Comput. Methods Appl. Mech. Engrg. 259 (2013) 77–92

Contents lists available at SciVerse ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

Distance fields on unstructured grids: Stable interpolation, assumed gradients, collision detection and gap function

Sebastian Wolff [a,b,*], Christian Bucher [a]

[a] Forschungsbereich für Baumechanik und Baudynamik, Technische Universität Wien, Karlsplatz 13/E2063, 1040 Wien, Austria
[b] Aside Vienna University of Technology:Dynardo Austria GmbH, Wagelseilgasse 14, 1120 Wien, Austria

ARTICLE INFO

Article history:
Received 30 January 2012
Received in revised form 23 October 2012
Accepted 27 February 2013
Available online 15 March 2013

Keywords:
Contact
Contact detection
Finite element
Level set
Distance field
Dynamic explicit

ABSTRACT

This article presents a novel approach to collision detection based on distance fields. A novel interpolation ensures stability of the distances in the vicinity of complex geometries. An assumed gradient formulation is introduced leading to a C^1-continuous distance function. The gap function is re-expressed allowing penalty and Lagrange multiplier formulations. The article introduces a node-to-element integration for first order elements, but also discusses signed distances, partial updates, intermediate surfaces, mortar methods and higher order elements. The algorithm is fast, simple and robust for complex geometries and self contact. The computed tractions conserve linear and angular momentum even in infeasible contact. Numerical examples illustrate the new algorithm in three dimensions.

1. Introduction

Many simulations involve unilateral contact/impact analysis. In dynamic simulations, notably in explicit dynamics, the number of time steps may be very large. A fast collision response is, therefore, required. This includes three components: the contact search, the generation of discrete contact conditions (constraint equations or penalty forces) and the solution of the dynamic response. Beside the numerical efficiency, accuracy of the contact algorithm is very important. It involves the geometrical accuracy of the actually measured interpenetration, the accuracy of the measured penetration depth and depth gradient, and the order of the spatial and temporal discretization scheme. This article focuses on the contact search and on the evaluation of the penetration depth and its linearization.

State-of-the-art methods are often based on a (smooth) gap function or non-smooth surface features. The gap function is a measure for the distance between a given contactor point and the target boundary surface. Many gap function algorithms rely on direct closest point projection of the contactor point onto the target surface [1,2]. These methods may become quite complex to ensure robustness of the detection. Problems may appear during the Newton iteration of the projection, when multiple candidates of contact segments appear or if the contactor point falls into a 'dead zone' (associated problems are nodes getting caught at corners and negative sliding energies in friction). These problems were first tackled by the inside-outside algorithm [3] which associates a normal vector to each contactor point that is obtained by averaging the normals of the adjacent segments. The dead-zone problem at corners and edges on the boundary is eliminated and no iterations are required. It still needs the creation of a halo, i.e. an artificial bounding volume around a surface segment, during the global search phase.

The gap function assumes smoothness of the contact boundaries. Some complex geometries, however, include non-smooth surface feature, for example corners or edges. For such problems, non-smooth contact search algorithms were developed [4–6]. Within the family of gap function algorithms, non-smooth surface features can be approached for a certain degree by replacing the boundary of the finite element mesh by a smoothed surface representation which is at least C^1-continuous and is typically based on splines, NURBS or NURBS-patches, see for example [7–15]. These formulations artificially smoothe the geometry of a corner such that a differentiable gap function is obtained. Further, instead of replacing the boundary of a FEM mesh by a smooth surface, one may use higher-continuous element formulations as found in isogeometric analysis [16–18] which explicitly define smooth boundaries. Boundary approximations are classified in [19] where one proposes the combination of point-to-point, point-to-edge and point-to-surface projections in order to treat C^0-continuous contact boundaries.

* Corresponding author at: Forschungsbereich für Baumechanik und Baudynamik, Technische Universität Wien, Karlsplatz 13/E2063, 1040 Wien, Austria.
E-mail addresses: sw@allmech.tuwien.ac.at, sebastian.wolff@dynardo.at (S. Wolff).

0045-7825
http://dx.doi.org/10.1016/j.cma.2013.02.015

1.1. Distance fields

An alternative approach to contact detection utilizing a gap function relies on the evaluation of distance fields. Distance fields provide an implicit representation of the closest point projection, see Fig. 1. The idea is that once the coordinate of a point is known, one does not compute the actual projection, but evaluates a scalar field to obtain the distance to a surface.

Algorithms regarding distance fields go back to the level set equation. The level set method was presented by Osher and Sethian [20] who described the temporal propagation of moving interfaces by numerical methods solving the Hamilton–Jacobi equation. This is performed by a finite difference scheme working on a rectangular grid in two or three dimensions. Information on normal vectors and curvature can be obtained. The fast marching method [21] provides an efficient numerical scheme of complexity $n \log n$ to compute the support values on the grid. It is a reinterpretation of the propagation process, i.e. the time where the interface passes a certain grid point is influenced only by those neighboring grid points which are previously passed by the interface. An overview on the theory of level set and fast marching methods and their applications to problems of various areas are given in [22,23], for example shape offsetting, computing distances, photolithography development, seismic travel times, etc. Distance fields are a special case of the level set equation where the absolute value of the advection velocity is 1.

The concept of distance fields was introduced to contact problems in [24] using first order tetrahedral finite elements. The distance field is generated on a supplementary grid and evaluated at the finite element nodes. Simplicity and robustness compared with closest point projection is emphasized, in particular no longer required smoothness conditions on the shape of the boundary. Self-contact, large deformations and deep interpenetrations may be treated easily. Exact intersection polygons are determined on which contact forces are computed by the penalty method [24]. More details on the employment of the distance field are provided in [25]. It focuses on the pre-computation of the distance field by fast marching. A simple partial update strategy during a time integration is proposed for regions where intersections actually occur. More details of the approach are presented in [26].

A supplementary grid is not required if the distance field is interpolated on the finite element mesh. This is constricted by the lack of efficient level set methods on unstructured meshes. A fast marching method is adopted to acute triangle meshes in [27]. The basic problem are instabilities which arise by propagating approximate levels along arbitrarily changing directions. Instead of propagating the approximate distance, [28] computes accurate distances of grid points to the initial interface, but propagates a reference to the surface patches to which the closest point projection refers to. The idea was adopted to tetrahedral finite element meshes [29] with application to collision detection eliminating the supplementary grid. A partial distance field update strategy is provided for simplex meshes therein. Although not related to distance fields, a notable partial update strategy is presented by Heidelberger et al. [30] which improves the robustness of closest point projection approximation in two dimensions. By identifying the actually intersected boundary as initial interface, the distances will only be computed with respect to the selected surface patches and the partial update is restricted to the finite element nodes which are actually in contact. The approximate distances and normal vectors are propagated similar to the original fast marching approach.

1.2. Objectives and outline

The objectives behind this article are to adopt the ideas behind distance fields to collision detection on unstructured grids. No supplementary grid should be used. Instead, the distance field will be interpolated using the finite element mesh, i.e. no distances outside the bodies can be represented. The discretization should be suitable for arbitrary finite element types. Further, the distance field is to be expressed in terms of the gap function which in turn enables penalty and Lagrange multiplier formulations.

Section 2 presents the basic ideas behind Eulerian distance fields and level sets on a supplementary grid and after that introduces the discretization being used in this article. The distance field is interpolated on arbitrary finite element types following the presentation in [31]. The formulation is, however, unstable. A correct distance distribution can not be represented by the formulations used in [29,31] when applied to complex geometries. This is pointed out in Section 3. Therein, a novel interpolation scheme of the distances is proposed which resolves the instabilities. Furthermore, the assumed gradient formulation of [31] will be modified to match the novel distance interpolation and to eliminate inaccuracies of the gradient field on the boundary. The presentation continues by reformulating the closest point projection through the distance field in Section 4. A simple and robust approximation to the projection will be derived from a linear expansion of the distance. Section 5 introduces some fundamentals of contact mechanics and interprets the gap function in terms of discrete distances. Numerical examples are presented in Section 6. They utilize a 'node-to-element' integration scheme on the boundaries of first order tetrahedral and hexahedral finite elements. Finally, Section 7 summarizes important properties of the new collision detection.

The appendix provides supplementary information that is required to complete the distance field algorithm. A presents Closest

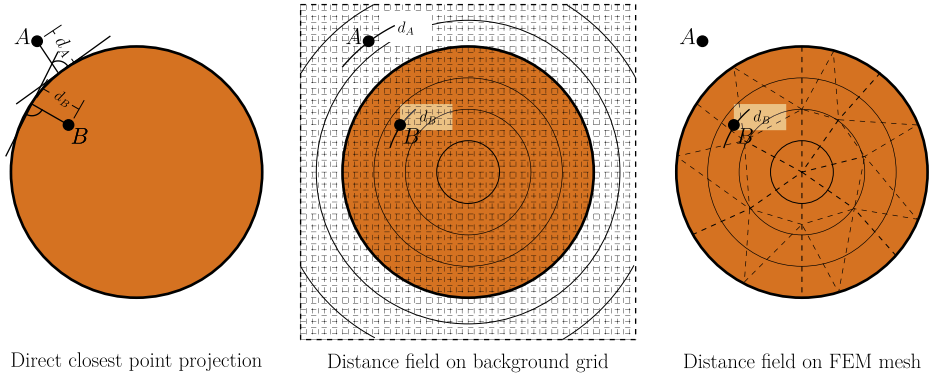

Fig. 1. Representations of the closest point projection of two points A, B onto the boundary of a circle: Direct projection, implicit projection by distance fields interpolated on rectangular supplementary grids or interpolated on unstructured grids.

Feature Front Marching (CFFM) which is adopted to compute the support values of the discrete distance field. B presents the local intersection test for tetrahedral and hexahedral elements. It further explains how global contact search algorithms can be incorporated. C presents how individual spatial discretization schemes can be employed, for example node-to-element, higher order mortar methods, intermediate surfaces. It further points out conceptual differences to closest point projection when using these discretization schemes.

2. Distance field

2.1. Level sets

The level set method is primarily used to implicitly describe propagating interfaces which are usually discretized on Cartesian grids, see [20,21]. The level set equation describes the evolution of a scalar field $d(\mathbf{x},t)$ through

$$\dot{d}(\mathbf{x},t) + v(\mathbf{x},t)\|\nabla_{\mathbf{x}} d\| = 0, \quad \forall(\mathbf{x},t) \in \Omega \times \mathcal{T} \tag{1}$$

where $v(\mathbf{x},t)$ is the advection velocity. This partial differential equation is used to describe the motion of an interface by associating its geometry with the zero-iso contour of d. Typically, d is initialized as the signed distance to the interface, satisfying $\|\nabla_{\mathbf{x}} d\| = 1 \Rightarrow \nabla_{\mathbf{x}}\|\nabla_{\mathbf{x}} d\| = \mathbf{0}$. The advection velocity v is often replaced by a corresponding extensional velocity v_e such that $\|\nabla_{\mathbf{x}} d\| = 1$ is maintained. The level set Eq. (1) of a distance field can then be recast as

$$\dot{d}(\mathbf{x},t) + v_e(\mathbf{x},t) = 0 \tag{2}$$

Consider a moving interface $\mathcal{F}$ which divides a domain Ω into two disjoint subsets Ω^+ and Ω^- to either side of $\mathcal{F}$. $\mathcal{F}$ is parametrized at the zero level set [20]

$$\mathcal{F}(t) = \{\mathbf{x} : d(\mathbf{x},t) = 0\} \tag{3}$$

where the velocity field is $v(\mathbf{x},t) = \mathbf{v}(\mathbf{x},t)\nabla_{\mathbf{x}} \cdot d(\mathbf{x},t)$ on $\mathcal{F}$. If the function d is a signed distance function [31] it becomes

$$d(\mathbf{x},0) = \begin{cases} \min\limits_{\mathbf{y}\in\mathcal{F}} \|\mathbf{x}-\mathbf{y}\| & \forall \mathbf{x} \in \Omega^+ \\ -\min\limits_{\mathbf{y}\in\mathcal{F}} \|\mathbf{x}-\mathbf{y}\| & \forall \mathbf{x} \in \Omega^- \end{cases} \tag{4}$$

with a velocity

$$v(\mathbf{x},t) = v_e \forall \mathbf{x} \in \mathcal{F} \quad \nabla_{\mathbf{x}} d(\mathbf{x},t) \cdot \nabla_{\mathbf{x}} v(\mathbf{x},t) = 0 \forall \mathbf{x} \in \Omega \tag{5}$$

which preserves the signed distance function, i.e. the length of the distance gradient is constant since $\frac{\partial}{\partial t}\|\nabla_{\mathbf{z}} d\|^2 = 0$.

