Consistent penetration depth estimation for deformable collision response

Bruno Heidelberger Matthias Teschner
Richard Keiser Matthias Müller Markus Gross

Computer Graphics Laboratory
ETH Zurich, Switzerland

Email: heidelberger@inf.ethz.ch

Abstract

Penalty approaches can be used to efficiently resolve collisions of dynamically simulated rigid and deformable objects. These approaches compute penalty forces based on the penetration depth of intersecting objects and there exist many algorithms for estimating the exact penetration depth. However, in discrete-time simulations, this information can cause non-plausible collision responses in case of large penetrations or due to the object discretization.

In this paper, we present a method to compute consistent penetration depth information in order to reduce collision response artifacts inherent to existing penetration depth approaches. The method considers a set of close surface features to avoid discontinuous penetration depths. Further, a propagation scheme is applied in case of large penetrations to avoid non-plausible, inconsistent penetration depth information.

1 Introduction

Interactive simulation environments with dynamically deforming objects play an important role in computational surgery and are also of growing interest to games. These environments require efficient and robust methods for basic simulation components, such as deformation, collision detection, and collision response. While efficient deformable models are well-investigated, the detection of collisions between deformable structures only recently gained increasing attention. A survey of current approaches to deformable collision detection can be found in [Tes04b]. Based on existing approaches to deformable modeling and collision detection, scenes consisting of several thousand volumetric elements can be simulated at interactive rates [Tes03, Tes04a].

In order to realistically simulate the behavior of colliding objects, an appropriate collision response has to be considered. One idea commonly used in discrete-time simulations is to generate forces which eventually separate colliding objects. These response or penalty forces are computed for penetrating object vertices as a function of their penetration depth which represents the distance and the direction to the surface of the penetrated object. In case of deformable objects, this force computation is intended to reflect the fact that real colliding objects deform each other. The deformation induces forces in the contact area which are approximated with penetration depth approaches in virtual environments. Response forces commonly consider additional features such as friction which is computed as a function of the relative velocity of colliding structures and their penetration depth.

Penetration depth approaches work very well for sufficiently dense sampled surfaces and in case of small penetrations. However, in interactive discrete-time simulations with discretized object representations, these two requirements are rarely met. Depending on the size of the simulation time step, large penetrations can occur which result in the computation of non-plausible penetration depths and directions. Fig. 1 illustrates this problem. Further, discrete surface representations can result in discontinuous penetration directions. These discontinuities illustrated in Fig. 2 degrade the stability of the response process.

Contribution. In this paper, we present a method to compute consistent penetration depths and directions for colliding tetrahedral meshes with triangulated surfaces. In contrast to approaches that only consider one closest surface feature, the pre-

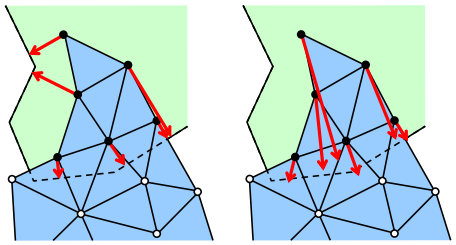

Figure 1: The presented approach addresses the problem of non-plausible penetration depth estimation. Instead of strictly computing minimal distances as illustrated in the left-hand image, the approach computes consistent penetration distances as illustrated in the right-hand image. Therefore, collision response artifacts in discrete-time simulations are significantly reduced.

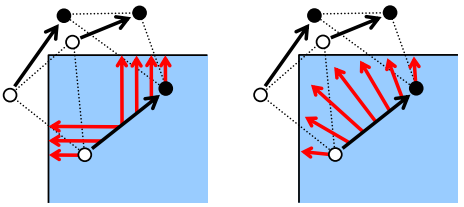

Figure 2: The presented approach also addresses the problem of discontinuous penetration depths for small displacements of penetrating vertices as illustrated in the left-hand image. Instead, smooth and plausible approximations are computed which reduce artifacts in the collision response scheme.

sented approach considers a set of close surface features to significantly reduce discontinuities of estimated penetration depth directions for small displacements of penetrating vertices. Further, a propagation scheme is introduced to approximate the penetration depth and direction for vertices with deep penetrations. This significantly reduces artifacts of the penetration direction in case of large penetrations.

