Using a signed distance function for the simulation of metal forming processes: formulation of the contact condition and mesh adaptation. From a Lagrangian approach to an Eulerian approach.

J. Bruchon,
`Corresponding author, E-mail address: bruchon@emse.fr`
Ecole Nationale Supérieure des Mines de Saint-Etienne
Centre Science des Matériaux et des Structures
LTDS CNRS UMR 5513
158 cours Fauriel, 42023 Saint-Etienne, France
H. Digonnet, T. Coupez
Mines ParisTech
Centre de Mise en Forme des Matériaux
CNRS UMR 7635
BP 207, 1 rue Claude Daunesse, 06904 Sophia Antipolis Cedex, France

October 29, 2008

Abstract

This paper proposes to use the metric properties of the distance function between two bodies in contact (or gap function), in simulations involving contact problems. First, the normal vectors, which are involved in the formulation of the contact condition, are defined through the gradient this distance function. This definition avoids to deal with the numerical penetration parameter, which is generally introduced otherwise. Furthermore, it allows the contact problem to be extended in a simple way to an Eulerian formulation. Second, this paper investigates two mesh adaptation strategies based on the properties of the distance function. The first strategy consists in building a size map according to the values of this function, in order to refine locally the mesh, and consequently to improve the description of the contact surface. The second strategy consists in adapting locally the mesh to the geometry of the contact surface. This anisotropic adaptation is performed by constructing a metric map which allows the mesh size to be imposed in the direction of the distance function gradient. A lot of elements are saved when compared to the isotropic case. Throughout this paper, many numerical simulations are presented in the context of the forging process: the deformable material is pressed between two rigid tools. Furthermore, the algorithm used to calculate the signed distance to a surface mesh is detailed in appendix of this paper.

KEY WORDS: signed distance function, no-penetration condition, mesh adaptation, anisotropic mesh, level-set methods.

1 Introduction

Signed distance functions are usually associated with level-set methods, used to capture interfaces when considering Eulerian approaches as in computational fluid dynamics [1]. Within this context, metric properties of these functions are well-known and largely used. For instance, we know that the normal vector to a surface described by the zero isovalue of a distance function is given by the gradient of this function, while the curvature is defined by the divergence of this normal [2]. On another hand, when considering simulations dealing with contact problems, as in [3], the approach adopted is generally Lagrangian. In such simulations, a 'gap function' is then defined to detect the contact surface between several bodies (that is, the contact between several meshes), and to express the no-penetration constraint between these bodies. This gap function is nothing else than the signed distance function between the bodies, computed generally at each surface node.

The idea of using signed distance functions to improve the solution to a contact problem is not something new. For instance, Belytschko *et al.* [4] replace a rough contact surface, described by a mesh, by a smooth surface, described by the zero isosurfaces of a distance function. In this paper, we first propose to investigate how using the metric properties of a signed distance function, in a Lagrangian context as in an Eulerian context, to formulate general expressions of the no-penetration condition. Furthermore, these metric properties allow one to improve the description of the contact surfaces. Hence, contrary to [4], these improvements are not performed through a smoothing procedure, but through mesh adaptation techniques, which are based on the distance function properties.

The formulation and the imposition of the impenetrability constraint in this paper is limited to a nodal enforcement using a penalty method, following the method described in Fourment *et al.* [5]. Of course, more sophisticated techniques could be used, as a Lagrange multiplier mixed formulation [3], or augmented Lagrangian methods as described in [6] and applied in [7]. Furthermore, for the sake of simplicity, only the contact between a deformable body and rigid tools is considered in the framework of the forging process simulation (except in Section 3.5). The material behavior is assumed to be described by a Norton-Hoff law, while the friction law is simplified through a linear Norton law. However, our approach remains accurate when considering contact between deformable bodies, or when considering more complex behaviors as elastoviscoplastic laws, or more complex friction laws.

This paper is organized as follows. The mechanical problem is established and described in Section 2. Furthermore, the contact condition is formulated in this section. Several numerical simulations carried out with this strategy are presented in Section 3. Section 4 gives two methods to adapt the mesh size in the vicinity of the contact surface. These mesh adaptation strategies are based on the metric properties of a distance function, and either provide an isotropic mesh, or an anisotropic mesh. Section 5 extends our approach of contact problems to an Eulerian context. Finally, Appendix A gives the algorithm which allows the signed distance to a surface to be computed with accuracy, when this surface is described by a mesh.

2 Mechanical and contact problem statement

In the framework of a forging process, let us consider Ω, a bounded region of $I\!R^d$ ($d = 2$ or 3) in contact with rigid tools, which are denoted by Ω_s. Figure 1 summarizes this situation when two tools are involved. Usually, the lower die does not move (the die velocity $\boldsymbol{v}_s$ vanishes on the lower die), while the upper die moves in the direction $-\mathbf{z}$ with the uniform velocity $\boldsymbol{v}_s$. The boundary of Ω, $\partial\Omega$, is composed of the sets $\Gamma_c = \partial\Omega \cap \partial\Omega_s$, the surface in contact with Ω_s, and $\partial\Omega\backslash\Gamma_c$, the free boundary.

Since a Lagrangian approach is adopted, the computational domain Ω represents the deformable body, which is deformed when the upper tool moves down. Both upper and lower dies are taken into account through the contact condition and through the friction law imposed over Γ_c. Hence, the mechanical problem consists in finding the velocity and the pressure fields $\boldsymbol{v}$ and p, which satisfy the mass and momentum conservation equations, under the constraint imposed by the contact condition of no-penetration.

2.1 Mass and momentum conservation equations

Since the thermomechanical coupling is not considered in this paper, the energy balance equation is not solved, and the temperature is assumed to be constant all along the forming process. The deformable body is assumed to be incompressible. The momentum and mass conservation laws are expressed by:

$$\begin{array}{rclcr}
\rho\dfrac{d\boldsymbol{v}}{dt} - \nabla\cdot\sigma & = & \rho\mathbf{g} & \text{in} & \Omega \\
\nabla\cdot\boldsymbol{v} & = & 0 & \text{in} & \Omega \\
\sigma\boldsymbol{n} & = & 0 & \text{over} & \partial\Omega\backslash\Gamma_c \\
\text{Contact condition} & & & \text{over} & \Gamma_c \\
\text{Friction law} & & & \text{over} & \Gamma_c
\end{array} \qquad (1)$$

where ρ is the material density. The Cauchy stress tensor σ can be decomposed into a deviatoric part s ($trace(s) = 0$) and a spherical part pI

($p = -trace(\sigma)/d$ and I is the identity tensor): $\sigma = s - pI$.

For many applications in the field of hot forging, the elastic deformation can be neglected and the classical Norton-Hoff law is used for incompressible isotropic materials:

$$s = 2K(\sqrt{3}\,\dot{\bar{\varepsilon}})^{m-1}\varepsilon(\boldsymbol{v}) \tag{2}$$

where K is the consistency of the material and m is the strain rate sensitivity coefficient. The strain rate tensor is defined by $\varepsilon(\boldsymbol{v}) = (\nabla \boldsymbol{v} + {}^t\nabla \boldsymbol{v})/2$, while $\dot{\bar{\varepsilon}} = \sqrt{\frac{2}{3}\varepsilon(\boldsymbol{v}) : \varepsilon(\boldsymbol{v})}$.

The contact condition and the friction law have now to be expressed to close system (1).

2.2 Formulation of the contact condition

To express the contact condition, we first consider $\alpha : \Omega \times I\!R^+ \to I\!R$ the signed distance function to the tool surface (see Figure 2),

$$\alpha(x,t) = \begin{cases} -\min\limits_{x_p \in \partial\Omega_s} \|x - x_p\| & \text{if} \quad x \notin \Omega_s \\ +\min\limits_{x_p \in \partial\Omega_s} \|x - x_p\| & \text{if} \quad x \in \Omega_s \end{cases} \tag{3}$$

The algorithm used to compute this signed distance function is detailed in Appendix A. For the moment, α is assumed to be known at each time $t > 0$. Let us consider a point $x \in \partial\Omega$. Regarding the definition (3), if this point verifies the equality $\alpha(x,t) = 0$, then it is exactly located at the rigid tool - deformable body interface. If the inequality $\alpha(x,t) > 0$ occurs, then the point x is penetrated of a distance $\alpha(x,t)$ into the tool. Consequently, the contact surface is defined by

$$\Gamma_c(t) = \{x \in \partial\Omega(t)\ ;\ \alpha(x,t) \geq 0\} \tag{4}$$

The evolution of the interface between the material and the rigid tool is governed by the no-penetration condition (*i.e.* by an unilateral contact). This condition is expressed by the following inequality constraint:

$$\alpha(x,t) \leq 0 \quad \forall (x,t) \in \Gamma_c \times I\!R^+ \tag{5}$$

Let us consider a partition $0 = t_0 < t_1 < \cdots < t_T = T$ of the time interval $[0,T]$. In this incremental context, the no-penetration condition can be reformulated as: at time t, find the velocity $\boldsymbol{v}^t$, solution of the mass and momentum balances, which also satisfies $\alpha(x, t + \Delta t) \leq 0$. This condition is then linearized in time:

$$\alpha(x, t+\Delta t) \simeq \alpha(x,t) + \Delta t\,\frac{\partial \alpha}{\partial t}(x,t) \leq 0 \quad \forall (x,t) \in {\Gamma_c}^t \times I\!R^+ \tag{6}$$

Recall the following differential relation: $d(\|x\|) = x \cdot dx/\|x\|$. The time derivative $\partial\alpha/\partial t$ is then expressed by differentiating definition (3):

$$\frac{\partial \alpha}{\partial t}(x,t) = (\boldsymbol{v}^t - \boldsymbol{v}_s) \cdot (x - x_p)\frac{1}{\alpha} = (\boldsymbol{v}^t - \boldsymbol{v}_s) \cdot \boldsymbol{n}_p(x_p) \tag{7}$$

where $\boldsymbol{v}_s$ is the uniform velocity of the rigid tool, *i.e.*, $\boldsymbol{v}_s = \partial x_p / \partial t$, and $\boldsymbol{n}_p$ is the interior unit normal to the tool surface.

The formulation of the contact condition is achieved by relating the vector $x - x_p$ to $\nabla\alpha$. Since the surface of the tools is described by the level set $\{x \in I\!R^d \,;\, \alpha(x,t) = 0\}$, the gradient $\nabla\alpha$ is related to the interior unit normal to this surface (according to the sign convention chosen in (3)) by

$$\boldsymbol{n}_p(x) = \frac{\nabla\alpha}{\|\nabla\alpha\|}(x) \tag{8}$$

Consequently,

$$x - x_p = \left(\alpha \frac{\nabla\alpha}{\|\nabla\alpha\|}\right)(x) \tag{9}$$

Both previous relations (8) and (9) hold for all point x located on the boundary $\partial\Omega_s$. These relations are sufficient to treat the continuous problem. However, when considering the discrete problem, the contact condition is not applied only at each point of the surface $\partial\Omega_s$, but more generally at each point (or node) of $\partial\Omega$ that satisfies $\alpha(x,t) \geq -\varepsilon$, where ε is a small positive numerical parameter. For this reason, relations (8) and (9) have to be extended in a neighborhood of $\partial\Omega_s$. In fact, in this case, and because α is a distance function, these relations hold at the first order. More precisely, taking x in the neighborhood of $\partial\Omega_s$, the first-order Taylor's expansion of α is written as:

$$\alpha(x+h) = \alpha(x) + \nabla\alpha(x) \cdot h + o(\|h\|) \tag{10}$$

Since the projection x_p of x on $\partial\Omega_s$ verifies $\alpha(x_p) = 0$, taking $h = x_p - x$ in expression (10) leads to:

$$\nabla\alpha(x) \cdot (x - x_p) = \alpha(x) + o(\|x - x_p\|) \tag{11}$$

By neglecting the second-order terms, we can write, in the neighborhood of the contact surface:

$$\nabla\alpha(x) \cdot (x - x_p) = \alpha(x) \tag{12}$$

Since $\|\nabla\alpha\| = 1$ (α is a distance function) and $|\alpha| = \|x - x_p\|$, Equation (12) implies $|\cos(\nabla\alpha, x - x_p)| = 1$, *i.e.*:

$$x - x_p = \alpha(x)\,\nabla\alpha(x) \tag{13}$$

This expression is similar to relation (9), except that it holds at the first order and at each point in a neighborhood of $\partial\Omega_s$. Finally, previous developments (6), (7) and (13) lead to the following formulation of the no-penetration constraint:

$$\frac{\alpha(x,t)}{\Delta t} + (\boldsymbol{v} - \boldsymbol{v}_s) \cdot \frac{\nabla\alpha}{\|\nabla\alpha\|}(x,t) \leq 0 \quad \forall (x,t) \in \Gamma_c \times I\!R^+ \tag{14}$$

This Equation requires some additional comments. First, the accuracy of the previous developments depends on the differentiability of the function α. Of course, if Ω_s is not a convex set, α is not differentiable everywhere. However, when $\partial\Omega_s$ is smooth enough, α is locally differentiable in a neighborhood of this boundary. See [8] for the theoretical aspects about this subject. The second point that has to be underlined is the geometrical meaning of the linearization (6): at each point x of Γ_c, the tool surface is locally approximated by the plane $(x, \nabla\alpha(x))$. In the simulations presented in this paper, this approximation is made at the beginning of the time step (see Section 3.1).

