Local Optimization for Robust Signed Distance Field Collision

MILES MACKLIN, NVIDIA and University of Copenhagen
KENNY ERLEBEN, University of Copenhagen
MATTHIAS MÜLLER, NVIDIA
NUTTAPONG CHENTANEZ, NVIDIA
STEFAN JESCHKE, NVIDIA
ZACH CORSE, NVIDIA

Signed distance fields (SDFs) are a popular shape representation for collision detection. This is due to their query efficiency, and the ability to provide robust inside/outside information. Although it is straightforward to test points for interpenetration with an SDF, it is not clear how to extend this to continuous surfaces, such as triangle meshes. In this paper, we propose a per-element local optimization to find the closest points between the SDF isosurface and mesh elements. This allows us to generate accurate contact points between sharp point-face pairs, and handle smoothly varying edge-edge contact. We compare three numerical methods for solving the local optimization problem: projected gradient descent, Frank-Wolfe, and golden-section search. Finally, we demonstrate the applicability of our method to a wide range of scenarios including collision of simulated cloth, rigid bodies, and deformable solids.

CCS Concepts: • **Computing methodologies** → **Simulation by animation**; *Interactive simulation*; • **Computer systems organization** → *Robotics.*

Additional Key Words and Phrases: simulation, collision detection, contact generation, signed distance fields

ACM Reference Format:
Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, and Zach Corse. 2020. Local Optimization for Robust Signed Distance Field Collision. *Proc. ACM Comput. Graph. Interact. Tech.* 3, 1 (May 2020), 9 pages. https://doi.org/10.1145/3384538

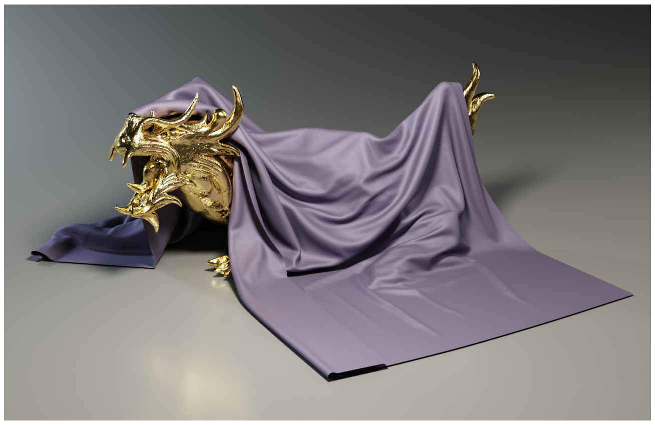
Fig. 1. Face-based optimization accurately captures contacts between cloth and sharp features in a signed distance function without discrete sampling of mesh geometry.

1 INTRODUCTION

In general a signed distance field (SDF) may be considered as a function $\phi(\mathbf{x}) : \mathbb{R}^3 \to \mathbb{R}$ where $\mathbf{x}$ is a point in some previously defined coordinate frame. The value $\phi(\mathbf{x})$ represents the signed Euclidean distance of $\mathbf{x}$ to a point on the surface, where $\phi = 0$. By convention we define the interior of an object as the set of points where $\phi(\mathbf{x}) < 0$. A consequence of this definition is that the gradient $\nabla\phi$ points in the direction of maximum distance increase away from the surface. Given an SDF, the closest point on the surface to $\mathbf{x}$ may be calculated directly as $\mathbf{y} = \mathbf{x} - \nabla\phi(\mathbf{x})\phi(\mathbf{x})$. Determining collision between a point $\mathbf{x}$ against an object represented by an SDF is then as simple as evaluating $\phi(\mathbf{x})$ and checking the sign. Depending on the choice of contact model, a contact normal may be defined as $\mathbf{n} = \nabla\phi$, which can be used as input for a penalty or constraint-based contact solver.

Although point-based contact is well suited for use cases such as particle systems, it is less clear how to extend the method to handle continuous surfaces such as triangle meshes. Previous work has proposed performing an offline sampling of the surface geometry at discrete points. Then, at runtime, checking each point for overlap as described above. While this approach is conceptually simple, it suffers from the problem that any discrete sampling may be insufficient to detect overlap for particularly sharp features, for example the spikes of a dragon, as shown in Figure 1, or the apex of a cone interpenetrating a piece of cloth, as shown in Figure 2 (a). In addition, point-sampling fails to capture the case of edge-edge contacts between bodies, for example between two boxes resting on each other, as in Figure 2 (c). While it is possible to increase point-sampling density, the contact locations remain fixed at discrete points on the surface which can cause jumps as contacts shift from one point to the next. Furthermore, as the number of samples increases, so does the number of contacts generated. This shifts the

Authors' addresses: Miles Macklin, NVIDIA, University of Copenhagen, mmacklin@nvidia.com; Kenny Erleben, University of Copenhagen, kenny@di.ku.dk; Matthias Müller, NVIDIA, matthiasm@nvidia.com; Nuttapong Chentanez, NVIDIA, nchentanez@nvidia.com; Stefan Jeschke, NVIDIA, sjeschke@nvidia.com; Zach Corse, NVIDIA, zcorse@nvidia.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
2577-6193/2020/5-ART $15.00
https://doi.org/10.1145/3384538

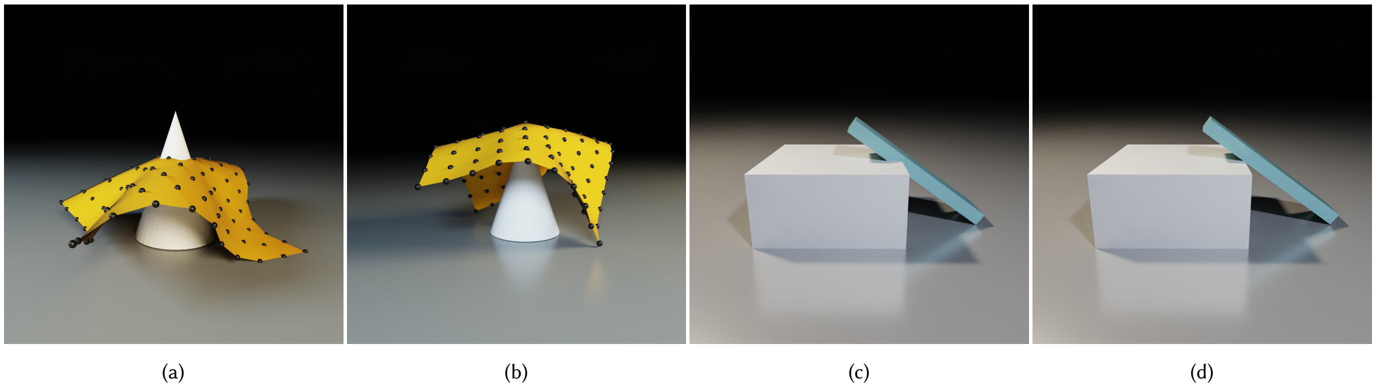

Fig. 2. Sampling based methods may miss contacts between sharp features in the SDF cone and the cloth geometry (a). Our method adaptively finds points of maximum penetration along each face, generating accurate face-vertex contacts (b). Point sampling also fails to capture edge-edge contacts between geometry and the SDF (c), our method naturally generates edge-edge contacts by optimizing over each face (d).

computational burden to the contact solver, which may have high computational complexity.