The method works with any underlying deformation model and any contact model that computes penalty forces based on a given penetration depth. The scheme requires a volumetric collision detection approach. Although the method is primarily intended to work with deformable objects, it can also be applied to rigid bodies.

The method has been integrated into a collision response scheme for dynamically deforming tetrahedral meshes. Experiments presented in Sec. 4 illustrate that the scheme significantly reduces artifacts compared to standard penetration depth approaches. It provides a plausible collision response for a wide range of simulation time steps even in case of large object penetrations. The scheme works with objects of any geometrical complexity, but is especially advantageous for coarsely sampled objects.

2 Related Work

Contact models and collision response for rigid and deformable bodies are well-investigated. Analytical methods for calculating the forces between dynamically colliding rigid bodies have been presented in [Moo88, Hah88, Bar89, Bar91, Bar93, Bar94, Fau96, Pau04]. These approaches solve inequality-constrained problems which are formulated as linear complementarity problems (LCP). In addition to analytical methods, a second class of collision response schemes is based on so-called penalty forces. These approaches calculate response forces based on penetration depths in order to resolve colliding objects. First solutions have been presented in [Ter87, Pla88]. Penalty-based approaches have been used in simulations with deformable objects, cloth and rigid bodies [Moo88, McK90, Des99]. A third approach which directly computes contact surfaces of colliding deformable objects is presented in [Gas93].

Due to their computational efficiency, penalty-based approaches are very appropriate for interactive simulations of deformable objects. They can consider various elasto-mechanical object properties. Friction and further surface characteristics can also be incorporated. Penalty forces are computed based on penetration depths and there exist many approaches that compute the exact or approximative penetration depth of two colliding objects which is defined as the minimum translation that one object undergoes to resolve the collision. Exact penetration depth computations can be based on Minkowski sums [Cam86, Gui86] or hierarchical object presentations [Dob93], while approximative solutions based on the GJK algorithm [Gil88] and iteratively expanding polytopes have been presented in [Cam97, Ber01]. Further approaches are

based on object space discretizations [Fis01], employ graphics hardware [Hof02, Sud04], or introduce incremental optimization steps [Kim04].

While existing approaches very efficiently compute the minimal penetration depth, they do not address inconsistency problems of the result in discrete-time simulations (see Figs. 1, 2). One solution to this problem are approaches to continuous collision detection [Red04]. However, these approaches are computationally expensive compared to discrete collision detection approaches and not appropriate for deformable objects. This paper presents an alternative solution to the inconsistency problems. The approach computes penetration depths which significantly reduce artifacts in the respective collision response scheme.

3 Method

This section provides an overview of the proposed algorithm followed by a detailed description of its four stages.

3.1 Algorithm Overview

The method takes a set of potentially colliding tetrahedral meshes as input and computes consistent penetration depths and directions for all colliding mesh points. The method proceeds in four consecutive stages:

Stage 1 detects all *colliding points* in the scene based on a spatial hashing approach (see Sec. 3.2).

Stage 2 identifies all colliding points adjacent to one or more non-colliding points as *border points*. Further, it detects all *intersecting edges* that contain one non-colliding point and one border point. The exact *intersection point* and corresponding surface normal of the penetrated surface are computed for each intersection edge (see Sec. 3.3).

Stage 3 approximates the penetration depth and direction for each border point based on the adjacent intersection points and surface normals obtained from the second stage (see Sec. 3.4).

Stage 4 propagates the penetration depth and direction to all colliding points that are not border points (see Sec. 3.5).

As a result of this algorithm, all colliding mesh points in the scene have an appropriate penetration depth and direction. This information can be used as input to any penalty-based collision response scheme. For our experiments in Sec. 4, we use linear response forces. Further, surface friction is considered.