Finally, the key point of this formulation is that the evaluation of the projected point x_p is not required to compute the vector $x - x_p$. Indeed, the projection on the tool surface of a point $x \in \partial\Omega$ being exactly in contact ($\alpha(x,t) = 0$), is the point itself ($x = x_p$). In this case, the penetration direction $x - x_p$ can not be directly evaluated. To overcome this difficulty, the approaches that evaluate the projection point x_p, can allow a small numerical penetration ε_p for the nodes in contact with the tool surface. See [5] for further details. In this case ε_p depends on the mesh size. Our approach allows one to avoid the introduction of this additional numerical parameter. Note that, computing the gradient $\nabla\alpha$, requires that the distance α is defined not only on the contact surface, but also in the volume Ω. Appendix A shows that the central processing unit (CPU) time induced by this additional calculation is limited when using a cellular representation of the tool meshes. Moreover, the mesh adaptation strategies of Section 4, as well as the Eulerian description of Section 5, are based on this volume representation of the distance function.

2.3 Friction law

Dealing with the friction laws is not the aim of this paper. However, in the forging process context, a rigid body motion can appear when only a few points are in contact in a frictionless simulation. For this reason, a simple linear Norton friction law is considered in the following. This law states the proportionality between the shear stress and the relative tangential velocity:

$$\tau = -fK\Delta\boldsymbol{v}_{tan} \quad \text{over } \Gamma_c \tag{15}$$

where parameter f is the coefficient of the friction law. The shear stress τ is defined by $\tau = \sigma\boldsymbol{n} - (\sigma\boldsymbol{n} \cdot \boldsymbol{n})\boldsymbol{n}$. For consistency reasons with the contact term, the unit normal pointing outward the material is $\boldsymbol{n} = \nabla\alpha / \|\nabla\alpha\|$. The relative tangential velocity is then written as $\Delta\boldsymbol{v}_{tan} = (\boldsymbol{v} - \boldsymbol{v}_s) - [(\boldsymbol{v} - \boldsymbol{v}_s) \cdot \boldsymbol{n}]\boldsymbol{n}$.

2.4 Weak form and discretization of the mechanical problem

The following functional spaces are first introduced: $\mathcal{V} = (H^1(\Omega))^d$ and $\mathcal{P} = L^2(\Omega)$. The weak mixed formulation of Equations (1) and (15), under the constraint (14), is written as (see [5]):

$$
\begin{aligned}
&\text{At time } t+\Delta t \text{, knowing } \Omega^t,\ \Gamma_c^t,\ \alpha^t \text{ and } \boldsymbol{v}^t \text{, Find } (\boldsymbol{v},p) \in \mathcal{V}\times\mathcal{P} \text{ such that}\\
&\frac{\rho}{\Delta t}\int_{\Omega^t} \boldsymbol{v}\cdot\boldsymbol{w}\, d\Omega + \int_{\Omega^t} 2K(\sqrt{3}\,\dot{\bar{\varepsilon}})^{m-1}\varepsilon(\boldsymbol{v}) : \varepsilon(\boldsymbol{w})\, d\Omega - \int_{\Omega^t} p\nabla\cdot\boldsymbol{w}\, d\Omega\\
&\qquad + \int_{\Gamma_c^t} fK\Delta\boldsymbol{v}_{tan}\cdot\boldsymbol{w}\, d\Gamma\\
&\qquad + r\int_{\Gamma_c^t} [\frac{\alpha^t}{\Delta t} + (\boldsymbol{v}-\boldsymbol{v}_s)\cdot\frac{\nabla\alpha^t}{\|\nabla\alpha^t\|}]^+ \boldsymbol{w}\cdot\frac{\nabla\alpha^t}{\|\nabla\alpha^t\|}\, d\Gamma_c\\
&\qquad = \frac{\rho}{\Delta t}\int_{\Omega^t} \boldsymbol{v}^t\cdot\boldsymbol{w}\, d\Omega + \int_{\Omega^t} \rho\mathbf{g}\cdot\boldsymbol{w}\\
&-\int_{\Omega^t} q\,\nabla\cdot\boldsymbol{v}\, d\Omega = 0\\
&\text{for any test functions } (\boldsymbol{w},q) \text{ in } \mathcal{V}\times\mathcal{P},
\end{aligned}
\tag{16}
$$

where r denotes the penalty coefficient ($r >> K$) used to impose the contact constrain. The operator $x \mapsto [x]^+$ is the positive part operator defined by $[x]^+ = \max(x,0)$. Furthermore, the time derivative $\partial\boldsymbol{v}/\partial t$ has been discretized by using an implicit Euler scheme. The superscript "$t+\Delta t$" has been omitted above the unknowns $\boldsymbol{v}$ and p. Finally, the measure $d\Gamma_c$ used in the contact term depends on the formulation adopted for the contact discretization and can be either the Lebesgue's measure $d\Gamma$ (integrated contact), or the discrete Dirac's measure δ_{node} (nodal contact).

The computational domain Ω is discretized with an unstructured simplex mesh $\mathcal{T}_h(\Omega)$. Within a Lagrangian context, the nodal update of this mesh is performed by an explicit Eulerian scheme (see Section 3.1):

$$\Omega^{t+\Delta t} = \Omega^t + \Delta t\, \boldsymbol{v}^{t+\Delta t} \tag{17}$$

The mixed formulation (16) is completely discretized by using the MINI-element. This finite element, introduced in [9], is a first-order unstructured element with a linear continuous interpolation of both pressure and velocity, and a bubble enrichment of the velocity. This bubble function, which is necessary to satisfy the LBB stability condition (c.f. [10]), can be seen as an enrichment of the element by four piecewise linear functions.

Furthermore, system (16) presents two nonlinear terms. The first one is due to the Norton-Hoff law (2) when $m \neq 1$. The second nonlinearity is induced by the no-penetration contact constraint ($[.]^+$ is a nonlinear operator). A Newton-Raphson method is then used to solve the discretized system. This method requires the differentiation of system (16) with respect to $\boldsymbol{v}$ and p. However, the functional $x \mapsto [x]^+$ is not differentiable in $x = 0$: its left-sided derivative in this point is equal to 0, while its right-sided derivative is equal to 1.

We choose to apply the Newton-Raphson method without treating this no-differentiability (see [5]). Following this way, we take:

$$\frac{d}{dx}[x]^+ = \begin{cases} 0 & \text{if} \quad x < 0 \\ 1 & \text{if} \quad x \geq 0 \end{cases} \tag{18}$$

This approach can lead to a decrease of the Newton-Raphson method residue which is not monotone. Especially, this residue can increase when passing from the first Newton method iteration to the second one. However, this approach let show a better behavior of the Newton-Raphson method than when using a regularized positive part operator (the regularized operator we used was defined as $[x]^+_\varepsilon = x(1 + sgn_\varepsilon(x))/2$, with $sgn_\varepsilon(x) = x/(|x| + \varepsilon)$).

2.5 Discretization of the contact condition

Let us consider α_h, a piecewise linear continuous function, which is the discretized distance to the tool surface ($\alpha_{h|K} \in P1(K), \forall K \in \mathcal{T}_h(\Omega)$), where $\mathcal{T}_h(\Omega)$ is an unstructured simplex mesh of the computational domain. The value of α_h at each node is defined by expression (3). A nodal approach of the contact imposition is considered in this paper. Hence, the discrete contact surface is defined as the following set of nodes:

$$\Gamma^h_c(t) = \{n \text{ node of } \partial\mathcal{T}_h(\Omega)\ ;\ \alpha_h(n) \geq -\varepsilon_c\} \tag{19}$$

where ε_c is a positive numerical precision parameter, which is chosen equal to one tenth of the mesh size in our simulations.

The no-penetration constraint (14) is therefore enforced for each node $n \in \Gamma^h_c$. This means that the contact term involved in system (16) is discretized by taking $d\Gamma^h_c = \sum_{n \in \Gamma^h_c} S_n \delta_n$, where $\delta_n(x)$ is the Dirac measure centered on the node n (equal to 1 on n and vanishing otherwise). The local contact surface S_n is defined as

$$S_n = \sum_{S \in \partial\mathcal{T}_h(\Omega) \cap \mathcal{S}_n} \int_S \phi_n d\Gamma$$

where $\mathcal{S}_n$ is the set of the faces that share the node n as a common vertex, while ϕ_n is the shape function associated to the node n of the face S. Considering the previous developments, the discrete contact term takes the form

$$r \sum_{n \in \Gamma^h_c} S_n [\frac{\alpha_h(n)}{\Delta t} + (\boldsymbol{v} - \boldsymbol{v}_s) \cdot \mathbf{n_c}(n)]^+ \mathbf{e}_i \cdot \mathbf{n_c}(n) \quad i = 1, \cdots, d \tag{20}$$

where $\mathbf{e}_i$ is the ith vector of the standard basis of $I\!R^d$. Furthermore, since $\alpha_{h|K} \in P1(K)$, the normal vector $\nabla\alpha_{h|K}$ is constant on an element K and can not be directly used in (20). For this reason, $\mathbf{n_c}(n)$, the unit normal to the node n has been introduced in (20). This vertex normal is computed as a weighted average of $\nabla\alpha_h$:

$$\mathbf{n_c}(n) = \frac{\sum\limits_{F \in \mathcal{C}_n} \nabla \alpha_{h|F}}{\| \sum\limits_{F \in \mathcal{C}_n} \nabla \alpha_{h|F} \|}, \tag{21}$$

with $\mathcal{C}_n = \{F \in \mathcal{S}_n \cap \partial \mathcal{T}_h(\Omega) \ ; \ F$ has $N_c(n)$ vertexes in $\Gamma_c^h\}$ and $N_c(n) = \max_{F \in \mathcal{S}_n \cap \partial \mathcal{T}_h(\Omega)} N_c(F)$, where $N_c(F)$ is the number of vertexes of the face F that are in contact, *i.e.* that belong to Γ_c^h. To conclude, we state that the restriction of the distance gradient on a face F of $\partial \mathcal{T}_h(\Omega)$, $\nabla \alpha_{h|F}$, is nothing else than the value of this gradient in the volume element whose F is a face, as explained in Figure 3.

3 Numerical applications

In this section, we present four simulations which involve contact problems, obtained by using the numerical strategy developed above. These simulations were carried out by using the CIMLIB finite element library. This C++ library, which is highly parallel, is developed at CEMEF by the team of T. Coupez (see [11]). The two first simulations are quite 'academic'. The results obtained with our code, referred as Forge++, are then compared to those obtained with the industrial software Forge3®. The third simulation involves more complex contact surfaces. Finally, the last simulation shows an example of contact between several deformable bodies.

The Forge3® software, which is developed in CEMEF for a long time, is now a reference for the forging process simulation in an industrial context. The contact formulation implemented in this code is close from the one of Forge++: the unpenetrability constraint is enforced at each node by a penalty method (see [5]). However, the gap function at the contact nodes in Forge3® is computed by evaluating the penetration direction $x - x_p$. For the reasons discussed at the end of Section 2.2, a small numerical penetration has then to be considered. The following comparisons aims to validate the 'signed distance function approach' developed in this paper to compute the penetration direction without dealing with such a numerical penetration.

3.1 Time stepping strategy

To summarize the numerical strategy developed in this paper, a simulation involving contact between a deformable material and rigid tools, is performed through the following steps.

For each time step, do:

1. Positioning the rigid tools at time t.
2. For each node n of the mesh $\mathcal{T}_h(\Omega)$, computation of the signed distance from n to the die surface, following the algorithm described in Appendix A.

3. Computation of $\boldsymbol{v}$ and p, solution of (16), by using a Newton-Raphson method.
4. Update of the coordinates of the nodes using Equation (17).
5. If necessary, computation of a new size map or metric map (see Section 4) and remeshing step. The fields defined on the mesh (as the equivalent strain) are transported from the old mesh to the new adapted mesh.

The stopping criterion can be the time or the final height of the upper die in a forging process. Note that in this scheme, the signed distance function is computed only once per time step. This choice corresponds to approximate locally the tool by its tangential plane computed at the considered contact point and at the beginning of the time step, as explained in Section 2.2 (see also [5]).

3.2 Flat-die forging process simulation

This simulation consists in compressing a workpiece between two flat dies, as described in Figure 4 (the flat dies are not shown in this figure). The upper die is pressed down onto the workpiece with a constant velocity $\boldsymbol{v}_s = 0.1\ m.s^{-1}$, while the lower die stays fixed. The material behavior obeys to the Norton-Hoff law (2) with $K = 10\ kPa.s$ and $m = 0.1$, while the friction law is linear with $\alpha_f = 0.3$ in (15). The initial height of the workpiece is $2 \times 10^{-1}\ m$, and the time step chosen for the simulation is 0.01 s.

Figure 4 compares the results obtained with Forge++ to those obtained with Forge3®, after 20 time steps, by using the same mesh (26 000 elements, 5 000 nodes) and without considering the remeshing step. The comparison point is the equivalent strain, which is defined, at time t, by $\overline{\varepsilon} = \int_0^t \dot{\overline{\varepsilon}} d\tau$. With the considered velocity discretization, the equivalent strain is piecewise constant on each mesh element. Figure 4(a) shows the isovalues of the equivalent strain, plotted on a cutting plane of the workpiece, for the case where the simulation is performed by using Forge++ (left) and for the case where Forge3® is used (right). To complete this description, the equivalent strains are plotted and compared along a vertical line passing through the center of the workpiece (see Figure 4(b)). The results obtained with Forge++ and Forge3® are shown to be almost identical. Of course, in these simulations, the normal vectors involved in the contact condition are uniform (equal to $(0, 0, 1)$ for the upper die and to $(0, 0, -1)$ for the lower die). In this sense, the flat-die process simulation is not completely relevant to assert the validity of our formulation of the contact condition. However, this first simulation allows one, on one hand, to check our mechanical solver for a non-linear behavior ($m = 0.1$), and, on another hand, to check the good imposition of the contact condition with the penalty method.