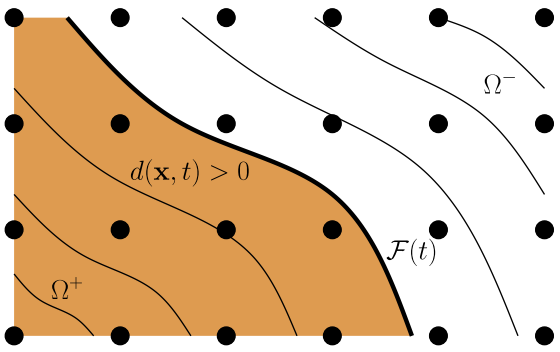

Fig. 2. Level set on a grid. Implicit representation of the surface $\mathcal{F}$ by the level set interpolated on a rectangular grid.

The level set $d(\mathbf{x},t)$ is usually discretized and interpolated using a Cartesian grid, see Fig. 2. It implicitly defines the position of the surface $\mathcal{F}$. The dynamics of the level set field is generally specified by Eulerian coordinates, i.e. the grid coordinates are constant in time. The motion of the surface $\mathcal{F}$ in time can be described using time stepping methods, i.e. the position of $\mathcal{F}$ can be determined at times $t > 0$.

In the context of collision detection [24,25] the surface $\mathcal{F}$ becomes the contact boundary $\partial_C\Omega$. Instead of propagating the dynamics of the boundary in time, the distance field is usually recomputed for regions of interest. Eulerian distance fields provide the following properties when applied to contact:

+ Eulerian distance fields provide a trivial strategy for inside–outside tests and can be used directly in collision tests.
+ The rectangular grid can be easily coupled with hierarchical or spatial partition collision detection procedures, for example octrees.
+ By interpolation on the grid, one obtains an approximation of the gap function $\hat{g}$ and the normal vector $\hat{v}$.
– High sampling rates are required to represent objects with fine detail. Accuracy is limited due to the nature of the Cartesian grid.
– In many applications, only some parts of given geometries require a fine resolution. In turn, distance fields may generate a lot of data slowing down data processing.

2.2. Discrete distance field on unstructured grids

Let the distance field be described in terms of Lagrangian coordinates, i.e. the distance d is expressed with respect to the reference configuration

$$d = d(\mathbf{X}) \tag{6}$$

with material coordinate $\mathbf{X}$. It describes the signed distance between $\mathbf{X}$ and the boundary $\partial_C\Omega$ of the considered body.

The distance field is now discretized with respect to the finite element nodes

$$d^h(\xi) = \sum_A N_A(\xi) d_A \tag{7}$$

with local material coordinate ξ, node index A and finite element shape function N_A. The discrete values d_A are stored at the nodes. The discrete field has the following properties:

- It is not defined outside of the finite element mesh, i.e.

$$\begin{aligned} d^h(\mathbf{X}) &> 0 \text{ if } \mathbf{X} \in \Omega \\ d^h(\mathbf{X}) &= 0 \text{ if } \mathbf{X} \in \partial_C\Omega \end{aligned} \tag{8}$$

- d^h is C^0-continuous.
- The condition $\left\|\nabla_{\mathbf{X}} d^h(\mathbf{X})\right\| = 1$ is generally not satisfied.
- The normal vector $\boldsymbol{v}$ being perpendicular to the tangential plane on the boundary at $\mathbf{X} \in \partial_C\Omega$ becomes

$$\boldsymbol{v}(\mathbf{X}) = -\frac{\nabla_{\mathbf{X}} d^h(\mathbf{X})}{\left\|\nabla_{\mathbf{X}} d^h(\mathbf{X})\right\|}, \ \forall\{\mathbf{X}, d^h(\mathbf{X}) = 0\} \tag{9}$$

- The interpolation returns reasonable values only in the case where the nodal values differ in at least one node of each element, i.e. $d_i \neq d_j$ (if $i \neq j$). This case must be intercepted by the mesh generator. Particularly, at least one node must not be part of the surface what may be circumvented by subdividing elements. If all nodes are associated with zero distances, the discrete field returns zero distances in the finite elements interior which is wrong, see Section 3 for more details.

3. Stable interpolation and assumed distance gradients

3.1. Illustration of instabilities

When using standard finite element shape functions, the representation of distances in the interior of elements may be erroneous. This happens whenever points of maximum distance are located in the considered element's interior. The maximum in the interior can not be captured by the interpolation functions if low order finite element shape functions are used. Such situations always appear for elements which are located at a body's skeleton, see Fig. 3 (let the body's skeleton be the set of those points where the closest point projection returns multiple projection points with identical minimal distance). But these elements are often not crucial since one is generally not interested in measuring very deep distances in collision detection. They are important, however, if the skeleton is close to the boundary, for example if more than one element face is part of the boundary or if individual nodes are on the boundary, but not the element faces they are part of. Some critical situations are illustrated in Fig. 4 for two and in Fig. 5 for three dimensions.

3.2. Stable interpolation of the distance

Obviously, an enrichment of the interpolation may stabilize the distance approximation. As seen in Fig. 5, the function space must contain at least edge, face and bubble shape functions. For hexahedral elements, a quadratic tensor-product interpolation with 27 nodes seems sufficient. For tetrahedra one requires at least 15 support points, i.e. a 10-noded quadratic tetrahedron enriched by 1 bubble and 4 area functions with supports in the element center and on the element faces. The two-dimensional interpolation is illustrated in Fig. 6.

3.3. Assumed interpolation of the distance gradient

An enrichment of the shape function space may stabilize the interpolation of the distance d^h, but the interpolation of the distance gradient $\nabla_{\mathbf{X}} d^h$ may still be insufficiently accurate. Consider, for example, a first order quadrilateral element where all element faces are on the boundary. The distance is interpolated by a 2nd order polynomial as in Fig. 6. Then, only the bubble shape function contributes nonzero terms to the distance field. The shape function gradient is zero at the interior support point (which is acceptable) and at the finite element nodes. The latter may lead to a wrong collision response because a zero gradient is measured in the vicinity of vertices and edges, see Fig. 7. The matter of nearly zero gradients may be approached in the first step by normalizing the gradient to unit length. This strategy decreases accuracy due to round-off errors and possible ill-conditioning of the normalization. Another strategy would be to enrich the elements by piecewise linear polynomials (instead of higher order polynomials). This is equivalent to subdividing finite elements into linear tetrahedra. The distance gradient will be nonzero everywhere, but discontinuous, see Fig. 8.

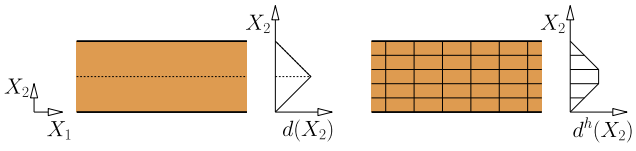

Fig. 3. Representation of skeletons. Left: The dotted line denotes the true skeleton. Right: The exact position of the skeleton can not be measured by the (linear) interpolation.

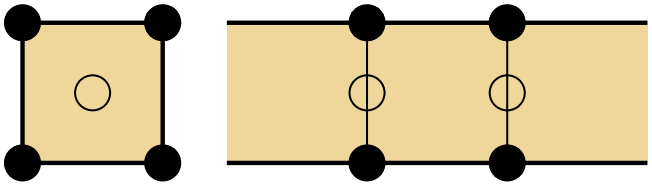

Fig. 4. Unstable discrete distance field in 2D, exemplified for a single 1st order quadrilateral element. Left: If all finite element faces are on the boundary, an additional support point in the element center stabilizes the interpolation. Right: If two opposite element faces are on the boundary, additional support points on the interior element edges stabilize the interpolation.

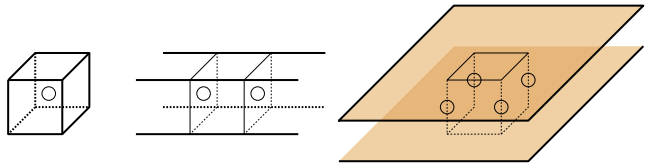

Fig. 5. Unstable discrete distance field in 3D, exemplified for a single 1st order hexahedral element. Left: If all finite element faces are on the boundary, an additional support point in the element center stabilizes the interpolation. Center: If only two opposite element faces are not on the boundary, additional support points on the non-boundary element faces stabilize the interpolation. Right: If two opposite element faces are on the boundary, additional support points on the interior element edges stabilize the interpolation.

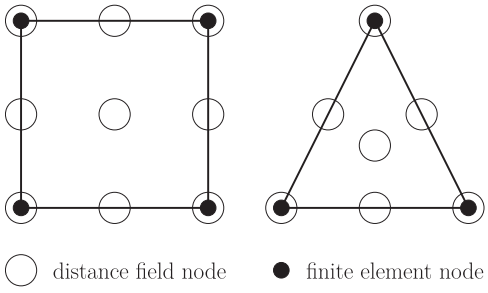

Fig. 6. Stable interpolation of the distance field in 1st order finite elements in two dimensions.

A stabilization of the gradient field can be realized through assumed gradients, see Fig. 8. The presented strategy extends ideas from [31]. Independent from the distance field, the distance gradient $\mathbf{G} = \nabla_{\mathbf{X}} d$ is interpolated by

$$\mathbf{G}^h(\mathbf{X}) = \sum_A N_A(\mathbf{X})\mathbf{G}_A, \quad \mathbf{G}_A \in \mathbb{R}^3 \tag{10}$$

Therein, $\mathbf{X}$ describes the material (undeformed) coordinate in the reference configuration. N_A denotes the interpolation function of the distance field with support point A. To ensure stability and accuracy of the assumed interpolation, N_A should be the same interpolation function as in Fig. 6. An inaccurate representation is shown in Fig. 9.

For support points which are not on the boundary, the discrete gradient $\mathbf{G}_A$ is determined from a least square problem which minimizes the error of the equivalence condition $\mathbf{G}^h(\mathbf{X}) \approx \nabla_{\mathbf{X}} d^h$:

$$\min_{\mathbf{G}_A} \int_\Omega \left\| \sum_A N_A(\mathbf{X})\mathbf{G}_A - \nabla_{\mathbf{X}} d^h \right\|^2 dV \tag{11}$$

Deriving by $\mathbf{G}_A$ and assuming the mass lumping condition

$$\int_\Omega N_A(\mathbf{X}) N_B(\mathbf{X}) dV \approx \delta_A^B \int_\Omega N_B(\mathbf{X}) dV \tag{12}$$

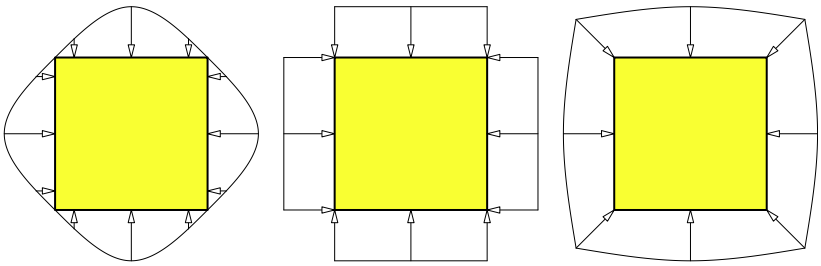

Fig. 7. Distance gradient field on the boundary of a single bilinear quadrilateral element. Left: Quadratic distance interpolation. Center: Piecewise linear enrichment of the bilinear distance interpolation. Right: Assumed gradient.