In this work we propose a method to generate contacts between triangle mesh faces and edges in a continuous manner. Our method is based on a local optimization over mesh edges or faces using constrained convex optimization. Unlike fixed, discrete sample points, this approach allows contacts to vary smoothly over the mesh elements, capturing sharp point-face and edge-edge contacts.

In summary, we make the following contributions:

- A method for generating smoothly varying contacts between mesh features and shapes represented by signed distance fields (SDFs)
- An analysis and comparison of three numerical methods for solving the local constrained optimization
- The application of our method to contact generation between thin-shells, rigid bodies, and deformable solids

2 RELATED WORK

Signed distance fields, and implicit surfaces have a long history in computer graphics for shape modeling and rendering [Blinn 1982; Bloomenthal and Wyvill 1997; Hart 1996; Wyvill et al. 1999a,b]. In this paper we focus on the use of SDFs for collision detection or contact generation, and discuss some previous work below.

Collision Detection. Fuhrmann et al [2003] leveraged SDFs for collision detection between cloth and complex geometry. They recognized the limitations of point-sampling the cloth surface, and proposed adding samples at the midpoint of each edge to minimize the chance of missed collisions. Our method generalizes this idea to adaptive and smoothly varying face and edge contacts. Guendelman [2003] showed how SDFs may be used for collision detection between non-convex rigid bodies with an impulse-based contact solver. They combined vertex sampling with edge-based isosurface intersection. We extend their work by capturing point-face contacts where the SDF may not have an embedded mesh representation. In addition, our method allows generating edge-edge contact constraints before intersection occurs through a local minimization. Xu et al. [2014] also used an SDF representation with point-sampling for rigid body contact, but with an implicit penalty-based contact model. Their work was later extended to the case of continuous collision detection (CCD) between shapes represented by SDFs [Xu and Barbic 2014]. Volume contact models [Allard et al. 2010; Wang et al. 2012] share some similarities with SDFs representations. Volume contact methods work by identifying the overlapping volume of shapes and introducing constraints to remove/minimize it. In this work we are concerned with how to generate contacts between an arbitrary mesh and a solid represented by an SDF. In contrast to volume based approaches, our method also handles the case of thin-shells, e.g.: cloth, colliding against a rigid body. Weidner et al. [2018] propose a hybrid Eulerian-on-Lagrangian with remeshing [Narain et al. 2012] to allow cloth to slide over sharp features and conform closely to the collision shape. For simplicity we do not perform remeshing, but generate smoothly varying contacts along the simulation mesh faces.

Deformable Bodies. There is a large body of work on collision detection between deformable bodies [Teschner et al. 2005]. Surface-based methods [Bridson et al. 2002; Harmon et al. 2008; Provot 1997; Selle et al. 2008; Stam 2009] typically aim to detect and prevent interpenetration between surface elements using a combination of continuous collision detection (CCD) and fail-safe methods. One advantage of SDFs is that, they provide robust inside/outside information, thus if penetration occurs it can often be recovered from. SDFs are usually considered static, although some authors have extended them to handle runtime deformations. Seyb et al. [2019] recently applied deformed sphere tracing for particle collisions. Fisher and Lin [2001] proposed a method to warp SDFs based on a tetrahedral embedding for collision against deformable objects. McAdams et al. [2011] use a similar idea with point sampling of a surface mesh to perform character self-collision. We apply our optimization-based contact generation method to these approaches for accurate and robust contact between deformable bodies and cloth.

Representation. SDFs are commonly discretized and stored on a grid, or volume texture. To avoid the memory overhead for dense volumes, adaptive storage methods have been proposed [Frisken

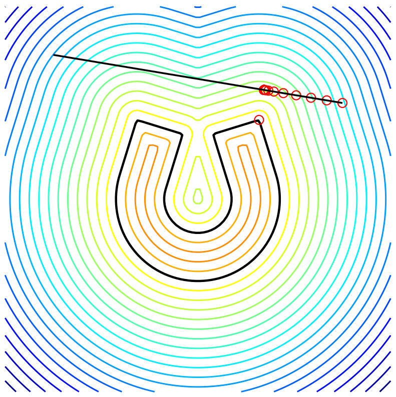

Fig. 3. We find the closest point between a mesh element (a line segment), and the surface of a shape represented by a signed distance field (horseshoe). Here we visualize the iterates (red dots) to show the progress of our optimization-based procedure.

et al. 2000; Liu and Kim 2013]. Non-manifold and sparse representations have also been proposed to allow representing thin features efficiently [Mitchell et al. 2015]. Koschier et al. [2016] proposed a highly accurate SDF representation based on hp-adaptive grids. They used the resulting representation for collision detection by point-sampling the surface. Our method is agnostic with regard to the underlying SDF representation, and may be used with analytic, dense, or sparse representations.

3 FACE CONTACT

We first consider the case of colliding a single triangle against a solid body represented as an SDF. We use the *closest point* methodology for generating contact points [Erleben 2018] which requires finding the closest points on the face and the SDF isosurface. For a triangle with vertices $\mathbf{p}, \mathbf{q}, \mathbf{r} \in \mathbb{R}^3$, the closest point on the triangle to the rigid body is given by the solution to the following minimization over the barycentric coordinates u, v, w,

$$\underset{u,v,w}{\text{argmin}} \quad \phi(u\mathbf{p} + v\mathbf{q} + w\mathbf{r}) \tag{1}$$

$$\text{s.t.} \quad u, v, w \geq 0 \tag{2}$$

$$u + v + w = 1. \tag{3}$$

The constraints ensure that the solution to this problem lies in the triangle interior or on its boundary. In the following sections we examine some numerical methods for solving this optimization problem.