3.3 Simulation of the deformation of a cube

This simulation, shown in Figure 5, consists in deforming the unit cube by pressing down its upper face with a rigid rod (as previously, the tools are not shown in this figure). The cube is standing on a flat lower die, as in the previous simulation. The rigid rod has a length larger than 1 (the cube edge length), a width equal to 0.5, and is centered on the cube upper face. The velocity of this upper die is constant and equal to -1 following the z axis. The cube material behaves as a Newtonian fluid ($m = 1$) with a consistency K equal to 10 (as previously). The friction law between the cube and the dies is linear, with $\alpha_f = 0.3$.

This simulation is carried out by using Forge++ and Forge3®, respectively. The time step is of 0.01. In both cases, the initial meshes are identical. However, because of the large distorsions of the mesh, a remeshing step is applied periodically all the 5 time steps. The remeshing strategies of Forge++ and Forge3® are different: Forge3® adapts the mesh with a special treatment of the surface elements and with respect to the mesh curvature, while Forge++ adopts the strategy described in Section 4.1. Hence, although their element numbers are approximately equal, the meshes involved in these simulations are different.

During the simulation, a part of the lower face of the cube loses progressively contact with the rigid die. The first key point of this simulation is that for both Forge++ and Forge3®, this loss of contact occurs at same time (at $t = 0,52$ s). Hence, exactly the same contact surface is obtained by using our code and by using Forge3®. Furthermore, the final geometries (at time $t = 0.69$) of the deformed cube, obtained with these approaches, are found quite similar. Finally, the same similarity is shown when comparing the equivalent strain (see Figure 5). The differences observed on the curves of Figure 5 are explained by the size map which differs from Forge++ to Forge3®.

This case remains relatively simple. However, it proves, in a case where the normal to the tool surfaces varies, that our numerical strategy allows one to predict correctly the evolution of the contact surface as well as the one of the deformable body geometry.

3.4 Simulation of a crankshaft forging process

To conclude with the contact between a deformable workpiece and rigid tools, we present in Figure 6 the simulation of a crankshaft forging process. This figure describes the evolution of the workpiece geometry, without showing the rigid tools. The isovalues of the distance to these tools are also shown. The parameters of the material behavior, as well as those of the friction law, are the same as in the cube deformation case.

This simulation (and others, like the forging of a spindle), performed with Forge++ was compared with success to Forge3®, according to the same criteria as previously (shape of the workpiece, contact surface, equivalent strain). Note that Forge3® presents a better behavior during the last step of the forging process, when flash starts to appear. This point, especially important for Forge3®, is not due to the approach adopted to treat the contact condition, but it is the consequence of a special treatment of the workpiece surface mesh, as well as of a special work for making the mechanical solver more robust in this special case. However, the purpose of this paper is not to treat the flash formation, but to give general methods based on the properties of the signed distance function to deal with numerical aspects of contact problems. Within this context, Section 4 will give methods to adapt, in a simple way, the mesh in the vicinity of the contact zone. Such methods could also be extended to treat the case of the flash formation (see the conclusion of this paper).

3.5 Simulation of the contact between three deformable bodies

Up to now, the different simulations presented in this paper were involving a deformable body in contact with several rigid tools. We conclude this section by showing that, as mentioned in the general introduction, our numerical strategy can be extended, without difficulty, to the contact between several deformable bodies. Of course, a specific algorithm has to be developed to project the surface nodes of a body on the surface of the other bodies, and the parallelized version of this algorithm is not straightforward to obtain. However, these aspects do not change our general approach of using the signed distance function in contact problems.

Without detailing the implementation of the contact between deformable bodies, we just mention that a Master - Slave strategy has been adopted. Moreover, one single disconnected mesh is used to descretize the different bodies (see Figure 7). The signed distance function α, defined on the whole mesh, is then computed connected component by connected component. On a connected component, α is equal to the signed distance (Equation 3) to the other components.

In the deformable body case, the unpenetrability constraint analogous to Equation (14), is expressed as

$$\frac{\alpha(x,t)}{\Delta t} + (\boldsymbol{v}(x) - \boldsymbol{v}(\Pi(x)) \cdot \frac{\nabla\alpha}{\|\nabla\alpha\|}(x,t) \leq 0 \quad \forall (x,t) \in \Gamma_c \times I\!R^+ \tag{22}$$

This constraint is enforced on the contact zone of the slave body, for all Master - Slave couple. That means that the nodes of the contact surface Γ_c (definition (4)) belong to the slave body, while Π is the orthogonal projection operator from the slave body to the master body.

A simulation carried out by using this strategy is presented in Figure 7: a deformable body (a cube), in contact with a moving flat upper die (not shown in the figure) comes to press down two other bodies which are standing on a flat lower die. The isovalues of the signed distance function are shown in this figure. Note that the mesh has been refined near the contact zone by using the method developed below, in Section 4.1.

4 A mesh adaptation strategy based on the properties of the signed distance function

The efficiency of a forging process simulation, especially when using the previous numerical strategy, depends on the degree of accuracy obtained for the contact area description. Furthermore, using a normal vector computed with the gradient of the distance function, as shown in previous sections, requires a mesh that is sufficiently thin with a good element quality in the neighborhood of the contact area.

In this section, we propose two strategies to adapt locally the mesh during a forming process simulation. Both strategies are based on the distance function to pre-adapt the mesh in the zones that are going to be in contact. The first approach uses only the values of the distance function to compute a size map and refine locally the mesh. In this case, the mesh is isotropic. The second approach consists in building a metric map which uses the gradient of the distance function to adapt the mesh according to the tool geometry. In this case, the mesh is anisotropic.

We outline that all the meshes shown in this paper have been generated through CIMLIB by the MTC mesher, developed by T. Coupez. This mesher, which is based on a topological optimization technique, is detailed in [12] and in [13] for the anisotropic mesh generation.

4.1 Construction of a size map according to the signed distance to the tool surface

The idea of this approach is to gradually refine the mesh when approaching the contact surface. In this way, the mesh becomes locally refined, whereas the gradient of the size map can be controlled: there is no a sudden jump of the size map from node to node.

The simplest way to perform such a mesh h-adaptation, is to consider a size map $\mathcal{S} : \Omega \rightarrow I\!R$, which is equal to a "background size" s_{max} when $\alpha_h \geq d_{max}$ and is equal to a minimal size s_{min} when $|\alpha_h| \leq d_{min}$. Moreover, $\mathcal{S}$ is chosen continue in Ω and linear when $d_{min} \leq |\alpha_h| \leq d_{max}$. The size map $\mathcal{S}$ is written as

$$\mathcal{S}(x) = \begin{cases} s_{min} & \text{if} \quad |\alpha_h(x)| \leq d_{min} \\ (1-t)s_{min} + ts_{max} & \text{if} \quad d_{min} \leq |\alpha_h(x)| \leq d_{max} \\ s_{max} & \text{if} \quad |\alpha_h(x)| \geq d_{max} \end{cases} \tag{23}$$

with

$$t = \frac{|\alpha_h(x)| - d_{min}}{d_{max} - d_{min}}$$

The gradient of $\mathcal{S}$ in the transition area (when $0 < t < 1$) is equal to $\pm(s_{max} - s_{min})/(d_{max} - d_{min})$, and is vanishing elsewhere. It has to be outlined that the parameters s_{max}, s_{min}, d_{max} and d_{min} are not calculated by using an error estimate. Their values are simply related to the characteristic lengths of the computational domain and to the desired element number. The mesh size provided by this approach is isotropic: it has the same value in all the directions.

Figure 8 shows the mesh of a crankshaft obtained by using the size map (23) with the following parameters: $d_{min} = 4.5$, $d_{max} = 14$, $s_{min} = 3$, $s_{max} = 12$. The mesh has 324 222 nodes and 1 597 592 elements. Note that the contact area is made visible by regarding the mesh size: almost all the surface nodes are in contact.

4.2 Construction of a metric map according to the tool surface geometry

When performing adaptations with previous method, only one parameter is taken into account, say the distance to the tool boundary. However, it can be interesting to consider a mesh which is, moreover, locally adapted to the tool geometry, *i.e.* to the geometry (the curvature) of the contact surface. In rough terms, for a same level of accuracy, the discretization of a curved surface with an isotropic mesh, requires much more elements than the discretization of a planar surface. Hence, using an anisotropic mesh, with elements stretched in a 'good' direction, could allow one to save a lot of elements.

Let us recall that an anisotropic mesh is built by considering a metric map $\mathcal{M}$ (see [13]). A 'metric' is a symmetric positive-definite matrix. It is well-known that with such properties, a new inner product and then a new norm can be defined in $I\!R^d$ ($\|x\|_{\mathcal{M}} = (x^T M x)^{1/2}$, $x \in I\!R^d$). The anisotropic mesh is then constructed by considering elements having a size equal to 1 in the $\|\cdot\|_{\mathcal{M}}$ norm. This strategy provides a mesh size equal to $1/\sqrt{\lambda_i}$ (in the usual Euclidian norm) in the direction ν_i, where λ_i is the ith eigenvalue of the metric, and ν_i the associated eigenvector.

The construction of a metric map $\mathcal{M}$ which takes into account the tool geometry, is discussed below. This metric is based on the gradient $\nabla\alpha$ of the distance function. It allows one to impose, in the contact surface vicinity, two different mesh sizes, in both directions $\nabla\alpha$ and $\nabla\alpha^{\perp}$ (the orthogonal directions defined by $\nabla\alpha \cdot \nabla\alpha^{\perp} = 0$).

More precisely, considering a vector $\mathbf{a} \in I\!R^d$, the matrix $A = \mathbf{a} \otimes \mathbf{a}$ (or with an indicial notation $A_{ij} = a_i a_j$) is first defined. This matrix is symmetric and positive ($x^T A x = x_i A_{ij} x_j = a_i a_j x_i x_j = (\sum_{i=1}^{d} a_i x_i)^2 \geq 0$ with the Einstein summation convention).

The first eigenvalue of A is $\lambda_1 = \|\mathbf{a}\|^2$, with the associated eigenvector $\nu_1 = \mathbf{a}$. Indeed, we have: $(A\nu_1)_i = a_i a_j \nu_{1j} = a_i a_j a_j = \|a\|^2 \nu_{1i}$.

However, the second eigenvalue of A is $\lambda_2 = 0$ with a multiplicity of $d-1$. Its associated eigenvector is $\nu_2 = \mathbf{a}^{\perp}$ ($(A\nu_2)_i = a_i a_j \nu_{2j} = 0$, since $\mathbf{a} \cdot \mathbf{a}^{\perp} = 0$). Hence, A is symmetric positive, but not symmetric positive-definite. Consequently A is not a metric. For this reason, the following regularized tensor is then introduced:

$$A_{\varepsilon} = m^2 A + \varepsilon^2 I$$

where I is the identity tensor, while m and ε are two positive real parameters. Now A_ε is symmetric positive-definite: it has the two same eigenvectors as A, $\mathbf{a}$ and $\mathbf{a}^\perp$, but the associated eigenvalues are $m^2\|\mathbf{a}\|^2+\varepsilon^2$ and ε^2, respectively. This means that, using the metric tensor A_ε leads to a mesh size of $1/\sqrt{m^2\|\mathbf{a}\|^2+\varepsilon^2}$ in the direction $\mathbf{a}$ and to a mesh size of $1/\varepsilon$ in the direction $\mathbf{a}^\perp$.

Following the same idea as in the isotropic case, we want to use a metric map which is equal to an isotropic metric when being far from the contact zone, and which is equal to A_ε in the vicinity of the contact zone. Moreover, this map is chosen continuous and linear in the transition area. In this way, we define $\mathcal{M} : \Omega \to I\!R^{d^2}$ by:

$$\mathcal{M}(x) = \frac{t}{s_{max}^2} I + (1-t)[m^2 \mathbf{a} \otimes \mathbf{a} + \epsilon^2 I] \tag{24}$$

where,

$$\begin{cases} t = 0 & \text{if} \quad |\alpha(x)| \le d_{min} \\ t = 1 & \text{if} \quad |\alpha(x)| \ge d_{max} \\ t = \dfrac{|\alpha| - d_{min}}{d_{max} - d_{min}} & \text{if} \quad d_{min} \le |\alpha(x)| \le d_{max} \end{cases}$$

In this expression, the parameters d_{min}, d_{max} and s_{max} have the same meaning as in Equation (23). In our simulations, $\mathcal{M}$ is defined node-to-node. The vector $\mathbf{a}$ is then taken as being the unit normal to the tool surface, defined for a node n by $\mathbf{a}(n) = \sum_{K\in\mathcal{K}_n} \nabla\alpha_{h|K} / \| \sum_{K\in\mathcal{K}_n} \nabla\alpha_{h|K}\|$, where $\mathcal{K}_n$ is the set of elements that have the node n as common vertex. Since $\|\mathbf{a}(n)\| = 1$, using the metric map (24) allows one to impose a mesh size of $1/\sqrt{m^2+\varepsilon^2}$ in the normal direction to the tool surface and a mesh size of $1/\varepsilon$ in the tangent plane to this surface.