3.2 Point Collisions

The first stage detects all object points that collide with any tetrahedral mesh in the scene. This volumetric collision detection is accomplished by the spatial hashing approach as presented in [Tes03].

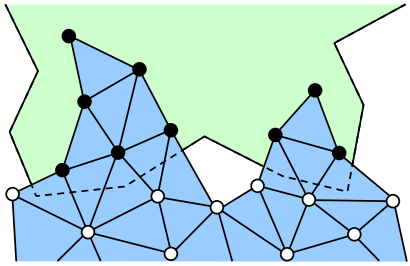

Figure 3: The first stage classifies all mesh points either as colliding points (black) or non-colliding points (white).

Spatial hashing implicitly subdivides $\mathbb{R}^3$ into a hash grid composed of small axis-aligned bounding boxes. The algorithm proceeds in two passes: First, all mesh points are classified with respect to the hash grid cells. Second, the same classification is applied to all tetrahedrons. If a tetrahedron interferes with a cell, all associated points of the cell are checked for collision with the tetrahedron. The actual collision test computes Barycentric coordinates of a point with respect to the tetrahedron in order to detect, whether a point collides with the tetrahedron or not. Refer to [Tes03] for further details. At the end of the first stage, all mesh points in the scene are either classified as colliding points or non-colliding points (see Fig. 3).

3.3 Edge Intersections

The second stage identifies all colliding points with at least one adjacent non-colliding point as border points. The underlying idea is to classify colliding points with respect to their penetration depth. Based on this information, the second stage finds all intersecting edges that contain one non-colliding point and one border point. Moreover, the exact intersection point of each of those edges with the surface

along with the corresponding surface normal of the penetrated mesh is computed. In order to efficiently compute this information, the original spatial hashing approach has been extended to handle collisions between edges and surfaces.

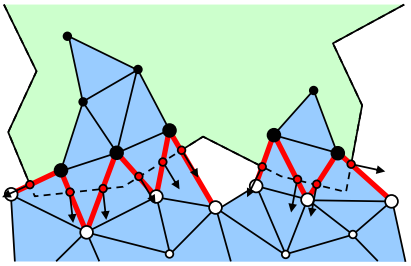

Figure 4: The second stage finds all intersecting edges (red) of the tetrahedral meshes that contain one non-colliding point (white) and one border point (black). Further, the exact intersection point and the corresponding surface normal are computed for each intersection edge.

The extended spatial hashing algorithm employs the same implicit subdivision of $\mathbb{R}^3$ as described in Sec. 3.2. In a first step, all intersecting edges are classified with respect to the hash grid cells by using an efficient voxel traversal technique, e. g. [Ama87]. In a second step, a simplified box-plane intersection test [Gre94] is performed to classify all mesh faces. If a face intersects with a hash grid cell, all associated edges of the cell are checked for intersection with the respective face. The actual intersection test computes Barycentric coordinates of the intersection point in order to detect, whether the edge intersects the face or not. In addition, the Barycentric coordinates can also be used to interpolate a smooth surface normal based on the three vertex normals of the face. This results in a smooth approximation of the penetration direction (see Sec. 3.4).

Each edge can possibly intersect with more than one mesh face. Therefore, only the intersection point nearest to the non-colliding point of the edge is considered in further stages.

At the end of the second stage, each border point is adjacent to one or more intersection edges. Further, all intersecting edges have an exact intersection point and a corresponding surface normal (see Fig. 4).

3.4 Penetration Depth and Direction

The third stage approximates the penetration depth and direction for all border points based on the adjacent intersection points and surface normals computed in the second stage.

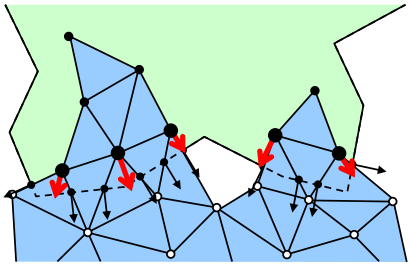

Figure 5: The third stage approximates the penetration depth and direction for all border points based on the adjacent intersection points and surface normals.