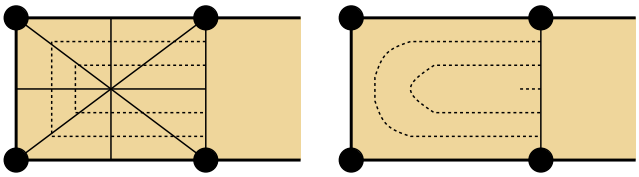

Fig. 8. Stable distance gradient for a quadrilateral element. Left: Piecewise linear interpolation/subdivision into simplexes. Dotted lines: contours of distance interpolation. Right: Assumed gradient interpolation. Dotted lines: contours of a distance function being associated with assumed gradients.

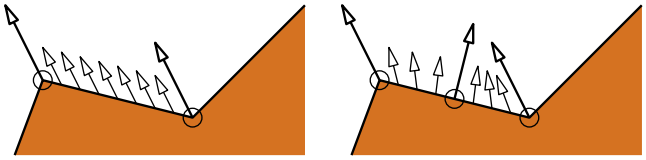

Fig. 9. Assumed interpolation of the surface normal for first order elements in two dimensions. Left: Using finite element shape functions. Right: Using the distance field interpolation functions.

with Kronecker delta δ_A^B leads to

$$\mathbf{G}_A = \sum_B \frac{\int_\Omega N_A(\mathbf{X}) \nabla_{\mathbf{X}} N_B(\mathbf{X}) d_B dV}{\int_\Omega N_A(\mathbf{X}) dV} \tag{13}$$

For support points which are part of the boundary, this strategy may lead to erroneous values because the distance field d^h itself contains insufficient data. The boundary gradients may be close to zero in the afore mentioned situations. The discrete gradients $\mathbf{G}_A$ of boundary points are, therefore, chosen to be the average of the surface normals of all surface patches F being adjacent to the considered node

$$\mathbf{G}_A = -\frac{\sum_{F \in A} v_F W_F}{\sum_{F \in A} W_F}, \quad A \in \partial_C \Omega \tag{14}$$

Therein, v_F denotes the normal vector of surface F at node A, which can be easily obtained from triangulation, and W_F represents a weighting factor, for example the area of the surface patch F. A similar procedure using average nodal normal vectors on surface nodes was used by Wang and Nakamachi [3] and Puso and Laursen [32].

The distance gradient $\gamma(\mathbf{X})$ in the deformed configuration $\mathbf{x} = \phi_t(\mathbf{X})$ can be computed through the chain rule. Introducing the deformation gradient $\mathbf{F}$ one obtains

$$\begin{aligned} \gamma_\alpha(\mathbf{X}) &= \frac{\partial}{\partial x_\alpha} d(\mathbf{X}) = \frac{\partial X_\beta}{\partial x_\alpha} \frac{\partial}{\partial X_\beta} d(\mathbf{X}) \\ \gamma(\mathbf{X}) &= \mathbf{F}^{-T}(\mathbf{X}) \mathbf{G}(\mathbf{X}) \end{aligned} \tag{15}$$

One can, therefore, store the nodal gradients in the reference configuration and map them into the deformed space during the simulation.

In order to obtain a C^0-continuous interpolation of γ one requires a C^0-continuous representation of the deformation gradient. Such is given if special element formulations are used, for example isogeometric analysis [33] or continuous assumed gradients [34].

The curvature of the distance field can be obtained through the interpolation function's derivatives, i.e.

$$\nabla_{\mathbf{X}}^2 d(\mathbf{X}) = \sum_A \nabla_{\mathbf{X}} N_A(\mathbf{X}) \mathbf{G}_A \tag{16}$$

4. Computing the closest point projection

The distance field can be used to efficiently compute the closest point projection of a point $\mathbf{x}$ to the boundary. Let this projection be $\mathbf{y} = \mathbf{x} + \mathbf{p}$, where $\mathbf{p}$ denotes the projection/distance vector from $\mathbf{x}$. The boundary is defined through $d(\mathbf{x}) = 0$. The closest point projection defines the projection vector $\mathbf{p}$ to be the shortest vector to the surface, i.e.

$$\min_{\mathbf{p}} \|\mathbf{p}\|, \text{ subject to } d(\mathbf{x} + \mathbf{p}) = 0 \tag{17}$$

which is transformed into an equivalent problem using the Lagrangian

$$L(\mathbf{p}, \lambda) = \|\mathbf{p}\|^2 + \lambda \cdot d(\mathbf{x} + \mathbf{p}) \rightarrow \min_{\mathbf{p}} \max_{\lambda} \tag{18}$$

The objective function is quadratic, but the distance may be highly nonlinear, the surface may be non-convex and non-smooth.

An approximate solution is obtained by replacing the distance function by a first order Taylor expansion

$$d(\mathbf{x} + \mathbf{p}) = d(\mathbf{x}) + \nabla_{\mathbf{x}} d(\mathbf{x})^T \mathbf{p} + \mathcal{O}(\mathbf{p}^2) \tag{19}$$

Substituting $\gamma(\mathbf{x}) = \nabla_{\mathbf{x}} d(\mathbf{x})$, one obtains

$$\mathbf{p} = -\frac{\gamma(\mathbf{x})}{\|\gamma(\mathbf{x})\|^2} d(\mathbf{x}) \tag{20}$$

The expansion $d(\mathbf{x} + \mathbf{p})$ approximates the tangential plane in $\mathbf{x} + \mathbf{p}$. This plane is perpendicular to $\mathbf{p}$, since the scalar product of $\mathbf{p}$ and any vector to a point $\hat{\mathbf{y}}$ on the approximated surface $0 = d(\mathbf{x}) + \nabla_{\mathbf{x}} d(\mathbf{x}) \cdot (\hat{\mathbf{y}} - \mathbf{x})$ is zero.

The advantages of using the distance field for closest point projection compared with exact projections are

1. Once the material coordinate $\mathbf{X}$ is found, the distance function $d(\mathbf{X})$ can be easily evaluated by interpolation. The evaluation time is independent of the complexity of the body's shape.
2. For small penetrations, the approximation is usually sufficiently accurate.
3. Directions are always feasible, independent from non-smooth geometries, non-convex surface features, neighborhood of corners and wedges, see Fig. 10.
4. The discrete distance field is C^0-continuous. When used in conjunction with assumed gradients, it is C^1-continuous.

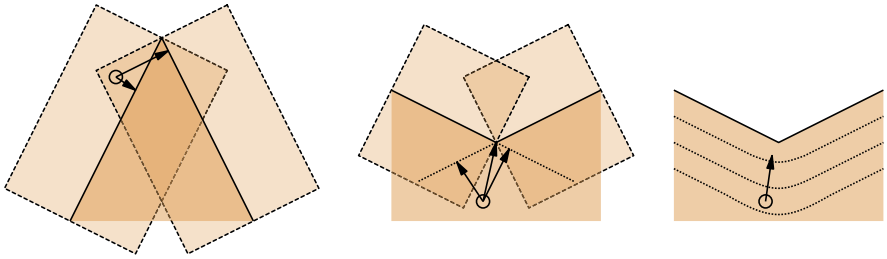

Fig. 10. Distance field and closest point projection. The highlighted rectangles around the surface patches are bounding volumes (halos) in which the search takes place. Left: Closest point projection at sharp corners. The point lies outside, but the right face returns a positive distance. Center: Closest point projection at non-convex corners. The point is outside of the halos and may not be used in detection. Multiple projections appear. Special considerations of edges and corners are required. Right: Distance field simply evaluates a distance and a smoothed gradient which automatically includes handling of sharp or non-convex geometries.

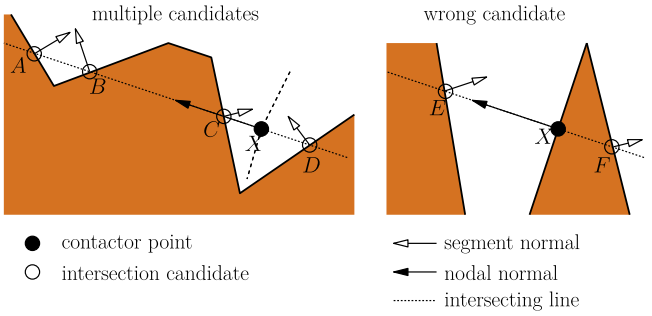

Fig. 11. Inside–outside test. Left: Multiple intersection candidates along the nodal normal. Right: Detection of penetration for point *F*.

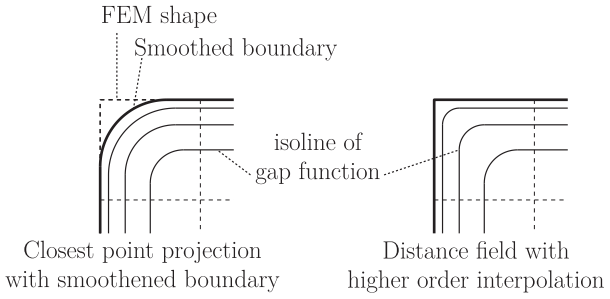

Fig. 12. Gap smoothening in corners. The effect of smoothening the boundary and of using higher order distance fields is very similar. The distance field, however, uses the exact finite element geometry for collision detection.

The direct closest point projection [1,2] often fails if the point to be projected is in a dead zone, see Fig. 10 center. Furthermore, in the vicinity of corners multiple segment candidates may be chosen leading to multiple solutions for the direction to the closest point, see Fig. 10 left. The problem of dead zones was eliminated by the inside-outside test [3] which associates a normal vector to the point to be projected. Then, one seeks for the intersection of a line along this normal with all neighboring segments. If the normal of the intersected segment shows into the opposite half space as the point normal and if the distance along the point normal to the projection point is negative then the point is inside the target body. Fig. 11 illustrates some remaining difficulties for the inside-outside test. The left figure in 11 shows multiple contact candidates along the point normal. The search algorithm must track the segment being closest to the point *X* with positive penetration condition.

Fig. 11 on the right is a case where the plain inside-outside test would detect a positive penetration distance of point *X* with reference point *F*. This case is an erroneously detected self-contact where the back side of the same body is chosen as reference. Using distance fields, however, the mentioned problems usually do not appear.

The dead-zone problem in Fig. 10 was eliminated in direct closest point projection by replacing the boundary by a smooth interpolation, for example in [9,12]. In fact, the representation of the gap function by smoothening and using a higher-order distance field are very similar, see Fig. 12.

5. Replacing the gap function in contact mechanics

5.1. Parametrization, kinematics and contact integral

Assume the existence of a contact frame C being defined as a surface which serves for parametrizing the contacting domain in three-dimensional two-body contact with local coordinate $\hat{\xi} \in \mathbb{R}^2$. All quantities referring to the contact frame are denoted by $(\hat{\cdot})$. Points in C are labeled by the position vector $\hat{\mathbf{x}} \in C$. One of the two bodies is denoted as contactor $\Omega_t^{(1)}$, the other as target $\Omega_t^{(2)}$.

The gap function $\hat{g}$ defines the signed distance between any point $\hat{\mathbf{x}}^{(1)}$ on the contactor boundary to the contact boundary $C^{(2)}$ of the target body $\Omega_t^{(2)}$ in the current configuration, see Fig. 13. It defines three possible states: If $\hat{g} < 0$ then $\hat{\mathbf{x}}^{(1)}$ is outside $\Omega_t^{(2)}$, if $\hat{g} = 0$ it is on the boundary $C^{(2)}$, else it is infeasible. Usually, the gap function is defined by the closest point projection between points on the contactor surface $C^{(1)}$ to the target boundary, i.e.

$$\hat{g}(\hat{\xi}) = \left(\hat{\mathbf{x}}^{(2)}(\hat{\xi}) - \hat{\mathbf{x}}^{(1)}(\hat{\xi})\right)^T \hat{v}(\hat{\xi}) \tag{21}$$

where the vector $\hat{v}$ is the normal to the tangential plane at the projection point $\hat{\mathbf{x}}^{(2)}$ and pointing outwards.