3.1 Projected Gradient Descent

SDFs may represent arbitrary shapes, as such the objective (1) in the preceding minimization is in general a nonlinear, and non-convex function. Global optimization of such functions is difficult, and may require sophisticated methods such as branch and bound approaches [Horst and Tuy 2013]. An alternative method that often works well to find a *local minimum* (or at least a stationary point) is projected gradient descent [Rosen 1960, 1961]. To apply gradient descent we need the derivative of the distance field with respect to the barycentric coordinates, which is given by:

$$\mathbf{d} = \begin{bmatrix} \frac{\partial \phi}{\partial u} & \frac{\partial \phi}{\partial v} & \frac{\partial \phi}{\partial w} \end{bmatrix}^T = \begin{bmatrix} \nabla\phi(\mathbf{x})\mathbf{p} \\ \nabla\phi(\mathbf{x})\mathbf{q} \\ \nabla\phi(\mathbf{x})\mathbf{r} \end{bmatrix} \tag{4}$$

where the spatial point $\mathbf{x}(u, v, w) = u\mathbf{p} + v\mathbf{q} + w\mathbf{r}$ is simply a barycentric interpolation of the triangle vertices. The SDF gradient $\nabla\phi(\mathbf{x})$ w.r.t. spatial coordinates may be computed on-demand using finite differences, analytic gradients, or precomputed and stored on a grid. After computing the gradient $\mathbf{d}$ w.r.t. barycentric coordinates, we iteratively update our candidate solution $\mathbf{c}$ according to a fixed step size α, and project onto the constraint manifold, using the fixed-point iteration

$$\mathbf{c}_{i+1} \leftarrow \mathbb{P}(\mathbf{c}_i - \alpha\mathbf{d}). \tag{5}$$

where $\mathbf{c} = [u, v, w]^T$ is the vector of barycentric coordinates, and i the iteration index. We visualize the progress of this method on a 2D problem in Figure 3.

The projection operator $\mathbb{P}$ maps a point to the closest point on the constraint manifold described by (2)-(3). Geometrically, these constraints represent a triangle centered around the origin in 3D with vertices $[1, 0, 0]$, $[0, 1, 0]$, $[0, 0, 1]$. The projection therefore, amounts to finding the closest point on this triangle from our current iterate, for which efficient codes exist [Ericson 2004]. We note that it is also possible to simplify the problem by expressing w in terms of u and v such that the closest point projection can now be performed in 2D instead of 3D.

3.2 Frank-Wolfe

The Frank-Wolfe, or *conditional gradient method* is an iterative algorithm for solving constrained convex optimization problems [Frank and Wolfe 1956; Jaggi 2013]. Similar to gradient descent, Frank-Wolfe only requires access to first order derivatives, however it has the appealing property that it does not require a projection step onto the constraint manifold. The Frank-Wolfe algorithm proceeds by iteratively solving the following minimization for the point $\mathbf{s}_i \in \mathbb{R}^3$,

$$\underset{\mathbf{s}_i}{\text{argmin}} \quad \mathbf{s}_i^T \nabla\phi(\mathbf{x}_i) \tag{6}$$

$$\text{s.t.} \quad \mathbf{s}_i \in \mathcal{D}, \tag{7}$$

followed by the update:

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \alpha(\mathbf{s}_i - \mathbf{x}_i). \tag{8}$$

here $\mathcal{D}$ is the domain to optimize over, in our case the triangle itself. Unlike projected gradient descent which optimizes over barycentric coordinates $\mathbf{c}$, for Frank-Wolfe it is more natural to optimize over spatial coordinates $(\mathbf{x})$ directly. Due to convexity of the triangle, the solution $\mathbf{s}_i$ must be one of the vertices $\mathbf{p}, \mathbf{q}, \mathbf{r}$, which can be computed by direct enumeration. This is similar to the problem of finding the support point for a given direction in the Gilbert-Johnson-Keerthi (GJK) algorithm [1988]. Once this extremal point $\mathbf{s}_i$ has been identified, the Frank-Wolfe method performs a step (8) where the step

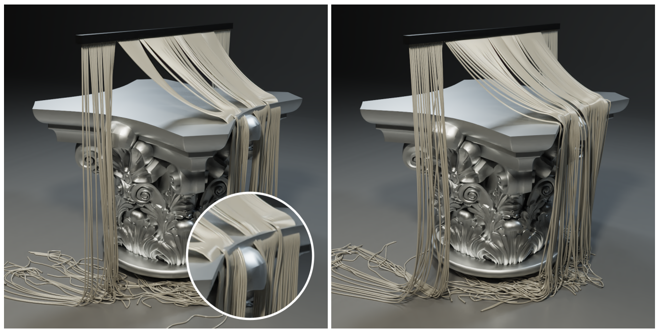

Fig. 4. An example showing collision of 128 ropes consisting of a string of 1D elements. Point-based sampling misses edge-edge contacts leading to interpenetration across the sharp SDF edge (left). We use golden-section search to find the closest point along each segment to the isosurface (right).

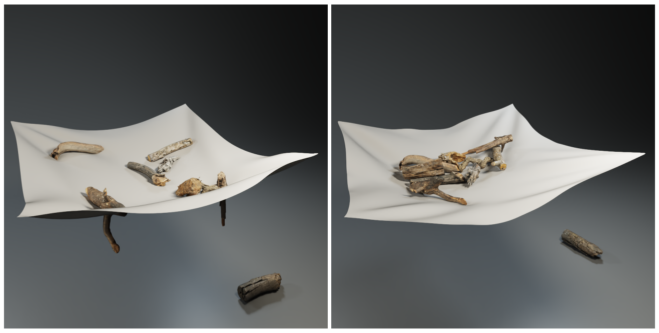

Fig. 5. A cloth net supporting sharp non-convex objects shows visible penetrations for simple point sampling approaches (left), while optimized face contacts were robust (right). In this example the cloth net consists of 4.8k faces, the rigid branches are represented as SDFs with embedded triangle meshes consisting of between 2-27k faces each. Collision detection time is around 350us per-timestep.

length has the particular form $\alpha = \frac{2}{i+2}$ to ensure convergence. Although Frank-Wolfe is designed for convex functions we found it to be effective for non-convex optimization.

3.3 Culling and Starting Iterate

To quickly reject triangles from processing we evaluate the distance at the centroid $\mathbf{x}_c$ of the triangle and check if it is less than the radius r of the bounding sphere centered at this point, i.e.: $\phi(\mathbf{x}_c) < r$. The same optimization applies for mesh edges as discussed in Section 4. We found this to be a highly effective early out for rigid body contact, however as may be expected, it is less effective for cloth in close contact with a body as illustrated in Figure 13.