First, the influence on a border point is computed for all adjacent intersection points. This influence is dependent on the distance between an intersection and a border point. The respective weighting function has to be positive for all non-zero distances and increasing for decreasing distances. Further, it has to ensure convergence to the penetration depth information with respect to a intersection point $\mathbf{x}_i$ if a colliding point $\mathbf{p}$ approaches $\mathbf{x}_i$. This leads to the following weighting function for the influence $\omega(\mathbf{x}_i, \mathbf{p})$:

$$\omega(\mathbf{x}_i, \mathbf{p}) = \frac{1}{\|\mathbf{x}_i - \mathbf{p}\|^2} \qquad (1)$$

with $\mathbf{x}_i$ denoting an intersection point and $\mathbf{p}$ denoting the border point. The weighting function does not have to be normalized, since this would not avoid any normalization steps in further processing. The weight is undefined for coinciding points. However, the first stage ensures that there is no collision detected in this case. The penetration depth $d(\mathbf{p})$ of a border point $\mathbf{p}$ is now computed based on the influences resulting from (1):

$$d(\mathbf{p}) = \frac{\sum_{i=1}^{k} (\omega(\mathbf{x}_i, \mathbf{p}) \cdot (\mathbf{x}_i - \mathbf{p}) \cdot \mathbf{n}_i)}{\sum_{i=1}^{k} \omega(\mathbf{x}_i, \mathbf{p})} \qquad (2)$$

with $\mathbf{n}_i$ denoting the unit surface normal of the penetrated object surface at the intersection point. The

number of intersection points adjacent to the border point $\mathbf{p}$ is given by k. Finally, the penetration direction $\hat{\mathbf{r}}(\mathbf{p})$ of a border point is computed as a weighted average of the surface normals

$$\hat{\mathbf{r}}(\mathbf{p}) = \frac{\sum_{i=1}^{k}(\omega(\mathbf{x}_i, \mathbf{p}) \cdot \mathbf{n}_i)}{\sum_{i=1}^{k} \omega(\mathbf{x}_i, \mathbf{p})} \tag{3}$$

and the normalized penetration direction $\mathbf{r}(\mathbf{p})$ is obtained as

$$\mathbf{r}(\mathbf{p}) = \frac{\hat{\mathbf{r}}(\mathbf{p})}{\|\hat{\mathbf{r}}(\mathbf{p})\|}. \tag{4}$$

At the end of the third stage, consistent penetration depths and directions have been computed for all border points (see Fig. 5). In contrast to existing penetration depth approaches that consider only one distance, the weighted averaging of distances and directions provides a continuous behavior of the penetration depth function for small displacements of colliding points and for colliding points that are adjacent to each other. Non-plausible penetration directions due to the surface discretization of the penetrated object are avoided.

3.5 Propagation

Based on the computed penetration depth information for border points, the fourth stage propagates the information to all other colliding points that are not border points (see Fig. 6). This is in contrast to existing penetration depth approaches that compute the penetration depth for all points independently. The idea of the propagation scheme is to avoid non-plausible penetration depths in case of large penetrations.

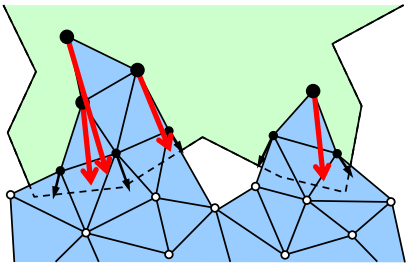

Figure 6: Stage 4 propagates the penetration depth and direction to all colliding points that are not border points.

The propagation is an iterative process that consists of the following two steps: First, the current border points are marked as *processed points*. Second, a new set of border points is identified from all colliding points that are adjacent to one or more processed points. The iteration is aborted, if no new border points are found. Otherwise, the penetration depth and direction for the new border points is computed based on the information available from all adjacent processed points.