Figure 9 shows the mesh of the same crankshaft as previously, obtained by using the size map (24) with the parameters $d_{min} = 4.5$, $s_{max} = 12$, $1/\sqrt{m^2+\varepsilon^2} = 2$ (size in the direction of $\nabla\alpha_h$) and $1/\varepsilon = 10$ (size in the orthogonal direction). The mesh that is obtained has 91 347 nodes and 468 946 elements. The anisotropy of this mesh appears clearly when regarding Figure 9. Furthermore, compared to the isotropic case of Figure 8, a lot of elements are saved, since the mesh passes from 1 597 592 elements to 468 946 elements for the same values of d_{min}, d_{max} and s_{max}, and a size in the direction of $\nabla\alpha_h$ smaller than s_{min}.

Meshes obtained during the simulation of a connecting rod, and shown in Figure 10, provide another comparison between both approaches presented below. The mesh size of the isotropic mesh makes the contact zone directly visible in Figure 10(a), while the tool geometry is clearly taken into account by the anisotropic mesh in Figure 10(b). Furthermore, from 474 382 for the isotropic mesh, the element number drops to 59 327 for the anisotropic mesh, thanks to a mesh size which is equal to 1 in all the directions in the isotropic case, but which is equal to 5 in the tangent plane and to 1 in the normal direction, in the anisotropic case.

5 Toward an Eulerian approach of the contact problem

Up to now, the simulations presented in this paper were carried out using a Lagrangian approach: the deformable body was identified to the computational domain Ω, while the rigid tools were represented only through their interactions with this deformable body. We propose to conclude this paper by extending our analysis to an Eulerian context. This last step seems quite natural, since using signed distance functions is usually associated to level set methods, which are interface capturing techniques used in the Eulerian frameworks. Developing an Eulerian approach for solving contact problems, remains a numerical challenge, even if a substantial progress has been made recently (see for example [14] and [15]). However, contrary to this literature, a 'fully' Eulerian approach is presented in this paper, in the sense that the contact surface is never rebuilt but remains known implicitly through the zero isovalue of the distance function. Note that the definition of the unit normal used in this paper (see Equation (8)) is appropriate for this fully Eulerian description, since this definition occurs not only on the mesh surface, but in the whole mesh volume where the distance α_h is defined. As it will be seen, this extension implies that the contact constraint, which is a surface constraint, can easily be turned into a volume condition in the neighborhood of the contact surface. The reader can refer to Sethian [1] and Osher *et al.* [2] to have an overview on the level set methods and of their applications. We point out that, in this section, the tools are still assumed to be rigid, without deformation (contrary, for example, to [16]). We are then focusing on how to formulate and impose the unpenetrability between deformable and rigid bodies, when these bodies are described by a level set method.

5.1 Mixed weak formulation of the mechanical problem

Let the computational domain Ω be composed of a deformable body Ω_f, a solid zone (rigid tools) Ω_s, and a gas part (the air) Ω_a, such that: $\Omega = \Omega_f \cup \Omega_s \cup \Omega_a$ (see Figure 11). As previously, the material behavior obeys to the Norton-Hoff law (2) of consistency K_f, while tools and air are considered as Newtonian fluids of consistencies K_s and K_a respectively. The densities of these domains are denoted by ρ_f, ρ_s and ρ_a.

The whole domain Ω is meshed with a fixed simplex mesh $\mathcal{T}_h(\Omega)$. Material and tool domains are characterized by the signed distance functions to their boundaries, α_f and α_s respectively (see definition (3)). The location of the air domain is then deduced by complementarity and does not require the introduction of an additional distance function.

Global density and consistency fields, ρ and K, are defined on the whole domain Ω by

$$\rho = H(\alpha_f)(\rho_f - \rho_a) + H(\alpha_s)(\rho_s - \rho_a) + \rho_a \tag{25}$$

and

$$K = H(\alpha_f)(K_f - K_a) + H(\alpha_s)(K_s - K_a) + K_a \tag{26}$$

where the characteristic function H is defined by

$$H(\alpha)(x) = \begin{cases} 0 & \text{if} \quad \alpha(x) < -\varepsilon \\ \frac{1}{2}(\frac{\alpha(x)}{\varepsilon} + 1) & \text{if} \quad |\alpha(x)| < \varepsilon \\ 1 & \text{if} \quad \alpha(x) > \varepsilon \end{cases} \tag{27}$$

Following this definition, the interface width is of 2ε. In our simulations, the numerical parameter ε is chosen such that $2\varepsilon = 1.5h$, where h is the mesh size. The global mixed velocity-pressure formulation of the mechanical problem (momentum and mass conservation) is finally formulated as:

At time $t + \Delta t$, α_f and α_s being known, find $(\boldsymbol{v}, p) \in \mathcal{V} \times \mathcal{P}$ that verify

$$\begin{array}{lll} \frac{1}{\Delta t} \int_\Omega \rho \boldsymbol{v} \cdot \boldsymbol{w} \, d\Omega & + & \int_\Omega \rho(\boldsymbol{v} \nabla \boldsymbol{v}) \cdot \boldsymbol{w} \, d\Omega \\ & + & \int_\Omega 2K(\sqrt{3}\,\dot{\bar{\varepsilon}})^{m-1} \varepsilon(\boldsymbol{v}) : \epsilon(\boldsymbol{w}) \, d\Omega - \int_\Omega p \nabla \cdot \boldsymbol{w} \, d\Omega \\ & + & \text{contact term} \; + \; \text{friction term} \\ & = & \frac{1}{\Delta t} \int_\Omega \rho \boldsymbol{v}^t \cdot \boldsymbol{w} \, d\Omega + \int_\Omega \rho \mathbf{g} \cdot \boldsymbol{w} \\ & & \\ -\int_\Omega q \, \nabla \cdot \boldsymbol{v} \, d\Omega & = & 0 \end{array} \quad \text{fof any test functions } (\boldsymbol{w}, q) \text{ in } \mathcal{V} \tag{28}$$

where $\boldsymbol{v}^t$ denotes the velocity field at the previous time step. The parameter m varies over Ω (typically, $m = 1$ in Ω_s and Ω_a). Furthermore, a full treatment of the advective term $\boldsymbol{v} \nabla \boldsymbol{v}$ would require a linearization of this term and a stabilization of this term by using, for example, a SUPG technique. However, we choose to perform our computations with an explicit approach of this advective term: $\boldsymbol{v} \nabla \boldsymbol{v}$ is evaluated at the previous time step and is considered only in the right member of system (28) (see [17]). Hence, the Newton-Raphson method is used only to linearize the contact term.

5.2 Eulerian formulation of the no-penetration constraint

In an Eulerian context, the contact surface between two bodies Ω_f and Ω_s is defined by

$$\Gamma_c(t) = \{x \in \Omega \; ; \; |\alpha_f(x,t)| + |\alpha_s(x,t)| = 0\} \tag{29}$$

The no-penetration condition is expressed by the inequality

$$\alpha_f(x,t) + \alpha_s(x,t) \leq 0 \quad \forall (x,t) \in \Gamma_c \times I\!R^+ \tag{30}$$

Following the same approach as in Section 2.2, linearizing the inequality (30) leads to

$$g(\alpha_f, \alpha_s) := \frac{\alpha_f(x,t)}{\Delta t} + \frac{\alpha_s(x,t)}{\Delta t} + (\boldsymbol{v} - \boldsymbol{v}_s) \cdot \boldsymbol{n}_s \leq 0 \quad \forall (x,t) \in \Gamma_c \times I\!R^+ \tag{31}$$

where $\boldsymbol{n}_s = \nabla \alpha_s / \|\nabla \alpha_s\|$. Hence the contact term can be written as

$$\int_{\Gamma_c} r[g(\alpha_f, \alpha_s)]^+ \, \boldsymbol{w} \cdot \boldsymbol{n}_s \, d\Gamma \tag{32}$$

where r is the penalty parameter. However, this surface integral is not computable within an Eulerian context. Indeed, the surface Γ_c is not known explicitly (it is not a set of mesh faces), but it is defined by the zero level set of the function $|\alpha_f| + |\alpha_s|$, which passes through the mesh elements. Consequently, surface integral (32) has to be turned into volume integral. Regarding [18] and [19], the function $\zeta : I\!R \to I\!R$ is first introduced:

$$\zeta(y) = \begin{cases} \frac{1}{2}(1 + \cos(\pi y)) & \text{if} \quad |y| < 1 \\ 0 & \text{if} \quad |y| \geq 1 \end{cases} \tag{33}$$

With this definition, the map $y \mapsto \frac{1}{\varepsilon}\zeta(\frac{\alpha}{\varepsilon})|\nabla \alpha|$ tends to the Dirac mass $\delta_{\{\alpha=0\}}$, in the dual space of continuous functions with compact support, when ε tends to zero (see [19]). Hence, the following approximation can be written:

$$\delta_{\Gamma_c} \approx \frac{1}{\varepsilon}\zeta(\frac{|\alpha_f| + |\alpha_s|}{\varepsilon})|\nabla|\alpha_f| + \nabla|\alpha_s|| \tag{34}$$

In the vicinity of the contact surface (that is, at a distance of Γ_c lower than ε), we can assume that $\alpha_f \simeq -\alpha_s$. Hence, $|\nabla|\alpha_f| + \nabla|\alpha_s|| \simeq 2|\nabla|\alpha_f|| = 2$, since $|\nabla \alpha_f| = 1$ by definition of a distance function. Finally, the contact term involved in (28) is written as:

$$\int_{\Omega} \frac{2r}{\varepsilon}[g(\alpha_f, \alpha_s)]^+ \, \zeta(\frac{|\alpha_f| + |\alpha_s|}{\varepsilon}) \boldsymbol{w} \cdot \boldsymbol{n}_s \, d\Omega \tag{35}$$

We point out that this volume formulation (35) of the contact term has been established thanks to the definition (8) of the normal vector $\boldsymbol{n}_s$.

The friction term of Equation (28) corresponds to the linear Norton law (15), turned into volume integral thanks to the approximation (34).

5.3 Advection of the signed distance functions

Once system (28) is solved at time t, the domains Ω_f and Ω_s are advected by the velocity $\boldsymbol{v}$, solution of (28). The deformable domain is advected through the distance function α_f, which is solution of a pure advection equation [1]:

$$\begin{array}{ll} \frac{d\alpha_f}{dt} = \frac{\partial \alpha_f}{\partial t} + \boldsymbol{v} \cdot \nabla \alpha_f = 0 & \forall t > 0 \\ \alpha_f(x, 0) = \alpha_0(x) & \forall x \in \Omega, \text{ at } t = 0 \end{array} \tag{36}$$

Note that, even though α_0 is initially a signed distance function (that is, $|\nabla\alpha_0| = 1$), the solution of Equation (36) does not generally conserve this property. Hence, steep gradients may progressively appear near the interface which becomes irregular. Consequently, after solving Equation (36), an additional re-initialization step is generally necessary, which consists in modifying the solution for obtaining $|\nabla\alpha_f| = 1$, without changing the zero isovalue, i.e. without perturbing the interface position (see Sussman and Fatemi [20]).

However, in our simulations, the advection step is carried out by using the approach described in [21] and [22]. Following this approach, the advection step and the re-initialization step are performed at the same time, by solving one single advection equation. Furthermore, instead of verifying the distance property $|\nabla\alpha_f| = 1$, the function α_f is chosen as being a sinusoidal function, *i.e.* α_f verifies $|\nabla\alpha_f| = \sqrt{1-(\pi\alpha_f/2E)^2}$. Consequently, α_f is the distance function only in the vicinity of the zero level set. In this way, α_f can be taken equal to a constant value $\pm 2E/\pi$, beyond a distance E from the zero level set.

Since the domain Ω_s is assumed to be rigid, the distance function α_s is not advected by solving an equation: it is directly recomputed at each node of the mesh according to the displacements of the tools, which are assumed to be known.

5.4 Numerical simulation of the deformation of a square

The simulation presented in this section is quite similar to the one shown in Section 3.3, except that it is a 2D case. The computational domain is a rectangle 4×1.5 (see Figure 12), discretized with a 'fixed mesh'. The term fixed mesh means that the mesh does not move with the material, and consequently can not become distorted. However, the Eulerian simulation presented in Figures 12 and 13, is performed by applying an h-adaptation technique, at each time step, in order to obtain an accurate description of the interfaces, while having a reasonable number of elements. This h-adaption technique is the one exposed in paragraph 4.1, applied with $\alpha_h = \max(\alpha_f, \alpha_s)$, $s_{min} = 0.01$, $d_{min} = 0.025$, $s_{max} = 0.08$ and $d_{max} = 0.15$. The solution to the finite element problem (velocity, pressure and distance function α_f) found on the 'old' mesh, is then simply re-interpolated on the new mesh by using the linear shape functions.

When the simulation starts, the deformable body is the unit square, standing on a fixed lower die (a rectangle 4×0.1). The upper die, a rigid disk of radius 0.1, whose center is initially located at coordinates $(2, 1.2)$, comes down to press the material with a constant velocity equal to 1. This tool velocity is imposed as a Dirichlet condition at the nodes which are 'interior' to the tools, *i.e.* at the nodes which verify $\alpha_s > 0.025$.

The material viscosity is constant, $\eta_f = 10$, the air 'viscosity' is $\eta_a = 0.001$, while the tool 'viscosity' is taken equal to $\eta_s = 0.001$ too. This choice can appear to be surprising, but when considering Equation (26), taking a more 'realistic' value of η_s (for instance, $\eta_s = 10000$), leads to a viscosity of same order at the material - tool interface, making appear an excessive numerical 'friction' between the tools and the deformable body. Furthermore, since the degrees of freedom of the velocity are blocked for the nodes which are interior to the tools, this tool 'consistency' η_s play a role only near the tool - deformable body interface. Note that the interface width defined by Equation (27), is equal to $2\varepsilon = 0.024$. The parameter f of the friction law is taken equal to 0.3.