The choice of starting iterate is important to the convergence of iterative methods. A simple heuristic we found effective is to evaluate ϕ at each triangle vertex and choose the point with the smallest value. This is also helpful in avoiding local minima, although it is not sufficient to guarantee a global solution.

3.4 Termination Conditions

Since a lower bound on the distance for each element is not known in advance we cannot define an absolute tolerance on the distance function $\phi(\mathbf{x})$. However, for projected gradient descent we may define a stopping criteria based on the magnitude of the gradient $\mathbf{d}$. For Frank-Wolfe the subproblem objective $\mathbf{s}_i^T \nabla\phi(\mathbf{x}_i)$ may be used, since for convex functions this is an upper bound on the primal error [Jaggi 2013]. After termination, if the minimum distance lies within some threshold margin $\phi(\mathbf{x}) < \delta$ then we create a contact constraint. Otherwise, the element is deemed not to collide and no constraint is generated.

4 EDGE CONTACT

The face-based optimization presented in the previous section will naturally find contacts that lie on edges. However in many cases we wish to optimize over edges directly, for example when simulating one dimensional objects such as hair, or rope, as in Figure 4. Considering an edge (line segment) defined between two points $\mathbf{p}, \mathbf{q} \in \mathbb{R}^3$ we can define the closest point to the isosurface of an SDF as the solution to the single variable optimization problem:

$$\underset{u}{\text{argmin}} \quad \phi(u\mathbf{p} + (1-u)\mathbf{q}) \tag{9}$$

$$\text{s.t.} \quad u \geq 0 \tag{10}$$

$$u \leq 1. \tag{11}$$

Projected gradient descent and Frank-Wolfe can be applied directly to this problem, however the lower dimensionality makes search based methods attractive. A simple and robust method for solving interval-constrained optimization problems is *golden-section search* [Kiefer 1953]. Golden-section search is a generalization of the root finding bisection method for minimizing unimodal functions. Although our SDF function may not be unimodal over the interval [0, 1], golden-section search will terminate on one of the minima, or the boundary of the interval. An advantage of golden-section search is that it does not require gradients of the objective function, making it particularly efficient.

5 MODELS

We now discuss how to apply our presented methods to the case of collision detection for a range of common simulation models in computer graphics.

5.1 Cloth Collision

To collide deformable cloth represented by a triangle mesh against shapes represented as SDFs, we perform the local face optimization of Section 3 for each triangle in the mesh. This approach generates one contact per-triangle, which we found was sufficient for our tests, as illustrated in Figures 5 & 6. To avoid creating repeated contacts between connected faces we use the representative triangle approach, which uniquely assigns mesh features (vertices, edges, faces) as to each element as a greedy preprocess [Curtis et al. 2008; Moravanszky et al. 2004]. After optimization a contact may be discarded if it lies on a feature not assigned to the element.

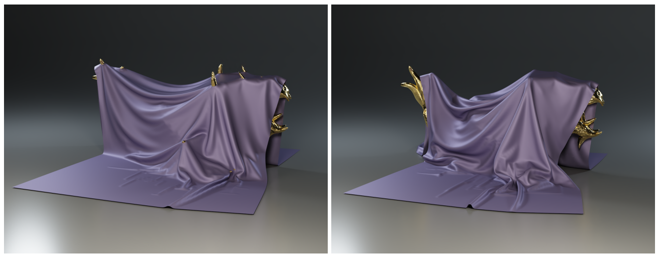

Fig. 6. Point sampling shows visible penetrations through the sharp features of a dragon (left), our method prevents interpenetration by creating optimized local contacts (right). In this example the cloth consists of 20k triangles, collision detection against the SDF dragon takes around 80us per-timestep.

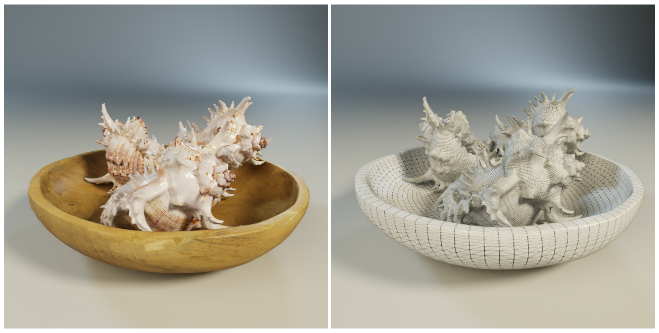

Fig. 7. A high resolution rigid body simulation. Each shell consists of 129k triangles and contains many sharp features. Collision detection time using per-face optimization is 0.4ms on average.

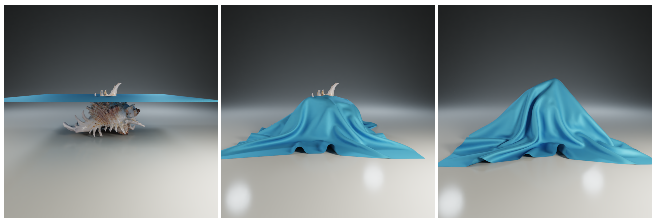

Fig. 8. Given an interpenetrating initial state (a), triangle-based collision will not resolve intersections (b). SDFs provide inside/outside information that allows them to reliably separate objects (c).

5.2 Rigid Body Collision

For collision between rigid bodies that are well tessellated e.g.: the shells in Figure 7, we found that optimization over faces was sufficient to generate good contact manifolds. However, for shapes where this was not true, e.g.: the elongated boxes in Figure 2, we found that descent based methods would converge slowly, particularly for edge-edge contacts with high curvature. To avoid this problem we can instead test vertices of both meshes directly as in Guendelman et al. [2003], followed by the search based optimization of Section 4 over each edge.

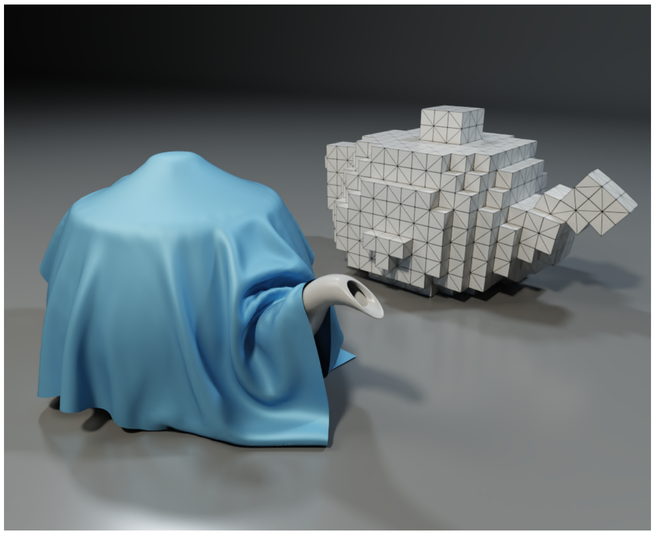

Fig. 9. Our method can be used with deformable SDF embeddings. Here cloth is collided against a deformable teapot using a tetrahedral FEM model. While the tetrahedral cage is relatively coarse (background), collision against the teapot is performed against the embedded high resolution SDF (foreground).