Similar to the method described in Sec. 3.4, a weighting function is used to compute the influence $\mu(\mathbf{p}_j, \mathbf{p})$ of an adjacent processed point $\mathbf{p}_j$ on a border point $\mathbf{p}$:

$$\mu(\mathbf{p}_j, \mathbf{p}) = \frac{1}{\|\mathbf{p}_j - \mathbf{p}\|^2}. \tag{5}$$

Based on the influences $\mu(\mathbf{p}_j, \mathbf{p})$, the penetration depth $d(\mathbf{p})$ of a border point $\mathbf{p}$ is computed as:

$$d(\mathbf{p}) = \frac{\sum_{j=1}^{l}(\mu(\mathbf{p}_j, \mathbf{p}) \cdot ((\mathbf{p}_j - \mathbf{p}) \cdot \mathbf{r}(\mathbf{p}_j) + d(\mathbf{p}_j)))}{\sum_{j=1}^{l} \mu(\mathbf{p}_j, \mathbf{p})}$$

with $\mathbf{r}(\mathbf{p}_j)$ denoting the normalized penetration direction of the processed point $\mathbf{p}_j$ and $d(\mathbf{p}_j)$ denoting its penetration depth. The number of processed points adjacent to the border point $\mathbf{p}$ is given by l.

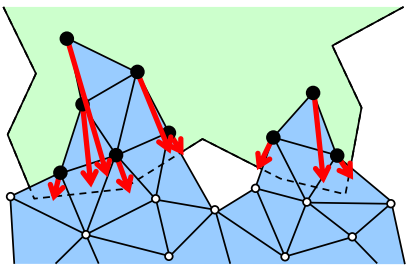

Figure 7: The algorithm computes consistent penetration depths and directions for all colliding points.

Finally, the penetration direction $\hat{\mathbf{r}}(\mathbf{p})$ is computed as a weighted average of the penetration direction of the processed points adjacent to the border point as

$$\hat{\mathbf{r}}(\mathbf{p}) = \frac{\sum_{i=1}^{l} \mu_j \mathbf{r}_j}{\sum_{i=1}^{l} \mu_j}. \tag{6}$$

and normalized

$$\mathbf{r}(\mathbf{p}) = \frac{\hat{\mathbf{r}}(\mathbf{p})}{\|\hat{\mathbf{r}}(\mathbf{p})\|}. \quad (7)$$

At the end of the fourth stage, all colliding points have a consistent penetration depth and direction assigned (see Fig. 7).

4 Results

We have integrated our method in a simulation environment for deformable objects based on [Tes04a]. Various experiments have been carried out to compare the quality and performance of the proposed method with the standard closest-feature approach. All test scenarios presented in this section have been performed on a PC Pentium 4, 3 GHz, GeForce FX Ultra 5900 GPU.

In a first test, two deformable cubes consisting of 1250 tetrahedrons are simulated. Large penetrations between the objects occur due to the high relative velocity and the discrete-time simulation. As illustrated in Fig. 8, the standard approach fails to compute a consistent penetration depth. This results in a non-plausible collision response. Employing our approach to the same scenario results in consistent, plausible penetration depth information.

The second scenario simulates 120 deformable spheres consisting of 2400 tetrahedrons. Starting from a random position, they build a stack of spheres. Computing the penetration depth with the standard approach leads to heavy artifacts. The spheres tend to stick together due to inconsistent handling of penetrated object points. In this case, inconsistent penetration depths and response forces cause non-plausible equilibrium states. By applying our approach, these response artifacts are avoided. Fig. 9 illustrates this second experiment.

A variety of deformable objects are simulated in the third scenario which is illustrated in Fig. 10. Our approach computes consistent penetration information throughout the entire simulation and resolves all collisions in a plausible way. At the end, a stable resting contact between all objects is maintained.

