle mesh in a coarse tetrahedral mesh with associated smooth surface representation. This gives a unique defined contact normal base from a continuous definition of surface normals.

3.2.4 Implicit Fields. Signed distance fields were used directly in Guendelman et al. (2003), Erleben (2007), Kaufman et al. (2008), and Smith et al. (2012) to generate penetration measures and normals at sample points for rigid body simulation. Articulated bodies were addressed in Weinstein et al. (2006). The regular grid on which the signed distance field is stored tends to smooth the surface yielding robust and well-defined normals as pointed out in Erleben (2005). Extensions to deforming distance fields have been proposed, too (Fisher and Lin 2001; Hirota et al. 2000, 2001). It is well known that bad contact normals can be generated due to choosing the normal that is dictated by the closest surface point. One suggested remedy by English et al. (2013) is to remove contacts where normals are not in the region defined by the conical combination of the normals of the faces meeting at the feature. This will imply loss of accuracy of the contact representation and potentially lead to severe penetration. Using approximate distance fields was explored in Marchal et al. (2004) and using the fast marching method (FMM) in Bridson et al. (2003) and Teran et al. (2005). The abstract notation of a gap function of moving points can be used to define non-penetration as for example done in Harmon et al. (2009) and Harmon et al. (2012).

3.2.5 Temporal Methods. Many works on CCD can be described as a temporal extension of the surface-based testing methods previously described. Often only pairwise feature (E, E) and (V, F) tests are necessary. For rigid bodies, algebraic equations under the assumption of screw motion and interval arithmetic have been studied (Redon et al. 2000, 2002). Interval testing has been extended to Separation Axis Tests (SATs) for OBB intersection testing (Redon et al. 2002) and for deforming tetrahedral meshes (Tang et al. 2011). Articulated bodies are presented in Zhang et al. (2007) as a generalization of the conservative advancement method by Mirtich (1996). When using deforming triangle and tetrahedral meshes it is often assumed that vertices have a linear motion (Moore and Wilhelms 1988; Bridson et al. 2002; Tang et al. 2014). This results in a cubic polynomial root search problem. Once the root is computed it is tested if closest points of (V, F) and (E, E) contacts are within proximity and the separation vector is used as the normal vector (Bridson et al. 2002; English et al. 2013). Follow-up works explore connectivity information to eliminate redundant elementary tests (Tang et al. 2008), failure proof methods (Wang 2014), feature level culling of redundant element-wise tests (Tang et al. 2011), using the Bernstein basis to ensure reliable testing (Tang et al. 2014), and extensions to deal with topological changes (He et al. 2015). Hence, past work for deforming meshes can be summarized as limited to linear motion of the vertices, much attention has been on solving the cubic root search problem reliable and fast culling methods of unneeded tests. Dealing with sliding and non-linear motion remains challenging. Tessellation of ambient space were explored in Misztal et al. (2012), Granados et al. (2014), and Müller et al. (2015) allowing for conforming contact manifolds that only need (V, V) feature pairs (Erleben 2014).

3.3 Normal Computation by Local Methods

Considering accurate simulations where penetrations are negligible, the actual intersection testing is rather straightforward for those delimitations and the major variation for exact geometric discrete local methods for triangle or tetrahedra meshes lies in how contact normals are generated. Table 3 summarizes major variations.

3.3.1 Normals from Features Types. Normals for intersecting (E, E) contacts are generated by edge cross-products and for intersecting (V, F) by plane normals (Moore and Wilhelms 1988; Hahn 1988; Baraff 1989), this is equivalent to SAT generation that uses the separation axis as normal direction (Baraff 1997). Both variants have natural CCD extensions (Redon et al. 2000, 2002), CCD adds complexity to queries as extra information about velocities is needed to be passed along to the collision detection system. Further, most CCD methods assume an initial separated state, which is hard to guarantee due to time-discretization, drift errors caused by inaccurate contact forces or user interaction. The combinatorial rules for intersecting features can be extended to cover penetration by considering (E, F) cases where the face normal will be used to define the normal (Mirtich 1998b). For SAT-based generation one often picks the minimum overlapping separation axis as the contact normal when penetration occurs. Penetration extensions are usually only done for DCD. In case of continuous collision detection, the normals can be generated by these methods by considering the intersecting features or separation axis at the first time of contact (Bridson et al. 2002; Tang et al. 2012). Both approaches can be applied to deformable and rigid objects in a straightforward manner. By decomposing tetrahedra into triangles, the normals may be generated with the intersection approach for surface meshes or using SAT-based testing directly for tetrahedra.

3.3.2 Normals from Closest Points. Using the separation vector between closest points of pairwise local features as the direction of the contact normal is inherently numerically unstable as objects come in closer and closer contact (Bridson et al. 2002; English et al. 2013; Brochu et al. 2012). If objects can be kept sufficiently separated at all times, then this yields a deterministic approach to finding contact normals. Hence, the major drawback is mostly a numerical concern. The approach needs certain modifications to deal with penetrations but can be applied in a straightforward manner to both deformable and non-deformable objects. The approach can be used naively in CCD approaches, too. Further, it generalizes to primitive shapes and even non-polygonal shape types. Hence, volume meshes are inherently supported.

3.3.3 Normals from Implicit Fields. Alternatively, contact normals can be defined by the gradient of some implicit representation describing object shapes (Guendelman et al. 2003). Distance fields are a quite popular choice. Using a signed distance field or other implicit function representation implies either auxiliary data for unstructured meshes or the application of implicit defined geometry. Many variations of gap functions exist but are in principle similar to the local "distance" metric defined by Harmon et al. (2009)

Table 3. Summary of Normal Generation Methods for Local Methods: Categories Are Normals from Features Types (FT), Closest Points (CP), Implicit Field (IF), and Points in Volumetric Elements (PV)

Method	Description	Category	Traits	Strong Limitations
Intersection	Applies rules to a pair of intersection feature types to pick normals from feature geometry.	FT	(2 , 2 , 2 , 2)	Only surface meshes.
Surface-SAT	Uses the minimum overlap or separated axis to determine the contact normal.	FT	(2 , 2 , 2 , 2)	Only surface meshes.
Volume-SAT	Similar to **Surface-SAT** except it is based on volumetric elements.	FT	(2 , 2 , 2 , 2)	Only volume meshes.
Closest points	Uses the separation vector between closest points to compute the normals.	CP	(2 , 2 , 2 , 2)	Numerically sensitive.
Implicit field	Computes the gradient using some auxiliary data structure.	IF	(0 , 1 , 1 , 1)	Storage overhead, cost of updating data structures, non-intuitive normals.
Vertex only	Computes the gradient from face normal of closest surface point.	PV	(0 , 0 , 2 , 2)	Only volume meshes, poor for coarse meshes.
Consistent vertex	Avoids non-intuitive normals by using motion entry points or some weighting directional vectors in place of the closest surface point.	PV	(0 , 0 , 2 , 2)	Only volume meshes, poor for coarse meshes.

Traits are defined as the supports tuple (Exact Geometric Representation, Continuous Collision Detection, Deformable Models, Topology Changes), and these capabilities are indicated by the numbers: 0 not possible, 1 can be supported but requires some effort, 2 trivially supported.

and Harmon et al. (2012). For instance, Volume contacts uses layered depth images to compute the volume overlap as a gap measure and the volume-overlap gradient as the normal (Allard et al. 2010). The approach often requires pre-computation and, for deformable objects, runtime updating, both result in computational overhead (Fisher and Lin 2001; Hirota et al. 2001). Once a field representation is available, one often has very fast runtime testing for point inclusion and normal computation. The approach is known to suffer from artifacts when dealing with edges, non-intuitive normals determined by closest surface points, and discontinuities of penetration depths. Using a regular sampling of distance fields may add a little smoothing that can help on the indeterminacy issue of contact normals (Erleben 2005). The method can be very expensive for deformable models as data need updating before each query (Marchal et al. 2004). Topological changes are not straightforward when keeping the distance field as an auxiliary data structure. Further, memory foot-print can be large. Even when using approximate or adaptive distance fields. The extension to CCD is not straightforward. One may use a point sampling of objects in combination with a signed distance field to determine when the trajectories of the points intersect the iso-surface. Drawbacks are that sliding motion is tricky and unsampled sharp features may cause deep penetrations (Xu and Barbič 2017). In summary, this class of methods can be used easily for rigid objects but requires overhead for deformable objects and topology changes.

3.3.4 Normals from Points Inside Volumetric Elements. Whenever a vertex/point is inside a volumetric element, one may use the face normal of the closest surface point of the volume as the contact normal or, as an alternative, determine the entry point on the surface of the element and use the face normal of this as the contact normal (Heidelberger et al. 2004; Spillmann and Teschner 2005). These methods are limited to volume meshes. The volume-based setting tends to add robustness in quick inside/outside testing. The volume meshes can be used both for simulation and collision detection although graphics work exists that uses coarse simulation meshes to gain performance. Edge-crossings are often an issue, but they are mostly ignored for animation. If not, then one may search for the deepest point on an edge or averaging of in-out surface intersection points on an edge. Normals are often chosen as the surface normal of the corresponding closest surface point or as a weighting of directional vectors from interior points to surface points.

These methods are limited to volume meshes but can handle deformables, rigids, and topological changes straightforwardly. They rely on actual penetration, and hence CCD is not applicable and the methods have known issues with coarse meshes. For small penetrations, the methods will yield the same contact information as obtained from a distance field approach (Hirota et al. 2000; Guendelman et al. 2003). For small penetrations, consistent vertex methods are similar to using a CCD method to find the first point of (V, F) contact and then trace forward the V position to the end of the timestep and use the F normal to generate the contact normal (Tang et al. 2012).

4 THE FUNDAMENTAL CASES

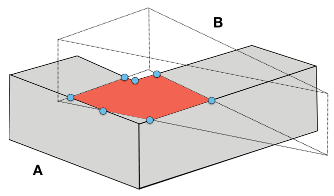
In terms of geometry, the different possible types of contact may be classified combinatorically as a pairing of local curvature classifications with respect to the local object surfaces. For simplicity, let us start our analysis in 2D. For a sufficiently small neighborhood at a point of contact on a smooth surface, the surface can locally be classified as being flat, convex, or concave, in 2D we can label the shape by the sign of the mean curvature. This local surface classification also holds when considering 2D polygon shapes. For starters, we consider an object's surface to be smooth. Hence, in 2D, we have six possible combinations to consider between two object surfaces $\mathcal{A}$ and $\mathcal{B}$: $(0, 0)$, $(+, 0)$, $(-, 0)$, $(+, +)$, $(-, +)$, and $(-, -)$.

We immediately notice that two of the cases $(-, 0)$ and $(-, -)$ are infeasible, as they would imply that the local geometry is penetrating. The $(0, 0)$ case is encountered at points interior to a contact area region, so-called planar regions. The $(+, 0)$ can occur, for instance, in case of a point touching a flat surface or at the boundary of a contact region touching another flat surface object (similar connectivity to a T-junction). The case of $(+, -)$ could be infeasible if the absolute value of the negative curvature is larger than the value of the positive curvature. This type of contact can occur in case of a wedge/point in a crack or ball in bowl. The $(+, +)$ case can occur if two points are in touching contact or at the co-located boundary point of two surfaces. The cases are illustrated in Figure 4. For 2D smooth objects in touching contact these form a closed set of cases. We may now straightforwardly extend the concept to piecewise continuous linear (PCL) shapes, which are simple polygons in 2D. Three-dimensional vertices poses more contact types than simple convex and concave vertices, namely mixed concave/convex vertices. For instance, saddle-points or mixed concave/convex edges incident to a strict concave or convex vertex.

PCL surfaces are often represented using polygonal meshes. In 2D these are connected line segments; that is, edges (E) connected by vertices (V) and in 3D a triangle mesh is an often-used example for representing PCLs consisting of triangle faces (F), edges (E), and vertices (V). This polygonal discretization/tessellation of the PCL shapes may lead to incorrect contact information. This is often experienced as contact is detected by locally testing pairwise features of the shapes. In 2D this implies finding all pairs of intersecting or touching edge segments and from these generating the contact point information. Contact point information is classically given as a point position and a unit-vector normal. Hence, in 2D from a local pair of geometric features (V, V), (V, E), or (E, E) one has to compute the intersection points and deduce proper normal information. In 3D, the approach for triangle or tetrahedral meshes is similar, although more pairwise types exist, (V, F), (E, F), and (F, F).

It is straightforward to determine intersection points for single-point-of-contact scenarios. For continuous regions of contacts in 2D, the intersection points representing the boundary points of the region must be found. In 3D, intersection points must be computed to ensure that the boundary of the region is exact. Our definition of exact geometry means that all strict convex and concave points of the boundary of the contact region must have a corresponding contact point otherwise one cannot capture phenomena such as sliding and tipping over an edge. One may include other boundary points or interior points in the discrete representation of a contact region. The strictly convex and concave points of the boundary are necessary for having exact geometry.

The concepts are illustrated on the right. Here a see-through triangle prism object is placed on top of an angle shaped object. The red planar contact region is given by the boundary polygon defined by the circular shaped boundary points. For the planar case these can be classified as convex or concave points. The illustration has one concave boundary point.



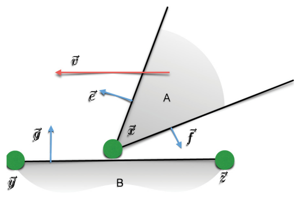
For smooth shapes, the contact normal is uniquely given by the co-parallel outward unit normals at a given point of contact. The smoothness ensures a surface point has a well-posed normal. Given the two normals are parallel, we represent the normal direction using the unit normal pointing from the object we label $\mathcal{A}$ to the object we label $\mathcal{B}$. This is merely a convention we apply. Determining normal information is more problematic for the PCL shapes for two reasons.

(1) The normals are not well posed at vertices, and a multi-set of normals can be associated with a vertex. This problem is due to the non-smooth geometry.
(2) The polygonal tessellation can allow for straight boundaries to be represented by several straight connected parallel edges. Causing internal vertices on flat pieces of the boundary.

The two problems extend trivially to 3D, with the added complexity that normals are not well defined for edges and tessellation can cause internal edges as well as internal vertices. Taking the tessellation and the ill determinacy of normals into account, we are now ready to study common cases for localized contact of polygonal based objects in 3D. We will identify cases that cover the types of contact we encountered in the closed 2D type classification scheme (see Figure 4). This will form a minimal set of necessary 3D cases that at least covers all 2D cases. We make no claims towards whether our fundamental tests are sufficient in the sense that they cover all possible 3D cases. Mixed curvature cases exist in 3D but have no 2D counterpart. Hence, we address mixed curvature as an extra 3D case. To illustrate the problems, we will consider the motion of objects. As we only care about relative information, we keep one object fixed to simplify explanations without loss of generality.

4.1 Sliding Point

First, we will study the case of a point on a planar support (Type 1). This is illustrated on the right. We name this case the **Sliding point**. Here local pairwise testing could imply that the contact normal direction would be determined by one of the edges from object $\mathcal{A}$. Hence, the simulation would see object $\mathcal{A}$ as standing on a hillside. If $\mathcal{A}$ is subject to gravity or sliding, then $\mathcal{A}$ will either penetrate object $\mathcal{B}$ when falling under gravity or jump off object $\mathcal{B}$. As one would test all intersecting pairwise local features, one could generate three normal directions, one for each of the contact edges **e**, **f**, and **g** in the drawing.



The physical consequence is that the simulator will think object $\mathcal{A}$ is at the bottom of a ridge created by **e** and **f**. Hence, object $\mathcal{A}$ would not be allowed to slide freely across the surface of object $\mathcal{B}$ as the drawing suggest. The only normal direction that would result in the expected motion is the straight up-down direction.

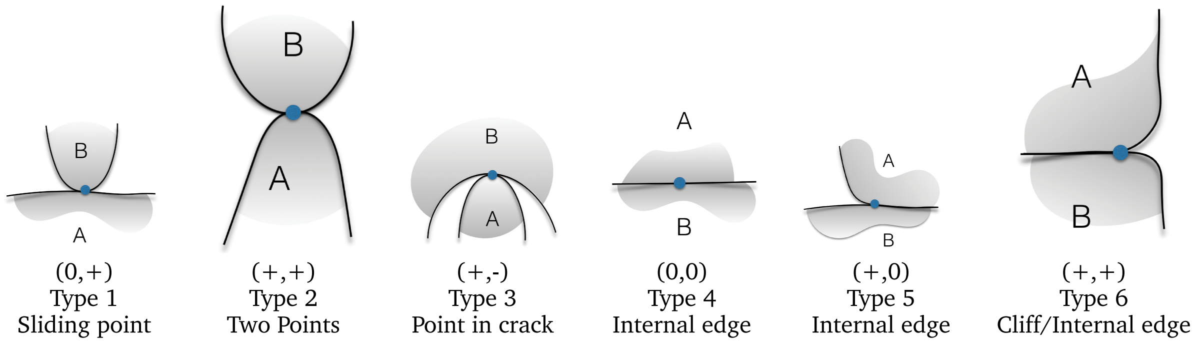
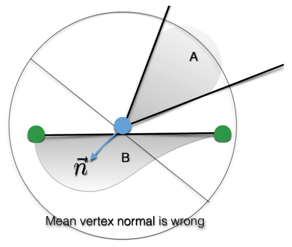


Fig. 4. Considering all possible curvature combinations of 2D smooth shapes labelled $\mathcal{A}$ and $\mathcal{B}$ results in six different cases categorizing different types of geometric contact. Notice when considering multiple points of contact then the (+, 0) and (+, +) appear as two variants classifying boundaries of contact regions.

Hence, only the **g** edge of object $\mathcal{B}$ should be used to define the normal direction.

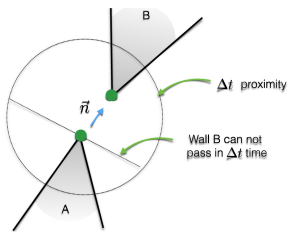
The problem is of course how one should figure out when to use the $\mathcal{B}$ edge and not any of the $\mathcal{A}$ edges. The core of the problem is that the local pairwise features do not know enough about the true boundary shape of the objects.



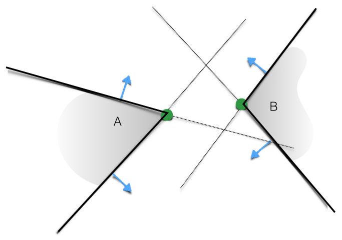
One could argue that the mean-normal of the vertex of object $\mathcal{A}$ would be a good normal direction for the sliding tip case. However, imagine that object $\mathcal{A}$ is slightly tilted, and then one would have the case of object $\mathcal{A}$ standing on a hill side as seen from the simulator viewpoint, as illustrated on the left. Hence the edge of object $\mathcal{B}$ is the only proper choice.