5.3 Soft Body Collision

SDFs may be used for collision against deformable shapes by embedding the distance field inside a deformable cage [Fisher and Lin 2001; McAdams et al. 2011]. Given a point in world space coordinates $\mathbf{x}_s$, we associate a material space point $\mathbf{x}_m = \mathbf{g}(\mathbf{x}_s)$ where $\mathbf{g} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is an operator that maps from spatial coordinates back to the material pose, i.e.: an inverse displacement function. Given such a mapping we can define a deformed distance field,

$$\phi_d = \phi_m(\mathbf{g}(\mathbf{x}_s)), \tag{12}$$

where ϕ_m is the (static) distance field defined in material space. We note that the deformed field may no longer satisfy the distance field property $\|\nabla\phi_d\| = 1$, however the isosurface is not changed by this transformation, and it is sufficient to find a local minimum and define contact constraints. To apply our method to deformable bodies we follow the approach of McAdams et al. [2011]. Given a deformed tetrahedral mesh, we first find the tetrahedron enclosing a world space point, and then compute the material space point associated with it. The enclosing tetrahedron with vertices $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ may be found efficiently using a bounding volume hierarchy (BVH) or other spatial data structure. Assuming linear shape functions, the projection back to material space is given by,

$$\mathbf{x}_m = \mathbf{g}(\mathbf{x}_s) = \mathbf{M}\mathbf{D}^{-1}(\mathbf{x}_s - \mathbf{v_0}_s) + \mathbf{v_0}_m. \tag{13}$$

We use the subscript s and m to denote world space and material space quantities respectively. Here $\mathbf{D} \in \mathbb{R}^{3\times3}$ is a basis for the enclosing deformed tetrahedron in world space, and $\mathbf{M} \in \mathbb{R}^{3\times3}$ is a basis for the undeformed tetrahedron in material space:

$$\mathbf{M} = \begin{bmatrix} \mathbf{v}_{1m} - \mathbf{v}_{0m} & \mathbf{v}_{2m} - \mathbf{v}_{0m} & \mathbf{v}_{3m} - \mathbf{v}_{0m} \end{bmatrix} \tag{14}$$

$$\mathbf{D} = \begin{bmatrix} \mathbf{v}_{1s} - \mathbf{v}_{0s} & \mathbf{v}_{2s} - \mathbf{v}_{0s} & \mathbf{v}_{3s} - \mathbf{v}_{0s} \end{bmatrix}. \tag{15}$$

Given the mapping function $\mathbf{g}$, and a world space point on a triangle defined by

$$\mathbf{x}_s(u, v, w) = u\mathbf{p} + v\mathbf{q} + w\mathbf{r}, \tag{16}$$

the gradient of $\phi_d(\mathbf{x}_s)$ w.r.t. the barycentric coordinates for optimization is given by,

$$\mathbf{d} = \begin{bmatrix} \frac{\partial \phi_d}{\partial u} & \frac{\partial \phi_d}{\partial v} & \frac{\partial \phi_d}{\partial w} \end{bmatrix}^T = \begin{bmatrix} \nabla\phi_m(\mathbf{x_m})\mathbf{G}(\mathbf{x_s})\mathbf{p} \\ \nabla\phi_m(\mathbf{x_m})\mathbf{G}(\mathbf{x_s})\mathbf{q} \\ \nabla\phi_m(\mathbf{x_m})\mathbf{G}(\mathbf{x_s})\mathbf{r} \end{bmatrix}, \tag{17}$$

where $\mathbf{G} = \nabla\mathbf{g} = \mathbf{M}\mathbf{D}^{-1}$. The matrix $\mathbf{G}$ may also be identified as the inverse of the deformation gradient commonly computed in tetrahedral FEM simulations [Sifakis and Barbic 2012]. As $\mathbf{D}$ is a basis for the deformed tetrahedron, if the element is extremely ill-conditioned calculating the inverse may not be possible. A robust implementation should detect this case and may choose to ignore such elements.

We note that $\mathbf{G}$ is a function of the world space point $\mathbf{x}_s$, and typically involves traversal of a BVH to determine the enclosing tetrahedron. To avoid performing this work for each optimization step, we evaluate $\mathbf{G}$ at the centroid of a triangle and treat it as constant over the course of the optimization. Following [McAdams et al. 2011], once we have found a minimum of ϕ_d we project the minimum point back onto the surface (in material space), and apply the deformation mapping back to world-space to define the contact point. We illustrate this approach in Figure 9, where cloth is draped over a deformable teapot. A key advantage of this method is that it decouples the elastic discretization from the collision representation. This allows us to use a regular and relatively coarse hexahedral grid (decomposed into tetrahedra) for the computation of internal elastic dynamics, while using a high resolution SDF representation to resolve collision.

6 DISCRETE DISTANCE FIELDS

SDFs are often stored as discrete samples on a regular or sparse grid [Koschier et al. 2016]. This approach allows representing complex non-linear functions. However using a finite extent discretization requires some special handling during contact generation. Since optimization can only proceed where there is valid SDF data, a problem occurs when there is not a sufficient margin between the surface of the shape and the extents of the SDF volume data. A possible solution to this is to project samples lying outside the volume's bounding box to the closest point on the box's surface. However we found it was sufficient to use a conservative sampling margin around the shape's isosurface, and for simple shapes to use the analytic approach suggested below.