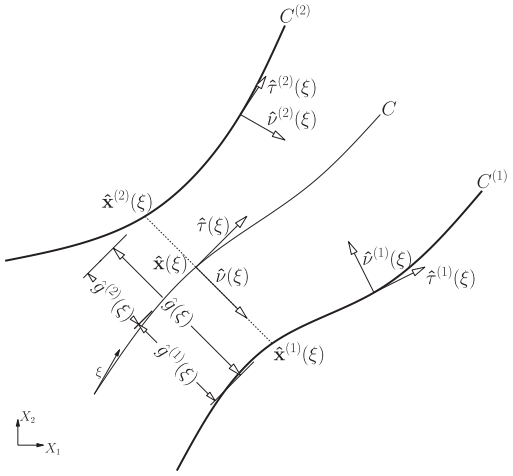

Fig. 13. Parameterization within the contact frame (non-penetrating case).

The contact traction $\hat{\mathbf{t}}^{(1)} = -\hat{\mathbf{t}}^{(2)}$ can be decomposed into normal and tangential components

$$\hat{\mathbf{t}}^{(1)} = \hat{t}_N \hat{v} + \hat{t}_{T\alpha} \hat{\tau}_\alpha \tag{22}$$

The variations $\delta\hat{\mathbf{x}}^{(i)}$ are expressed in terms of the the gap function and the tangential glide path with basis vectors $\hat{\tau}^\alpha$ being orthogonal to $\hat{v}$. The variation of the gap function is given by

$$\delta\hat{g} = \hat{v} \cdot \left(\delta\hat{\phi}^{(2)} - \delta\hat{\phi}^{(1)} \right) \tag{23}$$

while the variation of the tangential glide path is

$$\delta\hat{\xi}^\beta = \left(\delta\hat{\phi}^{(2)} - \delta\hat{\phi}^{(1)} \right) \cdot \hat{\tau}^\beta \tag{24}$$

Using these notations, the contact integral writes

$$\hat{G}_c(\hat{\mathbf{x}}, \delta\hat{\mathbf{x}}) = \int_C \left(\hat{t}_N \delta\hat{g} + \hat{t}_{T\alpha} \delta\hat{\xi}^\alpha \right) d\Gamma \tag{25}$$

The frictional response is characterized by the relative velocity between points on the contact surfaces. The relative velocity is decomposed using the dual basis

$$\dot{\hat{\mathbf{x}}}^{(1)} - \dot{\hat{\mathbf{x}}}^{(2)} = \hat{\mathbf{v}}_N + \hat{\mathbf{v}}_T = \hat{v}_N \hat{v} + \hat{v}^b_{T\alpha} \hat{\tau}^\alpha \tag{26}$$

5.2. Impenetrability constraint

The structure of the projection vector given through the distance field in Eq. (20) is similar to Eq. (21), i.e.

$$\frac{d(\mathbf{x})}{\|\gamma(\mathbf{x})\|} \frac{(-\gamma(\mathbf{x}))}{\|\gamma(\mathbf{x})\|} = \mathbf{p} \quad \leftrightarrow \quad \hat{g}(\hat{\xi}) \cdot \hat{v}(\hat{\xi}) = \hat{\mathbf{x}}^{(2)}(\hat{\xi}) - \hat{\mathbf{x}}^{(1)}(\hat{\xi}) \tag{27}$$

The definitions of the gap function and the surface normal can, therefore, be replaced by the distance function and its gradient. Let the gap function be the sum of the distances of a point $\hat{\mathbf{x}}$ on the contact frame to the boundaries of the two bodies, see Fig. 13,

$$\hat{g}(\hat{\xi}) = \sum_{i=1,2} \frac{d^{(i)}(\widehat{\mathbf{X}}(\hat{\xi}))}{\left\| \gamma^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\|} \tag{28}$$

Variation leads to

$$\delta\hat{g} = \sum_{i=1,2} \left(\frac{\gamma^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \cdot \left(\delta\hat{\phi}^{(i)} + \hat{\tau}^{(i)}_\beta \delta\xi^{(i)\beta} \right)}{\left\| \gamma^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\|} - \frac{d^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \delta \left\| \gamma^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\|}{\left\| \gamma^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\|^2} \right) \tag{29}$$

Due to perpendicularity of the basis it is $\gamma^{(i)} \cdot \hat{\tau}^{(i)}_\beta = 0$. The last term is assumed being zero since $\left\| \gamma^{(i)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\| = 1$ is satisfied by the continuous distance function. In feasible contact, $\hat{g} = 0$, assuming that two planes are in contact both tangential planes are identical and one has

$$\hat{v}(\hat{\xi}) = -\frac{\gamma^{(1)}(\widehat{\mathbf{X}}(\hat{\xi}))}{\left\| \gamma^{(1)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\|} = \frac{\gamma^{(2)}(\widehat{\mathbf{X}}(\hat{\xi}))}{\left\| \gamma^{(2)}(\widehat{\mathbf{X}}(\hat{\xi})) \right\|} \tag{30}$$

This assumption does not hold in point-to-curved-surface contact for which different contact models should be used, see for example [4]. Then Eq. (29) leads to

$$\delta\hat{g} = \hat{v} \cdot \left(\delta\hat{\phi}^{(2)} - \delta\hat{\phi}^{(1)} \right) \tag{31}$$

which is identical to the previous result (23).

The tangential velocity in Eq. (26) is obtained from the decomposition

$$\hat{\mathbf{v}}_T = (\mathbf{I} - \hat{v} \otimes \hat{v}) \hat{\mathbf{v}} \tag{32}$$

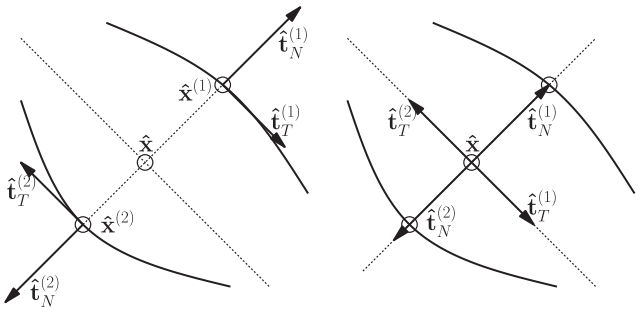

Fig. 14. Momentum preservation. Left: tractions applied to projection points $\hat{\mathbf{x}}^{(i)}$ in case of $\hat{g} > 0$. Right: tractions applied to material points $\widehat{\mathbf{X}}^{(i)}$ located at the same spatial coordinate $\hat{\mathbf{x}}$ on contact frame.

Hence, no assumptions on the smoothness of the contact surface are required for the computation of the tangential basis.

In frictionless contact, formulations based on closest point projections and distance fields preserve the balance of momentum: The normal tractions are applied to spatial points being located on the intersections of the two contact surfaces on a line along the normal vector $\hat{v}$ and the normal tractions $\hat{\mathbf{t}}_N$. This is different for tangential tractions: If the normal contact condition $\hat{g} = 0$ is not exactly enforced by the contact algorithm, the angular momentum is generally not preserved if the closest point projection is used, see Fig. 14 on the left. When using distance fields, the contact tractions are applied to material points with the same deformed coordinate. Therefore, balance of linear and angular momentum is satisfied even if the contact constraints are infeasible, i.e. $\hat{g} \neq 0$, see Fig. 14 on the right.

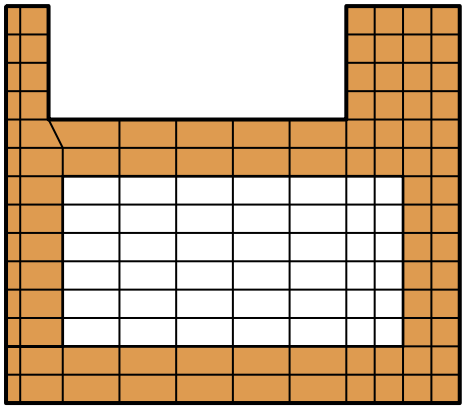

Fig. 15. Finite element layer used for distance field computation.

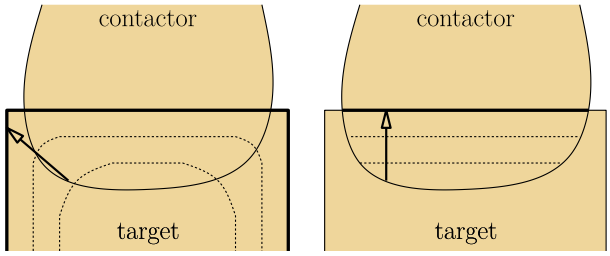

Fig. 16. Safe partial distance field update. Left: Distance with respect to the target's boundary. Right: Distance refers to actual intersection of the target's boundary with the contactor. Dotted lines: contours. Arrows: approximated projection vectors to the boundary.

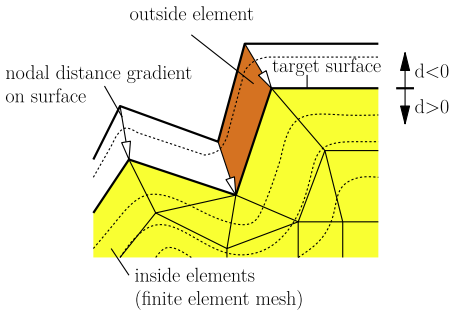

Fig. 17. Generation of outside elements that measure negative distances.

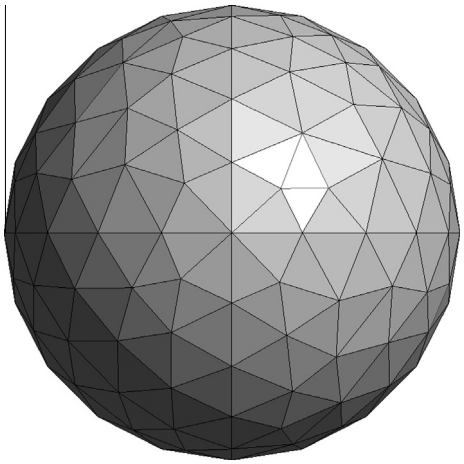

Fig. 18. FEM mesh of a sphere.

5.3. Distance field generation

In this work, the distance field is precomputed in the reference configuration [24]. Not all finite element nodes are associated with a discrete distance. Starting from the bounding surface, the two top layers of finite elements are identified, see Fig. 15. Only the nodes of these elements are considered during the distance field computation. This saves time and memory. Furthermore, it reduces the effort in collision detection. Since trial steps of a time stepping scheme should not exhibit deep penetrations, this assumption is feasible for many cases.

If the complete boundary of a body is used as reference surface **F**, then the distance gradient is not perpendicular to the boundary when measured at points close to corners and edges. Depending on the interpolation functions, some kind of smoothing appears. There are applications, however, where smoothing is not desired. This can be approached by careful definition of the reference surface **F**. In [30] a strategy for a safe partial distance field update is proposed which can be applied here. Therein, the boundary of one body (contactor) is intersected with the volume of the target. In the next step, all intersecting target surface patches are identified. These surface patches are used to define the initial set in CFFM, see A. The discrete distances of all nodes in the overlapping domain are recomputed at each deformed configuration. When evaluating the distance on the intersected contactor surface, it is measured with respect to the intersected target surface patches and the accuracy of the distance gradient can be improved, see Fig. 16. Other approaches exist to partial distance field updating which are more efficient, but less robust [25,29].

5.4. Negative distances

By associating the discrete distances to points located on the finite element mesh one is able to measure non-negative distances only. Some situations may require the evaluation of a distance and its gradient of a point to the target's boundary being located outside. Using the presented approach to distance fields, this is only possible by creating a supplementary mesh. Therein, for each surface segment one extracts the finite element nodes along the negative distance gradient, see Fig. 17. For quadrilateral segments, one obtains a hexahedron; for triangles one obtains a prism. These elements are then used for distance field generation and for contact search.