4.2 Two Points

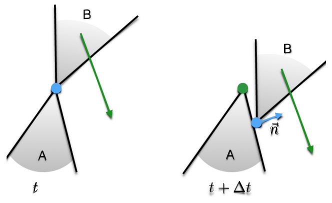
We name the point touching point (Type 2) contact case: **Two points**. This case is geometrically difficult due to the non-smooth nature of polygonal objects. A convex corner does not have a unique normal; instead, one may associate a whole set of normals to the convex surface point. This has been termed indeterminacy or degeneracy (Baraff 1989). One popular argument is that these types of contact are unlikely to be stable. Hence, they are often classified as short termed, and it can be argued that one, therefore, may have a lot of freedom in picking a normal as the contact will be broken in the following timestep, and it is inherently intermittent in nature.



One must be careful as large timesteps mean one is enforcing any chosen feasible normal direction in a proximity zone around the actual point. On the left, we show how the large timestep creates a pseudo wall in the simulation. Here the normal was generated by the closest points of a (V, V) type contact. The circle illustrates the motion bound of the vertices future trajectories within the given timestep size Δt. Even if object $\mathcal{B}$ does not touch object $\mathcal{A}$, the vertex of object $\mathcal{B}$ will hit the wall generated by normal **n**.

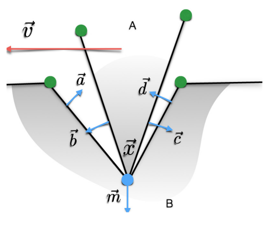


Using all face normals simultaneously as shown on the right may result in a so-called dual model trap (Williams et al. 2016). This implies the points get unnaturally stuck in the normal cones of each other. This can occur when (V, V) type of contacts are represented implicitly through multiple (V, F) type of contacts.



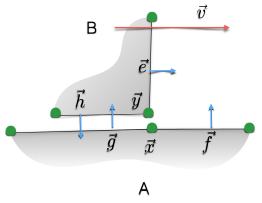
The difficulties are further enhanced, as this type of contact often suffers from indeterminacy due to the actual motion as illustrated on the right where the normal is not easily identified from geometry only and the relative motion must be taken into account to avoid creating a undesired point-wall. One approach to solve this is to combine the contact generation and dynamics models; we refer to Williams et al. (2016) for further details.

4.3 Point in Crack

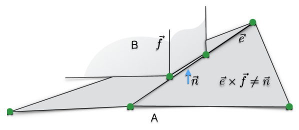


On the right, we analyze the case of a **Point in crack** (Type 3) contact. Here mean vertex normals correspond to $\mathcal{A}$ sliding on a plane; if any edge/side of the wedge is used, then $\mathcal{A}$ will be observed to appear to be virtually sliding up against its own surface and not the actual surface of $\mathcal{B}$. Hence, one must use edges/faces from $\mathcal{B}$ only to generate normal direction. Further, both sides of $\mathcal{B}$ will be needed; otherwise, if $\mathcal{A}$ was at rest, gravity would start pulling $\mathcal{A}$ into $\mathcal{B}$ when $\mathcal{A}$ slides down the side of $\mathcal{B}$ used to generate a single contact point. Notice the duality with the sliding point case. In that case, we only wanted a single unique normal. For this point in crack case, we really need two normals to be formed at the pointwise contact.

4.4 Internal Edges

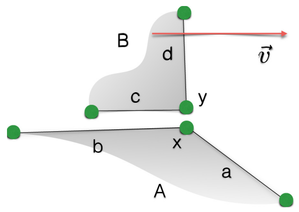


The **Internal edge** case as illustrated on the right suffers from strange artifacts when picking the normal direction from local information. In 2D, this may occur when the vertical edge of $\mathcal{B}$ dominates the decision. In the illustration the (V, V) type of contact (x, y) and (V, E) type of contacts (x, h), (y, f), (y, g), and (E, E) type of contacts (h, f) and (h, g) all result in a valid normal. All remaining combinations of mesh features will generate an incorrect normal.



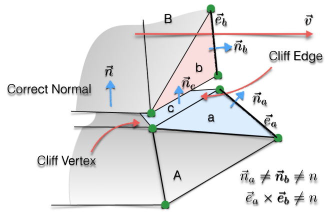
In 3D, the internal edge counterpart can in fact be created by two orthogonal edges from $\mathcal{A}$ and $\mathcal{B}$ leading to a horizontal contact normal as shown on the right. The horizontal normals are incorrect as they will cause a bounce and/or object $\mathcal{B}$ sinking into object $\mathcal{A}$. The internal edge can be seen as a special case of a flat cliff edge (to be described in the next section); it is the locality and tessellation that combined causes the problem. Cliff and internal edges cover the common failures that occur for local planar support, boundary point on a planar support, and co-located boundary points of planar support regions (Types 4, 5, and 6).

4.5 Cliff Edges

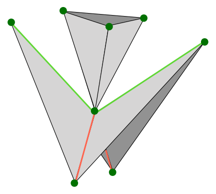


The **Cliff edge** case (Type 6) is illustrated on the right in 2D. The problem here occurs due to the local testing of pairwise edges and disregarding the true global shape of the objects. If, for instance, one is testing the boundary vertex of $\mathcal{A}$ with the vertical right edge of $\mathcal{B}$, then the edge dominates the (V, E) contact, and one would use the E to define a contact normal pointing to the right. Assume the $\mathcal{B}$-object is instantaneously moving to the right with a downward gravity; then the resulting motion after contact response will be a bounce so $\mathcal{B}$ moves to the left while falling under gravity. This is an unexpected motion. The cases where the horizontal edges of $\mathcal{A}$ and $\mathcal{B}$ dominate will lead to upward pointing normals. This will give rise to the right expected motion of $\mathcal{B}$. That is $\mathcal{B}$ sliding off $\mathcal{A}$ without bounce and without falling down into object $\mathcal{A}$. A case may exist where the normal direction can be chosen from the mean normal at the vertex of $\mathcal{A}$ or the inclined edge. In this case, the $\mathcal{B}$-object will jump off the object $\mathcal{A}$. If object $\mathcal{B}$ was not moving, then such a normal direction would result in $\mathcal{B}$ being kicked off the cliff. The kick-off direction could be caused in 3D by (E, E) type of contact if $\mathcal{B}$ did not have straight up and down vertical edges.

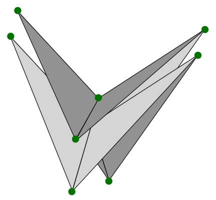
The analysis extends trivially to 3D, and results in two subcases of cliff vertices and cliff edges as shown on the left. Here one can favor (V, F) types of contact that involves face c or the opposing face on object $\mathcal{B}$ (not visible on drawing) to give proper normal directions. If face a or face b are involved in defining the contact normal, then the wrong motion would result. Notice that (E, E) type of contacts, too, can generate incorrect normals in 3D. For instance, the edge vectors cross-product $\mathbf{e}_b \times \mathbf{e}_a$ in the drawing will result in an incorrect normal.



4.6 Mixed Curvature



The added difficulty in 3D is mixed curvature. Some mixed curvature cases are covered by our previous cases. For instance, a spike hitting the center of saddle point can be broken down into a **Point in crack** case and a **Two points** case by observing that the saddle has a convex and concave ridge of edges. This is illustrated on the right. Here the green edges form a concave case, and the red edges form a convex case. For this particular case, all four-adjoining face normals are needed to prevent the spike from penetrating any of the faces. The convex part may cause a potential double trap. Kinematic information would be needed to resolve this correctly.



The mixed curvature cases can be an infinite source of difficult cases. On the left, we illustrate two saddle points coming into a (V, V) contact with each other. This case offers little intuition, and one can easily generalize the challenge by considering the same setup with generalized n-dip monkey saddles. Hence, our proposed cases cover only a small sub-set of all possible mixed curvature cases.

5 TEST CASES FOR QUALITY ASSESSMENT

For the study of the quality of normal generation with regard to cliff edges and internal edges, we propose to use the Pillar case study sketched in Figure 5. The structure is created such that we know the only feasible normal direction is in the straight up–down direction. The benefit is that one may simply assess normal quality by analyzing histograms of the absolute value of the up component of the normals. The optimal value is |1|. Doing simulation of a structured masonry structure initially at rest has several advantages to testing and measuring quality. The structured simple stacking enables quick visual inspection to determine robustness and stability. The stacking is such that a simulation will suffer from large mass ratios, provoking sagging motion of the pillar if contacts are poorly generated. The crown of the pillar adds a larger mass on a stick; this will amplify any asymmetries in simulation results, increasing chance of pillar tipping over.

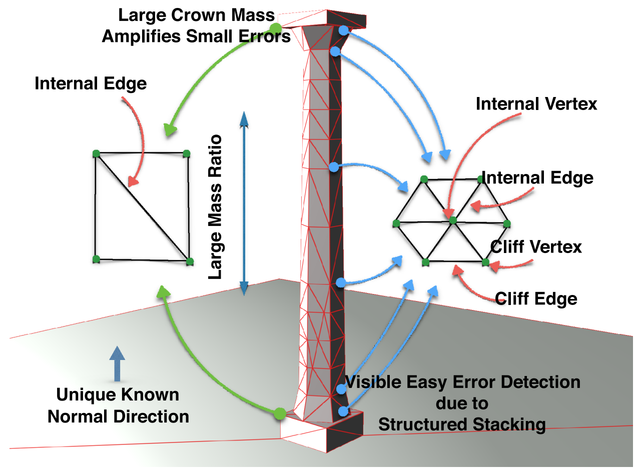


Fig. 5. The Pillar case study is careful designed to generate many cases of cliff edges and internal edges and allows for fast and trivial quality measure of normal vectors.

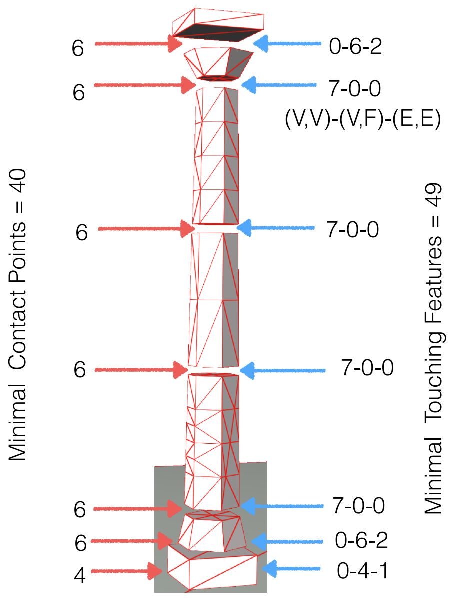


Fig. 6. Analysis of ideal contact count for the Pillar case. On the left is indicated the optimal minimum number of contact points that spans the contact region. On the right, the mesh tessellation is used to determine the minimum number of pairs of local touching features.

The geometry of the pillar is generated such that each circular cap consists of six triangles meeting at a vertex in the center of the circle. This specific choice will generate many cases of internal edges and internal vertices. Further, the vertical and inclined sides of the pillar generate many cliff edges/vertices, too. The choice of geometry means it is trivial to analyze exactly how many contact points an ideal method would result in. A careful analysis of touching (V, V), (E, E), and (V, F) cases ignoring coincident edges and faces reveals that an optimal method should produce 40 contacts to be minimal and no more than 49 contacts uniquely defined by touching features. The analysis is depicted in Figure 6. The benefit of knowing the ideal numbers allows us to gain intuition about the amount of excessive contact information. Notice that the coarse meshing will create multiple incident volumetric elements that are intersecting but do not necessarily share a common surface. This is intended to stress the contact generation algorithms with some highly irregular cases.

The correct result of a simulation would be that the pillar stays at rest, under the pre-assumption that only correct normal information has been provided. Incorrect contact information may in fact help over-stabilize the pillar, adding invisible walls or slopes keeping individual pillar segments in place. Hence, one must be careful to rule out false positives when drawing conclusions about stability when used in actual simulation.

The Pillar case is great for testing cliff and internal vertex/edge cases. The traits of the dynamic simulation seen in Figure 5 are in our opinion an advantage for stressing the robustness of the contact generation methods over time. The Pillar case only covers a small subset of the fundamental cases. Hence, we designed simple 3D exemplars of all fundamental cases except mixed curvature cases, as shown in Table 4. These seek to cover most feature combinations that are easily set up and with sufficiently simple geometries, such that given a known velocity of the top-most object one may from intuition deduce what the correct normals should be.

Observe that the three examples of the Point-in-crack case are testing all possible convex–concave interactions, concave versus concave is not possible unless mixed curvature surfaces are included. For completeness, the table includes individual test cases for internal and cliff edges.

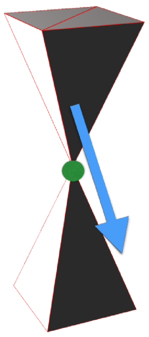
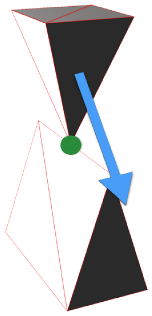
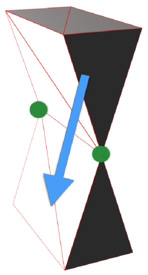
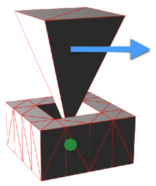
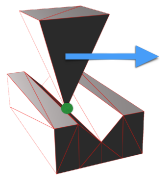
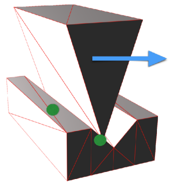
Observe that (‡) in Table 4 shows how mesh connectivity for internal vertices and edges have been designed to create two internal vertex contacts (V, V) and two internal vertex edge contacts (V, E). At each of the four contacts multiple (E, E) contacts exist that can generate incorrect normals.

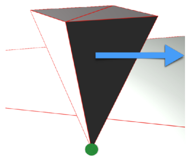
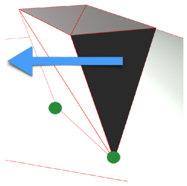
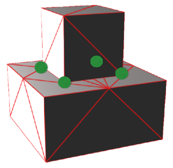
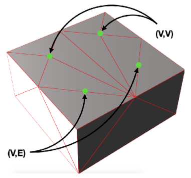
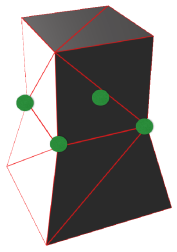
For the Pillar case, we exploited that only one unique normal direction was correct. For the fundamental cases, we have to consider that multiple normal directions may now form the desired solution space as the analysis from Table 4 shows. Hence, we suggest three measures to assess normal quality: a percentage quality measure telling how many of the generated normals belong to the set of feasible normals and a coverage percentage that shows how many of the feasible normals are obtained. Ideally, both of these measures should show 100% if all normal information is correct. Finally, we measure the excess factor of contact points by computing the multiple of the minimal known number of contact points a method has generated. A minimal information method would have an excess-factor of 1.0. Values less than 1.0 suggest inexact information, and values above 1.0 suggest over-determinacy in contact constraint information.

5.1 More Complex Examples

Studying more general examples is a challenge, because there exists no definition of a contact point in terms of a certificate or predicate that can verify if a normal or penetration distance is correctly computed. Hence, in an arbitrary simulation it is impractical at a

Table 4. The y-Axis Is Used as the World Up Vector

Two points (Strict convex cases)			Point in crack (Non-convex cases)		
					
Spikes	Spike & wedge	Wedges	Spike in hole	Spike in crack	Wedge in crack
$(\gamma,\beta,0)$ $(-\gamma,\beta,0)^{\dagger}$ $(0,\gamma,\beta)^{\dagger}$ $(0,-\gamma,\beta)^{\dagger}$	$(\gamma,\beta,0)$ $(-\gamma,\beta,0)^{\dagger}$	$(-\gamma,\beta,0)$ $(\gamma,\beta,0)^{\dagger}$	$(0,\alpha,\alpha)$ $(0,\alpha,-\alpha)$ $(\alpha,\alpha,0)$ $(-\alpha,\alpha,0)$	$(\alpha,\alpha,0)$ $(-\alpha,\alpha,0)$	$(\alpha,\alpha,0)$ $(-\alpha,\alpha,0)$
1	1	2	4	2	4

Sliding point (Flat cases)		Internal edges		Cliff edges
				
Sliding spike	Sliding wedge		Mesh Connectivity [‡]	
$(0,1,0)$	$(0,1,0)$	$(0,1,0)$	$2\times(V,V)$ & $2\times(V,E)$	$(0,1,0)$
1	2	4		4

The spike/wedges have base width of 1m and height 3m. Expected normal results from the 3D versions of the fundamental cases are listed below the case illustrations ($\alpha = \frac{\sqrt{2}}{2}$, $\beta = \frac{1}{\sqrt{10}}$, and $\gamma = \frac{3}{\sqrt{10}}$), the next row shows the minimum number of known contacts. In all cases of (†), the initial velocity of top object (illustrateda by blue arrow) is chosen such that only one unique normal should result. Location of expected contact points are shown with green circles. The symbol (‡) illustrates how mesh connectivity is designed to generate both internal edge and vertex contacts.

given instant in time to save all contact point information and then apply a predicate to test if all contacts are well defined.

From a practitioner viewpoint, one common workflow is to tune parameters interactively and then run simulations off-line. Afterwards, we import baked motion channels into a visualization tool and interactively inspect if the motions are correct. Further, we inspect energy plots and other profiling data in hope to verify correctness of the simulation. Visual clues of bad contact information are unexpected jittering, penetrations, and over-stabilization (i.e., objects stuck together). These visual clues and the point in time provide a hint for the spatial region to study the contacts in more detail. We then use our interactive tools to frame that region and study visualizations of the contact points. This workflow was applied to the simulations in Figure 7. It is quite time-consuming and requires, in our opinion, an understanding and experience of the fundamental cases to help identify the problems. Hence, it seems limited to experts to make the analysis. In Figure 7, we illustrate how our interactive tool visualizes the contact points as small blue spheres indicating the position and normals as small blue vectors. In Figure 8, a close-up is shown that helps identify cliff edges as a cause for sticky bricks.

Our didactic fundamental test cases and our Pillar case have the advantage that we by design can apply intuition that dictates unique solutions for contact positions and normals. This allows us to highly automate the process of identifying and quantifying errors in contact point generation. In a complete general setup, we have no obvious means to measure contact normal quality. Instead, we must use more controlled environments where we can apply intuition to determine the contact normals, based on a knowledge of physics. The Pillar case study is one such scenario.