7 ANALYTIC DISTANCE FIELDS

Closed form functions for the distance to simple shapes can often be found, and these can be efficiently combined to model complex

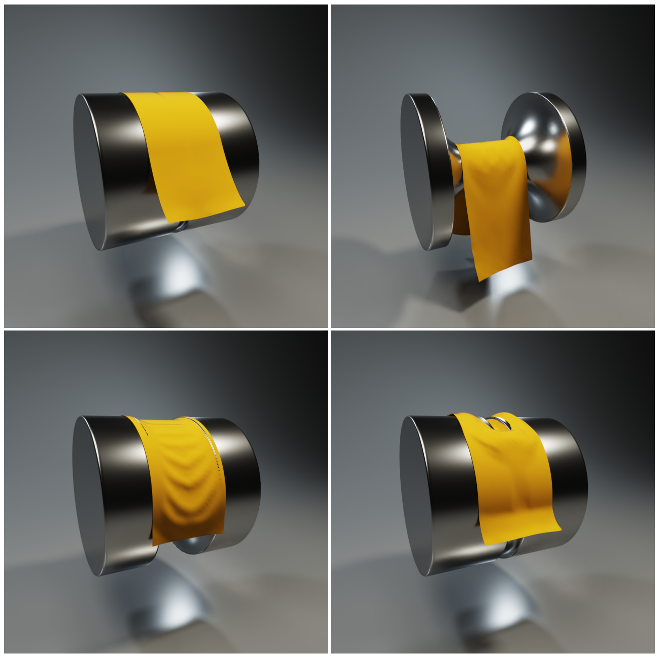
Fig. 10. Top Row: Our method naturally supports collision against analytic SDF functions without the requirement for an embedded mesh representation. In this case cloth collides against a deforming cylinder with smoothly varying face and edge contacts. Bottom Row: Point based sampling misses edge contacts, and eventually leads to cloth becoming separated and tangled.

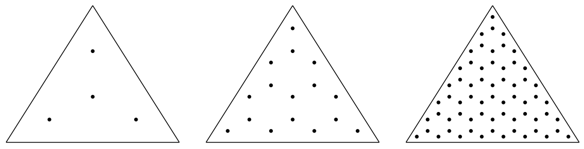
Fig. 11. Triangle point sampling for 4, 16, and 64 samples (left to right) using a low discrepancy sampling pattern. Despite providing well distributed coverage, discrete sampling may miss contacts between sharp features, and cannot provide smooth transitions for edge-edge contacts.

geometry. Our method is independent of the SDF representation, so we can naturally support analytic field definitions. This allows us to collide meshes against arbitrarily smooth surfaces, without the need to form a surface triangulation. In addition, parameterized shapes can be animated to deform collision geometry over time as illustrated in Figure 10.

8 RESULTS

Since our method naturally parallelizes over mesh features we have implemented it in CUDA and run it on an NVIDIA GTX 2080 Ti. For dynamics simulation we use the extended position-based dynamics (XPBD) method [Macklin et al. 2016; Müller et al. 2007] where contacts are handled as hard unilateral constraints.

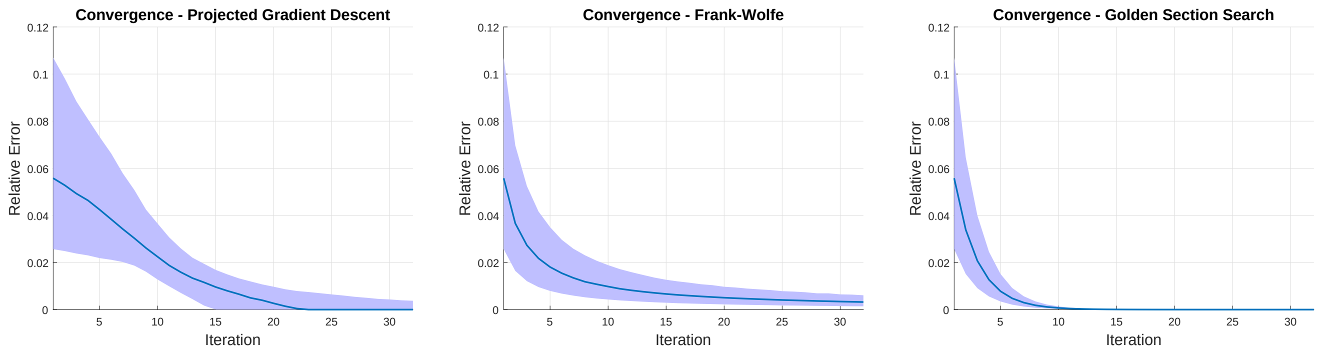

Fig. 12. We analyze the convergence of projected gradient descent (left), Frank-Wolfe (middle), and golden-section search (right) over a random distribution of 1000 closest point problems. We randomly position a 2D segment over the domain (possibly intersecting the SDF) and plot the median, first and third quartile (shaded area) of the relative forward error at each iteration. To compute the relative error we use a ground truth solution obtained using high resolution brute-force sampling over the entire element.

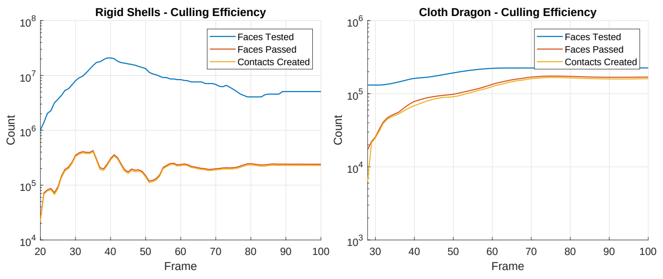

Fig. 13. Sphere culling faces against the SDF is highly effective for rigid body scenes (left). It is less effective for scenes where cloth is draped closely over an object (right). In both cases we see that the final number of contacts created is close to the number passing bounding sphere checks, indicating good culling efficiency.

8.1 Convergence

To compare the convergence of the three methods we position a line segment randomly over the domain shown in Figure 3, and record the error at each iteration. Figure 12 shows the error distribution over 1000 random tests. Comparing the gradient-based methods, we see that projected gradient descent often reaches a lower error after the same number of iterations as Frank-Wolfe. However we note that the behavior of gradient descent is quite sensitive to the step-length used. In contrast, Frank-Wolfe uses a decreasing step length, which slows convergence but removes the need to set parameters, making it attractive from an authoring perspective. Observing the error plot for golden-section search in Figure 12 (right), we see it converges quickly on average, and has the tightest error distribution. Hence, if we have a 1D optimization problem where search methods are applicable we believe they should be preferred over descent based methods.

Descent-based optimization occurs using fixed step lengths in barycentric coordinates, this means that convergence is independent of the world-space size of the triangle. Rather, convergence rate primarily depends on the complexity and conditioning of the SDF function over the mesh feature. As is typical for descent based methods, minima in flat regions of low curvature will cause slower convergence.

Table 1. Performance timings comparing a simple point-based approach (Simple), to our face-based optimization. Here we have fixed the number of iterations to 32 for all methods, and report Golden-section search numbers for the rope example only, since it is a 1D problem, while the other examples have used face-based optimization.

