TECHNISCHE UNIVERSITÄT CHEMNITZ

Sonderforschungsbereich 393

Parallele Numerische Simulation für Physik und Kontinuumsmechanik

Arnd Meyer Roman Unger

Projection methods for contact problems in elasticity

Preprint SFB393/04-04

Abstract

The aim of the paper is showing, how projection methods can be used for computing contact-problems in elasticity for different classes of obstacles.

Starting with the projection idea for handling hanging nodes in finite element discretizations the extension of the method for handling penetrated nodes in contact problems will be described for some obstacle classes.

Preprintreihe des Chemnitzer SFB 393

ISSN 1619-7178 (Print) ISSN 1619-7186 (Internet)

SFB393/04-04 April 2004

Author's addresses:

Arnd Meyer
TU Chemnitz
Fakultät für Mathematik
D-09107 Chemnitz
Germany

email: arnd.meyer@mathematik.tu-chemnitz.de

`http://www.tu-chemnitz.de/~amey`

Roman Unger
TU Chemnitz
Fakultät für Mathematik
D-09107 Chemnitz
Germany

email: roman.unger@mathematik.tu-chemnitz.de

`http://www.tu-chemnitz.de/~uro`

Contents

1 Introduction

The numerical simulation of frictionless contact problems between an elastic body and a rigid obstacle is almost done by solving the resulting variational inequalities for instance with penalty methods.
For some ideas of this methods see [KO88] or [Wri02]
In this paper we introduce a projection-method for solving the following problem in the $2D$-case.

Problem 1. *Find a valid dispacement field $u(x)$ for an elastic body Ω, such that the Lamé-Equation with the Lamé-constants λ, μ and the stress tensor σ*

$$\begin{aligned} -\mu\mathbf{\Delta}u &- (\lambda+\mu)\textbf{\textit{grad div}}\ u = f \\ u(x) &= g_D \quad on \quad \Gamma_D \\ \sigma(u)\cdot\mathbf{n} &= g_N \quad on \quad \Gamma_N \end{aligned}$$

is fulfilled, and all nodes of Ω stay outside a given rigid obstacle. (see figure 1)

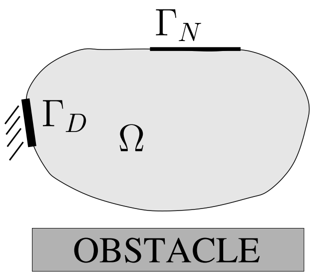


Figure 1: The contact-problem

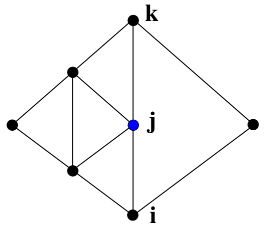


Figure 2: A hanging node example

For implementational details of an adaptive algorithm for solving the Lame'e problem without contact see the A2d Programmer's Manual [Mey01].

2 Projection methods for hanging nodes

The basic idea for using projectors comes from the problem of handling hanging nodes in adaptive finite element methods.
Because the used preconditioner needs hierarchical meshs, you fall in trouble by using a normal red-green refinement with temporally created green triangles which will removed before the next refinement step will performed. So the idea of using a projector, forcing conform values at the hanging node comes up.
As a simple example see figure 2. For guarateeing conformality the value u_j at the (mid-)node j is not free but restricted to

$$u_j = \frac{1}{2}(u_i + u_k) \tag{1}$$

.
From [Mey99] this restriction to a subspace of $\mathbb{R}^N$ is simply implemented by a projector like

$$P = \left[\begin{array}{ccccccc|c} & & & I & & & & \\ \cdots & \frac{1}{2} & \cdots & \frac{1}{2} & \cdots & & 0 & \\ & & & & & & & I \end{array} \right]$$

with $\frac{1}{2}$ at the columns i and k and 0 in all other columns of row j, working within the preconditioning step of the pcgm.
If you plug in such a projector in the preconditioner of the pcgm, i.e. changing the computing of the correction term w in the cg with the residuum r and preconditioner C from

$$w := C^{-1} r$$

to

$$w := P C^{-1} P^t r$$

and start with a conform initial guess $u^0 \in Im(P)$ all iterated u^k and so the solution u will stay in $Im(P)$.
For a detailed description of this method see [Mey99], in the following sections the extension of such projectors for handling some kinds of boundary conditions and the contact-problem will be described.