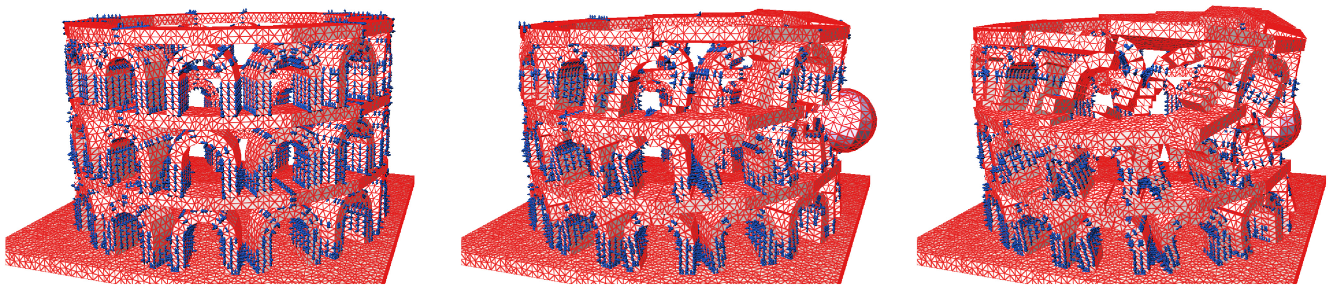


Fig. 7. A masonry structure impacting with a large rock. Observe that bricks of columns show incorrect normals (see Figure 8).

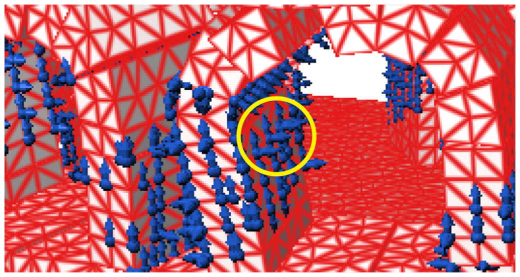


Fig. 8. A zoomed-in view on simulation step 94 of the masonry structure from Figure 7 reveals many unexpected normals pointing outward from arches and columns such as the ones encircled. These are generated by cliff edge cases and their combined effect is observed as bricks sticking together more than they really should.

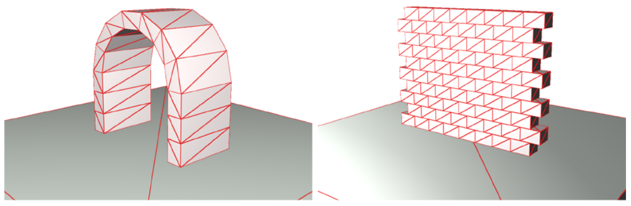


Fig. 9. The Arch case is shown on the left and the Wall case on the right. These cases have known normals from their setup.

We present two further complex cases named the Arch and the Wall cases. The cases are shown in Figure 9. For these cases, we use the same material settings as for the Pillar case.

5.1.1 The Wall Case. The Wall case is created to be 4m tall, 5m wide, and 0.5m thick. It has $L = 10$ layers of bricks, and each layer has a span of $S = 8$ bricks. The advantage of the Wall is that one can estimate the correct normal direction between any two neighboring bricks $\mathcal{A}$ and $\mathcal{B}$ only by looking at the y-coordinates $y_\mathcal{A}$ and $y_\mathcal{B}$ of their center of mass,

$$\mathbf{n}_{\text{wall}} \equiv \begin{cases} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T & \text{If } y_\mathcal{A} = y_\mathcal{B} \\ \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T & \text{Otherwise} \end{cases}. \tag{1}$$

This allows us to compute a normal quality measure by taking the absolute value of the dot product of a computed normal $\mathbf{n}$ with the given correct normal $\mathbf{n}_{\text{wall}}$. This gives a quality measure in the range of zero to 1, where the value one indicates best quality,

$$Q_{\text{wall}} \equiv |\mathbf{n} \cdot \mathbf{n}_{\text{wall}}| . \tag{2}$$

In a similar fashion, contact areas are known to be planar between bricks. Hence, one can take all contacts between two objects, project them into a 2D contact plane given by the normal from Equation (1). Next one may compute the area of the convex hull of the projected contacts and add up all those areas. That is, the total contact area is

$$A \equiv \sum_{\forall \mathcal{R}} \underbrace{\textsc{Area}(\textsc{ConvexHull}(\textsc{Projection}_{\mathbf{n}_{\text{wall}}}(\mathcal{R})))}_{\equiv A_\mathcal{R}} , \tag{3}$$

where $\mathcal{R}$ denotes the set of all contact points between two bricks $\mathcal{A}$ and $\mathcal{B}$. From the construction of the wall the correct contact area is known to be given by

$$\begin{aligned} A_{\text{wall}} \equiv S \left(\underbrace{B_D\, B_W}_{\equiv A_{\text{hor}}} \right) + L\,(S-1) \left(\underbrace{B_H\, B_D}_{\equiv A_{\text{gnd}}} \right) \\ + (L-1)\,(2\,S-1) \left(\underbrace{B_D \frac{B_W}{2}}_{\equiv A_{\text{ver}}} \right), \end{aligned} \tag{4}$$

where (B_W, B_H, B_D) are the brick width, height, and depth, respectively. A coverage value can be defined by

$$P_{\text{wall}} \equiv \frac{A}{A_{\text{wall}}}. \tag{5}$$

In our case, $A_{\text{hor}} = 0.2\text{m}^2$, $A_{\text{ver}} = 0.15625\text{m}^2$, and $A_{\text{gnd}} = 0.3125\text{m}^2$.

5.1.2 The Arch Case. The Arch is constructed to have a brick depth of 2m, there are three pillar stones and the two pillars have height of 2m. The outer and inner radius of the arch is 2m and 1.5m. The arch consists of exactly seven identical stones. The width of all stones is 0.5m. This guarantees that the expected contact area between any two brick stones should be 1m^2. By construction, the total contact area should be $A_{\text{arch}} \equiv 14$.

By construction, we may obtain a very good approximation to the true contact normal between any two neighboring stones $\mathcal{A}$ and $\mathcal{B}$ by using the unit direction vector between their center of mass positions, $\mathbf{r}_{\mathcal{A}}$ and $\mathbf{r}_{\mathcal{B}}$,

$$\mathbf{n}_{\text{arch}} \equiv \begin{cases} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T & \text{If } \mathcal{A} \text{ or } \mathcal{B} \text{ is ground} \\ \frac{\mathbf{r}_{\mathcal{B}} - \mathbf{r}_{\mathcal{A}}}{\|\mathbf{r}_{\mathcal{B}} - \mathbf{r}_{\mathcal{A}}\|} & \text{Otherwise} \end{cases}. \tag{6}$$

Observe that the ground object gives a single exception, that is trivial to handle. Knowing the true normal direction, we may reuse the concept of the quality measure from the Wall case,

$$Q_{\text{arch}} \equiv |\mathbf{n} \cdot \mathbf{n}_{\text{arch}}| \,. \tag{7}$$

The same goes for computing the contact area using Equation (3) with the Arch normal. This gives a coverage measure, too,

$$P_{\text{arch}} \equiv \frac{A}{A_{\text{arch}}} = \frac{A}{14}. \tag{8}$$

6 A FULL SPECTRUM OF LOCAL METHODS

We care about simulations with accurate display of interactions of the geometry needed in applications such as virtual prototyping, robotics, or training simulators. We seek a good representation of the true contact geometry to simulate accurately how objects tilt and tumble when they come in contact. Further, we find fixed-timestep methods attractive mostly due to performance considerations, being capable of advancing the whole simulation state in one step, avoiding possible infinite collapse and keeping well-posed problems due to the weak-form solutions. Hence exact geometric local contact point generation in the context of discrete collision detection is the scope of the domain of methods we choose to experimentally address in this work. In this context, the problem of contact point generation becomes an instantaneous geometry problem of simply computing the correct geometric information. It is the quality of this geometric information that are our focus and not the questions of overall animation quality or motion fidelity.

For all our simulations, we used a projected Gauss-Seidel method based on a proximal map formulation with isotropic Coulomb friction with coefficient of friction equal to 0.5. For full details on our simulator, we refer to Erleben (2017). To reduce possible side effects from the simulator intrinsic parts, we forced the solver to use 1,000 iterations in each simulation step. We turned off any stabilization to avoid stabilization side effects and applied density values for realistic materials, $2,000\ kg/m^3$. The pillar was designed to be roughly 3.5m in height.

Continuous collision detection has attracted much attention as a means to avoid tunneling for fast-moving objects. It is limited in the ways that a non-penetrating initial state is always assumed and further, unless an accurate event-driven scheme is created that can resolve what impact events goes first, it is guesswork in what order tunneling should be resolved. In any case, we focus on accurate simulation. This implies negligible penetrations and the speed of objects does not cause concern for tunneling. Besides our Pillar, Wall and Arch cases are scenes in equilibrium making continuous collision detection less meaningful. Even if tunneling was a concern, a CCD method often use intersection or SAT testing to generate the normals at the time of impact. Hence our comparison study is representative for contact generation as done in CCD methods as well.

Table 5. Taxonomy Overview of Our Selected Test Methods

Method Name	Taxonomy Classification
Growth	Volume-based, exact geometry, normals from closest points, continuous.
Opposing	Volume-based, approximate geometry, normals from features, discrete.
Intersection	Surface-based testing, normals from feature types, discrete.
Closest points	Surface-based testing, normals from closest points, discrete.
Vertex only	Volume-based testing, approximate geometry, normals from volumetric elements/implicit fields, discrete.
Consistent vertex	Volume-based testing, approximate geometry, normals from volumetric elements, discrete.
Surface-SAT	Surface-based testing, normals from feature types, discrete.
Volume-SAT	Volume-based testing, normals from feature types, discrete.

The table summarizes the variation of local contact point generation methods covered by our spectrum of test methods. We note that many normal feneration done by **Closest Points**, **Surface-SAT,** and **Volume-SAT** are representative to techniques used in CCD. Further, **Vertex Only** tields a normal representative of using signed distance fields.

Tetrahedral meshes are a convenient representation in our implementations, as they fit well with the application context (Schmidtke and Erleben 2017). We exploit this to quantify quality of both volume-based and surface-based contact point generation methods. The volumetric elements add robustness to inside-outside testing and overlap information.

We have selected and implemented eight different methods for generating contact points. We have attempted to cover the whole range of quality in all variations of methods from our taxonomy in Section 3. Table 5 summarizes our coverage.

Below we give relevant implementation details to ensure reproducibility by others. The **Opposing** method uses the most opposing surfaces to determine the normal direction, whereas the **Growth** method grows volumes until they touch and determines the normal from this location. The algorithms and implementation details are in Appendixes A and B.

Intersection method: We compute geometric intersections between a pair of triangles in the following way. If a vertex V is sufficiently close to a triangle face F and inside the edge-face planes of F, then a contact is generated with normal of F, position of V, and a depth equal to the signed distance of V with respect to F. After testing all combinations of (V, F), we proceed to test all (E, E) combinations. If edges are non-parallel and their cross-product does not form a SAT, then we test if the orthogonal projection of the closest points between the infinite lines defined by the edges are actually lying on the edges. If the test passes, then we generate a contact with the edge cross-product as normal and depth equal to the negative overlap measure.

Closest points method: We erode objects by the user-specified ε when using the closest points method to provide a suitable collision envelope that will not artificial enlarge object geometries.

Whenever the closest points between a pair of features are within the user specified threshold, $2\,\varepsilon$, then the average point of these generates the actual contact point and the separation vector between the closest points is used to find the unit contact normal. The length of the separation vector is used as the depth measure. One could extend this approach to deal with penetrations larger than $2\,\varepsilon$ by determining (E, F) contacts where E penetrates F and detect (V, F) contacts where the clipped V against F is beneath the face plane of F. For our accurate simulations, the penetrations should be negligible. The idealized setup we use implies that initially closest points are all separated by 2ε within numerical precision. One could argue that this is a rather synthetic, too-perfect setup. However, it serves the purpose of providing the most difficult case to test on. We have found that choosing ε is non-trivial and can be dependent on the configuration one is simulating. The erosion corresponds to a mathematical opening and hence yield a smooth version of the polygonal surface mesh.

In our **Vertex only** method, we test vertices for inclusion in tetrahedral volumes. If a vertex is found inside, then we generate a contact at the vertex position and using the normal of the closest surface face of the tetrahedron and a depth estimate equal to the distance to the face. Interior faces of the mesh are ignored.

In the implementation of the **Consistent vertex** method, we find all vertices inside a tetrahedron, and then we determine the surface triangle face of the tetrahedron with the closest edge intersection point and use this unit face normal as the contact normal. The contact position is set equal to the vertex position and depth is given by the distance to the chosen triangle face. We do not need to consider deep penetrations of vertices lying far inside other objects due to using an accurate simulation.

For our **Volume-SAT** method implementation, we determine the separation axis with minimum overlap between pairs of tetrahedra. Axes generated from faces, corresponds to (V, F), are given precedence over the (E, E) type of axes. Once an axis has been determined, we find all intersection points and use their average position to define a contact plane. Here after all intersection points are projected to the contact plane and each are reported as a contact point using the unit minimum overlap separation axis vector as contact normal and the minimum overlap as the depth estimate. Post-filtering is done to ensure that all reported generated contacts have unique positions within numerical precision.

The **Surface-SAT** method implementation works like the **Volume-SAT** method except with the condition that we restrict the search for a contact normal to only consider axes generated by features that are on the true surface of the object. This would correspond to using a triangle surface mesh.

In this work, we are more concerned with exact geometry and good normal quality and less with performance.

7 QUALITATIVE RESULTS

In this section, we present qualitative results from visual inspection of the contact points generated for the Pillar case. The surface meshes are displayed by red wireframe, and contact points are drawn as a blue sphere to mark the position and a small blue arrow to show the contact normal direction. The Pillar scene is a static scene with no initial penetrations. Hence, no visualization of the contact point depths was done. In all figures, we have stopped our simulation and interactively moved pillar stones out of the way to better observe the generated contacts. This is necessary to inspect contacts generated inside the interior of the contact regions.

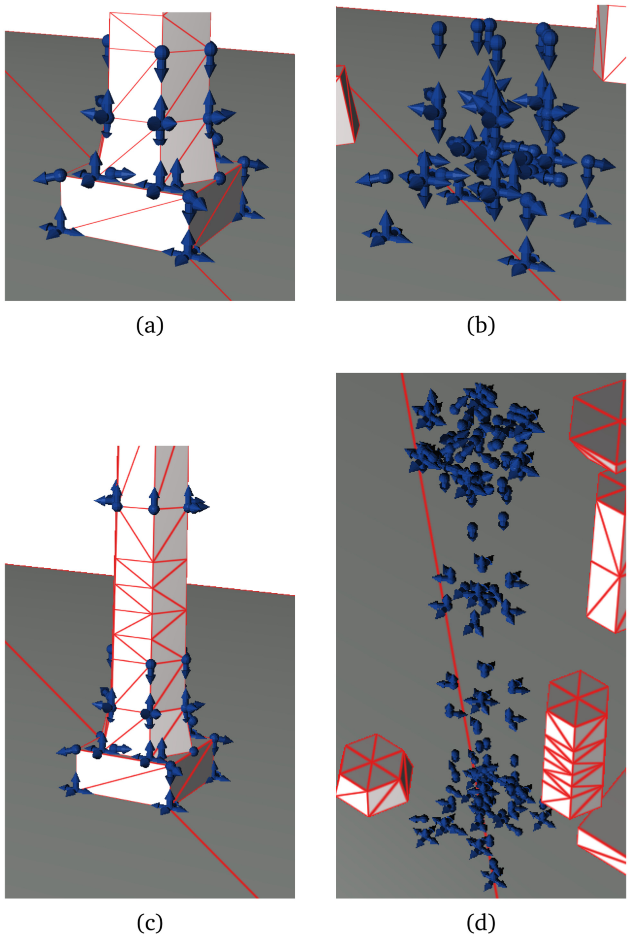


Fig. 10. **Intersection**-based contact point generation for the Pillar test case. Observe the framelike normals point in orthogonal directions.

7.1 The Intersection Method

Figure 10 shows the quality in contact points when using **Intersection** method for generating normal information. The **Intersection** method is very sensitive towards numerical thresholds. This becomes noticeable if one runs the simulation. At some point, the buildup of numerical errors can cause strange contact points originating from (E, E) cases. One may observe contact points floating at unexpected non-contact regions as shown in Figure 11(b). Objects need to sink within numerical threshold before contacts are generated; hence, the method can suffer slightly from sagging. Comparing Figure 11(a) with (c), the sagging motion is observed. Another contribution to sagging motion is inaccurate contact forces too weak to prevent penetration. This argument holds for all other methods, too. Even worse, if there are large aspect ratios between surface triangles and small penetrations occur then

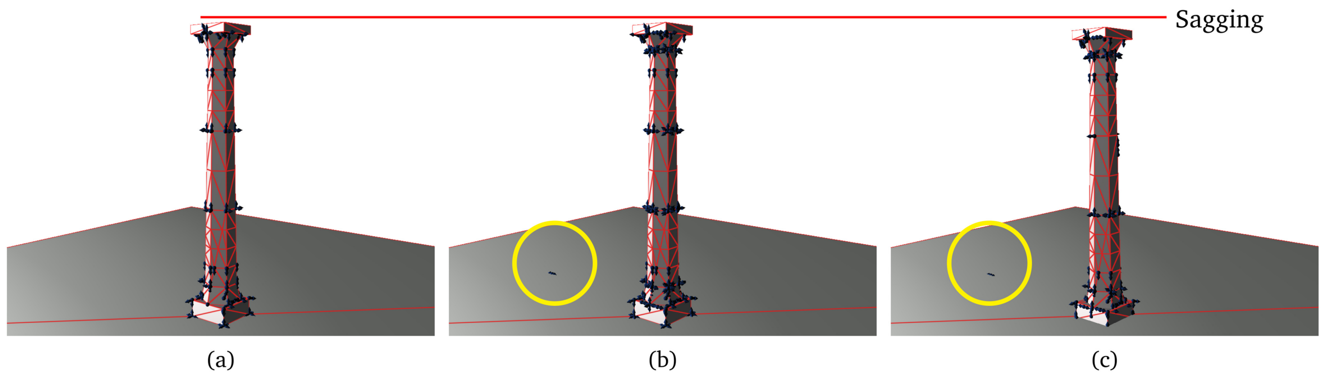
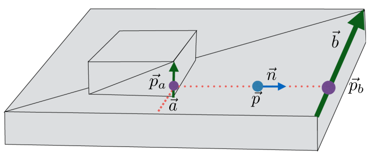


Fig. 11. From left to right, simulation steps 3, 13, and 500 of a simulation using the **Intersection** method is shown. Observe sensitivity to numerical inaccuracies caused by simulation. In step 13, an unexpected contact to the far left of the pillar with normal horizontal to the ground appears (yellow circle). Comparing the first simulation step against step 500 reveals subtle sagging motion.