Our approach scales linearly with the number of colliding points. In all experiments presented in this section, an average time of 35 μs is needed for resolving a colliding point. Most time is spent for detecting the edge intersections required by the second stage of the method (see Sec. 3.3). We experienced similar computational costs to calculate the closest feature in the standard approach.

5 Discussion

While the presented approach eliminates many collision response artifacts inherent to existing approaches, there still exist configurations where a plausible collision response can not be computed. If a colliding object is entirely enclosed by the penetrated object, the algorithm presented in this paper does not compute any penetration depth, since there are no border points. The response scheme would not generate any forces until at least one object point leaves the penetrated object. In contrast, standard approaches would compute penetration depth information for all object points and probably resolve the collision in an arbitrary direction. However, if at least one object point of a colliding object is outside the penetrated object, the presented approach is likely to compute plausible and consistent penetration depth information for all colliding points. Further, there exist cases of objects crossing each other, where neither the existing nor the proposed approach are able to compute useful penetration depth information.

The presented approach does not compute the penetration depth according to its definition. Instead of computing the shortest distance to the surface of the penetrated object, the approach approximates the penetration depth only for points close to the surface. For all colliding non-border points, the depth is propagated from border points without considering the penetrated object. This supports consistency, but leads to results that can differ significantly from the actual penetration depth according to the definition. However, this disregard of the definition eliminates many artifacts in the respective collision response scheme. Further, if colliding points converge to the surface of a penetrated object, the computed penetration depth converges to the exact penetration depth.

6 Conclusions

An approach to consistent penetration depth estimation has been presented. In discrete-time simulations, the method eliminates many collision response artifacts inherent to existing penetration depth approaches. Instead of computing only the

closest surface feature for colliding points, a set of surface features is considered to avoid dynamic discontinuities of the penetration depth function. Further, the penetration depth is only computed for colliding points close to the surface, whereas consistent information is propagated to colliding points with larger penetrations. In general, the algorithm is faster than standard penetration depth approaches due to the propagation process. Experiments with dynamically deforming objects have illustrated some advantages of the consistent penetration depth estimation compared to existing methods.

References

[Ama87] J. Amanatides, A. Woo, "A Fast Voxel Traversal Algorithm for Ray Tracing", *Proc. Eurographics EG'87*, pp. 3-9, 1987.

[Bar89] D. Baraff, "Analytical Methods for Dynamic Simulation of Non-penetrating Rigid Bodies", *Proc. Siggraph*, pp. 223-232, 1989.

[Bar91] D. Baraff, "Coping with Friction for Non-Penetrating Rigid Body Simulation", *Computer Graphics*, vol. 25, no. 4, pp. 31-40, 1991.

[Bar93] D. Baraff, "Issues in Computing Contact Forces for Non-Penetrating Rigid Bodies", *Algorithmica*, vol. 10, pp. 292-352, 1993.

[Bar94] D. Baraff, "Fast Contact Force Computation for Non-penetrating Rigid Bodies", *Proc. Siggraph*, pp. 23-34, 1994.

[Ber01] G. van den Bergen, "Proximity Queries and Penetration Depth Computation on 3D Game Objects", *Proc. Game Developers Conf.*, 2001.

[Cam86] S. A. Cameron, R. K. Culley, "Determining the Minimum Translational Distance Between Two Convex Polyhedra", *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 591-596, 1986.

[Cam97] S. Cameron, "Enhancing GJK: Computing Minimum and Penetration Distance Between Convex Polyhedra", *Proc. Int. Conf. Robotics and Automation*, pp. 3112-3117, 1997.

[Des99] M. Desbrun, P. Schröder, A. Barr, "Interactive Animation of Structured Deformable Objects", *Proc. Graphics Interface*, pp. 1-8, 1999.

[Dob93] D. Dobkin, J. Hershberger, D. Kirkpatrick, S. Suri, "Computing the Intersection Depth of Polyhedra", *Algorithmica*, vol. 9, pp. 518-533, 1993.