3 Projectors for handling some kinds of boundary conditions

Some kinds of boundary conditions can be handled by using projectors as well.
For instance periodic boundary conditions like $u_{left} = u_{right}$ on the left and right boundary parts in figure 3

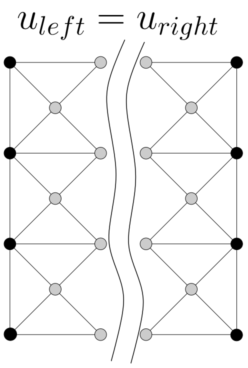


Figure 3: Periodic b.c.

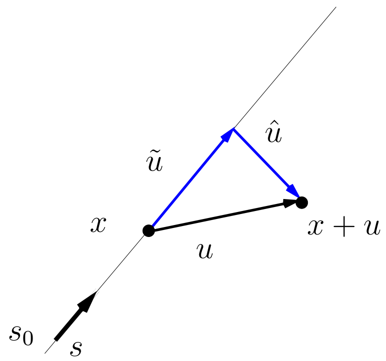


Figure 4: Slip- b.c.

Another kind of b.c. are the so called 'slip-boundary-conditions', where $u(x)$ is not totally fixed, but forced to stay in a 1-dimensional affine subspace. (See figure 4)
This kind of b.c. will later be used for the contact-handling, but at first some properties of the resulting projector should be revealed.
The slip-boundary is well defined by fixing the point s_0 and the direction s and assume $||s|| = 1$.
The displacement u can be decomposited in two orthogonal components $\tilde{u}$ along the boundary and $\hat{u}$ orthogonal to the boundary.
Now use $\tilde{u}$ as the valid displacement.

$$
\begin{aligned}
u &= \tilde{u} \oplus \hat{u} \\
\tilde{u} &= \frac{\langle u, s\rangle}{\langle s, s\rangle} s \\
&= \langle u, s\rangle\, s \quad \text{for} \quad ||s|| = 1 \\
&= ss^T u
\end{aligned}
\qquad \Rightarrow \qquad P = \begin{bmatrix} I & & \\ & ss^T & \\ & & I \end{bmatrix}
$$

So for all nodes x_i on the slip-boundary the Projector is a block-diagonal matrix with 2×2 blocks ss^T where s is the slip direction of node x_i.
In the next section this idea will extend to a method, handling infinite planar obstacles.

4 Planar obstacles

4.1 Infinite planar obstacles

In the same way as for slip-boundary-conditions let the half-space-obstacle be defined by a point s_0 and direction s with $||s|| = 1$.

Definition 1 (Half-space-obstacle). *Let $s_0 \in \mathbb{R}^2$ be a fixed point in the plane and $s \in \mathbb{R}^2$ a given direction with $||s|| = 1$.*
The straight line, given by s_0 and s divides the $\mathbb{R}^2$ in two half-spaces. We enforce the body to stay completly in one of this half-spaces and call the other one half-space-obstacle.
If a node of the body violates this condition we say the node penetrates.

The contact- handling consists of two parts, the penetration test and correction of the displacement with selective switching of possible projectors.
Let x be one node at the boundary of Ω that can possible come into contact with the obstacle, let n be the inner normal of the obstacle and $a := (x + u) - s_0$.

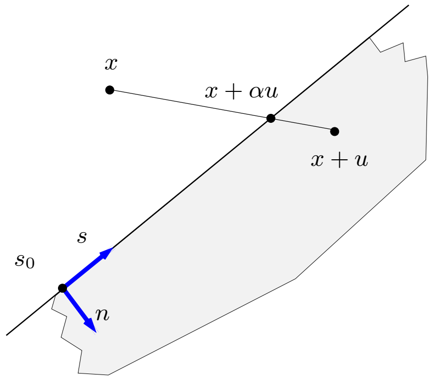


Figure 5: Infinite Planar obstacles

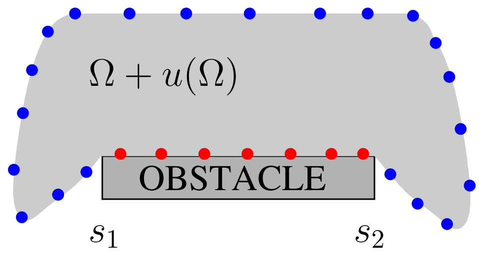


Figure 6: Finite planar obstacles

Then, the penetration test is done by computing $\langle n, a\rangle$:

$$(x + u) \quad \text{penetrates} \quad \Leftrightarrow \quad \langle n, a\rangle > 0 \tag{2}$$