In Figures 12 and 13(a), the interfaces are directly made visible by the mesh size. However, the zero level set of α_f and α_s are also plotted in Figures 12(b) and 13(a). The key point of this simulation is that, as in Section 3.3, a part of the lower edge of the deformable body loses progressively contact with the lower die (see Figure 13(a)). This loss of contact occurs exactly at the same time when using a Lagrangian approach (see Figure 13(b)). Furthermore, the shape of the deformable body obtained by using the Eulerian and the Lagrangian approaches are shown to be comparable. These results show the ability of our Eulerian approach to describe the interaction between several bodies, when each body is characterized by a distance function which evolves through a fixed mesh.

6 Conclusion

This paper presents several ways to use the properties of the signed distance functions in simulations with contact problems, within a Lagrangian context, as well as within an Eulerian context. First, the distance function, usually used only to detect the contact zone, allows one to formulate the contact condition. Indeed, the normal vector involved in the contact expression can be expressed by computing the gradient of the distance function. In a Lagrangian framework, following this approach, the small numerical penetration that is usually allowed, is not necessary any more. Furthermore, this definition of the normal vector, allows the contact condition formulation to be extended in a simple way to an Eulerian framework. The Lagrangian simulations are compared with success to the Forge3® software, while the Eulerian approach is confronted, still with success, to the Lagrangian approach.

Second, we have presented two mesh adaptation strategies based on the distance function to the tool surface. The mesh adaptation aims to refine locally the mesh in the vicinity of the contact surface, in order to improve the description of this surface, while having a reasonable number of elements. The first strategy simply consists in refining the mesh according to the distance function values. The mesh is therefore isotropic. The second strategy consists in calculating a metric map to impose the mesh size in the direction of the gradient of the distance function and in the associated orthogonal direction. In this approach, the mesh is anisotropic and takes into account the geometry of the contact surface. Hence a lot of elements are saved when compared to the isotropic case. Finally, we give in appendix a fast algorithm to compute the signed distance function to a surface mesh.

Several outlooks and applications can be envisaged for this work. First, in the Lagrangian approach, only the distance to the tool surface has been considered to perform mesh adaptation. However, it could be interesting to take into account the geometry of the deformable body, by constructing a metric map based on the gradient of the distance to the computational domain boundary. This approach could improve the treatment of the flash during a forging process simulation. Second, the Eulerian formulation of the contact condition can be applied to model more realistic conditions at the fluid - mold interface, in simulations of injection or extrusion processes, for example (see [23]). Finally, the Eulerian formulation of the contact could allow the topological changes (as the self-contact or the crack propagation) to be 'implicitly' considered. In this perspective, the mesh adaptation strategies developed in this paper, could be combined with a thinner imposition of the contact condition using, for example, Lagrange multipliers.

APPENDIX

A Calculation of the signed distance function

This appendix details the algorithm we have developed to calculate the signed distance function to the tool surface in the previous Lagrangian simulations. Since the tool surface is discretized by a simplex mesh (a set of triangles for three-dimensional simulations or a set of segments for two-dimensional simulations), the problem is the computation of the signed distance from a given point (a node of the computational domain) to a surface mesh. This makes appear two 'difficulties'. The first one is to determine the distance sign, as explained in [24]. In other words, it is a question of determining whether the point is inside or outside of a complex oriented surface. The second difficulty that has to be overcame, is the reduction as much as possible of the computation time of the distance function. Indeed, this computation is performed at each time step.

A.1 Evaluation of the sign of the distance function

Let us consider a point $x \in I\!R^d$ (a node of the mesh $\mathcal{T}_h(\Omega)$), and $\mathcal{S}_h = \{F_n\}_{n=1,\cdots,nbF}$ a set of two- or three-dimensional oriented 'faces', triangles or segments (*i.e.* a surface mesh). By denoting $|d(x,F)|$ the unsigned distance from x to a face F, the unsigned distance from x to $\mathcal{S}_h$ is simply defined by

$$|d(x,\mathcal{S}_h)| = \min_{F \in \mathcal{S}_h} |d(x,F)| \tag{37}$$

The algorithm used to calculate $|d(x,F)|$ is classical and detailed in [25]. Here, our purpose is to evaluate the signed distance $d(x,\mathcal{S}_h)$. Since a face $F \in \mathcal{S}_h$ is oriented, the sign of $d(x,F)$ is easily obtained by using $\boldsymbol{n}_F$, the outward-pointing normal to this face. Following the convention (3), we have:

$$sign(d(x,F)) = \begin{cases} +1 & \text{if} \quad (x-x_p)\cdot\boldsymbol{n}_F < 0 \\ -1 & \text{if} \quad (x-x_p)\cdot\boldsymbol{n}_F > 0 \end{cases} \tag{38}$$

where x_p is the projection of x on F. However, applying this strategy to determine the sign of $d(x,\mathcal{S}_h)$ is not straightforward, as shown in Figure 14. In this two-dimensional figure, the set $\mathcal{S}_h$ is composed of three oriented segments: $\mathcal{S}_h = \{[A,B],[B,C],[C,A]\}$. The projection on $\mathcal{S}_h$ of a point x located in the dashed area coincides with the point B ($x_p = B$), which belongs to both segments $[A,B]$ and $[B,C]$. Furthermore, the inner product $(x-B)\cdot\boldsymbol{n}_{AB}$ is positive, while $(x-B)\cdot\boldsymbol{n}_{BC}$ is negative. Consequently, the sign of $d(x,\mathcal{S}_h)$ can not be evaluated directly.

To overcome this difficulty, we introduce a new parameter, the 'quality' of a projection, defined by

$$Q(x,F) = |(x-x_p)\cdot\boldsymbol{n}_F| \tag{39}$$

Then, we choose the sign obtained with the face that maximizes the quality parameter. Hence, in the example shown in Figure 14, we have $|(x-B)\cdot \boldsymbol{n}_{AB}| > |(x-B)\cdot \boldsymbol{n}_{BC}|$. Consequently, the signed distance from a point x located in the dashed zone, to the set $\mathcal{S}_h$ is negative: $d(x,\mathcal{S}_h) = -\|x-B\|$.

The algorithm described below is based on the quality parameter to compute the signed distance $d(x,\mathcal{S}_h)$. Two distances are first computed: d^+, a positive distance, and d^-, a negative distance, with their associated qualities, Q^+ and Q^-, respectively. Finally, $d(x,\mathcal{S}_h)$ is taken equal to d^+ or d^- according to the absolute value of these distances and according to their qualities. Considering a point $x \in I\!R^d$, the algorithm that computes the signed distance $d(x,\mathcal{S}_h)$, is as following:

```
    $begin$
// Initialization step
$real\ d^- \leftarrow -\infty$
$real\ d^+ \leftarrow +\infty$
$real\ Q^- \leftarrow 0$
$real\ Q^+ \leftarrow 0$
$real\ d_{aux}, Q_{aux}$ // auxiliary distance and quality

// Loop over the elements of the surface mesh
$integer\ i$
$for\ (i = 1, \cdots, nbF)$ {
    $getDistanceAndQualityToFace(i, x, d_{aux}, Q_{aux})$
    $if\ (d_{aux} < 0)$ {
        $real\ d_{relat} = (d_{aux} - d^-)/|d^-|$
        $if\ (\ (d_{relat} > \varepsilon)\ OR\ (\ (|d_{relat}| \le \varepsilon)\ AND\ (Q_{aux} > Q^-)\ )\ )$ {
            $d^- \leftarrow d_{aux}$
            $Q^- \leftarrow Q_{aux}$
        }
    }
    $else$ {   // $d_{aux}$ is positive
        $real\ d_{relat} = (d^+ - d_{aux})/d^+$
        $if\ (\ (d_{relat} > \varepsilon)\ OR\ (\ (|d_{relat}| \le \varepsilon)\ AND\ (Q_{aux} > Q^+)\ )\ )$ {
            $d^+ \leftarrow d_{aux}$
            $Q^+ \leftarrow Q_{aux}$
        }
    }
}   // end of loop

// Now we have to choose between $d^+$ and $d^-$:
$real\ distance$
$if\ (|d^+ + d^-|/d^+ < \varepsilon)$ { // $d^+$ and $|d^-|$ are of same order
    $if\ (Q^- > Q^+)$
        $distance \leftarrow d^-$
    $else$
```

```
        $distance \leftarrow d^+$
}
else {
    if $(d^+ + d^- < 0)$
        $distance \leftarrow d^+$
    else
        $distance \leftarrow d^-$
}
end
```

At the end of this algorithm, the variable '*distance*' is equal to $d(x, \mathcal{S})$. The numerical precision parameter ε is used to compare the distances d^+, d^- and d_{aux}. It is equal to 1.0×10^{-10} in our computations. Furthermore, only the relative distances (obtained by dividing the distances by d^+ or $|d^-|$) are compared in order to have a method that does not depend on the mesh units. Finally, the function $getDistanceAndQualityToFace(i, x, d_{aux}, Q_{aux})$ calculates the distance from the point x to the ith face of $\mathcal{S}$ following the method described in [25] and with the sign given by expression (38). This distance and its associated quality are set in the auxiliary variables d_{aux} and Q_{aux}.

A.2 A hierarchical representation of the surface mesh

Following the previous algorithm, the determination of the distance $d(x, \mathcal{S}_h)$ requires to evaluate the 'auxiliary' distances $d(x, F)$ for the nbF faces $F \in \mathcal{S}_h$. The CPU time cost of this kind of algorithm is typically linear with respect to nbF (see Figures 16(a) and 17(a) which will be explained below). In order to decrease this cost, a hierarchical representation of $\mathcal{S}_h$ is built at the beginning of the simulation. Such a representation consists in packing the elements of $\mathcal{S}_h$ into a set of recursively defined cells (that is, a set of rectangles in 2D and a set of cubes in 3D). In this way, the distance between a point x and a cell C_i is first evaluated, before possibly computing the distance between x and the subcells of C_i, or between x and the elements contained in C_i.

More precisely, the hierarchical representation of the surface mesh corresponds to the following tree data structure (a quadtree in 2D, an octree in 3D):

- The root C_0 of the tree is the minimal box (rectangle or cube) which contains the whole mesh $\mathcal{S}_h$.
- Each node of the tree is a cell C_i. If the number of elements contained in C_i exceeds a prescribed number, this cell is split into 4 (in 2D) or 8 (in 3D) subcells of equal sizes. These subcells are the new nodes of the tree, considered as the children of C_i. This process is the applied recursively until each (sub-)cell reaches the prescribed number of elements.

Once the hierarchical structure is built, the process used to compute the signed distance $d(x, \mathcal{S}_h)$ is defined recursively as following. Let us consider all the cells which belong to the same level in the hierarchical representation. First, the minimal and maximal distances from x to each of these cells are calculated (this step requires only the computation of the distances between x and each corners of the cells, and therefore is performed very quickly). Second, some of these cells are removed from the process. Indeed, let A and B be two cells which belong to the same level as depicted in Figure 15, and let $d_A^{max}, d_A^{min}, d_B^{max}$ and d_B^{min} be the maximal and minimal distances from x to A and B respectively. If $d_B^{min} > d_A^{max}$, then the cell B is removed from the process because the distance from x to any element of B is still larger than the distance from x to any element of A. This situation is illustrated in Figure 15, where all the grey cells are removed from the process. Finally, the same selection process is applied recursively to the subcells (or daughter cells) of the remaining cells. When the last level of the tree is reached, the distance between x and the elements which belong to the cells of this last level, is computed by using the algorithm described in the previous section.

Figures 16 and 17 show the evolution of the signed distance computation time with respect to the number of elements of the surface mesh. The distance evaluation is performed without and with using the hierarchical representation of the surface mesh. When using the hierarchical representation, the maximal number of elements per cell (per leaf, in fact) is fixed to 4 in 2D and to 128 in 3D. Note that this number depends on the mesh (in 3D) and must be larger than the degree of connectivity of the mesh (*i.e.* the maximal number of elements which share a same node as a common vertex). The improvement involved by this hierarchical representation appears clearly when comparing the CPU time scales of Figures 16(a) and 16(b) on one hand, and 17(a) and 17(b) on the other hand. Furthermore, when using the hierarchical representation in the 3D case 17(b), the CPU time variation which is shown when passing from a surface mesh of 20 000 faces to a surface mesh of 80 000 faces, is only due to the 'internal' properties of the mesh such as the degree of connectivity. This means that the CPU time cost of the method in this range of number of elements is approximately constant.

References

[1] Sethian JA. Level Sets Methods and Fast Marching Methods. *Cambridge Monograph on Applied and Computational Mathematics*, vol. 3. 1986.

[2] Osher S., Fedkiw F. Level Set Methods: An Overview and Some Recent Results. *Journal of Computational Physics* 2001; **169**(2):463–502.

[3] Bathe K.J., Brezzi F. Stability of finite element mixed interpolation for contact problems. *Rendiconti Lincei Matematica E Applicazioni* 2001; **12**(3):167–183.

[4] Belytschko T., Daniel W.J.T., Ventura G. An monolithic smoothing-gap algorithm for contact-impact based on the signed distance function. *International Journal for Numerical Methods in Engineering* 2002; **55**(1):101–125.

[5] Fourment L, Chenot J.L., Mocellin K. Numerical formulations and algorithms for solving contact problems in metal forming simulation. *International Journal for Numerical Methods in Engineering* 1999; **46**(9):1435–1462.