[Fau96] F. Faure, "An Energy-Based Method for Contact Force Computation", *Proc. Eurographics*, pp. 357-366, 1996.

[Fis01] S. Fisher, M. C. Lin, "Deformed Distance Fields for Simulation of Non-Penetrating Flexible Bodies", *Proc. Workshop Computer Animation and Simulation*, 2001.

[Gas93] M.-P. Gascuel, "An Implicit Formulation for Precise Contact Modeling Between Flexible Solids", *Proc. Siggraph*, pp. 313-320, 1993.

[Gil88] E. G. Gilbert, D. W. Johnson, S. S. Keerthi, "A Fast Procedure for Computing the Distance Between Objects in Three-Dimensional Space", *IEE Journal Robotics and Automation*, vol. 4, pp. 193-203, 1988.

[Gre94] N. Greene, "Detecting Intersection of a Rectangular Solid and a Convex Polyhedron", *Graphics Gems IV*, pp. 74-82, 1994.

[Gui86] L. Guibas, R. Seidel, "Computing Convolutions by Reciprocal Search", *Proc. Symposium on Computational Geometry*, pp. 90-99, 1986.

[Hah88] J. K. Hahn, "Realistic Animation of Rigid Bodies", *Proc. Siggraph*, pp. 299-308, 1988.

[Hof02] K. Hoff, A. Zaferakis, M. Lin, D. Manocha, "Fast 3D Geometric Proximity Queries Between Rigid and Deformable Models Using Graphics Hardware Acceleration", *Technical Report University of North Carolina, Computer Science*, 2002.

[Kim04] Y. J. Kim, M. C. Lin, D. Manocha, "Incremental Penetration Depth Estimation Between Convex Polytopes Using Dual-Space Expansion", *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 2, 2004.

[McK90] M. McKenna, D. Zeltzer, "Dynamic Simulation of Autonomous Legged Locomotion" *Proc. Siggraph*, pp. 29-38, 1990.

[Moo88] M. Moore, J. Wilhelms, "Collision Detection and Response for Computer Animation", *Proc. Siggraph*, pp. 289-298, 1988.

[Pau04] M. Pauly, D. K. Pai, L. J. Guibas, "Quasi-Rigid Objects in Contact", *Proc. Symposium on Computer Animation*, 2004, to appear.

[Pla88] J. C. Platt, A. H. Barr, "Constraint Methods for Flexible Models", *Proc. Siggraph*, pp. 279-288, 1988.

[Red04] S. Redon, Y. J. Kim, M. C. Lin, D. Manocha, "Fast Continuous Collision Detection for Articulated Models", *Proc. Symposium Solid Modeling and Applications*, 2004.

[Sud04] A. Sud, M. A. Otaduy, D. Manocha, "DiFi: Fast 3D Distance Field Computation Using Graphics Hardware", *Proc. Eurographics*, 2004, to appear.

[Ter87] D. Terzopoulos, J. C. Platt, A. H. Barr, "Elastically Deformable Models", *Proc. Siggraph*, pp. 205-214, 1987.

[Tes03] M. Teschner, B. Heidelberger, M. Müller, D. Pomeranets, M. Gross."Optimized spatial hashing for collision detection of deformable objects", *Proc. Vision, Modeling, Visualization VMV'03*, pp. 47-54, 2003.

[Tes04a] M. Teschner, B. Heidelberger, M. Müller, M. Gross, "A Versatile and Robust Model for Geometrically Complex Deformable Solids", *Proc. Computer Graphics International*, pp. 312-319, 2004.

[Tes04b] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnetat-Thalmann, W. Strasser, "Collision Detection for Deformable Objects", Proc. Eurographics, State-of-the-Art Report, 2004, to appear.