(E, E) contacts can lead to both incorrect normals and positions as explained in detail later. The **Volume-SAT** and **Surface-SAT** methods do not suffer from this as they filter out the minimum overlap pair of features. The naive **Intersection** would simply just test all (V, F) and (E, E) contacts and hence would not pick a single face-normal case over an edge-edge normal case.



Small penetrations occur due to sagging. When this occurs (E, E) contacts in the **Intersection** method can generate contact points outside the contact region. This is illustrated on the right. Here a small box is sagging into a larger box. When testing edge a and edge b (green arrows) one will find the closest points $\mathbf{p}_a$ and $\mathbf{p}_b$ these result in contact position outside the intersection point set of the small and large boxes. Observe the mid-point (blue circle) are not even close to the small box. Further the closest points give a normal direction pointing to the right. That is in the wrong direction.

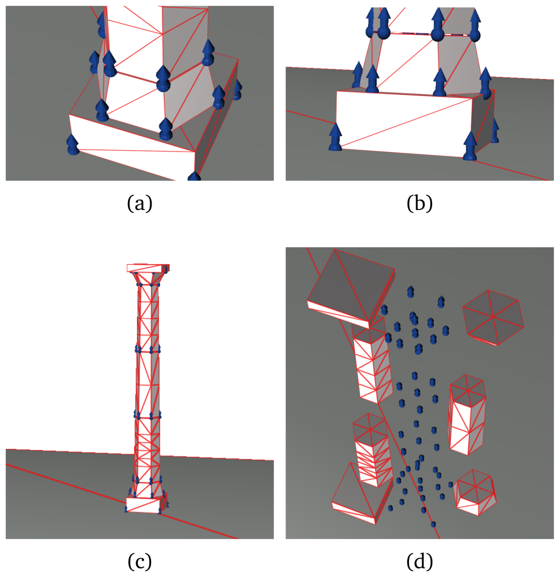


Fig. 12. **Closest points**–based contact point generation for the Pillar test case. Observe the apparent perfect result due to perfect accurately aligned geometries.

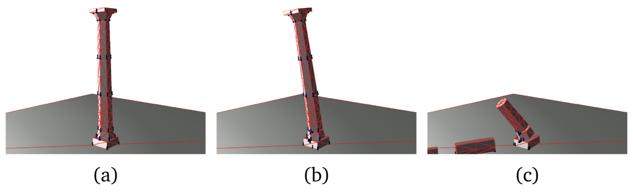


Fig. 13. **Closest points** is sensitive towards numerical errors and eventually cause an unexpected collapse of the Pillar case.

7.2 The Closest Points Method

Figure 12 shows results for the **Closest points** method. The results are quite impressive. The method has issues dealing with objects sinking deep into their envelopes. This may easily occur during simulations as seen in our supplementary video. The static test case benefits tremendously from us being able to setup the geometry in a perfect starting state.

During simulation, one will observe that seemingly perfect results by **Closest points** are very sensitive to the accumulation of simulation errors. As shown in Figure 13, the simulation will break down over sufficiently long time. The major cause to the breakdown is parameter dependencies.

During simulation, sagging will occur. Due to the iterative nature of our solver, this may not occur symmetrically, and slight numerical perturbations can be seen in this kind of solvers. Hence, some contacts might numerically break or sink too deep into their proximity zones. When this occurs, the contacts vanish and the support polygons needed for a stable simulation disappear. Another drawback of this method can be seen in Figure 13(c), where the falling objects have observable penetrations. This is amplified due to us using discrete timestepping and not using stabilization terms in the simulator.

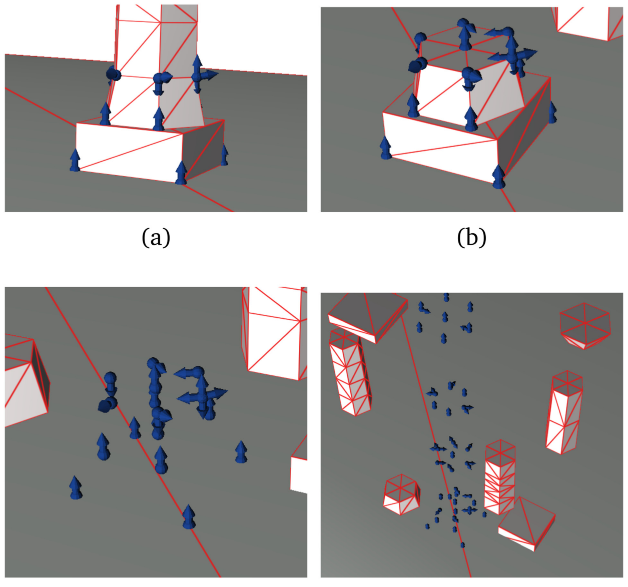


Fig. 14. **Vertex only** uses vertex inside only–based contact point generation for the Pillar case. Observe that few contacts are present and that cliff edges prove to be a problem.

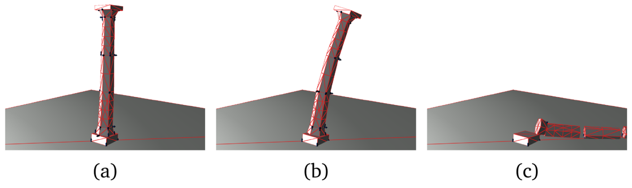
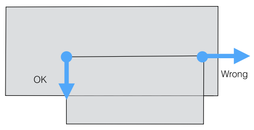


Fig. 15. Over time the **Vertex only** method is sensitive towards numerical errors and cause an unexpected collapse of the Pillar test case.

7.3 Vertex Only and Consistent Vertex methods

Using **Vertex only** to generate contacts will give the results shown in Figure 14. This approach suffers from vertices being locally attracted to nearby surfaces.

The artifact is illustrated in more details on the right. Here a small box is deeply penetrating a large box. The corners generate contact points and choosing the normal direction from closest surface point on the large box lead to unexpected normal direction for the right most corner point of the small box.



The artifact can have severe consequences when considering side effects of stabilization caused by discontinuous normals. Imagine drift errors build up and when penetrations are too deep the normal direction jumps to a different direction.

Choosing the closest surface for the vertices based on edge-penetration information leads to a more consistent method with results as shown in Figure 16.

Neither **Vertex only** nor **Consistent vertex** give the full exact geometric information of the contact regions, as they only see contacts when vertices are touching/penetrating the surface. If only touching the true surface of the object, then the normals tend to be good. Even for slight penetrations the true surface normals can be quite bad, and for large penetrations inner surfaces give no clue about the true real outer surface of the object.

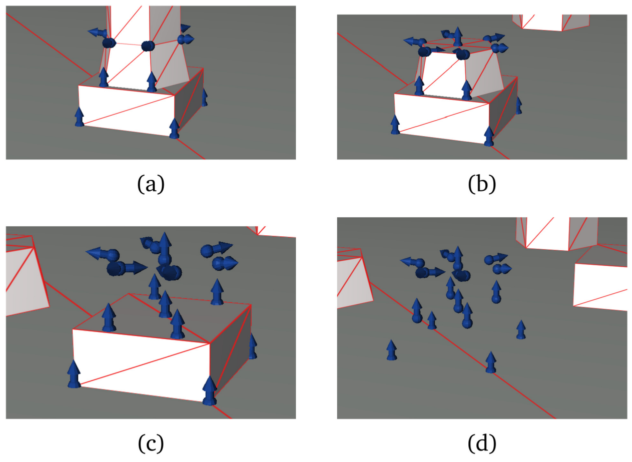


Fig. 16. Results from **Consistent vertex**–based contact point generation for the Pillar test case. Observe that the more consistent normal information does not resolve cliff edges.

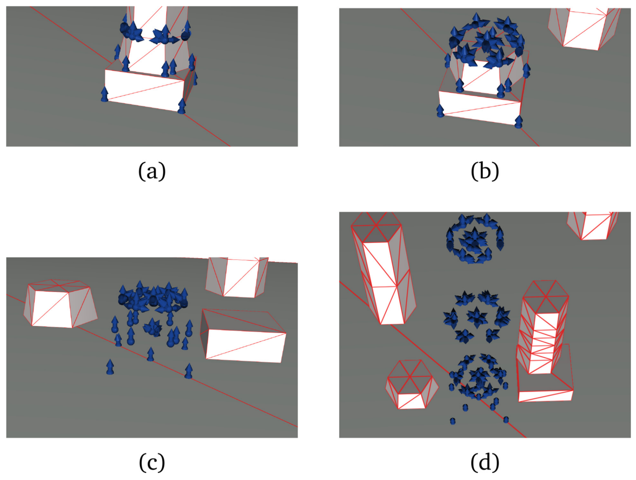


Fig. 17. Results from the **Volume-SAT**–based contact point generation for the Pillar test case. Observe the morning starlike shape of normals appearing at (V, V) type of contacts.

We found that **Consistent vertex** gave us a stable simulation result, whereas **Vertex only** resulted in a collapse shown in Figure 15.

7.4 Surface and Volume SAT Methods

Next, we study quality of the SAT-based approaches. The morning star shape effect seen in Figure 17 is caused when (E, E) type separation axes are yielding slightly lower overlap than any of the (V, F) type separation axes and from (E, E) pairs of contacting elements that do not capture the real surface of the objects.

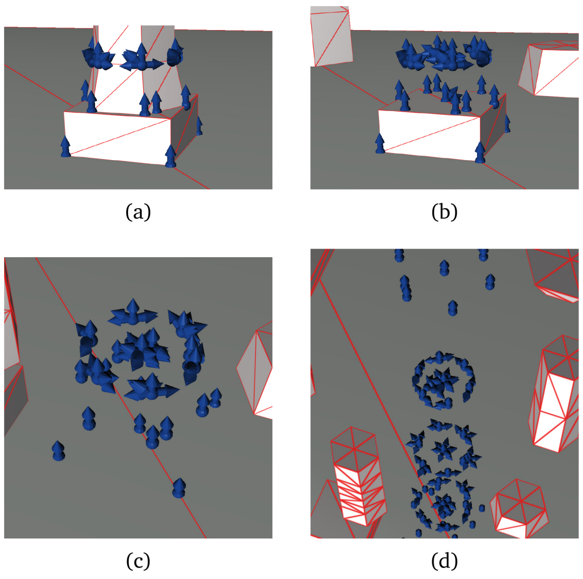
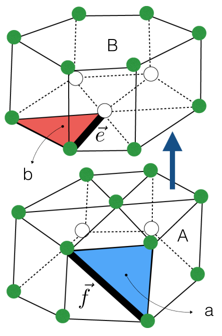


Fig. 18. Quality of **Surface-SAT**–based contact point generation for the Pillar case. No noticeable difference between **Volume-SAT** and **Surface-SAT** can be seen.

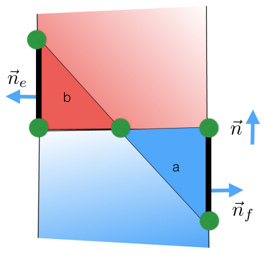


On the right, we illustrate the problem caused by using local elements. Here we observe that local element-wise (E, E) pairs may not reflect the true shape of the objects. The illustration shows how the internal edges e and f generates a SAT for the local elements a and b. The resulting normal from the SAT direction does not agree with expected upward pointing normal direction. Such cases cause the morning star effect for **Volume – SAT** and the fan-shape for **Surface – SAT**.

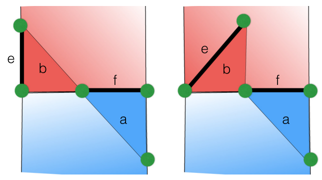
Observe that the artifact is caused by internal edges of the mesh. Hence, one remedy we did not explore as it required caching of further data could be to pre-preprocess the mesh before a query and flag internal edges so they could be skipped during testing. Further, treating elements locally without knowing more is common place in physics engines such as Bullet (Coumans 2010), our approach is hence representable of industry standard.

Restricting the SAT axes to only be feasible if generated by true surface features does not help much as observed by comparing Figure 17 and 18. The added normal information will restrict objects from sliding causing an artificially too stable simulation. Neither **Volume-SAT** or **Surface-SAT** collapse when simulating. This is in fact a false positive result.

Notice that **Surface-SAT** generates a morning-star shape effect at the center points of the circular pillar. This is unexpected as only parallel surface triangles will meet at this point. Hence, all face normals are in the desired up–down direction and all edge-cross products will be in the up–down direction. The undesired directions are generated due to us using the tetrahedral volumetric elements to determine the minimum overlapping axes (as is common practice) and the fact that we deliberately generated a coarse mesh to provoke extreme cases.



The illustration on the right shows a 2D cross-sectional view of two touching objects. Observe how two local elements a and b each can generate SAT directions from local face normals $\mathbf{n}_e$ and $\mathbf{n}_f$ that will be orthogonal to the expected upward normal direction. The two local elements are in a point-to-point touching state. These types of incorrect normals account for the horizontal normals in Figures 17 and 18.

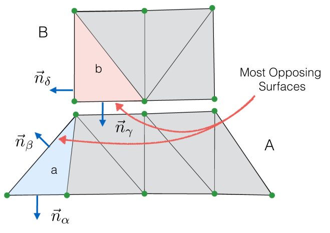


Not only face normals contribute to SAT directions, but also edge-edge cross products, too, are a source to incorrect normals as shown in the illustration on the left. Again, we are looking at a cross-sectional view of two objects in contact. Here cross-products of the thick edges e and f will give directions that are different from the expected upward direction. The right case illustrate that cross-products might not be orthogonal but can be tilted, depending on the direction of the edge e. Observe the tilted normals in Figures 17 and 18.

One remedy may be to only use triangle surface elements for detecting the minimum overlap axis. This will essentially convert the approach to the equivalent of the **Intersection** method. Keeping volumetric elements allow us to robustly compute inside outside information, this advantage is lost in a pure surface-based approach.

7.5 Opposing and Growth Methods

Figure 19 shows the qualitative results obtained by picking the contact normal from the most opposing surface faces only. As observed, the **Opposing** method has issues with cliff edges. This is due to the fact that selecting the most opposing surfaces for local elements in a degenerate contact lead to wrong assumptions.



This is illustrated on the right using a cross-sectional view of two objects $\mathcal{A}$ and $\mathcal{B}$. The most opposing surfaces of the local elements a and b (blue and red) suggest that $\mathbf{n}_\beta$ should be part of generating the normal. Notice that a and b are locally in a point to point scenario.

As seen in Figure 20, the growth-based approach tends to transform (F, F) contacts into (V, E) contacts and get incorrect normals in those cases. For many (V, V) types of contacts the shrinking tilts the normals a little along the center-line between the two objects. Hence, the **Growth** model works best for elements where the center line is orthogonal to the desired contact normal line. The most-opposing surface strategy tends to be robust towards this.

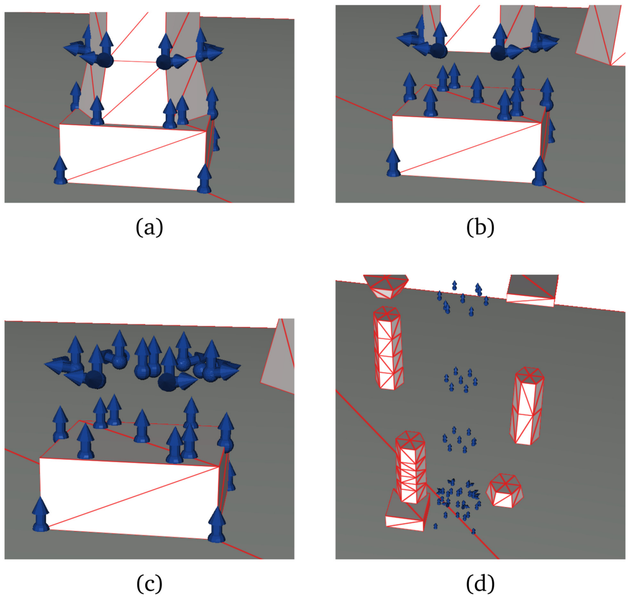


Fig. 19. Most **Opposing** surface–based contact point generation for the Pillar test case. Observe that only a few cliff edges have incorrect normals, and all other contacts are as we expect.

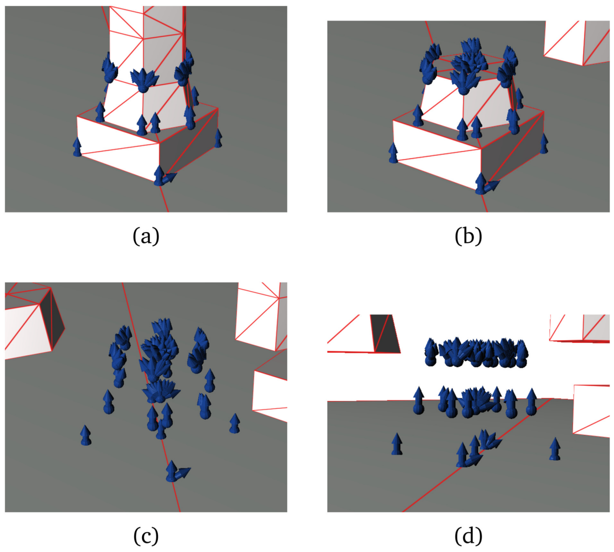
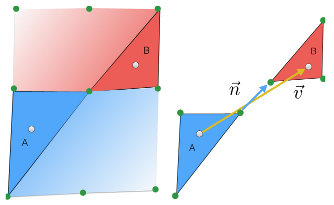


Fig. 20. **Growth**-based contact point generation for Pillar test case. Observe the fanlike normals appearing at (V, V) type of contacts and the non-intuitive normals at (V, E) contacts. Notice that cliff edges do not have outward point normals but upward pointing fans instead.

The fanlike structures seen for growth model is caused by a tilting artifact for (V, V) type of contacts as explained next. The fan-shape is a direct result of how the tetrahedral mesh is tessellated. Hence, growth model seems to deal better with cliff edges and internal edges creating fans at those places.