[6] Fortin M., Glowinski R. Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems. *Amsterdam-New York, North-Holland Publ. Co.* 1983.

[7] Baba O.A., Radi B., Gelin J.-C. An augmented Lagrangian treatment of the metal forming process. *Mathematical and Computer Modelling* 2000; vol. **32**(10):1171–1179.

[8] Poliquin R.A., Rockafellar R.T., Thibault L. Local differentiability of distance functions. *Transactions of the American Mathematical Society* 2000; **352**(11):5231–5241.

[9] Arnold D.N., Brezzi F., Fortin M. A stable finite element for the Stokes equations. *Calcolo* 1983; **21**(4):337–344.

[10] Fortin M., Brezzi F. *Mixed and Hybrid Finite Element Method.* Springer: Berlin, 1991.

[11] Digonnet H, Coupez T. Object-oriented programming for 'fast-and-easy' development of parallel applications in forming processes simulation. *In Second MIT Conference on Computational Fluid and Solid Mechanics, Bathe KJ (ed.), Massachusset Institute. Elsevier: Amsterdam*, 2003; 1922–1924.

[12] Coupez T. Génération de maillage et adaptation de maillage par optimisation locale. *Revue Européenne des éléments finis* 2000; **49**(4):403–423.

[13] Gruau C., Coupez T. 3D tetrahedral, unstructured and anisotropic mesh generation with adaptation to natural and multidomain metric. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**(48):4951–4976.

[14] Benson D. J., Okazawa S. Contact in a multi-material Eulerian finite element formulation. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**(39-41):4277–4298.

[15] Vitali E., Benson D. J. An extended finite element formulation for contact in multi-material arbitrary Lagrangian-Eulerian calculations. *International Journal for Numerical Methods in Engineering* 2006; **67**(10):1420–1444.

[16] Legay A., Chessa J., Belytschko T. An Eulerian-Lagrangian method for fluid-structure interaction based on level sets. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(17-18):2070–2087.

[17] Coupez T., Basset O., Digonnet H. Stabilisation for capturing moving surface and incompressible Navier Stokes solution. *In 8th ESAFORM Conference on Material Forming, D. Banabic (Ed.), Publishing House of Romanian Academy, Bucarest, Romania*, 2005; **1**.

[18] Cottet G.-H., Maitre E. A level-set formulation of immersed boundary method for fluid-structure interaction problems. *Comptes Rendus Mathematique* 2004; **338**(7):581–586.

[19] Cottet G.-H., Maitre E. A level set method for fluid-structure interactions with immersed surfaces. *Mathematical Models and Methods in Applied Sciences* 2006; **16**(3):415–438.

[20] Sussman M, Fatemi E. An efficient interface preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM Journal on Scientific Computing* 1999; **20**(4):1165–1191.

[21] Coupez T. Réinitialisation convective et locale des fonctions Level Set pour le mouvement de surfaces et d'interfaces. *Journées Activités Universitaires de Mécanique, La Rochelle* 2006;ISBN 2-9526-8123-8:33–41.

[22] Bernacki M., Chastel Y., Coupez T., Logé R.E. Level set framework for the numerical modelling of primary recrystallization in polycrystalline materials. *Scripta Materialia* 2008; **58**(12):1129–1132.

[23] Valette R., Bruchon J., Digonnet H., Laure P., Leboeuf M., Silva L., Vergnes B., Coupez T. Méthodes d'interaction fluide-structure pour la simulation multi-échelles des procédés de mélange. *Mécanique et Industries* 2007; **8**(3):251–258.

[24] Bærentzen J. A., Aanæs H. Signed Distance Computation Using the Angle Weighted Pseudonormal. *IEE Transactions on Visualization and Computer Graphics* 2005;**11**(3):243–253.

[25] Schneider P., Eberly D. Geometric Tools for Computer Graphics. *Morgan Kaufmann series in Computer Graphics.* San Francisco, 2002 (http://www.geometrictools.com).

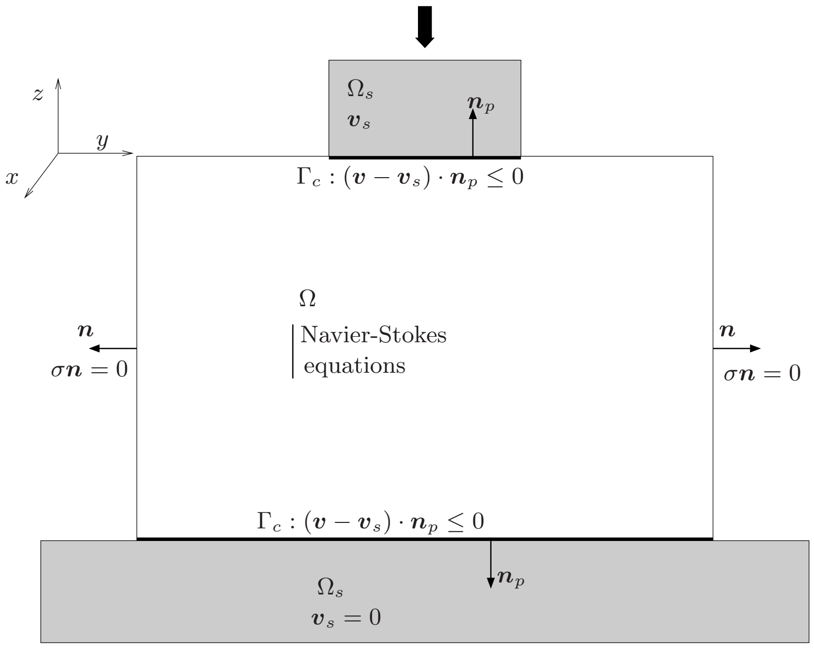

z
y
x
Ω_s
$\boldsymbol{v}_s$
$\boldsymbol{n}_p$
$\Gamma_c : (\boldsymbol{v} - \boldsymbol{v}_s) \cdot \boldsymbol{n}_p \leq 0$
Ω
Navier-Stokes equations
$\boldsymbol{n}$
$\sigma\boldsymbol{n} = 0$
$\boldsymbol{n}$
$\sigma\boldsymbol{n} = 0$
$\Gamma_c : (\boldsymbol{v} - \boldsymbol{v}_s) \cdot \boldsymbol{n}_p \leq 0$
Ω_s
$\boldsymbol{v}_s = 0$
$\boldsymbol{n}_p$

Figure 1: Definition of the forming problem.

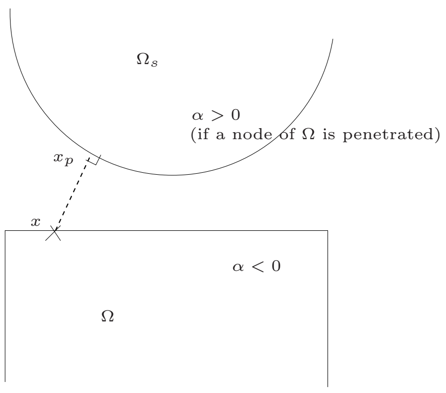


Figure 2: Definition of the signed distance function (Equation (3))

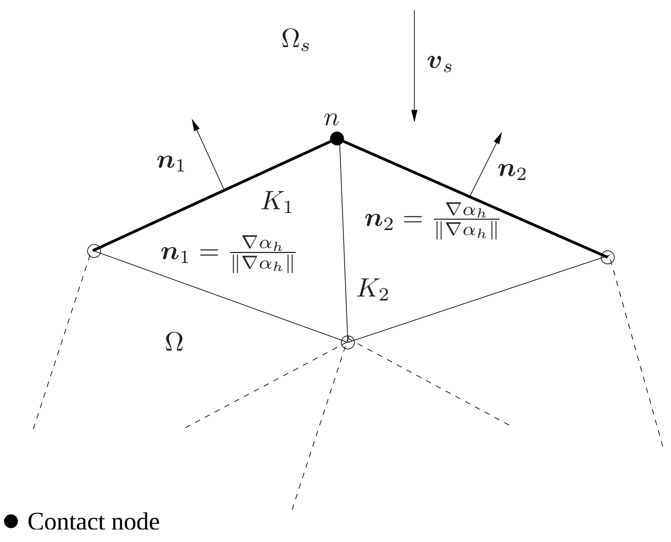
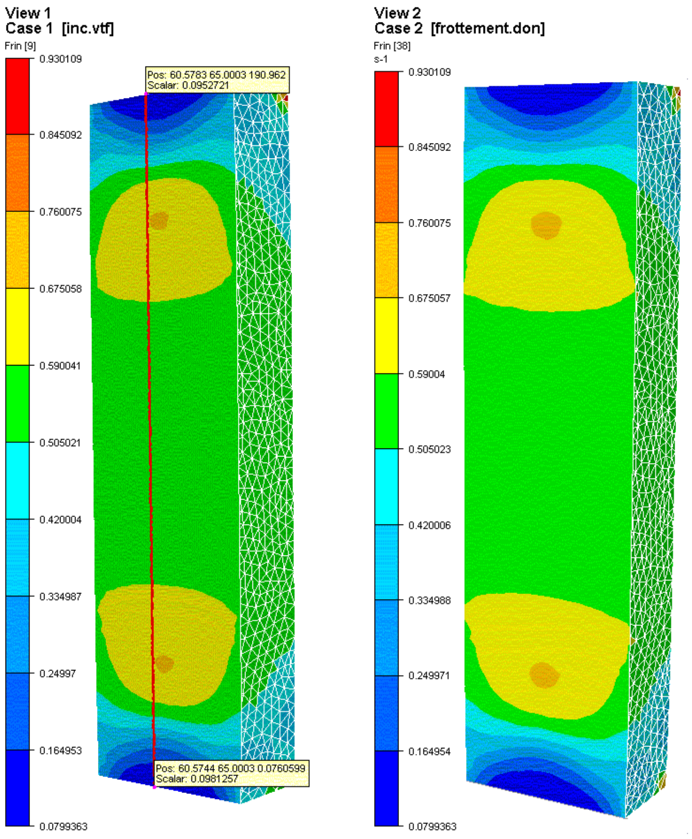
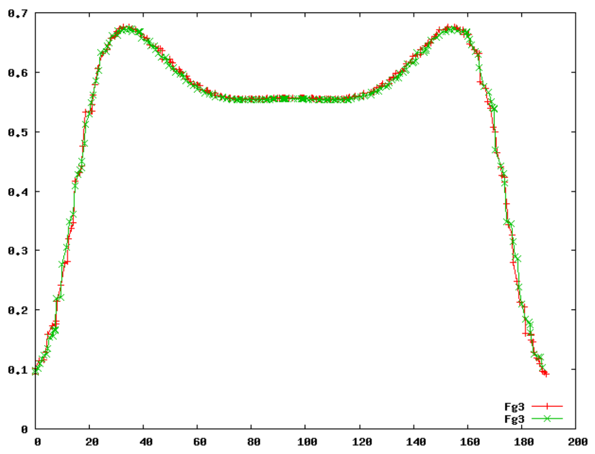

Ω_s
$\boldsymbol{v}_s$
n
$\boldsymbol{n}_1$
$\boldsymbol{n}_2$
K_1
$\boldsymbol{n}_2 = \frac{\nabla \alpha_h}{\|\nabla \alpha_h\|}$
$\boldsymbol{n}_1 = \frac{\nabla \alpha_h}{\|\nabla \alpha_h\|}$
K_2
Ω
● Contact node

Figure 3: Face normal and vertex normal.

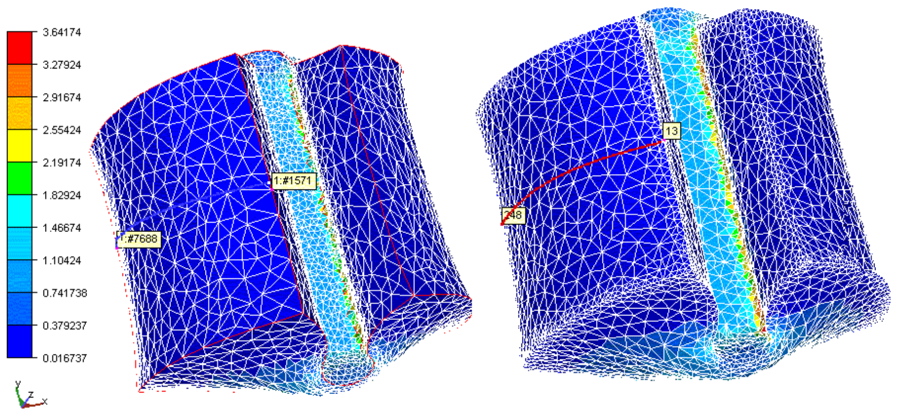


(a) Cutting plane - Isovalues of the equivalent strain.

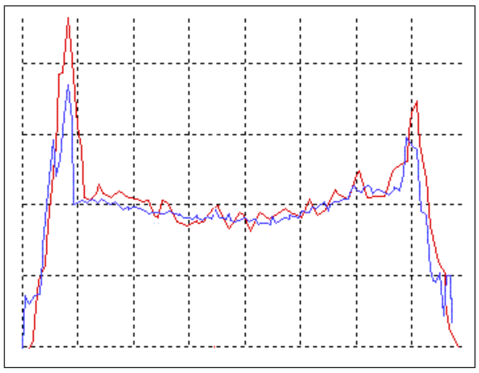


(b) Equivalent strain plotted on the red line of Figure 4(a).

Figure 4: Comparison of the equivalent strain obtained with Forge++ (left) and Forge3® (right) after the deformation between two flat dies of a solid workpiece.



(a) Isovalues of the equivalent strain.