6. Numerical examples

The following tests are performed by a node-to-element integration where the finite element nodes on the contactor boundary are the numerical integration points of the contact integral. If not

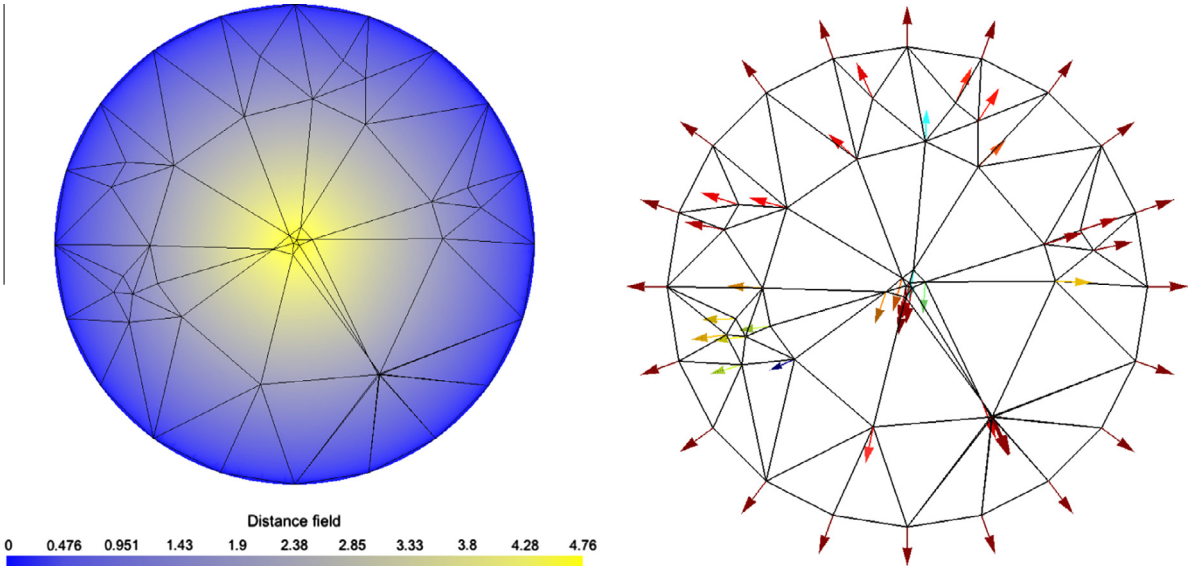

Fig. 19. Discrete distance field of a sphere. Left: interpolated distance field. Right: Assumed normal vector (at nodes).

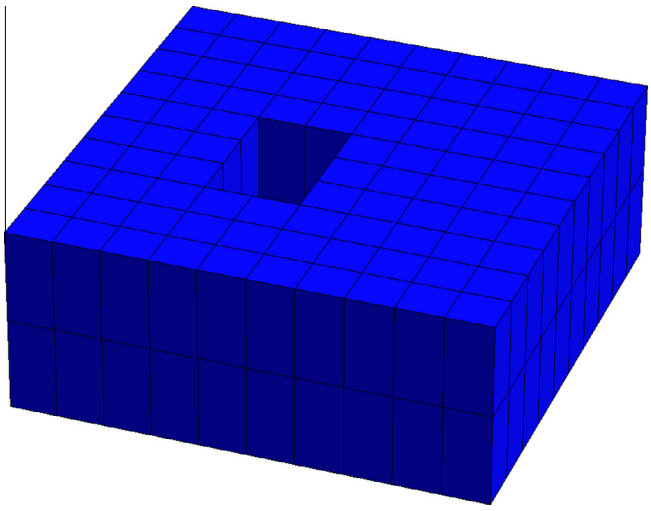

Fig. 20. FEM mesh of a block.

explicitly denoted, the FEM meshes were generated by GMSH [35] using linear tetrahedral elements. A general contact detection strategy is employed, i.e. no extra knowledge of the problem is used by the algorithm, such as individual contact pairs or maximum search distances.

6.1. Distance field interpolation for selected geometries

A sphere with very rough discretization in the center and finer discretization at the boundary is shown in Fig. 18. The radius is 5, the cell size is 1.5. The mesh contains 1041 nodes and 900 elements. To illustrate the interpolation of the distance field, the sphere is cut along a plane through its center point, see Fig. 19 on the left. The interpolation is continuous and of higher order due to the enhanced interpolation of the distance to the boundary. Hence the distance field appears smooth although the finite elements are presented by a piecewise linear geometry. The negative assumed gradient of the distance field points towards the boundary and is illustrated on the right hand side of the same figure. The interpolation of distance and gradient are sufficiently accurate even for large depths.

Another example is a rectangular block with an non-symmetric hole in its interior using a regular grid of first order hexahedral elements, see Fig. 20. The width and depth are 10, the height is 4. The hole has a width of 3 and a depth of 5 within the horizontal plane. The interpolated distance field is illustrated in Fig. 21 on the left (GMSH plots triangles in the cutting plane, but the elements are hexahedra). Because the interpolation function is quadratic, the field appears rather smooth. In the vicinity of corners, a smoothing of the natural gradient is explicitly obtained, but there is no natural gradient at the outer vertices. A gradient field being continuous being defined everywhere in the domain is shown in Fig. 21 on the right using an interpolation of the negative assumed gradient. The interpolations are plausible near convex and non-convex surface features. Note, the three nodes which are not associated with a gradient in the figure are on a symmetry plane such that the closest surface patches are either on top or on the bottom of the structure.

The next example is a hollow sphere. It illustrates the distance interpolation for a thin-walled structure with non-convex geometrical features. The radius is 1. The wall thickness is 0.1. The average size of the tetrahedra is 0.6. There are 23,220 elements and 6983 nodes in the structure. The structure is subdivided into two parts. The lower and upper parts are connected by a clamp involving a unilateral contact. Fig. 22 illustrates the interpolation of the distance field and the finite element boundaries on a cutting plane through the center point of the sphere. The discretization is too rough for obtaining a reasonable mechanical response: Along the sphere wall 1–2 element layers are used. This is chosen to illustrate the stability of the distance interpolation. At the clamp, only one element layer is used with convex and concave corners, see Fig. 23. As in the other examples, the discrete distance field covers the most important features of the geometry. The skeleton of the structure is clearly identified as the hyperface of maximum distance.

6.2. Impact of a clamped sphere on a rigid obstacle

The geometry of this example is the hollow sphere of the last section, rotated by 45°, see Fig. 24. A St. Venant material is chosen

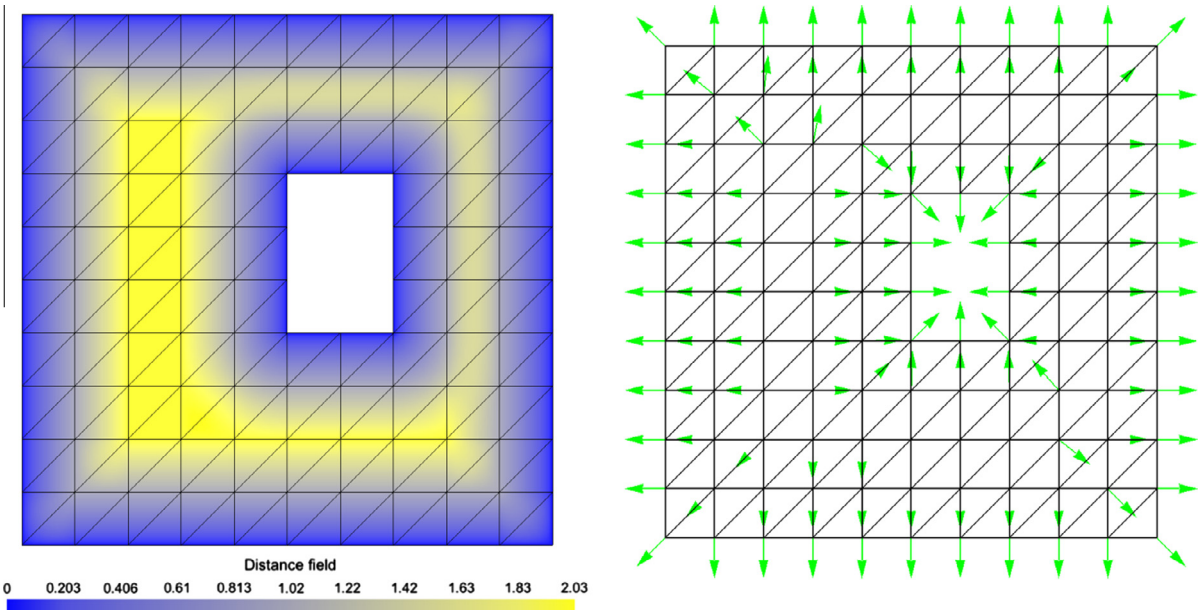

Fig. 21. Discrete distance field of a block. Left: interpolated distance field. Right: Assumed normal vector (at nodes).

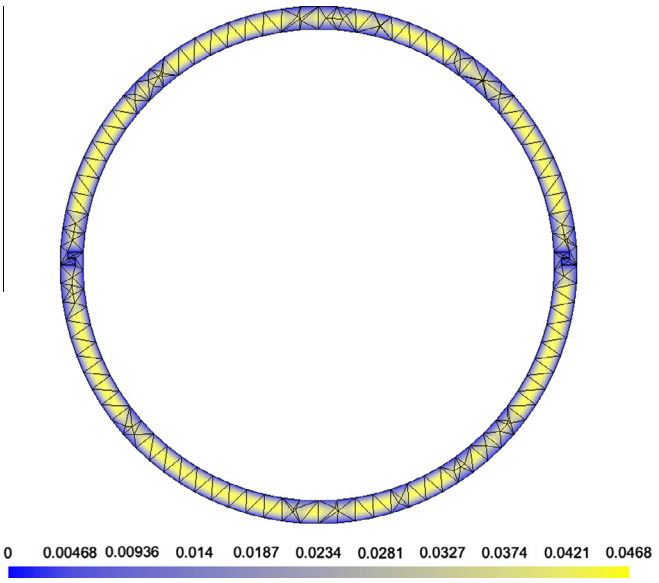

Fig. 22. Interpolated discrete distance field of a hollow sphere. Bottom: Detail around the clamp joint.

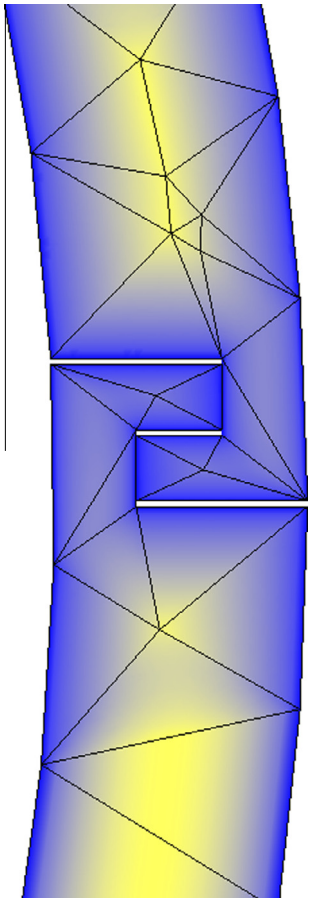

Fig. 23. Interpolated discrete distance field of a hollow sphere. Detail around the clamp joint.

with elastic modulus 1, Poisson ratio 0.3 and mass density 1e−6. The clamps between both half-spheres are tied by unilateral contact. The initial velocity is given by $v_x = 1000$ and $v_z = -500$. A Coulomb friction model is used between the two half-spheres

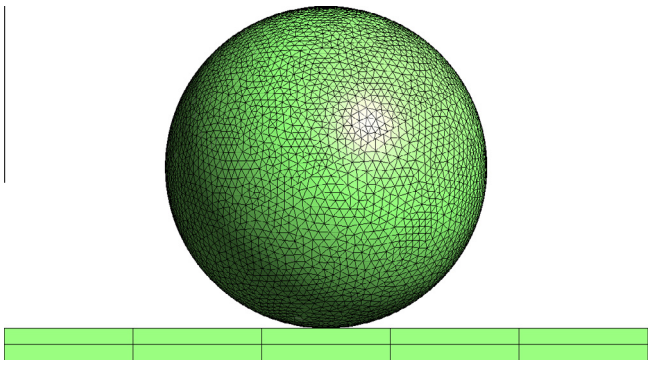

Fig. 24. Initial configuration of the tied sphere.