Example	SDF	Points	Elements	Simple	PGD	FW	GS
Unit	#	#	#	μs	μs	μs	μs
Dragon	256^3	20,000	39,402	20	85	77	-
Net	256^3	51,161	81,194	336	373	321	-
Teapot	128^3	5,281	6,962	112	123	115	-
Shells	512^3	635,020	1,269,880	401	445	437	-
Rope	256^3	4224	4096	98	102	99	99
Analytic	-	10,000	19,502	48	48	48	-

8.2 Performance

In Table 1 we report the per-time step average collision times for our examples. Despite performing a local optimization loop per-triangle, collision detection times are still a fraction of a millisecond (we report timings in microseconds) for complex geometry. A surprising finding is that in many cases overall collision detection performance was not significantly more costly than simple point sampling. We attribute this partly to effective early culling as shown Figure 13, but also to the fact that once SDF data enters the GPU cache it is fast to sample. This is even more apparent with the analytic test case shown in Figure 10, which has identical performance for all three methods. This is explained by the fact that, in this example, the optimization is purely compute-based (no memory fetches) and so total cost is small compared to the rest of the kernel and falls below our ability to measure it.

8.3 Comparison To Sampling Approaches

One way to improve point sampled collision is to simply increase the number of samples. To compare our optimization method against this approach we generate face samples using the low-discrepancy sampling scheme of Basu and Owen [2015] as illustrated in Figure 11. Using 64 face samples we were unable to obtain an interpenetration free result for the cone-cloth example. This illustrates how, for sharp features, even high sampling frequencies are insufficient to guarantee an interpenetration free state. On the cloth dragon example, which has a higher-resolution base cloth mesh, we found 4 additional face samples was sufficient to prevent interpenetration, however this approach generated approximately 80k contacts, compared with the 20k generated by face-based optimization.

8.4 Comparison To Surface Approaches

A popular method for collision detection is to perform contact generation between mesh surface features by examining triangle pairs, either in proximity, or using CCD checks. We implemented a surface-based contact generation scheme where point-face and edge-edge pairs are tested for proximity and a contact generated if the closest points lie below a distance threshold. We employ a GPU AABB tree [Karras 2012] to cull triangle pairs. On the rigid shells example this mesh-based collision took approximately 15ms per-step, compared to < 0.5ms using SDF-based contact. The performance difference may be explained by the fact each shell contains approximately 129k triangles, which results in deep hierarchy traversals. This is not a perfect comparison, since SDFs implicitly resample the surface geometry (possibly losing detail), and because our approach generates exactly one contact per-triangle rather than one per-feature pair. Nevertheless, we found this result to be representative on a range of high resolution meshes. A more concerning limitation of surface-based approaches is that once interpenetration has occurred it is difficult to recover from. A primary advantage of SDFs is their ability to provide inside/outside information, allowing points to be projected to the surface. In Figure 8 we illustrate this by embedding cloth in a complex shape. Triangle-based contact remains entangled, while SDF-based contact can reliably separate both objects.

9 LIMITATIONS

While our method improves the robustness of mesh contact against SDFs, some issues remain. Since our method is local, it does not generate a global minimum translation distance (MTD) that would guarantee separation of already interpenetrating elements. In practice we found this was not a significant problem, and that deep penetrations are typically resolved within a few time-steps. A second limitation occurs when a triangle or edge is resting on multiple features (repeated local minima). Since we generate one contact per-element this may result in interpenetration. A possible solution would be to perform an implicit subdivision of each face. If local minima are isolated they can be used to bracket the element into smaller pieces and recursively search on these smaller subspaces.

10 CONCLUSION AND FUTURE WORK

Due to their efficiency, flexibility, and robustness we believe SDFs are well suited to shape representation for simulation. We have presented a local optimization-based technique that improves the quality of contact generation for meshes against SDFs. We have focused on first-order methods for their simplicity. However, second-order methods such as Newton, quasi-Newton, or accelerated methods could also be employed to improve convergence rate. For future work we plan to explore contact generation between two solids both represented purely as SDFs, without the requirement for any mesh-based surface representation. Producing the intersection of two SDFs is straightforward, which suggests they may be well suited for volume-based contact approaches.

ACKNOWLEDGMENTS

The authors wish to thank Tae-Yong Kim and Richard Tonge for valuable discussions and feedback on the manuscript. We thank the NVIDIA Omniverse team for their help preparing the visualizations in this paper. In addition, we wish to thank Inigo Quilez for the horseshoe SDF used in Figure 3, and the Stanford Computer Graphics Laboratory for providing XYZRGB dragon model.

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 (2010), 82.

Kinjal Basu and Art B Owen. 2015. Low discrepancy constructions in the triangle. *SIAM J. Numer. Anal.* 53, 2 (2015), 743–761.

James F Blinn. 1982. A generalization of algebraic surface drawing. *ACM Trans. Graph.* 1, 3 (1982), 235–256.

Jules Bloomenthal and Brian Wyvill. 1997. *Introduction to Implicit Surfaces.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques* (San Antonio, Texas) *(SIGGRAPH 02).* Association for Computing Machinery, New York, NY, USA, 594–603. https://doi.org/10.1145/566570.566623

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* (Redwood City, California) *(I3D '08).* Association for Computing Machinery, New York, NY, USA, 61–69. https://doi.org/10.1145/1342250.1342260

Christer Ericson. 2004. *Real-Time Collision Detection.* CRC Press, Inc., USA.

Kenny Erleben. 2018. Methodology for Assessing Mesh-Based Contact Point Methods. *ACM Transactions on Graphics (TOG)* 37, 3 (2018), 39.

Susan Fisher and Ming C. Lin. 2001. Deformed Distance Fields for Simulation of Non-Penetrating Flexible Bodies. In *Computer Animation and Simulation 2001*, Nadia Magnenat-Thalmann and Daniel Thalmann (Eds.). Springer Vienna, Vienna, 99–111.

Marguerite Frank and Philip Wolfe. 1956. An algorithm for quadratic programming. *Naval research logistics quarterly* 3, 1-2 (1956), 95–110.

Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. 2000. Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00).* ACM Press/Addison-Wesley Publishing Co., USA, 249–254. https://doi.org/10.1145/344779.344899

Arnulph Fuhrmann, Gerrit Sobotka, and Clemens Gross. 2003. Distance fields for rapid collision detection in physically based modeling. In *International Conference on Computer Graphics and Vision '03. Proceedings.* Eurographics, Moscow, Russia.

Elmer G Gilbert, Daniel W Johnson, and S Sathiya Keerthi. 1988. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal on Robotics and Automation* 4, 2 (1988), 193–203.

Eran Guendelman, Robert Bridson, and Ronald Fedkiw. 2003. Nonconvex Rigid Bodies with Stacking. *ACM Trans. Graph.* 22, 3 (July 2003), 871–878. https://doi.org/10.1145/882262.882358

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

John C Hart. 1996. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer* 12, 10 (1996), 527–545.