(b) Equivalent strain plotted on the lines of Figure 5(a) (red = Forge3®, blue = Forge++).

Figure 5: Deformation of the unit cube. Comparison between Forge++ (left) and Forge3® (right) of the equivalent strain $\bar{\varepsilon}$ obtained after 70 time steps: isovalues and comparison along a line (graph on the right).

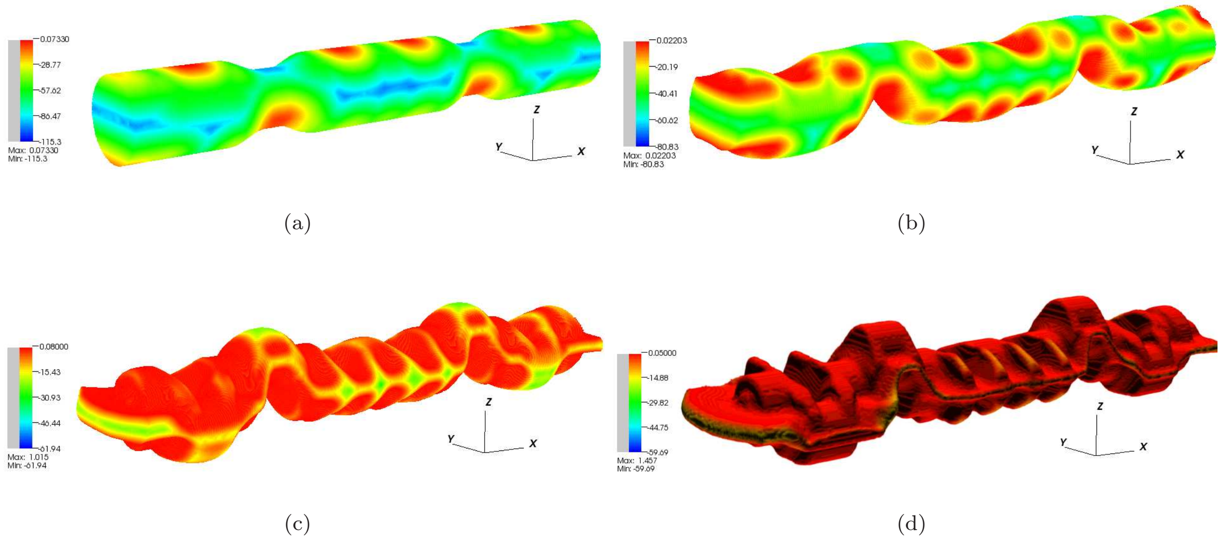
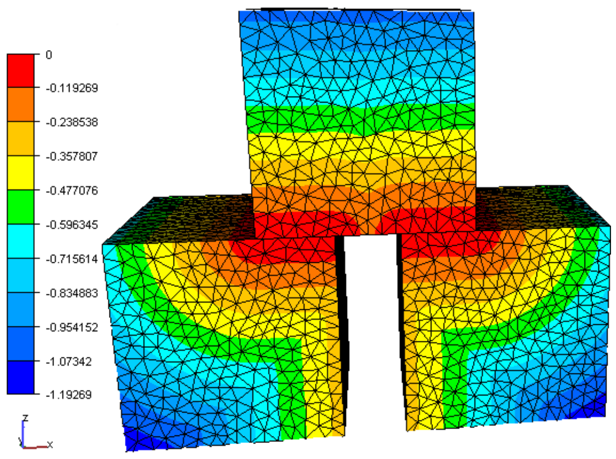
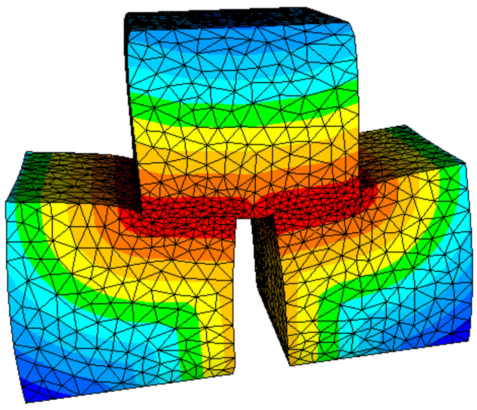


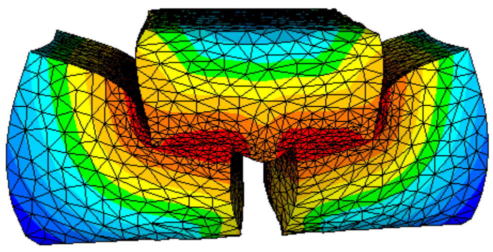
Figure 6: Simulation of the forging process of a crankshaft, carried out by using parallel computing on 11 processors: evolution of the shape of the workpiece all along the process, and isovalues of the distance function to the tool surface.



(a) t=0

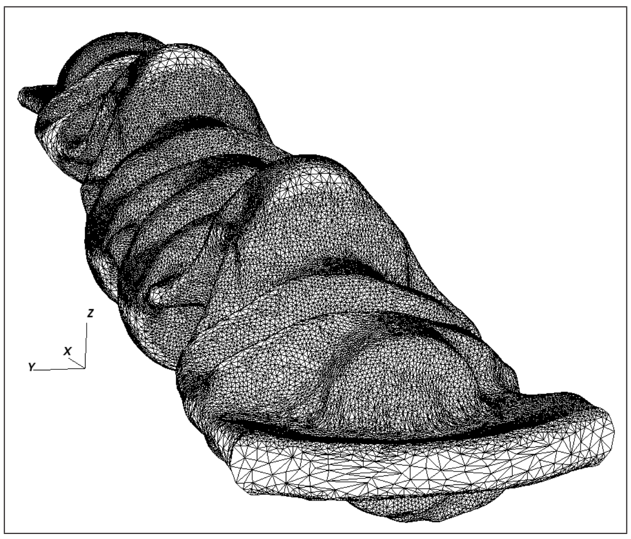


(b) t=50

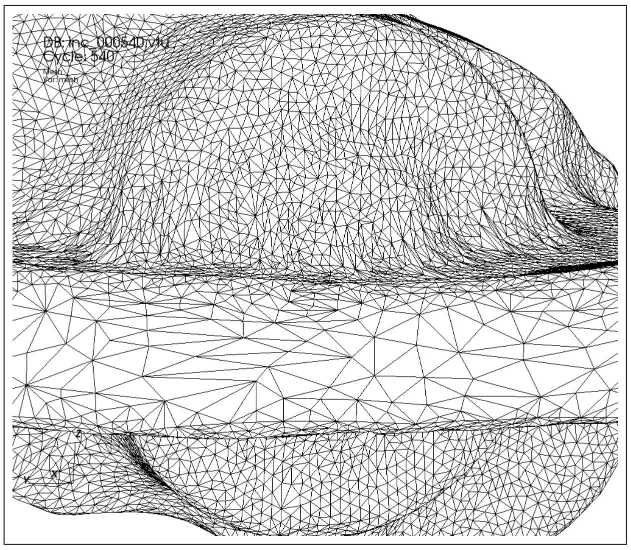


(c) t=172

Figure 7: Simulation of the contact between three deformable bodies: isovalues of the signed distance function and mesh (which is refined in the vicinity of the contact zone.)

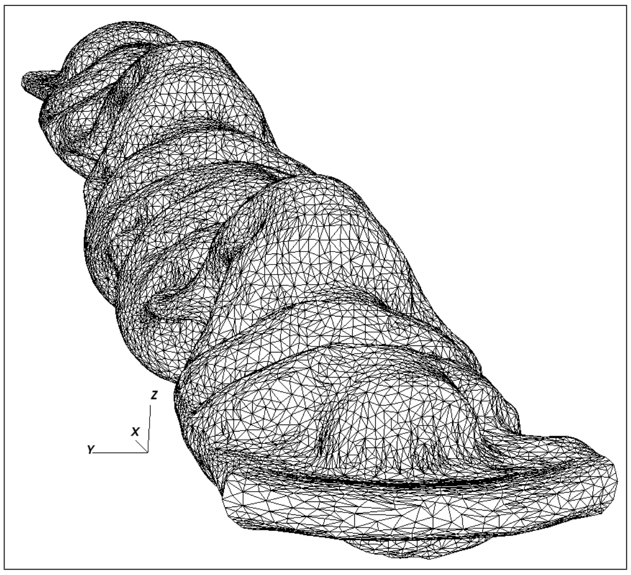


(a) Whole mesh

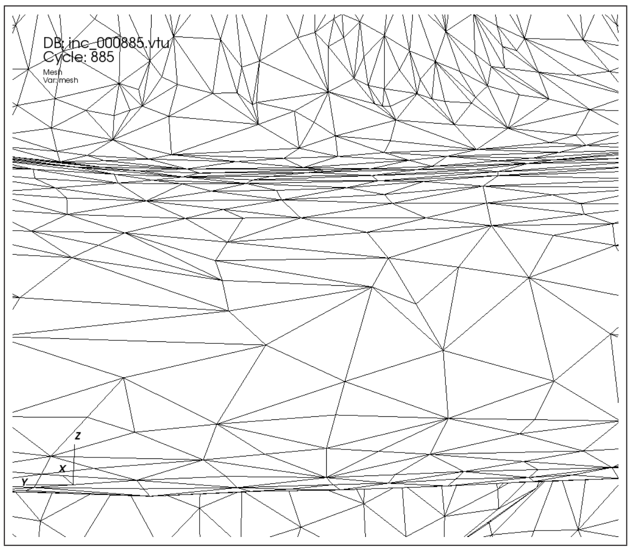


(b) Zoom

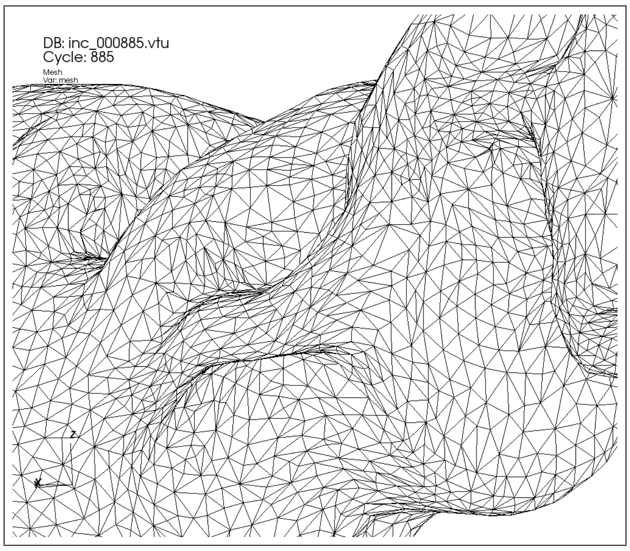
Figure 8: Isotropic mesh of 324 222 nodes and 1 597 592 elements, generated according to the distance function to the tool surface, by using the following parameters in mesh size expression (23): $d_{min} = 4.5$, $d_{max} = 14$, $s_{min} = 3$, $s_{max} = 12$.



(a) Whole mesh

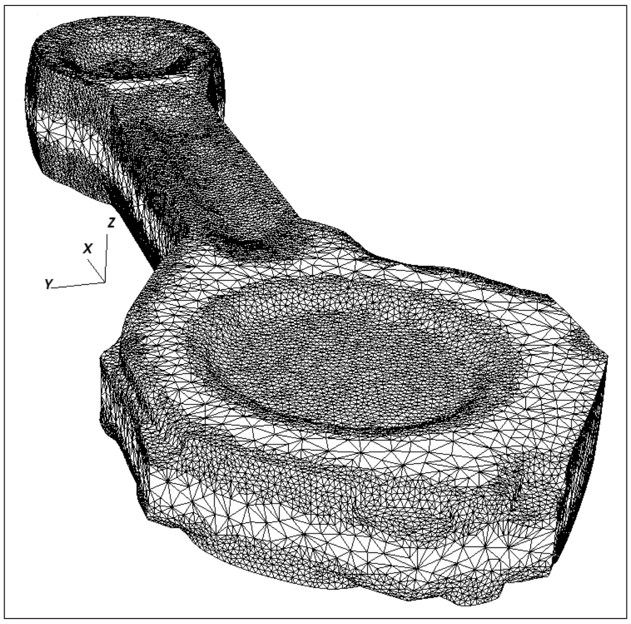


(b) Zoom 1

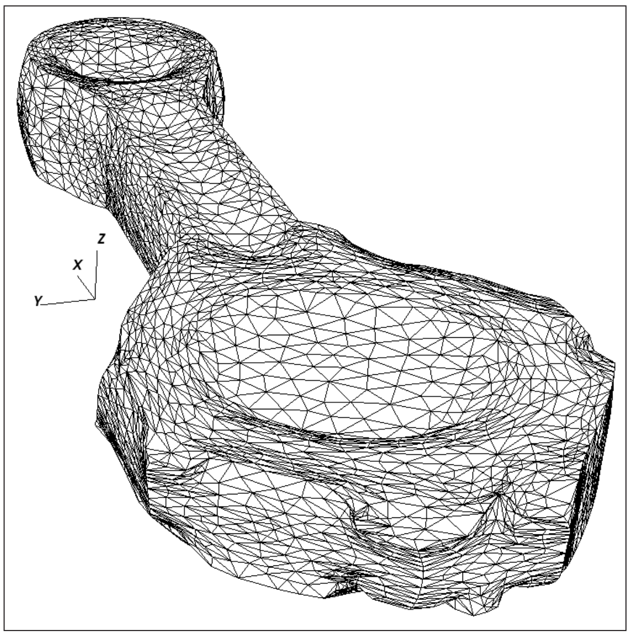


(c) Zoom 2

Figure 9: Anisotropic mesh of 91 347 nodes and 468 946 elements, generated according to the tool surface geometry, by using the following parameters in metric map expression (24): $d_{min} = 4.5$, $d_{max} = 14$, $s_{max} = 12$, size in the $\nabla \boldsymbol{n}$ direction = 2, size in the orthogonal direction = 10.