The illustrations on the right explain the artifact of the **Growth** method. The first drawing shows a cross-section view of red and blue objects in contact. Two local elements A and B are in a point-to-point contact. The next drawing shows down scaling of the elements. Observe that the **Growth** method give a sensible normal for the two down-scaled elements. Comparing the down-scaled drawing with the non-scaled drawing suggests that the normal direction will cause a fan-like shape of generated normals at the common surface point.



In conclusion, both the **Growth** and **Opposing** methods can be fooled to provide incorrect normal information. When used for simulation **Growth** and **Opposing** both result in a static pillar. We interpret this as a gradually better result than **Volume-SAT** or **Surface-SAT**, as there is less incorrect normal information compared to the fan and morning-star shape normals.

7.6 Qualitative Results of The Fundamental Cases

We continue the qualitative analysis with the fundamental cases from Table 4. We summarize our observations from the supplementary video. **Intersection**, **Closest points**, and **Growth** provide good temporal results for all three sub-cases of the **Point in crack** case. Both SAT methods come close but end in false-positive results for the **Spike in hole** case. **Vertex only** and **Consistent vertex** fail on all accounts, and **Growth** are too stable on **Spike and spike** and **Spike in hole** cases. All methods show reasonable expected behavior for the **Sliding point** case. Although the coarse meshing challenges the vertex-based methods when the **Spike & wedge** come in contact. We note that the **Intersection** method artificially interlocks the wedge and spike edges near the end of the motion. The SAT methods behave quite well for all three sub-cases of the **Two points** case, whereas the **Vertex only** and **Consistent vertex** methods fail on all accounts. The **Growth** method comes very close to the SAT methods. The **Closest points** method suffers from artificially locking leaving the **Spike & wedge** case hanging in mid-air. The **Opposing** method shows a too stable **Spikes** case.

8 QUANTITATIVE RESULTS

We study more quantitative measures starting with the Pillar case, then we study the Arch and the Wall cases before looking at the fundamental cases. Last, we will study effects of mesh resolution, timestep sizes and perturbation.

8.1 The Pillar Case

Table 6 and 7 and Figure 21 show the quality measure of the generated normals for each method. Apparently, **Closest points** appear to be ideal, although its ε parameter and threshold testing make it very sensitive. As a consequence, this requires intelligent parameter selection. We exploited that we have a specific Pillar test case that we could apply prior knowledge about to select ε to a meaningful scale compared to the object sizes. Further, we exploited our knowledge of this to set up the pillar in an ideal initial state. Overall, for actual simulation, the method did not show robust results. Looking for a secondary method that provided robust simulation results although with lesser quality of contact normals, we find

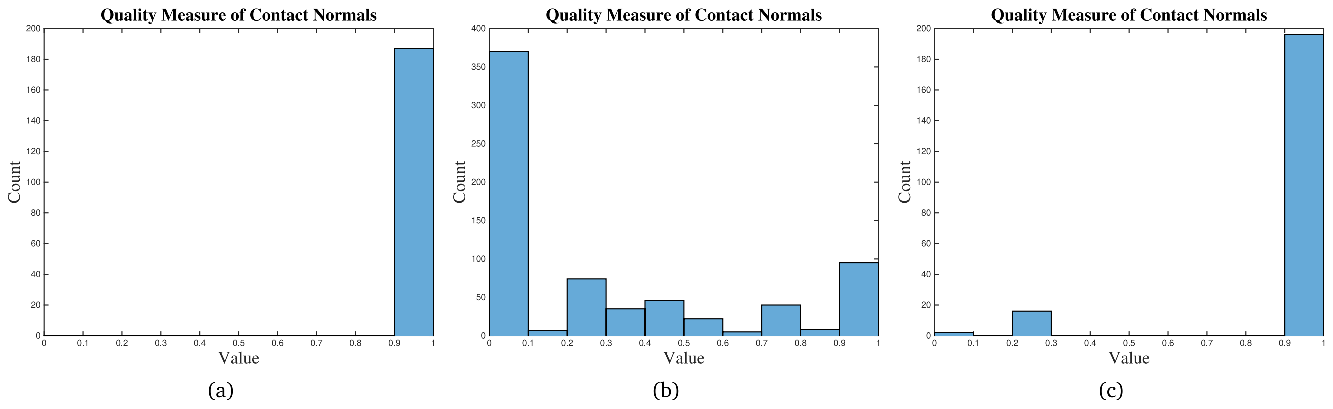


Fig. 21. Normal quality measure for the Pillar test for selected contact point generation methods: **Closest points** (a), **Intersection** (b), and **Opposing** (c). Due to space considerations, these three plots were selected to illustrate the variation.

Table 6. Contact Statistics for the Pillar Case

Method Name	Mean (#)	Min (#)	Max (#)	Std. Dev.
Closest points	267.11	102	375	59.40
Intersection	867.97	504	1332	163.07
Vertex only	65.94	6	119	19.62
Consistent vertex	93.26	32	113	10.44
Volume-SAT	1248.52	339	1399	69.37
Surface-SAT	1186.44	333	1362	89.06
Opposing	403.12	117	459	24.83
Growth	1257.45	400	1539	162.75

Showing Mean, Minimum, Maximum and Standard Deviation of the Contact Count for the First 500 Simulation Steps.

Table 7. Overall Quality Measures for the Pillar Case from Simulation Step 3, Showing Percentage of Contacts with Correct Normals, Number of Unique Contacts, and the Multiple-factor of the Minimal Contact Point Count

Method Name	Quality (%)	Contacts (#)	Excess (#)
Closest points	97.9	188	4.7
Intersection	15.4	722	18.1
Vertex only	60.8	51	1.3
Consistent vertex	35.3	34	0.8
Volume-SAT	20.9	540	13.5
Surface-SAT	52.9	607	15.2
Opposing	91.6	214	5.3
Growth	24.6	622	15.6

An exact geometric optimal method would have quality measure 100%, count 40 (for the Pillar Test Case), and an Excess-factor of 1.

Opposing to be near optimal. The bad values in the diagram are due to the cliff edge issue explained in Section 7.5.

In all the Pillar tests, we applied a post filtering of contact points that removed redundant contact information. That is contact points and normals that were the same within numerical precision. This was to validate how many unique contact points each method generated. Table 6 presents counts for each method for the first 500 simulation steps. Observe from Figure 22 that the number of contacts vary greatly from method to method. Simulation results are very sensitive towards the quality in the contact points. Hence, the number of contacts vary greatly over the duration of the simulations. A near-optimal result would suggest that contacts do not vary at all between simulation steps.

As expected, the vertex-based approaches result in far less contacts (65–90 contacts). The **Volume-SAT**, **Surface-SAT**, and **Growth** are among the highest, of approximately 1, 200 contacts. **Opposing** is on the level of 400 contacts, slightly above **Closest points** with an order of approximately 270 contacts. **Intersection** is the most erratic method of all and varies from low 500 up to 1, 300 contacts. By comparing contact counts with Figure 6, we observe that in all cases the contact count numbers are quite large for the Pillar case. The reasons for our contact count to exceed the ideal counts are as follows: imprecision when computing the same contact points with respect to different pairs of local elements. Hence, the imprecision is too large for our post filtering to identify the redundant contacts. Generation of false normal information leads to a significant number of contacts with incorrect contact information. This is a major source to increased contact numbers. Another major source is redundant generation of the same contact point by different pairs of local elements. Our post-filtering removes some of this redundancy, but testing on numerical values is not very robust. In the literature, the redundant generation of contacts from different local elements has been addressed (Curtis et al. 2008; Tang et al. 2008). The attribution of element-wise tests that generated incorrect normal information has not been addressed as a source to generating too many contact points.

8.2 The Wall Case

The **Closest points** and **Opposing** methods appear to be good candidates for the Pillar case. Hence, we wish to examine these for other test scenes too. Figure 23 shows a visualization of the contact points computed by the two methods. We observe a more regular structure for the **Opposing** method although it is clear that in simulation step 3 some contact regions are no longer found due to propagation of numerical round off and precision errors. The **Closest points** method appears more complete but shows incorrect normal directions.

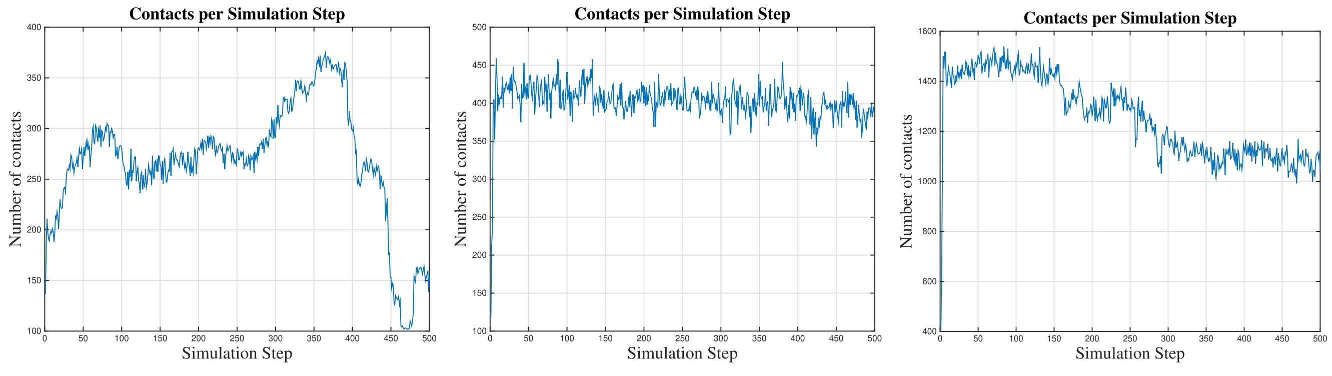


Fig. 22. Number of contacts per simulation step for selected contact point generation methods. **Closest points** (a), **Opposing** (b), and **Growth** (c). Due to space considerations, these three plots were selected to illustrate the variation.

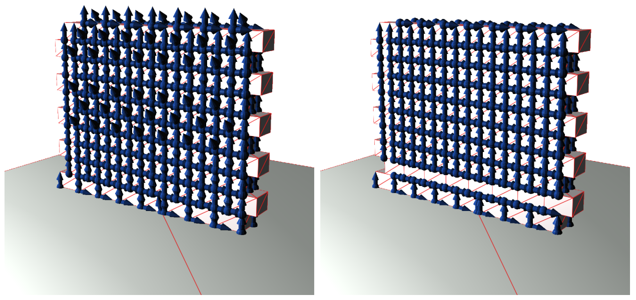


Fig. 23. Visualization of contact points for the Wall case. Top shows results of simulation step 3 for **Closest points** and bottom for **Opposing**.

Figure 24 shows that 20% of the normals for the **Closest points** method is incorrect, whereas not a single normal is incorrect with the **Opposing** method.

From Figure 25 we expect to see three plateaus corresponding to the values $A_{\text{hor}} = 0.2\,m^2$, $A_{\text{ver}} = 0.15625\,m^2$, and $A_{\text{gnd}} = 0.3125\,m^2$ (see Section 5.1.1). It is clear that for **Closest points** we observe far too many smaller area values, suggesting we do not have full coverage of the contact areas. For **Opposing**, we only observe a single incorrect area value. The coverage values rounded to two decimals for the **Opposing** method is $P_{\text{wall}} = 0.93$ and for **Closest points** it is $P_{\text{wall}} = 0.89$.

Not surprisingly, the **Opposing** method deals nicely with the very regular wall structure and **Closest points** appears to have less normal and coverage quality.

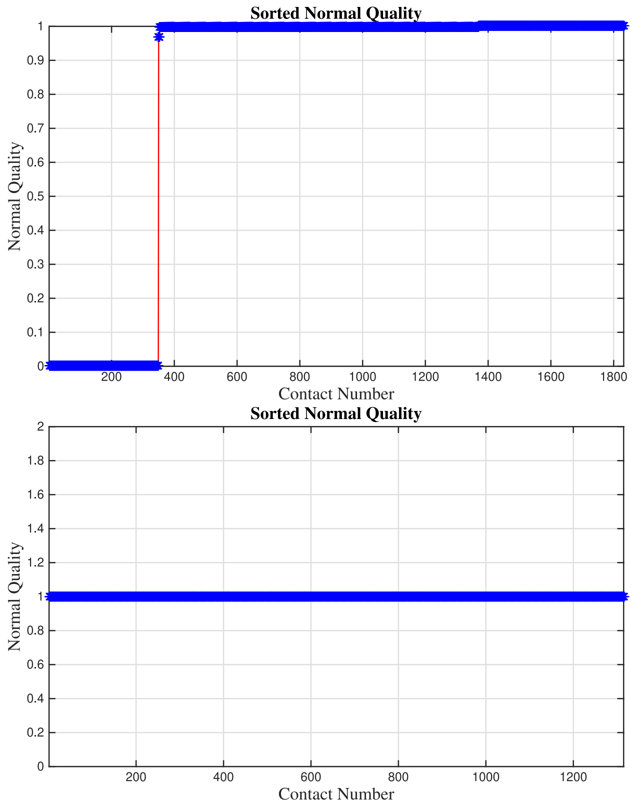


Fig. 24. Sorted plots of Q_{wall}-normal quality measure for the Wall case as defined by Equation (2). Top shows results of simulation step 3 for **Closest points** and bottom for **Opposing**. The value 1 is optimal.

8.3 The Arch Case

Figure 26 shows contact points for the Arch case. The figure suggests that the arch and key stones only touch near the inner parts of the arch for the **Closest points** method. This is expected as stone bricks initially made smaller by moving vertices along their inward vertex normals by a distance of $\varepsilon = 0.01$m. This is to ensure sufficient separation for the **Closest points** method to have a separation vector. This implies the stones are sinking into each other's collision envelopes in the first few steps of the simulation. We therefore studied a case where we artificially created overlapping stones by displacing along the outward unit normal. As Figure 26 suggests this, identifies far more contact points at the missing areas. In comparison, the **Opposing** method appears to capture the areas much more easily.

In Figure 27, we show sorted plots of the quality measure defined by Equation (7). Here we notice a tendency for **Closest points** to have a much smaller fraction of approximately 30% bad normals,

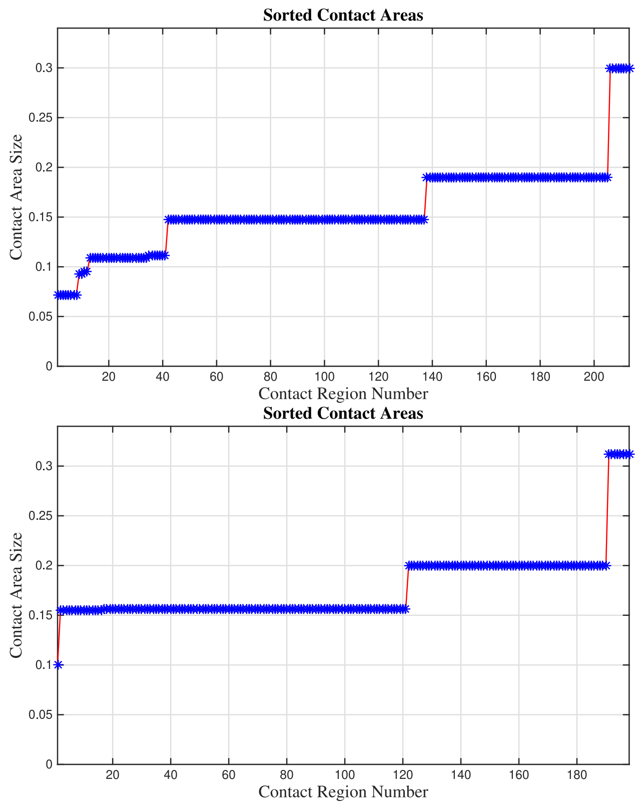


Fig. 25. Sorted plots of contact areas $A_{\mathcal{R}}$ for the Wall case as defined by Equation (3). Top shows results of simulation step 3 for **Closest points** and bottom for **Opposing**. There should be three plateaus of values as explained in Section 5.1.1.

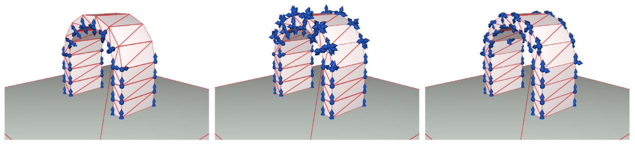


Fig. 26. Visualization of contact points for simulation step 3 of the Arch case. From left to right, we see **Closest points** first with no initial penetrations, then with artificially created small penetrations and finally **Opposing** with no penetrations. Observe **Closest points** is very sensitive.

whereas the **Opposing** method is closer to 50% incorrect normals. We notice that the **Opposing** method generated about five times more contact points than the **Closest points** method and this is after we have done our post-filtering step to only keep sufficiently unique contact points.

Figure 28 shows sorted plots of the contact areas as defined by Equation (3). This shows that the **Opposing** method has a better coverage than the **Closest points** method. The coverage values rounded to two decimals for **Closest points** is $P_{\text{arch}} = 0.64$ and for **Opposing**$P_{\text{arch}} = 0.90$. Hence, in this case, **Closest points** generates better normals and fewer contacts than **Opposing** but has less coverage of the contact area.

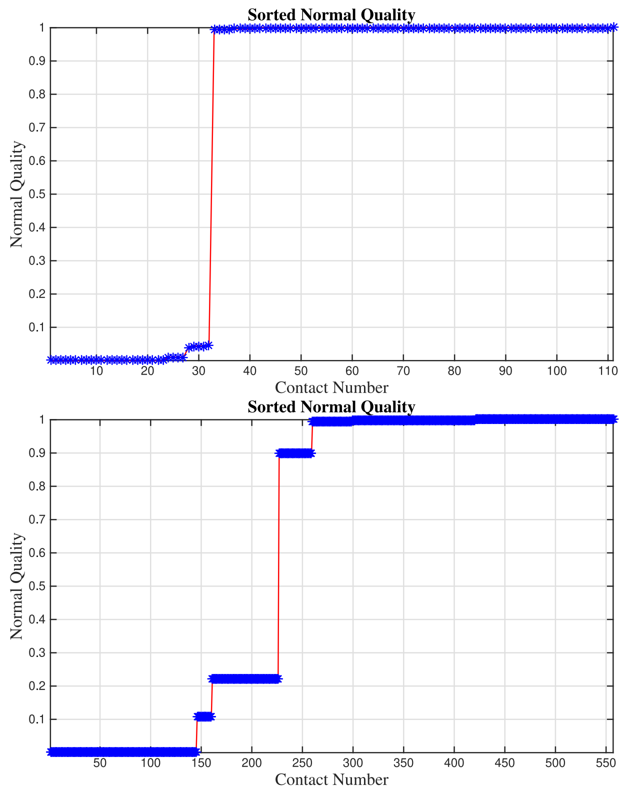


Fig. 27. Sorted plots of Q_{arch}-normal quality measure for the Arch case as defined by Equation (7). Top shows results of simulation step 3 for **Closest points** and bottom for **Opposing**. The value 1 is best.