Reiner Horst and Hoang Tuy. 2013. *Global optimization: Deterministic approaches.* Springer Science & Business Media, Berlin Heidelberg.

Martin Jaggi. 2013. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In *Proceedings of the 30th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Sanjoy Dasgupta and David McAllester (Eds.), Vol. 28. PMLR, Atlanta, Georgia, USA, 427–435.

Tero Karras. 2012. Maximizing Parallelism in the Construction of BVHs, Octrees, and k-d Trees. In *Proceedings of the Fourth ACM SIGGRAPH / Eurographics Conference on High-Performance Graphics* (Paris, France) *(EGGH-HPG'12).* Eurographics Association, Goslar, DEU, 33–37.

Jack Kiefer. 1953. Sequential minimax search for a maximum. *Proceedings of the American mathematical society* 4, 3 (1953), 502–506.

Dan Koschier, Crispin Deul, and Jan Bender. 2016. Hierarchical Hp-Adaptive Signed Distance Fields. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Zurich, Switzerland) *(SCA '16)*. Eurographics Association, Goslar, DEU, 189–198.

Fuchang Liu and Young J Kim. 2013. Exact and adaptive signed distance fields computation for rigid and deformable models on gpus. *IEEE transactions on visualization and computer graphics* 20, 5 (2013), 714–725.

Miles Macklin, Matthias Müller, and Nuttapong Chentanez. 2016. XPBD: Position-Based Simulation of Compliant Constrained Dynamics. In *Proceedings of the 9th International Conference on Motion in Games* (Burlingame, California) *(MIG '16)*. Association for Computing Machinery, New York, NY, USA, 49–54. https://doi.org/10.1145/2994258.2994272

Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. 2011. Efficient Elasticity for Character Skinning with Contact and Collisions. *ACM Trans. Graph.* 30, 4, Article 37 (July 2011), 12 pages. https://doi.org/10.1145/2010324.1964932

Nathan Mitchell, Mridul Aanjaneya, Rajsekhar Setaluri, and Eftychios Sifakis. 2015. Non-manifold level sets: A multivalued implicit surface representation with applications to self-collision processing. *ACM Trans. Graph.* 34, 6 (2015), 247.

Adam Moravanszky, Pierre Terdiman, and A Kirmse. 2004. Fast contact reduction for dynamics simulation. *Game programming gems* 4 (2004), 253–263.

Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position based dynamics. *J. Vis. Comun. Image Represent.* 18, 2 (April 2007), 109–118.

Rahul Narain, Armin Samii, and James F O'Brien. 2012. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph.* 31, 6 (2012), 152.

Xavier Provot. 1997. Collision and self-collision handling in cloth model dedicated to design garments. In *Computer Animation and Simulation '97*, Daniel Thalmann and Michiel van de Panne (Eds.). Springer Vienna, Vienna, 177–189.

J. B. Rosen. 1960. The Gradient Projection Method for Nonlinear Programming. Part I. Linear Constraints. *J. Soc. Indust. Appl. Math.* 8, 1 (1960), 181–217.

J. B. Rosen. 1961. The Gradient Projection Method for Nonlinear Programming. Part II. Nonlinear Constraints. *J. Soc. Indust. Appl. Math.* 9, 4 (1961), 514–532.

Andrew Selle, Jonathan Su, Geoffrey Irving, and Ronald Fedkiw. 2008. Robust high-resolution cloth using parallelism, history-based collisions, and accurate friction. *IEEE transactions on visualization and computer graphics* 15, 2 (2008), 339–350.

Dario Seyb, Alec Jacobson, Derek Nowrouzezahrai, and Wojciech Jarosz. 2019. Non-linear Sphere Tracing for Rendering Deformed Signed Distance Fields. *ACM Trans. Graph.* 38, 6, Article 229 (Nov. 2019), 12 pages.

Eftychios Sifakis and Jernej Barbic. 2012. FEM Simulation of 3D Deformable Solids: A Practitioner's Guide to Theory, Discretization and Model Reduction. In *ACM SIGGRAPH 2012 Courses* (Los Angeles, California) *(SIGGRAPH '12)*. Association for Computing Machinery, New York, NY, USA, Article 20, 50 pages. https://doi.org/10.1145/2343483.2343501

J. Stam. 2009. Nucleus: Towards a unified dynamics solver for computer graphics. In *2009 11th IEEE International Conference on Computer-Aided Design and Computer Graphics*. IEEE, USA, 1–11. https://doi.org/10.1109/CADCG.2009.5246818

Matthias Teschner, Stefan Kimmerle, Bruno Heidelberger, Gabriel Zachmann, Laks Raghupathi, Arnulph Fuhrmann, M-P Cani, François Faure, Nadia Magnenat-Thalmann, Wolfgang Strasser, et al. 2005. Collision detection for deformable objects. In *Computer graphics forum*, Vol. 24. Wiley Online Library, USA, 61–81.

Bin Wang, François Faure, and Dinesh K. Pai. 2012. Adaptive Image-based Intersection Volume. *ACM Trans. Graph.* 31, 4, Article 97 (July 2012), 9 pages.

Nicholas J Weidner, Kyle Piddington, David IW Levin, and Shinjiro Sueda. 2018. Eulerian-on-lagrangian cloth simulation. *ACM Trans. Graph.* 37, 4 (2018), 50.

Brian Wyvill, Andrew Guy, and Eric Galin. 1999a. Extending the CSG Tree. Warping, Blending and Boolean Operations in an Implicit Surface Modeling System. *Computer Graphics Forum* 18, 2 (1999), 149–158. https://doi.org/10.1111/1467-8659.00365

Brian Wyvill, Andrew Guy, and Eric Galin. 1999b. Extending The CSG Tree. Warping, Blending and Boolean Operations in an Implicit Surface Modeling System. *Comput. Graph. Forum* 18 (06 1999), 149–158.

Hongyi Xu and Jernej Barbic. 2014. Continuous Collision Detection Between Points and Signed Distance Fields. In *Workshop on Virtual Reality Interaction and Physical Simulation*, Jan Bender, Christian Duriez, Fabrice Jaillet, and Gabriel Zachmann (Eds.). The Eurographics Association, Bremen, Germany. https://doi.org/10.2312/vriphys.20141218

Hongyi Xu, Yili Zhao, and Jernej Barbic. 2014. Implicit multibody penalty-based distributed contact. *IEEE Trans. Vis. Comput. Graph.* 20, 9 (2014), 1266–1279.