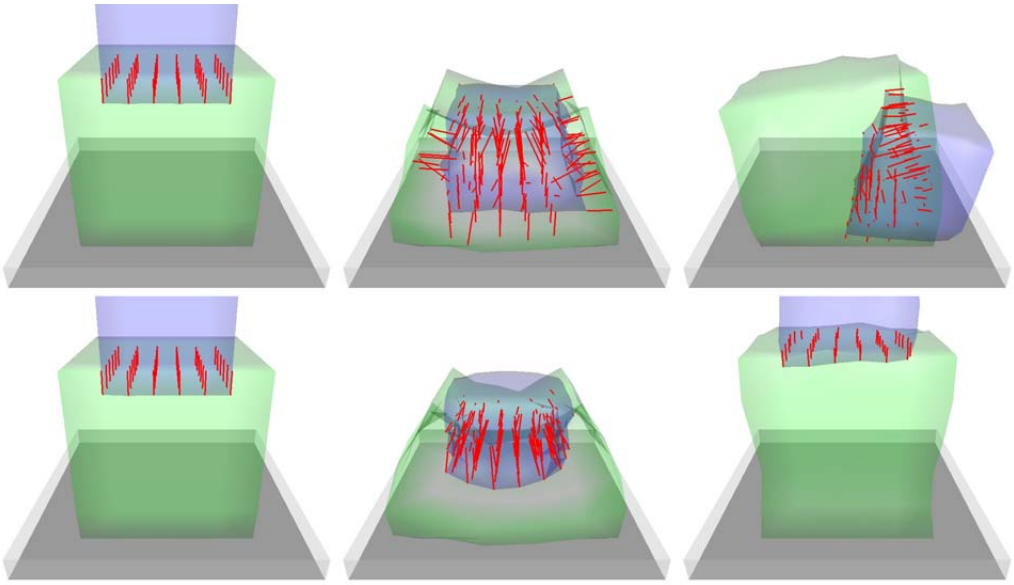

Figure 8: Two colliding deformable cubes. The standard closest-feature approach shown in the first row causes non-plausible penetration depth information in case of large penetrations. This causes artifacts in the collision response scheme which are eliminated with the presented approach illustrated in the second row.

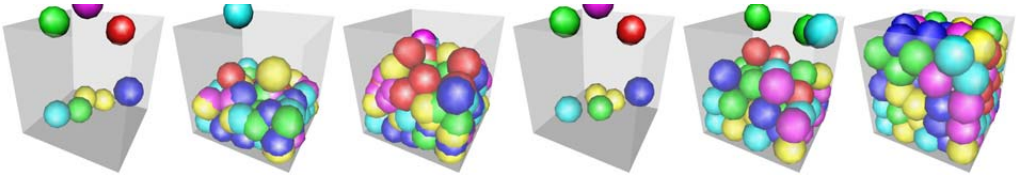

Figure 9: 120 colliding deformable spheres. The first three images illustrate the sticking artifact of the standard penetration depth approach. These non-plausible equilibrium states are avoided with the presented approach as shown in the three images on the right-hand side.

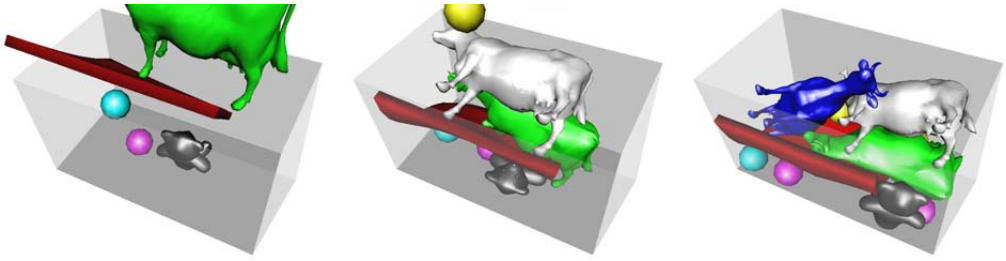

Figure 10: Various colliding deformable objects. This scenario illustrates that the presented approach can be used for realistic collision response for objects with varying characteristics. In this scene, convex and concave objects of different size are simulated.