8.4 The Fundamental Cases

Table 8 shows the quality measures for the cases in Table 4. For the **Point in crack** case, we observe from the first three columns that **Closest points** are optimal, followed by **Growth** as a second choice, and third comes SAT-based methods. The **Sliding point** case, columns four and five, suggest that **Closest points** is the best and that **Growth**, **Opposing** and SAT-based methods give the same quality but with more redundant information. **Intersection** does the worst job here. The **Two points** cases (the last three columns) show that **Closest points** are non-optimal and the **Opposing** method does better. None of the SAT-based approaches do well here. One should remark that the case quality measurements illustrate the quality one may obtain with an ideal starting state.

8.5 Effects of Perturbation

Practitioners often advocate adding a little noise to a simulation to break perfect symmetries or eliminate chances of singularities. The idea can be applied to contact point generation as well. Adding a small random displacement to the objects in a scene will likely break all (V, V) type of contacts and remove all parallel (E, E) type of contacts. Intuitively speaking, such singular contact types will change into (V, F) type of contacts or (E, F) type of contacts. The added benefit is that the indefiniteness will hopefully be less likely to occur.

Table 8. Quantitive Results of the Cases in Table 4

Method Name	Spike in crack	Spike in hole	Wedge in crack	Sliding spike	Sliding wedge	Spikes	Spike and wedge	Wedges	Cliff edges	Internal edges
	Coverage (%)									
Closest points	100.0	100.0	100.0	100.0	100.0	0.0	0.0	0.0	100.0	100.0
Intersection	0.0	0.0	0.0	0.0	0.0	100.0	100.0	0.0	100.0	100.0
Vertex only	50.0	0.0	50.0	100.0	100.0	0.0	0.0	0.0	100.0	100.0
Consistent vertex	50.0	0.0	50.0	100.0	100.0	0.0	100.0	100.0	100.0	100.0
Volume-SAT	50.0	25.0	50.0	100.0	100.0	0.0	0.0	0.0	100.0	100.0
Surface-SAT	50.0	25.0	50.0	100.0	100.0	0.0	0.0	0.0	100.0	100.0
Growth	50.0	25.0	50.0	100.0	100.0	0.0	0.0	0.0	100.0	100.0
Opposing	0.0	0.0	0.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Quality (%)									
Closest points	100.0	100.0	100.0	100.0	100.0	0.0	0.0	0.0	62.5	58.3
Intersection	0.0	0.0	0.0	0.0	0.0	10.0	33.3	0.0	55.6	50.0
Vertex only	100.0	0.0	100.0	100.0	100.0	0.0	0.0	0.0	60.0	83.3
Consistent vertex	100.0	0.0	100.0	100.0	100.0	0.0	50.0	25.0	50.0	100.0
Volume-SAT	100.0	100.0	100.0	100.0	100.0	0.0	0.0	0.0	61.5	65.2
Surface-SAT	100.0	100.0	100.0	100.0	100.0	0.0	0.0	0.0	70.0	83.3
Growth	100.0	100.0	100.0	100.0	100.0	0.0	0.0	0.0	37.5	63.0
Opposing	0.0	0.0	0.0	100.0	100.0	22.2	55.6	47.4	73.5.0	93.3
	Excess (%)									
Closest points	1.0	1.0	1.0	1.0	1.0	10.0	3.0	3.5	2.0	3.0
Intersection	0.5	1.2	1.5	3.0	0.5	20.0	6.0	3.5	2.2	3.0
Vertex only	0.5	(*)	0.5	1.0	0.5	2.0	1.0	1.0	1.2	1.5
Consistent vertex	0.5	(*)	0.5	1.0	0.5	4.0	2.0	2.0	1.5	1.0
Volume-SAT	2.0	1.8	2.2	5.0	1.0	14.0	9.0	8.5	3.2	5.8
Surface-SAT	2.0	1.8	2.2	5.0	1.0	14.0	9.0	9.0	2.5	4.5
Growth	2.0	2.2	1.5	5.0	1.0	15.0	11.0	10.5	6.0	6.8
Opposing	3.0	2.2	1.8	5.0	1.0	18.0	9.0	9.5	8.5	3.8

In (*) no contacts were found due to the spike vertex moving downward along a shared face of a pair of tetrahedra. Hence, the vertex is never really inside any tetrahedra.

We have chosen to study how the normal quality of the **Opposing**, **Growth**, and **Closest points** methods change when we increase the amount of random perturbation applied to the x and y center of mass positions in the Pillar case. We use the random displacements of magnitude 0.01m, 0.02m, and 0.03m. The displacement has to be large enough to be able to break (V, V) contacts but still not be visible to the common observer; 0.01m is on the scale of the collision envelope used for the **Closest points** method and at 0.03m displacements are visible to us. Figure 29 shows the contact points generated at simulation step number 3 for the three methods when using the highest noise setting.

We compare the sorted normal quality plots of the Pillar case in Figure 30. The plots show that all methods benefit from larger perturbations. We observe that the **Growth** method outperforms any of the others in terms of getting better quality. It is a close call between **Opposing** and **Closest points**. Strictly speaking, **Opposing** has more normals with quality one for all perturbation levels. Although **Closest points** shows much larger variation in quality for normals with quality less than 1.

Adding perturbations has benefits in terms of increasing quality. The drawbacks are that there is no guarantee that all bad normals are removed. Further, adding perturbations without end-user knowing can have fatal consequences when used in digital prototyping.

8.6 Effects of Mesh Refinement

For tetramesh creation, we use TetGen with a default setting of quality ratio 2.0, no maximum volume setting, and suppressing splitting of surface faces. To study how normal quality changes with finer mesh resolution, we forced TetGen to generate much finer meshes by setting the quality ratio to 1.015, allowing surface face splitting, and adding a maximum volume constraint on non-conical shapes of 0.015 and for conical shapes we used a much lower setting of 0.00015.

Figure 31 shows the refined mesh examples of simulation step 3 for the Pillar case. One can see the increase in number of generated contacts. It is visually clear that for the three local methods shown

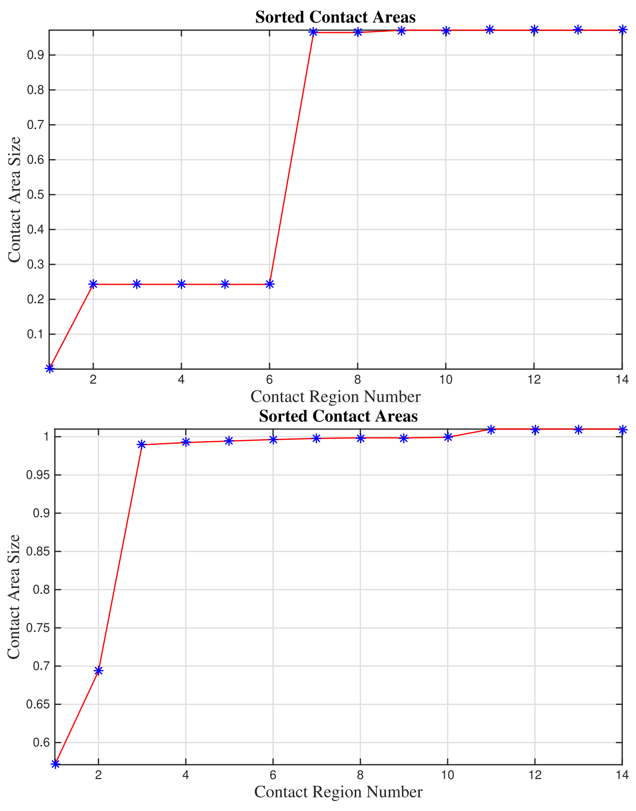


Fig. 28. Sorted plots of contact areas $A_{\mathfrak{R}}$ for the Arch case as defined by Equation (3). Top shows results of simulation step 3 for **Closest points** and bottom for **Opposing**. The value 1 is correct.

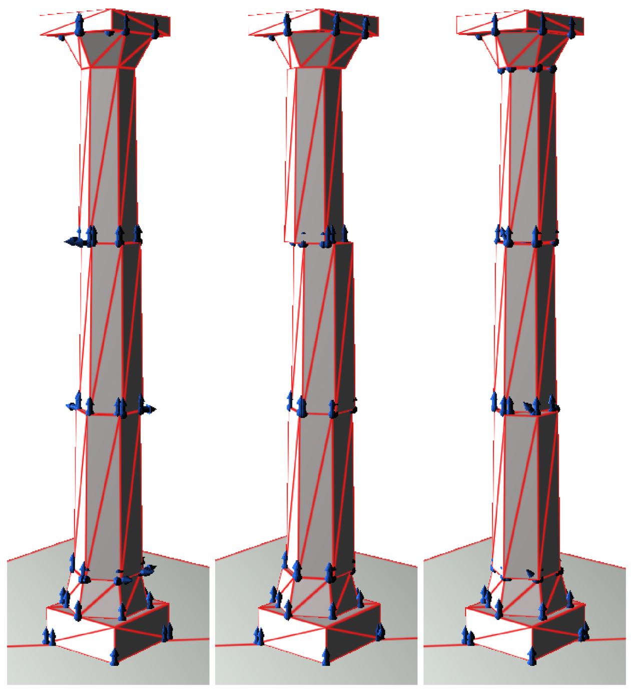


Fig. 29. Increasing amount of initial perturbation reduces (V, V) contacts. Left shows **Opposing**, middle **Growth**, and right **Closest points**. Observe the large perturbation value of 0.03m is visibly noticeable.

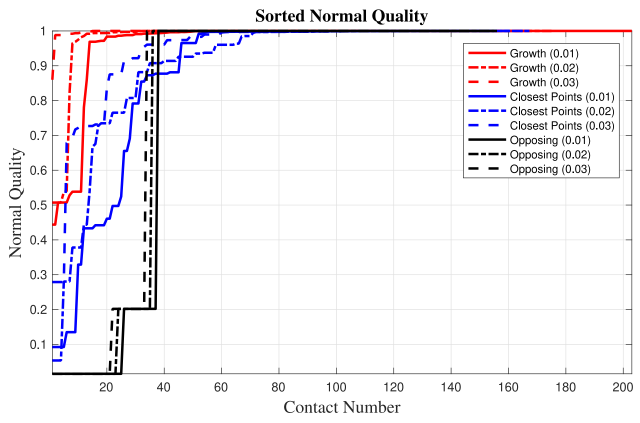


Fig. 30. Comparison of sorted Pillar normal quality plots for increasing amount of initial perturbation, Using random x and y displacements of the body center of masses 0.01m, 0.02m, 0.03m. All methods benefit from the perturbation.

all have incorrect contact normals, but coverage of contact regions appears to be equally good.

In Figure 32, we show sorted plots of the Pillar normal quality measures, the absolute value of the y-component of the normals. The best value is 1. **Growth** and **Closest points** generate roughly 1, 500 contact points more than the **Opposing** method. Interestingly, the quality of **Closest points** is much worse than the other methods. **Closest points** has only 30% good normals and a much wider spread of different bad normals. The **Opposing** method has little more than 90% good contact normals, and **Growth** close to 70% good normals. The **Opposing** method benefits most from higher resolution.

The results are less positive when considering accurate contact force computations as it only takes one single incorrect normal to get the wrong result. In our experience, approximate iterative methods are less sensitive to a few incorrect normals although they, too, will be affected by incorrect normals.

8.7 Effects of Time-Step Size

We study the influence of changing the timestep size for the **Growth**, **Closest points**, and **Opposing** methods. We use the timestep sizes 0.001s, 0.01s, and 0.1s. Further, we choose simulation step 4 for studying the normal quality. This is to allow for propagation of errors caused by bad contacts to have an influence on the overall simulation results.

Figure 33 shows the contact points at simulation step 4 in case of using the largest timestep size. It is evident that the **Opposing** method starts to collapse and the results looks different from the results obtained with the other two methods.

In Figure 34, we compare quality plots across timestep sizes and methods. We observe that the **Growth** method is highly sensitive towards changing the timestep size.

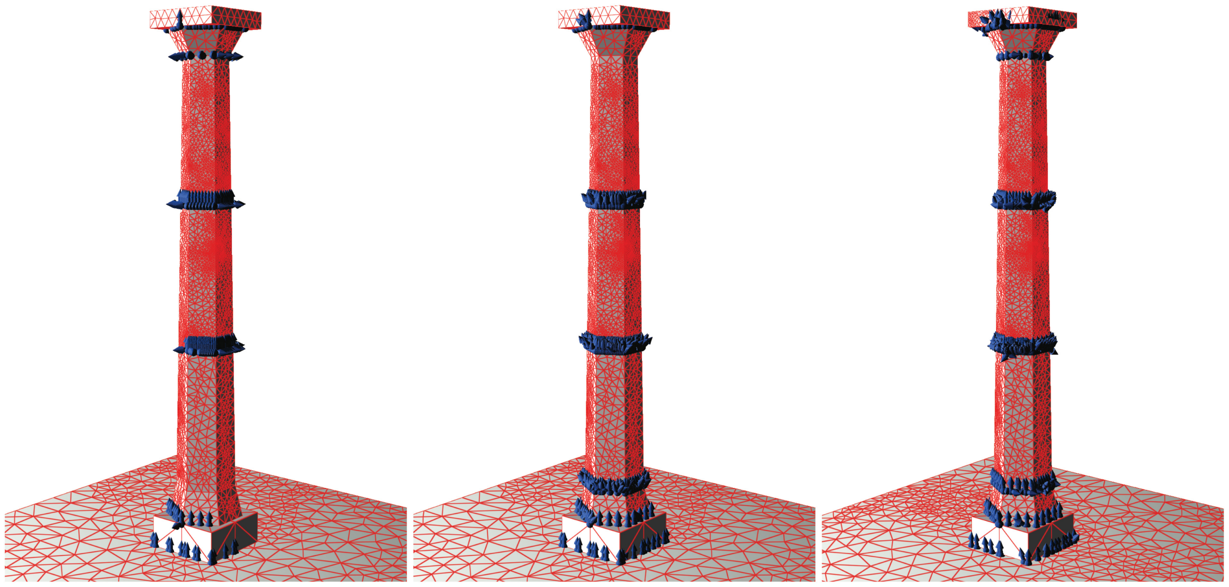


Fig. 31. Contact point visualization for fine resolution meshes. Left shows **Opposing**, middle **Growth**, and right **Closest points**. Observe the fine resolution does not remove badly formed normals.

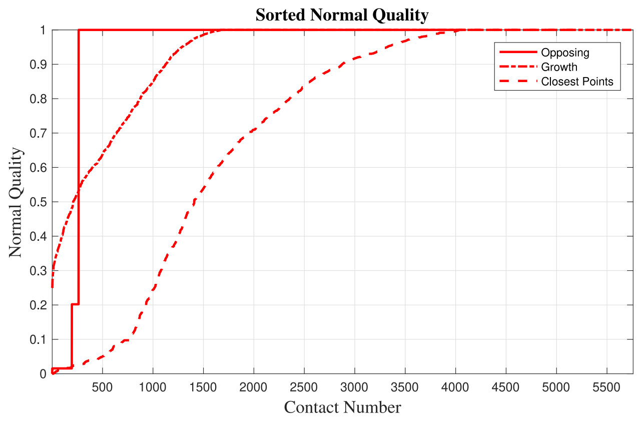


Fig. 32. Comparison of sorted plots of Pillar normal quality. Observe the **Opposing** method handled the fine resolution meshes better than the other methods. **Closest points** performs worse than the other methods.

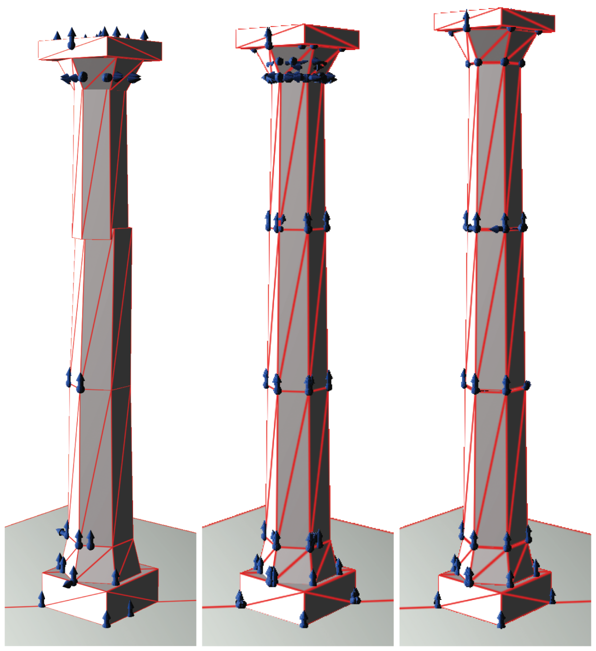


Fig. 33. Contact point visualization for very large timestep sizes (0.1s). Left shows **Opposing**, middle **Growth**, and right **Closest points**. Observe that the **Growth** method causes collapse at this timestep setting.

Not surprisingly, timestep size has a huge impact on the overall simulation results. It is contrived to relate this directly to contact point generation that is considered an instantaneous local geometric problem. The large timestep sizes often cause deep penetrations or tunneling artifacts or causes instabilities to large discretization errors. Hence, timestep size influences many aspects of the overall simulation results.

The major reason for differences in our simulations is because the chosen methods do not produce the exact same contact information for the contact force solver. The contact force solutions are sensitive to even a slight change in the normal direction or the addition or removal of a contact point. This can potentially change the overall motion of the objects involved.

9 CONCLUSION

In this article, we have reviewed past work on contact point generation methods and created a taxonomy for their classification. Our system is a combinatorial matrix of independent trait descriptors: (local, semi-local, or global) × (exact or approximate geometry) ×

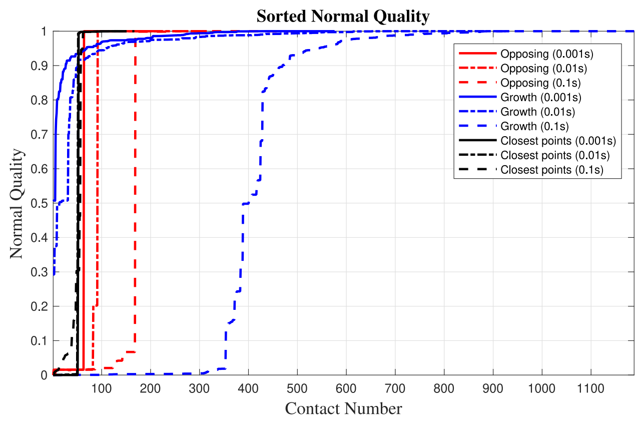


Fig. 34. Comparison of sorted plots of Pillar normal quality. Observe the **Opposing** and **Closest points** methods appear invariant to these timestep settings, whereas **Growth** method is very sensitive. Plots are taken from simulation step 4 but using different timestep sizes of 0.001s, 0.01s, and 0.1s.