(a) Isotropic mesh (92 836 nodes, 474382 elements)



(b) Anisotropic mesh (12335 nodes, 59327 elements)

Figure 10: Comparison between isotropic and anisotropic meshes obtained during the simulation of the forging process of a connecting rod by using expressions (23) and (24), respectively. The parameters used to calculate the isotropic mesh size are $d_{min} = 3$, $d_{max} = 11$, $s_{min} = 1$, $s_{max} = 8$, while those used for the metric map are $d_{min} = 3$, $d_{max} = 11$, $s_{max} = 8$, size in the $\nabla \boldsymbol{n}$ direction = 1, size in the orthogonal direction = 5.

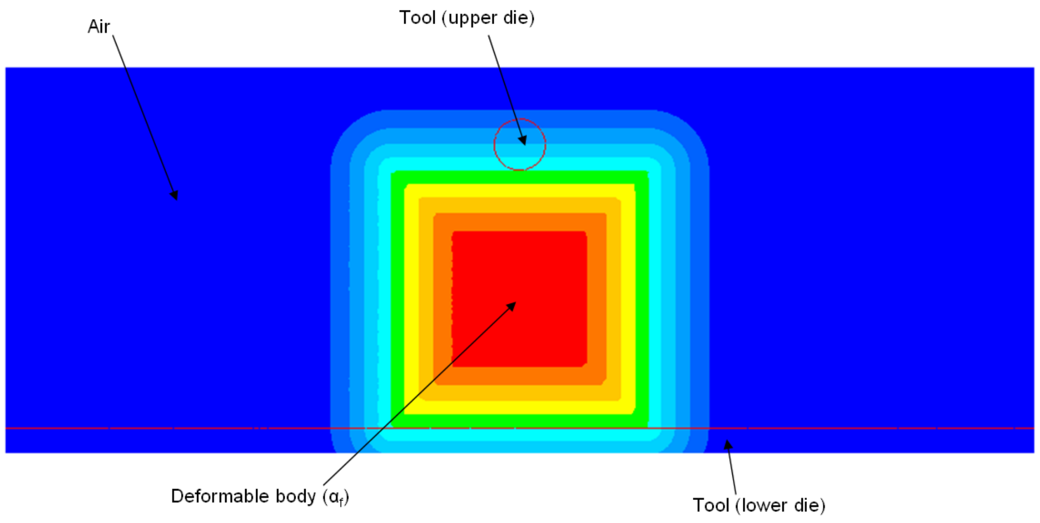
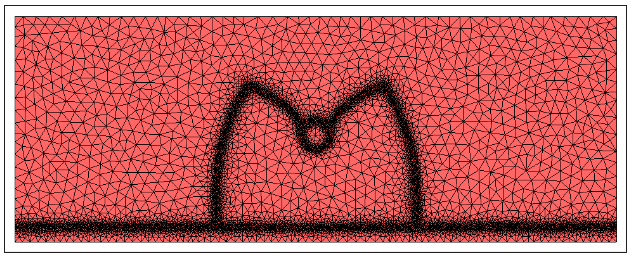
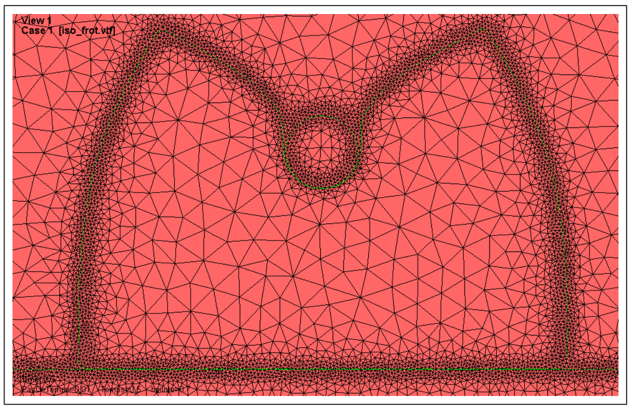


Figure 11: Simulation of the deformation of a square within an Eulerian context: initial configuration (level set 0 of α_s and isovalues of α_s).

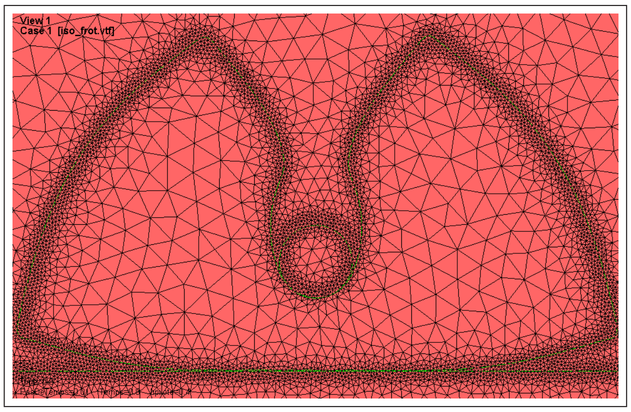


(a) Whole mesh of the computational domain

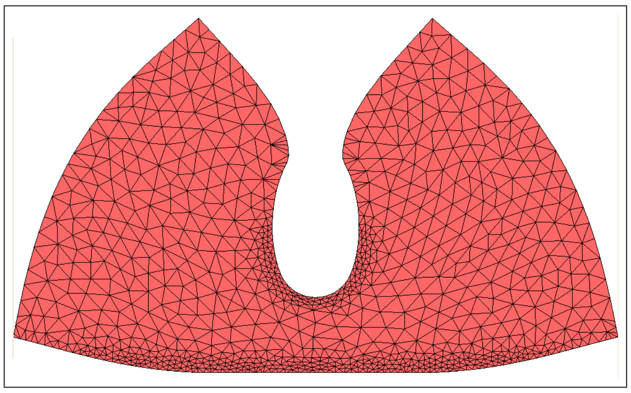


(b) Zoom on the deformable body

Figure 12: Eulerian simulation of the deformation of a square pressed out by a rigid disk: mesh of the computational domain at t=0.5



(a) Eulerian approach (mesh and zero level set of α_f and α_s)



(b) Lagrangian approach

Figure 13: Comparison between the Eulerian and Lagrangian simulations of the deformation of a square pressed out by a rigid disk: mesh of the computational domain at t=0.8

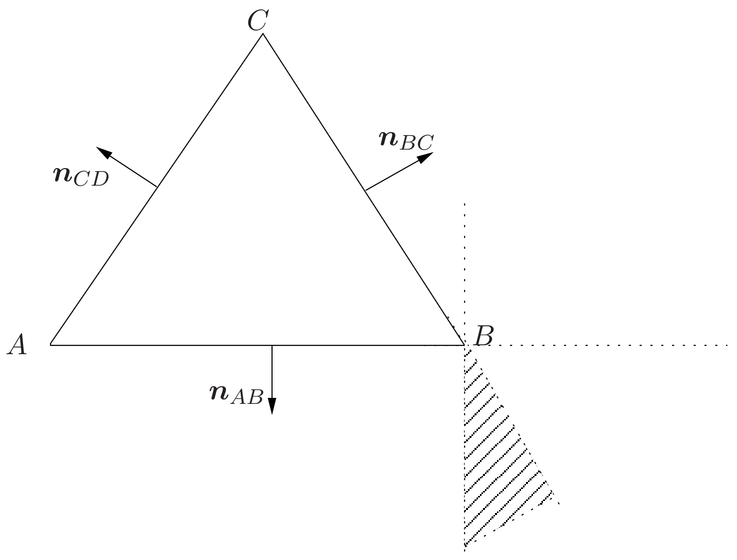


Figure 14: Computation of the signed distance to the set of segments $\{[A,B],[B,C],[C,A]\}$ (a surface mesh in a two-dimensional case).

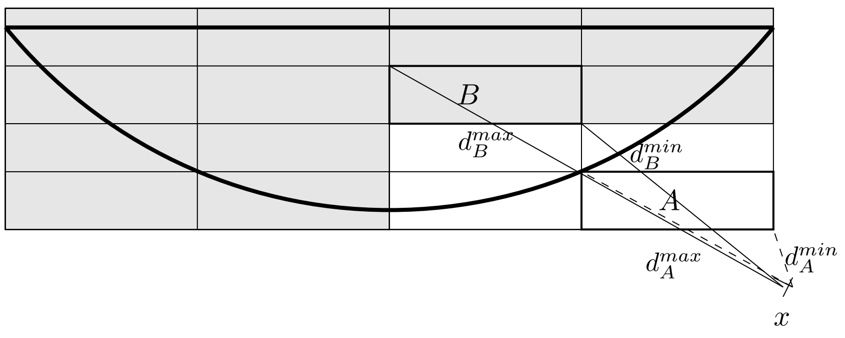
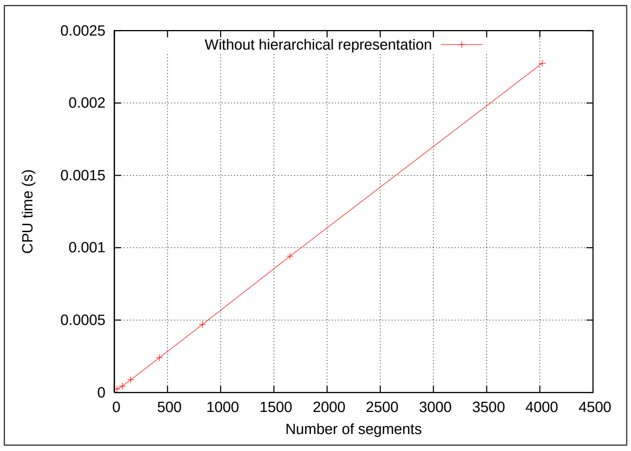
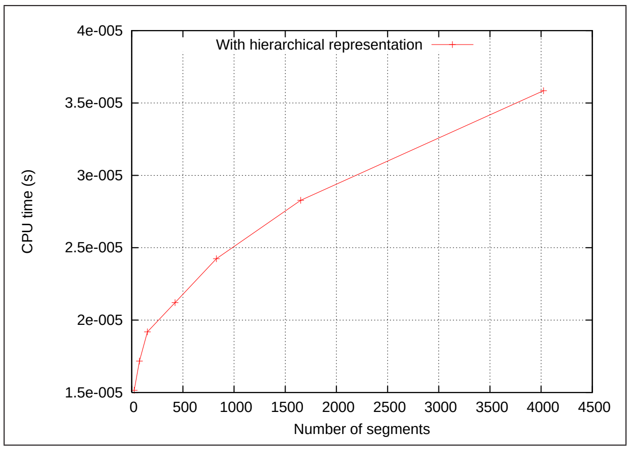


Figure 15: Hierarchical representation of level 2 in 2D (16 subcells) of a tool (arc of a circle in bold).

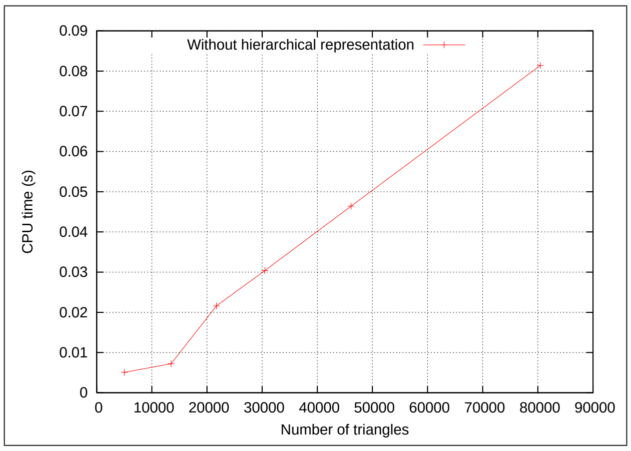


(a) Linear mesh representation

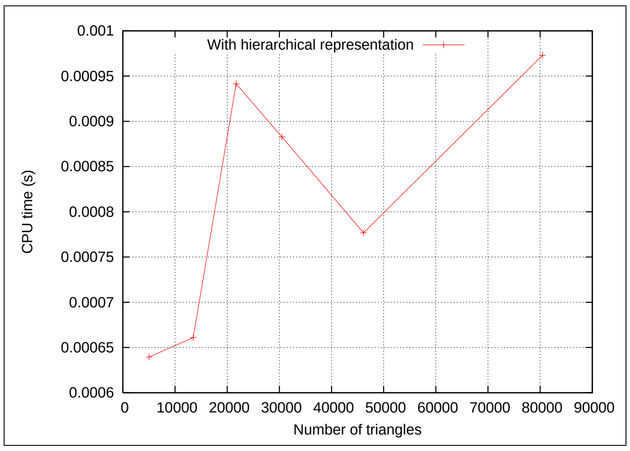


(b) Hierarchical mesh representation (a leaf contains no more than 4 elements)

Figure 16: Duration (in seconds) of the computation of the signed distance from a point to a surface mesh (set of segments) - 2D case: without 16(a) and with 16(b) the hierarchical representation of the surface mesh.



(a) Linear mesh representation



(b) Hierarchical mesh representation (a leaf contains no more than 128 elements).

Figure 17: Duration (in seconds) of the computation of the signed distance from a point to a surface mesh (set of triangles) - 3D case: without 17(a) and with 17(b) the hierarchical representation of the surface mesh.