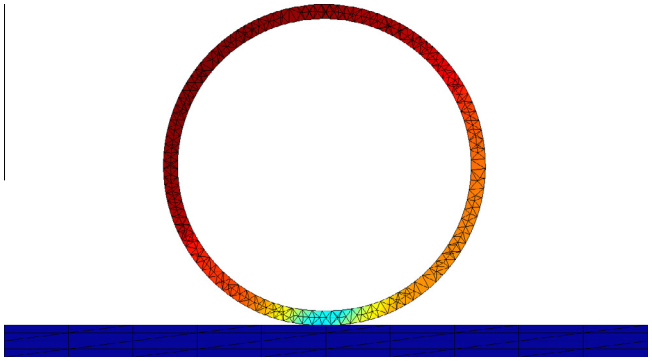

Fig. 25. Tied sphere at $t = 4e{-}5$.

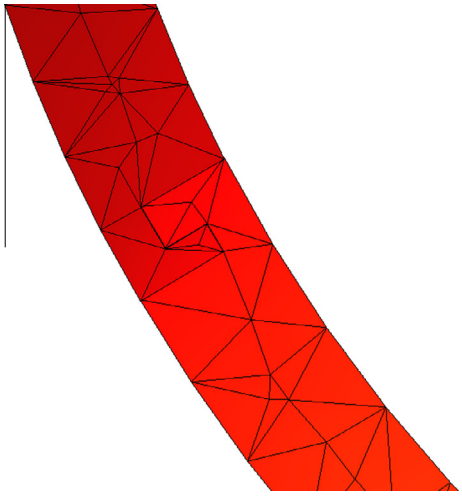

Fig. 26. Detail of a tied sphere at $t = 4e{-}5$.

and between the sphere and the rigid obstacle with coefficient 0.2. An explicit time integrator is used with a time step $\Delta t_{crit}/2$ and a simulation time $T = 0.0004$. For the dynamic contact algorithm, an asynchronous decomposition contact response is employed [36,6].

Fig. 25 illustrates a cutting plane through the sphere at time 4e-5 at the approximate peak of the impact. A detail is shown in Fig. 26 showing that the clamp exhibits no interpenetration due to erroneous detected collisions. The solution trajectory from $t = 0$ to $t = 0.0004$ is plotted by superimposing multiple configurations within this interval in Fig. 27.

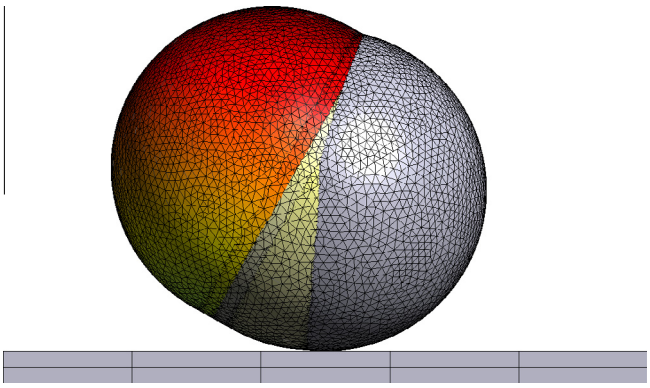

Fig. 27. Various configurations of a tied sphere in time interval [0..0.0004].

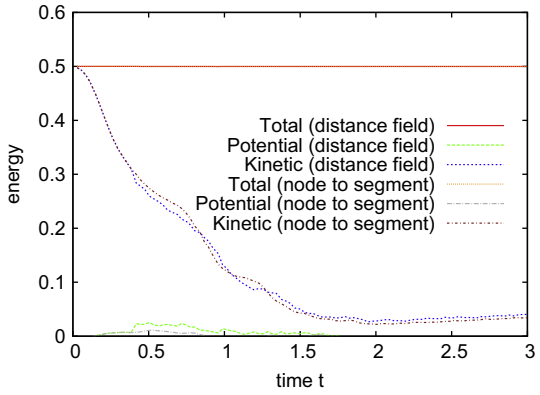

Fig. 30. Sliding block: Energy balance over time.

6.3. Elastic block sliding on obstacle

This example serves to compare the collision detection based on distance fields with node-to-segment projection when treating Coulomb friction. A block with dimensions $1 \times 1 \times 1$ is sliding on a rigid plane subject to gravitation. The block is discretized by $3 \times 3 \times 3$ 1st order hexahedral elements. The obstacle is discretized by $5 \times 5 \times 1$ elements and has the dimension $5 \times 2 \times 0.2$. Fig. 28 illustrates the geometry at the beginning and at the end of the simulation. The material is linear elastic with Young's modulus $E = 100$, Poisson's ratio $\nu = 0$, mass density $\rho = 1$. The block is subject to a constant vertical body force $F = 1$.

All nodes of the basement are supposed to be fixed. We want to measure the error in linear and angular momentum and, therefore, the nodes of the obstacle are not restrained. Instead, the mass density and elastic modulus of the plate are increased, i.e. $\rho = 100,\ E = 10,000$.

The initial horizontal velocity is $v_x = 1$. The Coulomb parameter of friction is $\mu = 0.5$. The total simulation time is $T = 3$. For time integration we use the explicit symplectic-momentum preserving Velocity Verlet scheme. The collisions are resolved by the Decomposition Contact Response [6] that modifies the velocities discontinuously through a simple projection. The time step ratio related to the critical time step is $\beta = 0.5$.

Fig. 29 presents the horizontal displacements and velocities at the block's bottom. A single value of the displacements is obtained by averaging the nodal values at the bottom surface. The results are in good agreement with the analytical solution of a rigid block sliding on a fixed obstacle: The displacements describe a parabola with end displacement $u_x = 1$ while the velocity decreases linearly until time $t = 2$. The distance field is compared with direct node-to-segment projection [1] and the inside-outside algorithm [3]. The solutions of the node-to-segment algorithm and the distance field detection are in good agreement. When comparing the required cpu time, both are nearly identical (the search distance in node-to-segment projection is limited to 0.1).

The inside-outside algorithm does not converge. The reason is the averaging of the nodal normal vector on the boundary. The decisive error appears in the first time step. The nodal normal of the nodes at the bottom left corner of the block (as in Fig. 28)

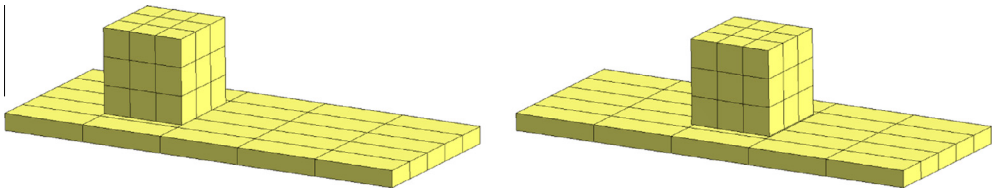

Fig. 28. Sliding block: Start and end geometry.

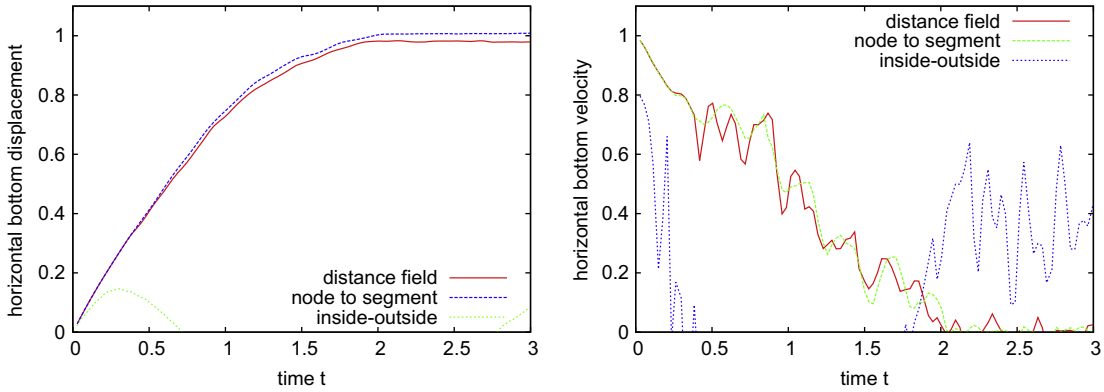

Fig. 29. Displacement and velocities over time. Left: Displacement. Right: Velocity.

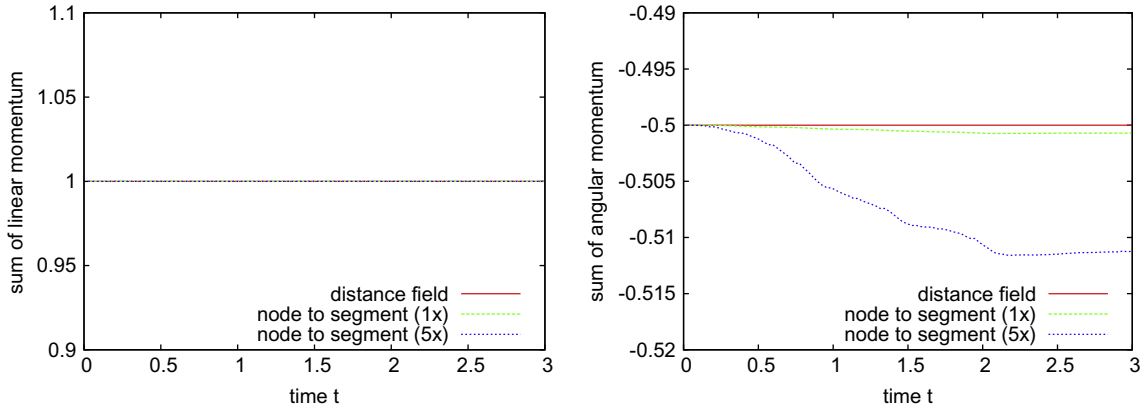

Fig. 31. Momentum preservation over time. Left: Sum of linear momentum. Right: Sum of angular momentum.

points downwards and to the left while the normal of the bottom right nodes shows downwards and to the right. When measuring the gap rate at the predictor step, the decomposition contact response measures an inactive contact at the bottom left nodes and an active contact for all other nodes. The friction force of the nodes in the bottom's center is very small because it is limited by the normal force which is related to the gravity $F = -1$. The response of the bottom right corner nodes, however, depends on the initial velocity due to the inexact normal vector. Hence, the nodes obtain a rather large impulse which starts a rolling motion of the block. The true solution, however, is a sliding motion. Although the distance field, as tested herein, also implements an averaged gradient on the block's boundary, it does not affect the convergence at least in this example.

Fig. 30 presents the energy balance. The energy dissipated by the friction grows until almost no energy is left in the system. The total energy is nearly preserved by the algorithm. Again, both solutions are nearly identical.

The balance of linear and angular momentum is presented in Fig. 31. The angular momentum is computed with respect to the origin, i.e. $x = 0$ at the block's left side, $y = 0$ at the block's center and $z = 0$ at the block's bottom. As predicted, the linear momentum is preserved by both algorithms while the angular momentum is only preserved by the distance field. The magnitude of the error in angular momentum grows with the residual of the penetration. This is illustrated by application of the collision response every 5 time steps (instead of each time step). This leads to a larger drift in the angular momentum as illustrated in Fig. 31.