(discrete or continuous). We sub-classified local methods into subgroups based on their geometric representations: surface-based, volume-based, implicit fields, approximations, and temporal approaches. Complementary to this, we identified unique normal generation approaches based on intersecting elements, SAT-tests, vertex-in-elements (similar to distance fields), and closest points.

From an analysis of local geometric shape, we carefully designed five fundamental cases that we named, **Sliding point**, **Two points**, **Point in crack**, **Internal edge**, and **Cliff edge** that widely spans the challenges in computing normal information. Our cases are limited in the sense that they only partially cover cases with mixed curvature. We presented specific simple examples with explicit defined quality measures to provide the community with a specific tool for comparison. We selected eight methods to implement that in our opinion span all possible normal generation methods for local methods. We subjected our eight implementations to our test cases and performed both quantitative and qualitative studies. Through examples, we have demonstrated that simulations are very sensitive towards contact point generation, and particular correct normal information is challenging to provide given the limitations of a using a local method.

No clear winner could be found when only considering local methods. All methods show different traits and tradeoffs. Without exception, all our selected methods could be fooled to yield incorrect normal information in cases of static setups, or they do not have the ability to select proper normal information when kinematics must be included.

The **Closest points** method behaved quantitively nice in a majority of cases but suffered from lack of temporal robustness. All the vertex in element variations we tested could not provide sufficient exact geometric information, and SAT-based methods often yielded false-positive results or incorrect normals. The Intersection-based approach seemed most erratic in behavior and the **Growth** method (a volume-based/CCD/closest points variant) and the **Opposing** method (a volume-based SAT variant using a post-normal search method) showed to be competitive, although their excessive factors are large compared with the **Closest points** method.

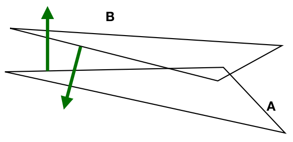
Hence, we are led to conclude that local methods can never provide 100% correct normal quality in all cases. Our fundamental cases analysis and test results are in our opinion an indication that semi-local methods are the only viable option that can provide enough local information to compute high quality normal information.

APPENDIX

A THE MOST OPPOSING SURFACE METHOD

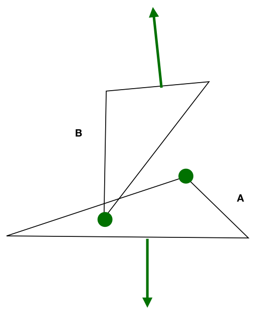
For accurate modeling, fine-resolution meshes are often desired. For such meshes, the contact regions are expected to often be larger flat regions. Hence, cases such as **Sliding point**, **Point in crack**, or **Two points** are not dominating the contact information. It tends to be more (F, F) types of contact. This idea was originally posed in Niebe (2013), we develop it further here. The idea behind favoring (F, F) information is supported by our observation that many other methods as we have shown in Section 7 tend to be fooled by (E, E) and (V, F) types of contact. The intuition is that if an opposing (F, F) pair can be found, then it is a very good estimation of the contact normal.

Given two volumetric elements, we first test if the elements have surface triangles. Only if they have surface triangles can the elements contribute to the contact information. Afterwards, we perform a SAT test to detect overlap or not. If elements are separated, then we bail out and the elements do not contribute with contact information.

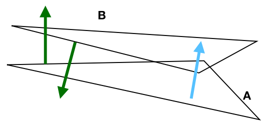


Now, given two overlapping volumetric elements with surface information, we now search for a pair of surface triangles that are mostly opposing each other as illustrated on the left. That is, their unit normal product is closest to -1. If no negative dot-products are found, then the volumes do not contribute to contact information. One may add an upper acceptable threshold to avoid detecting surface faces that are too close to become orthogonal.

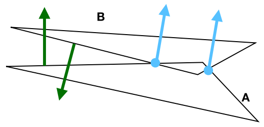
Further, it is required that part of $\mathcal{A}$ is in front of the surface triangle of $\mathcal{B}$ and vice versa. As we use tetrahedral elements, this is done by testing if any vertices of $\mathcal{A}$ are in front of the surface face from $\mathcal{B}$ and vice versa. This is to ensure that $\mathcal{A}$ and $\mathcal{B}$ are not sandwiched between the opposing surface triangles as the illustration on the right shows. Green arrows mark the most opposing surface triangle. One could have applied simple intersection testing to verify that the opposing faces are close by and not separated by the interior of the elements. In case of penetrations, the intersection test might miss the contact, whereas our choice would allow to deal with penetrations at the scale of the element sizes. Further, our choice of tetrahedral elements can be exploited to easily identify the opposing vertex of a surface



triangle and then test if that vertex is in front of the surface triangle from the other element.



At this point, we would either have found a pair of most opposing surface triangle faces or discarded the tetrahedral elements for further processing. To generate the normal, we take the unit vector of the average triangle normal directions as our resulting normal direction as shown on the left in the illustration. Here the two green face arrows on the left in the drawing give the average light blue normal shown near the apexes of the triangles $\mathcal{A}$ and $\mathcal{B}$.



Once the normal direction has been generated, all intersection points of all surface triangles of the volume elements are computed. They are then projected onto the generated normal direction, and the extreme points are used to estimate penetration depth and the mean point of the extreme points defines a contact plane where all intersection points are projected onto before their projected positions are used to generate the actual contact points as illustrated on the right.

This method will give exact geometry when having flat planar regions, but the most opposing (F, F) normal generation can be fooled for coarse meshes near cliff edges, as we showed in Section 7.5. Obviously, for **Sliding point** case the configuration does not fulfill the expectation of contact regions having a larger semi-flat area. For **Point in crack** cases, two normals will be generated although the normal directions might not be accurate. The opposing method can be interpreted as a variation of averaging/smoothing ideas (Civit-Flores and Susín 2015) without needing a multi-scale representation and taking simpler choices.

In Algorithm 1, we present the pseudo code for the implementation we used in this work. It has been specifically tailored for tetrahedra. Knowing we have tetrahedra is useful when initializing the points in lines 7 to 8. These points are needed for the sandwich test just after line 8. As we work with tetrahedral meshes in this work, we make sure to only use the triangles of the tetrahedra that are part of the object surface.

ALGORITHM 1: MostOpposingFacesNormal: This method searches for most opposing surface faces and use these to generate the resulting normal to be used. It is assumed that $\mathcal{A}$ and $\mathcal{B}$ overlap.

```
Data: $\mathcal{A}, \mathcal{B}$: a pair of tetrahedra
Result: $\mathbf{n}$: The resulting contact normal
1  $\gamma \leftarrow 0$;
2  foreach surface triangle $T_a \in \mathcal{A}$ do
3      $\mathbf{n}_a \leftarrow$ GetNormalOfTriangle($T_{\mathcal{A}}$);
4      foreach surface triangle $T_b \in \mathcal{B}$ do
5          $\mathbf{n}_b \leftarrow$ GetNormalOfTriangle($T_{\mathcal{B}}$);
6          if $\mathbf{n}_a \cdot \mathbf{n}_b < \gamma$ then
7              $\mathbf{p}_a \leftarrow$ GetOpposingNodeOfTriangle($T_a, \mathcal{A}$);
8              $\mathbf{p}_b \leftarrow$ GetOpposingNodeOfTriangle($T_b, \mathcal{B}$);
9              if $\mathbf{p}_a$ in front of $T_b$ and $\mathbf{p}_b$ in front of $T_a$ then
10                 $\gamma \leftarrow \mathbf{n}_a \cdot \mathbf{n}_b$;
11                 $\mathbf{n} \leftarrow$ Unit($\mathbf{n}_a - \mathbf{n}_b$);
12             end
13         end
14     end
15 end
16 return $\mathbf{n}$;
```

B THE GROWTH DISTANCE METHOD

Most methods based on using the separation vector to determine the contact normal suffer from the fact that as objects come in close static contact, then the closest points approach each other, and the separation vector tends to vanish. This is often countered by using a collision envelope that forms a proximity zone around each object surface, and anything inside the zone is defined as being in contact. As objects are locked in static contact or get pushed together, for instance, as caused by large mass ratios in stacks, then the objects sink deep into the proximity zones and possibly even small penetrations can occur due to time-discretization and drifting errors. Hence, all naive closest points-based approaches are doomed to fail to determine contact normals in such scenarios. Our solution is to replace the naive-approach to using the separation vector with a more intelligent and robust method. The idea is to find a way to robustly, virtually separate the objects such that the separation vector can be robustly computed from the closest points. Second, rather than using the separation vector directly to find the normal direction, we use it as a prior to search for the best matching normal given by the shape features.

First, we will present the separation approach in a rather general setting of any kind of convex objects. Hence, our approach is not limited to triangle/tetrahedral meshes but can be applied to general convex shapes. Given two convex point sets $\mathcal{A} \subseteq \mathbb{R}^3$ and $\mathcal{B} \subseteq \mathbb{R}^3$ we define the Minkowski sum to be the point set given by

$$\mathcal{A} \oplus \mathcal{B} \equiv \{\mathbf{p}_{\mathcal{A}} + \mathbf{p}_{\mathcal{B}} \,|\, \forall \mathbf{p}_{\mathcal{A}} \in \mathcal{A} \,\wedge\, \forall \mathbf{p}_{\mathcal{B}} \in \mathcal{B}\}\,. \tag{9}$$

The Minkowski difference is defined as

$$\mathcal{A} \ominus \mathcal{B} \equiv \mathcal{A} \oplus -\mathcal{B} \equiv \{\mathbf{p}_{\mathcal{A}} - \mathbf{p}_{\mathcal{B}} \,|\, \forall \mathbf{p}_{\mathcal{A}} \in \mathcal{A} \,\wedge\, \forall \mathbf{p}_{\mathcal{B}} \in \mathcal{B}\}\,. \tag{10}$$

A collision between $\mathcal{A}$ and $\mathcal{B}$ means (Cameron 1997; Gilbert et al. 1988; van Bergen 2001)

$$\mathbf{0} \in \mathcal{A} \ominus \mathcal{B}\,. \tag{11}$$

We define the uniform growth scaled versions of $\mathcal{A}$ and $\mathcal{B}$ as

$$\mathcal{A}(\alpha) \equiv \{\alpha\,(\mathbf{p}_{\mathcal{A}} - \mathbf{c}_{\mathcal{A}}) + \mathbf{c}_{\mathcal{A}} \,|\, \forall \mathbf{p}_{\mathcal{A}} \in \mathcal{A} \,\wedge\, \mathbf{c}_{\mathcal{A}} \in \mathcal{A}^{\circ}\}\,, \tag{12}$$

$$\mathcal{B}(\alpha) \equiv \{\alpha\,(\mathbf{p}_{\mathcal{B}} - \mathbf{c}_{\mathcal{B}}) + \mathbf{c}_{\mathcal{B}} \,|\, \forall \mathbf{p}_{\mathcal{B}} \in \mathcal{B} \,\wedge\, \mathbf{c}_{\mathcal{B}} \in \mathcal{B}^{\circ}\}\,, \tag{13}$$

where $0 < \alpha \leq 1$ is termed the growth scale (Ong and Gilbert 1994, 1996) and the interior of $\mathcal{A}$ and $\mathcal{B}$ are given by $\mathcal{A}^{\circ}$ and $\mathcal{B}^{\circ}$. Observe that $\mathcal{A}(0) = \{\mathbf{c}_{\mathcal{A}}\}$ and $\mathcal{A}(1) = \mathcal{A}$ similar holds for $\mathcal{B}(0)$ and $\mathcal{B}(1)$. We use the geometric centroids for the interior center points $\mathbf{c}_{\mathcal{A}}$ and $\mathbf{c}_{\mathcal{B}}$.

Assume we are given two colliding sets and $\mathbf{c}_{\mathcal{A}} \neq \mathbf{c}_{\mathcal{B}}$; then we wish to find a growth scale α^* such that

$$\alpha^* \equiv \arg\min_{\alpha > 0} \alpha \quad \text{s.t.} \quad g(\alpha) = \delta, \tag{14}$$

where $\delta > 0$ is a user-specified desired separation distance and the g-function is defined such that it gives us the minimum norm of any point in the Minkowski difference,

$$g(\alpha) \equiv \min \{ \| \mathbf{p} \| \mid \mathbf{p} \in \{\mathcal{A}(\alpha) \ominus \mathcal{B}(\alpha)\}\} \, . \tag{15}$$

If we have a value of $\alpha^\dagger$ such that $g(\alpha^\dagger) = 0$, then $\mathbf{0} \in \mathcal{A}(\alpha^\dagger) \ominus \mathcal{B}(\alpha^\dagger)$ and the scaled growth objects would collide. This means $\alpha^\dagger$ is a conservative upper bound for feasible α-values and that $\alpha^* < \alpha^\dagger$. For $\delta = 0$, we have that $\alpha^\dagger$ is the supremum. We may re-interpret the growth scale formulation by some re-writing using $\mathbf{v} = \mathbf{c}_\mathcal{A} - \mathbf{c}_\mathcal{B}$,

$$\mathcal{A}(\alpha) \ominus \mathcal{B}(\alpha) = \alpha \left(\{\mathcal{A} \ominus \mathcal{B}\} - \mathbf{v}\right) + \mathbf{v}, \tag{16}$$

$$= \alpha \left(\{\mathcal{A} \ominus \mathcal{B}\} - \mathbf{v} \underbrace{\left(1 - \frac{1}{\alpha}\right)}_{\tau} \right). \tag{17}$$

Since $\alpha > 0$, we always have $-\infty < \tau < 0$ and as $\alpha \to 0$, we have $\tau \to -\infty$. Since $\alpha > 0$ and we are looking for α^* such that $g(\alpha^*) \to \delta$, that implies we are equivalently trying to find a supremum $\tau^\dagger$ value such that

$$\{\mathcal{A} \ominus \mathcal{B}\} - \mathbf{v}\,\tau^\dagger = \mathbf{0}. \tag{18}$$

From this, we can re-interpret the problem of finding a suitable growth scale as a ray-cast problem,

$$\mathbf{v}\,\tau \in \{\mathcal{A} \ominus \mathcal{B}\} \, . \tag{19}$$

Alternatively, as a continuous linear motion, here we express it as $\mathcal{B}$ moving with velocity $-\mathbf{v}$ and τ becomes the pseudo time of impact between $\mathcal{A}$ and $\mathcal{B}$,

$$\mathbf{0} \in \{\mathcal{A} \ominus \{\mathcal{B} - \mathbf{v}\,\tau^\dagger\}\}. \tag{20}$$

Hence, we see that to determine the growth scale is in fact equivalent to performing CCD under the assumption that the relative linear motion is given by the velocity $\mathbf{v}$. Of course, $\mathbf{v}$ may not coincide with the real relative velocity of the objects. Instead, it provides the advantage of using a CCD technique to resolve penetrating objects in static scenarios where using real velocities would imply no motion and hence loss of ability to resolve the penetration.

Besides providing us with analytical insight in our approach our three equivalent problem formulations (14), (19), and (20) provide us with several possible numerical choices for finding a method that can compute the proper scaling, ray-length, or time-of-impact. Any method that can robustly compute closest points or penetration state can be utilized. In our experiments, we used a direct method utilizing a case-by-case approach specialized for tetrahedra. In a preliminary version, we experimented with using ray-casting with GJK to support general convex primitives (Silcowitz et al. 2010).

Algorithm 2 illustrates the implementation used in this work. In lines 1 to 4, we perform a back-tracking line search on the growth scale to quickly find a down-scaled version of the shapes that do not overlap. This serves as the initial ray length for a ray-casting GJK-like algorithm starting in line 7. Observe how we convert to ray-cast parameter to growth scale in line 10. Once the ray-cast method has enlarged the objects such that their closest points are δ distance apart (we use one hundredth of the diameter of the smallest shape as δ-value), then we can search the surfaces of the shapes to find the best shape normal. We give precedence to higher dimensionality features, and hence a face normal will be preferred over a vertex normal. After having used our algorithm for computing the normal, then one can compute all intersection points between the two shapes and use these as contact points. One may project these points onto the found normal and compute the extent along the normal to get a measure of penetration.

ALGORITHM 2: GrowthDistanceNormal: This algorithm finds a down scaled version of the given shape with a sufficient separation. Further, the sub-routine BestNormal searches the input shapes for the vertex, edge or face features whos surface normals arethe closest to the specified direction. The feature with highest dimension identifies the final normal.

```
Data: $\mathcal{A}, \mathcal{B}$: two convex shapes, $\delta$:desired separation, $\beta = 0.9$:
      damping factor
Result: $\mathbf{n}$: The contact normal
1  $\alpha_{\min} \leftarrow 1$;
2  while Overlap($\mathcal{A}(\alpha_{\min}), \mathcal{B}(\alpha_{\min})$) do
3      $\alpha_{\min} \leftarrow \alpha_{\min}\,\beta$;
4  end
5  $\mathbf{p}_\mathcal{A}, \mathbf{p}_\mathcal{B} \leftarrow$ ClosestPoints($\mathcal{A}(\alpha_{\min}), \mathcal{B}(\alpha_{\min})$);
6  $s \leftarrow \mathbf{p}_\mathcal{B} - \mathbf{p}_\mathcal{A}$;
7  $\tau \leftarrow 1 - \frac{1}{\alpha_{\min}}$;
8  while $\| s \| > \delta$ do
9      $\tau \leftarrow \tau + \beta\,\frac{\mathbf{s}\cdot\mathbf{v}}{\mathbf{v}\cdot\mathbf{v}}$;
10     $\alpha \leftarrow 1 - \frac{1}{\tau}$;
11     $\mathbf{p}_\mathcal{A}, \mathbf{p}_\mathcal{B} \leftarrow$ ClosestPoints($\mathcal{A}(\alpha), \mathcal{B}(\alpha)$);
12     $s \leftarrow \mathbf{p}_\mathcal{B} - \mathbf{p}_\mathcal{A}$;
13 end
14 $\mathbf{n}_\mathcal{A}, f_\mathcal{A} \leftarrow$ BestNormal($s, \mathcal{A}$);
15 $\mathbf{n}_\mathcal{B}, f_\mathcal{B} \leftarrow$ BestNormal($-s, \mathcal{B}$);
16 return Dim($f_\mathcal{A}$) > Dim($f_\mathcal{B}$) ? $\mathbf{n}_\mathcal{A}$ : $\mathbf{n}_\mathcal{B}$;
```

REFERENCES

Jérémie Allard, François Faure, Hadrien Courtecuisse, Florent Falipou, Christian Duriez, and Paul G. Kry. 2010. Volume contact constraints at arbitrary resolution. *ACM Trans. Graph.* 29, 4, Article 82 (July 2010), 10 pages.

George Baciu and Wingo Sai-Keung Wong. 2004. Image-based collision detection. In *Integrated Image and Graphics Technologies*, David D. Zhang, Mohamed Kamel, and George Baciu (Eds.). Kluwer Academic Publishers, Norwell, MA, 75–94.

David Baraff. 1989. Analytical methods for dynamic simulation of non-penetrating rigid bodies. *SIGGRAPH Comput. Graph.* 23, 3 (1989), 223–232.

David Baraff. 1991. Coping with friction for non-penetrating rigid body simulation. *SIGGRAPH Comput. Graph.* 25, 4 (July 1991), 31–41.

David Baraff. 1994. Fast contact force computation for nonpenetrating rigid bodies. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'94)*. ACM, New York, NY, 23–34.

David Baraff. 1997. An introduction to physically based modeling: Rigid body simulation I unconstrained rigid body dynamics. In *ACM SIGGRAPH 1997 Courses (SIGGRAPH'97)*. ACM.

David Baraff and Andrew Witkin. 1998. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'98)*. ACM, New York, NY, 43–54.

David Baraff, Andrew Witkin, and Michael Kass. 2003. Untangling cloth. *ACM Trans. Graph.* 22, 3 (July 2003), 862–870.

Jan Bender, Kenny Erleben, and Jeff Trinkle. 2014. Interactive simulation of rigid body dynamics in computer graphics. *Comput. Graph. Forum* 33, 1 (2014), 246–270.

Bedřich Beneš and Nestor Goméz Villanueva. 2005. GI-COLLIDE: Collision detection with geometry images. In *Proceedings of the 21st Spring Conference on Computer Graphics (SCCG'05)*. ACM, New York, NY, 95–102.

Gareth Bradshaw and Carol O'Sullivan. 2002. Sphere-tree construction using dynamic medial axis approximation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'02)*. ACM, New York, NY, 33–40.

Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph.* 21, 3 (July 2002), 594–603.

R. Bridson, S. Marino, and R. Fedkiw. 2003. Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'03)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 28–36.

Tyson Brochu, Essex Edwards, and Robert Bridson. 2012. Efficient geometrically exact continuous collision detection. *ACM Trans. Graph.* 31, 4, Article 96 (July 2012), 7 pages.

S. Cameron. 1997. Enhancing GJK: Computing minimum and penetration distances between convex polyhedra. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation, 1997*, Vol. 4. 3112–3117.

O. Civit-Flores and A. Susín. 2015. Fast contact determination for intersecting deformable solids. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games (MIG'15)*. ACM, New York, NY, 205–214.

Jonathan D. Cohen, Ming C. Lin, Dinesh Manocha, and Madhav Ponamgi. 1995. I-COLLIDE: An interactive and exact collision detection system for large-scale environments. In *Proceedings of the 1995 Symposium on Interactive 3D Graphics (I3D'95)*. ACM, New York, NY, 189–ff.

Erwin Coumans. 2010. Contact Generation. *Online slides; Game Developers Conference 2010*. Retrieved from https://bullet.googlecode.com/files/GDC10_Coumans_Erwin_Contact.pdf.

Sean Curtis, Rasmus Tamstorf, and Dinesh Manocha. 2008. Fast collision detection for deformable models using representative-triangles. In *Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games (I3D'08)*. ACM, New York, NY, 61–69.

Gilles Daviet, Florence Bertails-Descoubes, and Laurence Boissieux. 2011. A hybrid iterative solver for robustly capturing coulomb friction in hair dynamics. *ACM Trans. Graph.* 30, 6, Article 139 (Dec. 2011), 12 pages.

R. E. English, M. Lentine, and R. Fedkiw. 2013. Interpenetration free simulation of thin shell rigid bodies. *IEEE Trans. Vis. Comput. Graph.* 19, 6 (June 2013), 991–1004.

Kenny Erleben. 2005. *Stable, Robust, and Versatile Multibody Dynamics Animation.* Ph.D. Dissertation. Department of Computer Science, University of Copenhagen, Denmark.

Kenny Erleben. 2007. Velocity-based shock propagation for multibody dynamics animation. *ACM Trans. Graph.* 26, 2, Article 12 (June 2007).

Kenny Erleben. 2014. Moving conforming contact manifolds and related numerical problems. *Video from workshop "Computational Contact Mechanics: Advances and Frontiers in Modeling Contact" at Banff International Research Station (BIRS)*. Retrieved from http://videos.birs.ca/2014/14w5147/201402200946-Erleben.mp4.

Kenny Erleben. 2017. Rigid body contact problems using proximal operators. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'17)*. ACM, New York, NY.

Kenny Erleben, Jon Sporring, Knud Henriksen, and Henrik Dohlman. 2005. *Physics-based Animation (Graphics Series)*. Charles River Media, Inc., Rockland, MA.

François Faure, Sébastien Barbier, Jérémie Allard, and Florent Falipou. 2008. Image-based collision detection and response between arbitrary volume objects. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'08)*. Eurographics Association, Aire-la-Ville, Switzerland, 155–162.

Susan Fisher and Ming C. Lin. 2001. Deformed distance fields for simulation of non-penetrating flexible bodies. In *Proceedings of the Eurographic Workshop on Computer Animation and Simulation*. Springer-Verlag New York, Inc., New York, NY, 99–111.

E. G. Gilbert, D. W. Johnson, and S. S. Keerthi. 1988. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE J. Robot. Automat.* 4, 2 (Apr 1988), 193–203.

S. Gottschalk, M. C. Lin, and D. Manocha. 1996. OBBTree: A hierarchical structure for rapid interference detection. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'96)*. ACM, New York, NY, 171–180.

Naga K. Govindaraju, David Knott, Nitin Jain, Ilknur Kabul, Rasmus Tamstorf, Russell Gayle, Ming C. Lin, and Dinesh Manocha. 2005. Interactive collision detection between deformable models using chromatic decomposition. *ACM Trans. Graph.* 24, 3 (July 2005), 991–999.

Alba Granados, Jonas Brunskog, Marek Krzysztof Misztal, Vincent Visseq, and Kenny Erleben. 2014. Finite element modeling of the vocal folds with deformable interface tracking. In *Proceedings of the 7th Forum AcusticumForum Acusticum*. European Acoustics Association, Krakow, Polen.

Eran Guendelman, Robert Bridson, and Ronald Fedkiw. 2003. Nonconvex rigid bodies with stacking. *ACM Trans. Graph.* 22, 3 (July 2003), 871–878.

James K. Hahn. 1988. Realistic animation of rigid bodies. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'88)*. ACM, New York, NY, 299–308.

David Harmon, Etienne Vouga, Breannan Smith, Rasmus Tamstorf, and Eitan Grinspun. 2009. Asynchronous contact mechanics. *ACM Trans. Graph.* 28, 3, Article 87 (July 2009), 12 pages.

David Harmon, Etienne Vouga, Breannan Smith, Rasmus Tamstorf, and Eitan Grinspun. 2012. Asynchronous contact mechanics. *Commun. ACM* 55, 4 (Apr. 2012), 102–109.

David Harmon, Etienne Vouga, Rasmus Tamstorf, and Eitan Grinspun. 2008. Robust treatment of simultaneous collisions. *ACM Trans. Graph.* 27, 3, Article 23 (Aug. 2008), 4 pages.

Liang He, Ricardo Ortiz, Andinet Enquobahrie, and Dinesh Manocha. 2015. Interactive continuous collision detection for topology changing models using dynamic clustering. In *Proceedings of the 19th Symposium on Interactive 3D Graphics and Games (i3D'15)*. ACM, New York, NY, 47–54.

Bruno Heidelberger, Matthias Teschner, Richard Keiser, Matthias Müller, and Markus H. Gross. 2004. Consistent penetration depth estimation for deformable collision response. In *Proceedings of the Vision, Modeling and Visualization 2004 Conference*, Bernd Girod, Marcus A. Magnor, and Hans-Peter Seidel (Eds.). Berlin, 339–346.

Gentaro Hirota, Susan Fisher, and Ming Lin. 2000. *Simulation of Non-penetrating Elastic Bodies Using Distance Fields.* Technical Report. University of North Carolina at Chapel Hill, Chapel Hill, NC.

Gentaro Hirota, Susan Fisher, Andrei State, Chris Lee, and Henry Fuchs. 2001. An implicit finite element method for elastic solids in contact. In *Proceedings of the 14th Conference on Computer Animation.* 136–254.

Philip M. Hubbard. 1996. Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans. Graph.* 15, 3 (July 1996), 179–210.

Thomas C. Hudson, Ming C. Lin, Jonathan Cohen, Stefan Gottschalk, and Dinesh Manocha. 1997. V-COLLIDE: Accelerated collision detection for VRML. In *Proceedings of the 2nd Symposium on Virtual Reality Modeling Language (VRML'97)*. ACM, New York, NY, 117–ff.

Doug L. James and Dinesh K. Pai. 2004. BD-tree: Output-sensitive collision detection for reduced deformable models. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 393–398.

Michel Jean. 1999. The non-smooth contact dynamics method. *Comput. Methods Appl. Mech. Eng.* 177, 3–4 (July 1999), 235–257.

Franck Jourdan, Pierre Alart, and Michel Jean. 1998. A gauss-seidel like algorithm to solve frictional contact problems. *Comput. Methods Appl. Mech. Eng.* 155, 1 (1998), 31–47.

Danny M. Kaufman, Timothy Edmunds, and Dinesh K. Pai. 2005. Fast frictional dynamics for rigid bodies. *ACM Trans. Graph.* 24, 3 (July 2005), 946–956.

Danny M. Kaufman, Shinjiro Sueda, Doug L. James, and Dinesh K. Pai. 2008. Staggered projections for frictional contact in multibody systems. *ACM Trans. Graph.* 27, 5, Article 164 (Dec. 2008), 11 pages.

Young J. Kim, Miguel A. Otaduy, Ming C. Lin, and Dinesh Manocha. 2002. Fast penetration depth computation for physically-based animation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'02)*. ACM, New York, NY, 23–31.

James T. Klosowski, Martin Held, Joseph S. B. Mitchell, Henry Sowizral, and Karel Zikan. 1998. Efficient collision detection using bounding volume hierarchies of k-DOPs. *IEEE Trans. Vis. Comput. Graph.* 4, 1 (Jan. 1998), 21–36.

Ming Chieh Lin. 1993. *Efficient Collision Detection for Animation and Robotics.* Ph.D. Dissertation. University of California, Berkeley. AAI9430587.

Ming C. Lin and Stefan Gottschalk. 1998. Collision detection between geometric models: A survey. In *Proceedings of the IMA Conference on Mathematics of Surfaces.* 37–56.

D. Marchal, F. Aubert, and C. Chaillou. 2004. Collision between deformable objects using fast-marching on tetrahedral models. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'04)*. Eurographics Association, Aire-la-Ville, Switzerland, 121–129.

Hammad Mazhar, Toby Heyn, Dan Negrut, and Alessandro Tasora. 2015. Using nesterov's method to accelerate multibody dynamics with friction and contact. *ACM Trans. Graph.* 34, 3, Article 32 (May 2015), 14 pages.

Brian Mirtich. 1998a. *Rigid Body Contact: Collision Detection to Force Computation.* Technical Report TR98-01. Mitsubishi Electric Research Laboratories.

Brian Mirtich. 1998b. V-Clip: Fast and robust polyhedral collision detection. *ACM Trans. Graph.* 17, 3 (July 1998), 177–208.

Brian Mirtich. 2000. Timewarp rigid body simulation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'00)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, 193–200.

Brian Vincent Mirtich. 1996. *Impulse-based Dynamic Simulation of Rigid Body Systems.* Ph.D. Dissertation. University of California, Berkeley.

M. K. Misztal, K. Erleben, A. Bargteil, J. Fursund, B. Bunch Christensen, J. A. Bærentzen, and R. Bridson. 2012. Multiphase flow of immiscible fluids on unstructured moving meshes. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'12)*. Eurographics Association, Aire-la-Ville, Switzerland, 97–106.

Matthew Moore and Jane Wilhelms. 1988. Collision detection and response for computer animationr3. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'88)*. ACM, New York, NY, 289–298.

Matthias Müller, Nuttapong Chentanez, Tae-Yong Kim, and Miles Macklin. 2015. Air meshes for robust collision handling. *ACM Trans. Graph.* 34, 4, Article 133 (July 2015), 9 pages.

Sarah Maria Niebe. 2013. *Rigid Bodies in Contact: And Everything in Between.* Ph.D. Dissertation. Department of Computer Science, Faculty of Science, University of Copenhagen.

Chong Jin Ong and E. G. Gilbert. 1996. Growth distances: New measures for object separation and penetration. *IEEE Trans. Robot. Automat.* 12, 6 (Dec 1996), 888–903.

Chong Jin Ong and Elmer G. Gilbert. 1994. Robot path planning with penetration growth distance. In *Proceedings of the International Conference on Robotics and Automation (ICRA'94)*. 2146–2152.

Eric G. Parker and James F. O'Brien. 2009. Real-time deformation and fracture in a game environment. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'09)*. ACM, New York, NY, 165–175.

Stéphane Redon, Abderrahmane Kheddar, and Sabine Coquillart. 2000. An algebraic solution to the problem of collision detection for rigid polyhedral objects. In *Proceedings of the IEEE Conference on Robotics and Automation*, Vol. 4. San Fransisco, CA. 3733–3738.

Stephane Redon, Abderrahmane Kheddar, and Sabine Coquillart. 2002. Fast continuous collision detection between rigid bodies. *Comput. Graph. Forum* 21, 3 (2002), 279–287.

Robert Schmidtke and Kenny Erleben. 2017. Chunked bounding volume hierarchies for fast digital prototyping using volumetric meshes. *IEEE Trans. Vis. Comput. Graph.* 99 (2017), 1–1. https://ieeexplore.ieee.org/document/8219711/.

Dylan A. Shell and Evan Drumwright. 2009. Precise generalized contact point and normal determination for rigid body simulation. In *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC'09)*. ACM, New York, NY, 2107–2108.

Morten Silcowitz, Sarah Niebe, and Kenny Erleben. 2010. Contact point generation for convex polytopes in interactive rigid body dynamics. *Poster at SCA 10'.* (2010).

Breannan Smith, Danny M. Kaufman, Etienne Vouga, Rasmus Tamstorf, and Eitan Grinspun. 2012. Reflections on simultaneous impact. *ACM Trans. Graph.* 31, 4, Article 106 (July 2012), 12 pages.

Jonas Spillmann and Mathias Teschner. 2005. Contact surface computation for coarsely sampled deformable objects. In *Proceedings of the Conference on Vision, Modeling, Visualization (VMV'05)*. 189–296.

Avneesh Sud, Naga Govindaraju, Russell Gayle, Ilknur Kabul, and Dinesh Manocha. 2006. Fast proximity computation among deformable models using discrete voronoi diagrams. *ACM Trans. Graph.* 25, 3 (July 2006), 1144–1153.

Min Tang, Sean Curtis, Sung-Eui Yoon, and Dinesh Manocha. 2008. Interactive continuous collision detection between deformable models using connectivity-based culling. In *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling (SPM'08)*. ACM, New York, NY, 25–36.

Min Tang, Dinesh Manocha, Miguel A. Otaduy, and Ruofeng Tong. 2012. Continuous penalty forces. *ACM Trans. Graph.* 31, 4, Article 107 (July 2012), 9 pages.

Min Tang, Dinesh Manocha, and Ruofeng Tong. 2010. MCCD: Multi-core collision detection between deformable models using front-based decomposition. *Graph. Models* 72, 2 (2010), 7–23.

Min Tang, Dinesh Manocha, Sung-Eui Yoon, Peng Du, Jae-Pil Heo, and Ruo-Feng Tong. 2011. VolCCD: Fast continuous collision culling between deforming volume meshes. *ACM Trans. Graph.* 30, 5, Article 111 (Oct. 2011), 15 pages.

Min Tang, Ruofeng Tong, Zhendong Wang, and Dinesh Manocha. 2014. Fast and exact continuous collision detection with bernstein sign classification. *ACM Trans. Graph.* 33, 6, Article 186 (Nov. 2014), 8 pages.

Joseph Teran, Eftychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. 2005. Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'05)*. ACM, New York, NY, 181–190.

M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. 2005. Collision detection for deformable objects. *Computer Graphics Forum* 24, 1 (2005), 61–81.

Richard Tonge, Feodor Benevolenski, and Andrey Voroshilov. 2012. Mass splitting for jitter-free parallel rigid body simulation. *ACM Trans. Graph.* 31, 4, Article 105 (July 2012), 8 pages.

Gino van Bergen. 2001. Proximity queries and penetration depth computation on 3D game objects. In *Proceedings of Game Developers Conference (GDC'01)*.

Gino van den Bergen. 1998. Efficient collision detection of complex deformable models using AABB trees. *J. Graph. Tools* 2, 4 (Jan. 1998), 1–13.

Huamin Wang. 2014. Defending continuous collision detection against errors. *ACM Trans. Graph.* 33, 4, Article 122 (July 2014), 10 pages.

Rachel Weinstein, Joseph Teran, and Ron Fedkiw. 2006. Dynamic simulation of articulated rigid bodies with contact and collision. *IEEE Trans. Vis. Comput. Graph.* 12, 3 (May 2006), 365–374.

Jedediyah Williams, Ying Lu, and J. C. Trinkle. 2016. PEG: A geometrically exact contact model for polytopes in multi-rigid-body simulation. *J. Comput. Nonlinear Dynam* 12, 2 (Dec 2016), 14 pages.

H. Xu and J. Barbie. 2017. 6-DoF haptic rendering using continuous collision detection between points and signed distance fields. *IEEE Trans. Haptics* 10, 2 (April 2017), 151–161.

Xinyu Zhang, Stephane Redon, Minkyoung Lee, and Young J. Kim. 2007. Continuous collision detection for articulated models using taylor models and temporal culling. *ACM Trans. Graph.* 26, 3, Article 15 (July 2007).

Received February 2017; revised November 2017; accepted March 2018