7. Conclusions

Distance fields on unstructured grids may help to improve collision detection algorithms that rely on a smooth gap function in several areas. The key features are summarized below:

- very simple local intersection search, see Appendix B,
- safe and robust local search (no problems with dead zones, multiple segments as projection candidates, self contact, caught nodes, etc.),
- no user specified input required (for example, a search depth that defines the halo size),
- no generation of artificial volume elements around surface segments necessary,
- simple computation of gap function and direction by interpolation,
- suitable directions close to complex geometrical features,
- exact conservation of linear and angular momentum without post-processing, even for infeasible contact in friction,
- constraint and constraint gradient are continuous functions. This may improve convergence of iterative solution algorithms if an integration point moves from one target element to the other,
- distances outside of bodies can not be measured unless a supplementary mesh is generated,
- thin-walled structures can only be treated for penetrations $< 0.5\times$ tool size (in automatic contact),
- thin-walled structures with deeper penetrations only by careful contact pair definition and outside meshing,
- the global contact search must generally treat more finite elements than segment boundary volumes when using projection algorithms,
- compared with natural gradients, the assumed gradient field may lead to improper treatment of friction at corners, but may improve the treatment of normal contact in few situations.

This article utilizes a numerical integration scheme for the contact integral where the integration points are the contactor nodes. For this strategy, we introduce the term "node-to-element integration" for its conceptual differences to node-to-segment approaches. The distance field can be easily incorporated in higher-order mortar methods introducing the notation "segment-to-element integration". When using exact area integration, the distance field may produce nearly exact discrete gradients [37]. For the construction of higher order integration schemes one may start with the ideas in Section C.

One key aspect of the distance field being utilized in this article is the enriched interpolation when using low-order finite elements. It should be noted that the proposed interpolation schemes for hexahedral and tetrahedral elements contain the minimal number of support points to ensure a stable interpolation in the vicinity of complex geometries. Hexahedral elements with tensor-product interpolation of at least second order and tetrahedral elements of at least third order can be used directly for distance interpolation. Quadratic tetrahedra need five additional support points, one in the element center and four in the face centers.

Possible future research may be

- to explore segment-to-element integration schemes like Mortar methods
- to adopt the distance field to isogeometric analysis where usually a direct closest point projection is employed
- to compare the numerical efficiency of the local search with direct closest point projection more deeply for distorted higher order elements. The convergence behavior of the direct closest point projection was already studied in [38].

Acknowledgements

This research has been funded by the company Dynardo GmbH and by the Austrian Science Fund (FWF) under project number P20419-N14.

Appendix A. Computing discrete distances by closest feature front marching

The computation of a distance field may be a very expensive application. To be accurate, for each discrete point of the mesh the distance to all parts of the related surface must be computed and the smallest is chosen, yielding a complexity of $\mathcal{O}(n \cdot m)$ (number of nodes n, number of faces m). The complexity of this approach can be reduced by first identifying the closest discrete points on the related surface for each mesh point and then computing the distance to the adjacent faces of these closest points only. When interpreting distance fields as level sets, they may be efficiently computed using level set methods like Fast Marching [23].

The Fast Marching Method was originally developed for Cartesian grids, see Fig. A.32. The idea is that distances of points lying on (or near) the reference surface are already known. Considering a point with an unknown value, its distance may be approximated if the distances of neighboring points are known. The level set value of an arbitrary point is, therefore, not the exact distance to the surface, but a good approximation while only the information of a few neighboring points is used, yielding an algorithm of complexity $O(n \cdot \log(n))$ [23]. In [25] the distance field of a finite element structure was computed by creating a Cartesian grid occupying the same space as the finite element mesh. The actual distance values are then obtained by interpolating the values of the underlying Cartesian grid. The reason for this approach is, that although stable modifications of the Fast Marching Method for acute triangular meshes [27] exist, there is no stable version for arbitrary tetrahedral meshes in three dimensions. The nodal distance to be determined in each step depends on the discrete distances of the current marching front. As a result, an error is induced in each step which grows during the propagation.

Instead of propagating the distance through the body, one could remember the surface patch which is closest to a considered mesh point, obtaining the Closest Feature Fast Marching (CFFM) method. The approach was presented in [28] and first applied to arbitrary tetrahedral meshes in [29]. A closest feature denotes either the point, the edge or the surface patch of the elements defining the surface which is closest to the considered point. The algorithm requires a division of all spatial points into 3 mutually exclusive sets: ALIVE, ACTIVE and FAR. Additionally, each point stores two properties, the distance to the surface as well as a reference to the closest feature. At initialization, all points are marked FAR, except those being on the body's surface and their neighbors. The points on the surface are marked ALIVE, their distance is set to zero and their closest feature becomes the set of surface patches they belong to. The adjacent points of the initial ALIVE points are set being ACTIVE; they form the so called narrow band. The narrow band is then propagated through the body, starting from its boundary and moving towards the interior skeleton until no FAR points exist. The procedure applied to finite continuum elements is summarized in Algorithm 1. The resulting method is of complexity $\mathcal{O}(n \cdot \log(n))$. It is more accurate than the original Fast Marching Method because it computes the accurate distances to the boundary instead of propagating approximate values. Errors may appear if a node stores the wrong closest feature it is referring to. Still, the nodal distance may be a good approximation of the exact value. Furthermore, the error can be corrected at deeper nodes during the propagation.

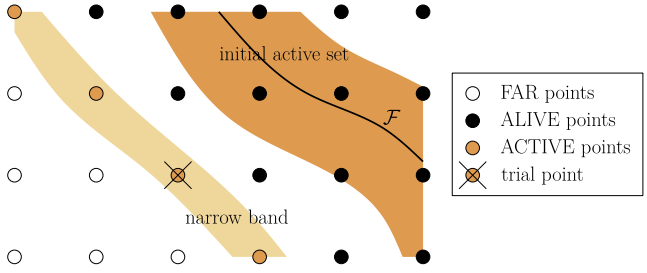

Fig. A.32. Fast Marching method on rectangular grid.

Algorithm 1. Closest Feature Front Marching

Given is a set of faces and a set of finite elements.
for all finite element nodes A being part
  of the given elements**do**
  $\text{state}_A = \text{FAR}$
end for
Create a graph which associates each finite element node
  to neighboring nodes whereby the adjacency is
  defined through the elements.
Create an empty
  set of active nodes (the narrow band) being sorted
  by their distance.
for all nodes A which are part of
  the surface**do**
  $d_A := 0,\ \text{state}_A := \text{ACTIVE}$
  Determine the associated feature.
  Take all surface patches this node is part of,
  triangulate them and collect
  the triangle patches
  to a feature.
  Add node A to the narrow band.
end for
Define the function $\text{proj}(A, f)$ which returns the
  (nonnegative) smallest
  distance between node A and the triangles of feature f.
while narrow band is not empty **do**
  Find the active node A with smallest
  distance and remove it
  from the narrow band.
  $\text{state}_A = \text{ALIVE}$
  for all neighbors B of node A**do**
    if $\text{state}_B == \text{ACTIVE}$ **then**
      Compute $g = \text{proj}(B, \text{feature}_A)$
      if $g < d_B$ **then**
        $d_B := g,\ \text{feature}_B := \text{feature}_A$
        Resort the set of active nodes.
      end if
    else if $\text{state}_B == \text{FAR}$ **then**
      $d_B := \text{proj}(B, \text{feature}_A), \text{feature}_B := \text{feature}_A,$
      $\text{state}_B := \text{ACTIVE}$
      Insert node B into the set of active nodes.
      end if
    end for
  end while
If there are any nodes A left with $\text{state}_A == \text{FAR}$ then A
  belongs to another body than the given surface patches.

Appendix B. Collision detection

During a dynamic simulation, the contact conditions must be satisfied at discrete points in time. Depending on the temporal discretization of the contact constraints, this may require an iterative solution process involving the evaluation of the constraints at predictor configurations. For the solution algorithm one generally requires information on the activity of the constraints, the current residuum at the predictor configuration and eventually derivatives of the residuum. Given a predictor coordinate vector $\mathbf{x}$ one requires the following steps in a contact algorithm:

1. Global search: One has to identify the local element coordinates $\xi^{(i)}$ on the contactor and target side for each given $\mathbf{x}$. This is equivalent to the inverse mapping ϕ^{-1} with $\mathbf{x} = \phi(\mathbf{X}(\xi))$. A brute force approach would compare all finite elements with $\mathbf{x}$ which unnecessarily increases numerical complexity. Instead, a global collision phase is done before the actual contact detection. In this phase, the set of possible collision candidates is reduced to a reasonably small number by means of very fast methods. These are potential contact pairs consisting of a contactor point and a target element/face being sufficiently close to the other.
2. Local search: The local search performs an accurate inside–outside test for the specified contact pair and finds the local coordinates $\xi^{(i)} = \xi^{(i)}(\mathbf{x})$.
3. Generation of constraint equations: For each positive detection, the discrete constraints are evaluated and temporarily stored (including discrete gradients, etc.).
4. Computation of response: Given the active set of constraints the response is computed.

If an iterative solution procedure is used, one repeats the process until the desired accuracy is obtained.

Global search

Using closest point projection, one needs to create a bounding volume around each target face. The global collision detection then intersects the bounding volume with numerical integration points located on the contactor boundary. This is different for distance fields where one can use the target finite elements directly. Hence, one does not need to specify the size of the halo (for example, the search distance along the surface normal). Global collision detection algorithms can then be used without modification. These procedures are spatial partition schemes and hierarchical representations often used to localize the regions where the actual collision appears or to delimit the domain where the exact collision test must be performed. Such representations approximate the topology of an object at different levels of detail. These include bounding volume hierarchies like sphere trees [39,40], OBB-trees (Oriented Bounding Boxes) [41], AABB (axis-oriented bounding box) trees [42] and hierarchies of k-DOPs (Discrete Orientation Polytopes) [43], as well as spatial partitioning like octrees (octant trees) [44], kd-trees [45] and position code algorithms [46–48].

Tracking algorithms may support the global detection in time stepping schemes. Therein, one seeks the new position of a point within the neighboring elements of the target element being intersected by the same point at the last time step, see for example [3].

Local search

Direct closest point projection Within closest point projection the local search solves

$$\|\mathbf{y}(\xi) - \hat{\mathbf{x}}\| \to \min, \quad \mathbf{y}(\xi) = \sum_A N_A^F(\xi)\mathbf{x}_A \tag{B.1}$$

which finds the point $\mathbf{y}$ on some finite element face which is closest to a given point $\hat{\mathbf{x}}$ on the contact frame. Here, ξ are two-dimensional coordinates defining the parameterization on the finite element face. The face is interpolated using the shape functions N_A^F. The optimality conditions lead to

$$0 = \sum_{A,B} N_A^F(\xi)\nabla_\xi N_B^F(\xi)\mathbf{x}_A \cdot \mathbf{x}_B - \sum_A \nabla_\xi N_A^F(\xi)\mathbf{x}_A \cdot \hat{\mathbf{x}} \tag{B.2}$$

which is solved iteratively.

Distance field

Compare these equations with the application of the distance field. It only involves the identification of the local finite element coordinate $\xi \in \mathbb{R}^3$ of a contactor point within a target element. The target finite elements can be used directly in global collision detection. In case of isoparametric continuum elements, the local search solves

$$0 = \mathbf{y}(\xi) - \hat{\mathbf{x}}, \quad \mathbf{y}(\xi) = \sum_A N_A(\xi)\mathbf{x}_A \tag{B.3}$$

with finite element shape function N_A.

When comparing Eqs. (B.2) and (B.3), the latter is generally faster to solve by iterative methods. Consider, for example, distorted first order hexahedral elements or distorted second order elements. Then the degree of nonlinearity is higher in (B.2). Hence, one usually requires more iterations in order to obtain similar accuracy. Furthermore, one usually must check more faces being close to $\hat{\mathbf{x}}$ than elements being intersected by $\hat{\mathbf{x}}$. The complexity of projection finding is increased if the contacting boundaries are replaced by smooth surface representations and if non-convex or non-smooth boundaries are present.

The number of iterations should be limited, for example to 5. After obtaining the local element coordinate, the actual intersection test takes place. A point with local coordinate ξ intersects a tetrahedron if $\xi_1 \geqslant 0,\ \xi_2 \geqslant 0,\ \xi_3 \geqslant 0,\ 1 - \xi_1 - \xi_2 - \xi_3 \geqslant 0$ and a hexahedron if $-1 \leqslant \xi_\alpha \leqslant 1, \alpha = \{1,2,3\}$ Thereafter, the local finite element coordinate is used to evaluate the distance field gradient given by Eq. (10).

Appendix C. Spatial discretization of the contact interface

The integration domain C in Eq. (25) is defined as the intersection of the contactor boundary and the target elements, see Fig. C.33. This is in contrast to direct closest point projection where the integration domains are the boundaries of both bodies. In feasible contact, however, C becomes the intersection of both boundaries.

Algorithms developed for node-to-segment and segment-to-segment integration can be applied directly, but the local coordinate of a numerical integration point on the target side does not refer to a target segment, but to the interior of a target element. Then

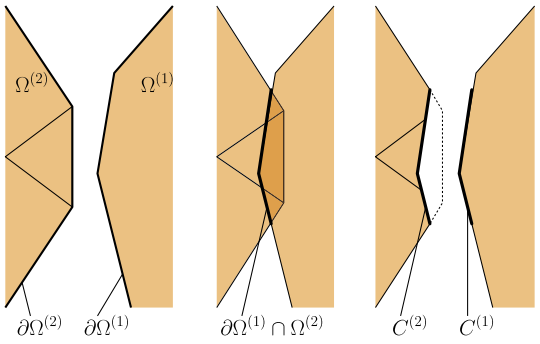

Fig. C.33. The contact interface. Left: contactor $\Omega^{(1)}$ and target $\Omega^{(2)}$. Center: Intersection of the contactor boundary with the target elements. Right: contact interfaces on contactor and target side.

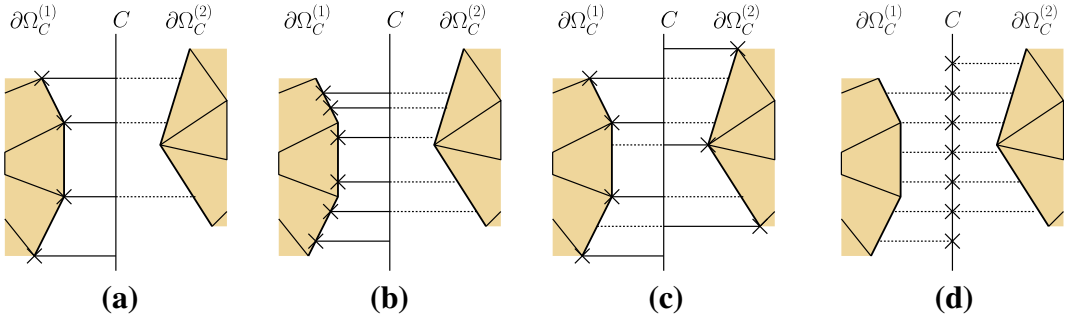

Fig. C.34. Integration point strategies on the contact frame. (a) one-sided node-to-segment. (b) segment-to-segment on one boundary. (c) both-sided node-to-segment/node-based segment-to-segment. (d) segment-to-segment on the contact frame.

one can chose from various integration schemes found in the literature, see for example Fig. C.34.

References

[1] J.O. Hallquist, G.L. Goudreau, D.J. Benson, Sliding interfaces with contact-impact in large-scale lagrangian computations, Comput. Methods Appl. Mech. Engrg. 51 (1985) 107–137.
[2] D.J. Benson, J.O. Hallquist, A single surface contact algorithm for the post-buckling analysis of shell structures, Comput. Methods Appl. Mech. Engrg. 78 (1990) 141–163.
[3] S.P. Wang, E. Nakamachi, The inside–outside contact search algorithm for finite element analysis, Int. J. Numer. Methods Engrg. 40 (19) (1997) 3665–3685.
[4] C. Kane, E.A. Repetto, M. Ortiz, J.E. Marsden, Finite element analysis of nonsmooth contact, Comput. Methods Appl. Mech. Engrg. 180 (1999) 1–26.
[5] A. Pandolfi, C. Kane, J. Marsden, M. Ortiz, Time-discretized variational formulation of non-smooth frictional contact, Int. J. Numer. Methods Engrg. 53 (2002) 1801–1829.
[6] F. Cirak, M. West, Decomposition contact response (DCR) for explicit finite element dynamics, Int. J. Numer. Methods Engrg. 64 (8) (2005) 1078–1110.
[7] N. El-Abbasi, S.A. Meguid, A. Czekanski, On the modelling of smooth contact surfaces using cubic splines, Int. J. Numer. Methods Engrg. 50 (4) (2001) 953–967.
[8] L. Krstulovic-Opara, P. Wriggers, J. Korelc, A c1-continuous formulation for 3d finite deformation frictional contact, Comput. Mech. 29 (2002) 27–42.
[9] P. Wriggers, L. Krstulovic-Opara, J. Korelc, Smooth C1-interpolations for two-dimensional frictional contact problems, Int. J. Numer. Methods Engrg. 51 (2001) 1469–1495.
[10] T. Belytschko, W.J.T. Daniel, G. Ventura, A monolithic smoothing-gap algorithm for contact-impact based on the signed distance function, Int. J. Numer. Methods Engrg. 55 (1) (2002) 101–125.
[11] M.A. Puso, T.A. Laursen, A 3d contact smoothing method using gregory patches, Int. J. Numer. Methods Engrg. 54 (8) (2002) 1161–1194.
[12] M. Stadler, G.A. Holzapfel, J. Korelc, Cn continuous modelling of smooth contact surfaces using nurbs and application to 2d problems, Int. J. Numer. Methods Engrg. 57 (15) (2003) 2177–2203.
[13] M. Stadler, G. Holzapfel, Subdivision schemes for smooth contact surfaces of arbitrary mesh topology in 3d, Int. J. Numer. Meth. Engng. 60 (7) (2004) 1161–1195.
[14] B. Flemisch, J.M. Melenk, B.I. Wohlmuth, Mortar methods with curved interfaces, Appl. Numer. Math. 54 (3–4) (2005) 339–361.
[15] B. Flemisch, B. Wohlmuth, Stable Lagrange multipliers for quadrilateral meshes of curved interfaces in 3D, Comput. Methods Appl. Mech. Engrg. 196 (8) (2007) 1589–1602.
[16] I. Temizer, P. Wriggers, T. Hughes, Contact treatment in isogeometric analysis with NURBS, Comput. Methods Appl. Mech. Engrg. 200 (2011) 1100–1112.
[17] I. Temizer, P. Wriggers, T. Hughes, Three-dimensional mortar-based frictional contact treatment in isogeometric analysis with NURBS, Comput. Methods Appl. Mech. Engrg. 209 (2012) 115–128.
[18] L. de Lorenzis, P. Wriggers, G. Zavarise, Isogeometric analysis of 3d large deformation contact problems with the augmented lagrangian formulation, Comput. Mech. 49 (2012) 1–20.
[19] A. Konyukhov, K. Schweizerhof, On the solvability of closest point projection procedures in contact analysis: analysis and solution strategy for surfaces of arbitrary geometry, Comput. Methods Appl. Mech. Engrg. 197 (2008) 3045–3056.
[20] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.
[21] J.A. Sethian, A fast marching level set method for monotonically advancing fronts, in: Proc. Nat. Acad. Sci, 1996, pp. 1591–1595.
[22] J.A. Sethian, Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science on Applied and Computational Mathematics, second Ed., Cambridge University Press, 1999.
[23] J.A. Sethian, Fast marching methods, SIAM Rev. 41 (1999) 199–235.
[24] G. Hirota, S. Fisher, A. State, C. Lee, H. Fuchs, An implicit finite element method for elastic solids in contact, in: Computer Animation 2001. The Fourteenth Conference on Computer Animation Proceedings, 2001, pp. 136–147.
[25] S. Fisher, M.C. Lin, Deformed distance fields for simulation of non-penetrating flexible bodies, in: Proceedings of the Eurographic Workshop on Computer Animation and Simulation, New York, NY, USA, 2001, pp. 99–111.
[26] G. Hirota, An improved finite element contact model for anatomical simulations, Ph.D. thesis, The University of North Carolina at Chapel Hill (2002).
[27] T.J. Barth, J.A. Sethian, Numerical schemes for the Hamilton–Jacobi and level set equations on triangulated domains, J. Comput. Phys. 145 (1998) 1–40.
[28] S. Mauch, D. Breen, A fast marching method of computing closest points, <http://www.cco.caltech.edu/sean/closestpoint/closept.html>, 2006.
[29] D. Marchal, F. Aubert, C. Chaillou, Collision between deformable objects using fast-marching on tetrahedral models, in: SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, ACM Press, New York, NY, USA, 2004, pp. 121–129.
[30] B. Heidelberger, M. Teschner, R. Keiser, M. Muller, M. Gross, Consistent penetration depth estimation for deformable collision response, in: Proceedings of Vision, Modeling, Visualization VMV, vol. 04, 2004, pp. 339–346.
[31] H.M. Mourad, J. Dolbow, K. Garikipati, An assumed-gradient finite element method for the level set equation, Int. J. Numer. Methods Engrg. 64 (8) (2005) 1009–1032.
[32] M. Puso, T. Laursen, A mortar segment-to-segment contact method for large deformation solid mechanics, Comput. Methods Appl. Mech. Engrg. 193 (2004) 601–629.
[33] T.J.R. Hughes, J.A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements NURBS, exact geometry and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4135–4195.
[34] S. Wolff, C. Bucher, A finite element method based on C0-continuous assumed gradients, Int. J. Numer. Methods Engrg. 86 (2011) 876–914.
[35] C. Geuzaine, J.-F. Remacle, Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities, Int. J. Numer. Methods Engrg. 79 (2009) 1309–1331.
[36] S. Wolff, C. Bucher, Asynchronous collision integrators: Explicit treatment of unilateral contact with friction and nodal restraints, International Journal for Numerical Methods in Engineering (2013), in press.
[37] S. Wolff, A dual mortar segment-to-element contact method: formulation and application to three-dimensional explicit dynamics, in: M. Papadrakakis, D. Charmpis, N. Lagaros, Y. Tsompanakis (Eds.), Proceedings of of the 2007 ECCOMAS Thematic Conference on Computational Methods in Structural Dynamics and Earthquake Engineering, ECCOMAS, Rethymno, Greece, 2007, p. 12.
[38] D. Gabriel, J. Kopacka, J. Plesek, M. Ulbin, Computational modelling of contact-impact problems in explicit transient dynamic analysis, in: Proceedings of ECCOMAS 2012, 6th European Congress on Computational Methods in Applied Sciences and Engineering, Vienna, Austria, 2012, p. 12.
[39] G. Bradshaw, C. O'Sullivan, Sphere-tree construction using dynamic medial axis approximation, ACM SIGGRAPH Symposium on Computer Animation (2002) 33–40.
[40] P.M. Hubbard, Collision detection for interactive graphics applications, IEEE Trans. Visual. Comput. Graphics 1 (3) (1995) 218–230.
[41] S. Gottschalk, M.C. Lin, D. Manocha, OBBTree: A hierarchical structure for rapid interference detection, Comput. Graphics 30 (Annual Conference Series) (1996) 171–180.
[42] G. van den Bergen, Efficient collision detection of complex deformable models using aabb trees, J. Graph. Tools 2 (4) (1997) 1–13.
[43] J.T. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, K. Zikan, Efficient collision detection using bounding volume hierarchies of k-DOPs, IEEE Trans. Visual. Comput. Graphics 4 (1) (1998) 21–36.

[44] M.S. Warren, J.K. Salmon, A parallel hashed oct-tree n-body algorithm, in: Supercomputing, 1993, pp. 12–21.
[45] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (1975) 509–517.
[46] R. Diekmann, J. Hungershöfer, M. Lux, L. Taenzer, J.-M. Wierum, Using space filling curves for efficient contact searching, in: Proceedings of IMACS, 2000.
[47] R. Diekmann, J. Hungershöfer, M. Lux, L. Taenzer, J.-M. Wierum, Efficient contact search for finite element analysis, in: Proceedings of ECCOMAS, 2000.
[48] M. Oldenburg, L. Nilsson, The position code algorithm for contact searching, Int. J. Numer. Methods Engrg. 37 (3) (1994) 359–386.