For an effective implementation of this test and for computing the correction the test is better done in the following (equivalent) form.
Let $\alpha_1 := \langle (s_0 - x), n\rangle$ and $\alpha_2 := \langle (u, n\rangle$, then $\langle n, a\rangle > 0 \leftrightarrow \alpha_1 < \alpha_2$
Whenever (2) is fulfilled, the actual displacement u is not admissible. So we correct $u(x)$ to $\alpha u(x)$ such that $x + \alpha u$ lies on the obstacle boundary.
A way to do this is forcing

$$\langle (x + \alpha u) - s_0, n\rangle = 0 \tag{3}$$

which implies

$$\begin{aligned} 0 &= \langle x, n \rangle + \alpha \langle u, n \rangle - \langle s_0, n \rangle \\ 0 &= \langle x - s_0, n \rangle + \alpha \langle u, n \rangle \\ \alpha &= \frac{\langle s_0 - x, n \rangle}{\langle u, n \rangle} \\ \alpha &= \frac{\alpha_1}{\alpha_2} \end{aligned}$$

This two steps of the whole algorithm enforce the fulfilling of the non-penetration condition but can lead to clamping of some nodes in the end of the (a priori unknown) contact zone.
To avoid this, a second test after solving the linear system with active projectors must be performed.
An easy way to do this, is to compute the contact pressure along the actual contact zone and to free all nodes (i.e. switch off the projectors) with a negative contact pressure.
Note that the pcgm-solver starts with a starting vector, that is admissible and uses the onedimensional projectors for contact nodes. Hence the solution stays to be admissible.

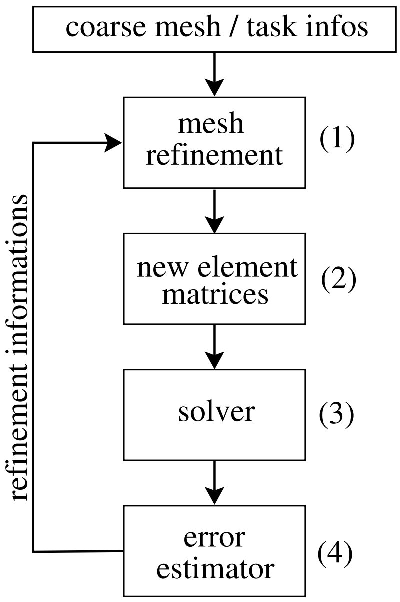


Figure 7: The normal solver cycle

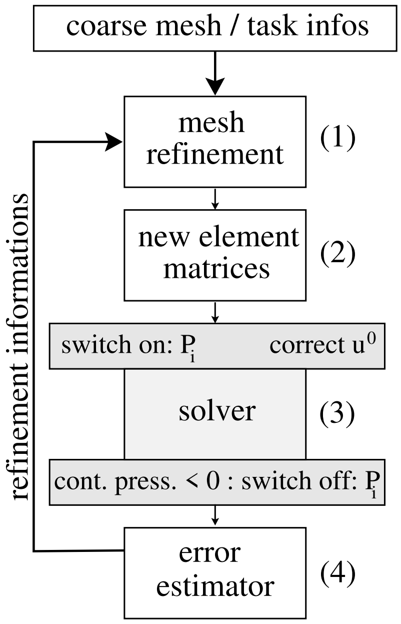


Figure 8: The whole solver cycle

The whole cycle of the algorithm with penetration test, projector switching and contact pressure checking is shown in figure 8, where in figure 7 a normal solver-cycle for an adaptive finite-element algorithm is shown.

4.2 Finite planar obstacles

For the handling of finite planar obstacles (see figure 6) the algorithm is changed a little bit in the penetration test. Only for nodes between the obstacle boundaries s_1 and s_2 the test returns the result 'penetration', all other parts of the algorithm stay unchanged.
For some pictures of computed examples see the appendix A.1.

5 Obstacle description by implicit functions

To enrich the class of obstacles with easy performable penetration tests implicit functions for the description of the obstacle are useful.
Let $F : \mathbb{R}^2 \to \mathbb{R}$ a given function of $x \in \Omega$ and define the obstacle in the following sense:

$$F(x) \begin{cases} = 0 & \text{on obstacle boundary} \\ < 0 & \text{inside the obstacle} \\ > 0 & \text{outside the obstacle} \end{cases}$$

So the penetration test for $x + u$ easily can be done by calculation of $F(x + u)$.
If the test is positive, a proper parameter α, such that $x + \alpha u$ is on the obstacle boundary is to compute by solving the following onedimensional nonlinear equation

$$\sigma := F(x + \alpha u) = 0 \tag{4}$$

for instance with a bisection-method.
Good starting values for α are $\alpha = 0$ and $\alpha = 1$, because this values enforce different signs of the value of σ.
With this parameter α a one dimensional affine subspace for the the corresponding node can defined by forcing the node staying on the tangent in point $x + \alpha u$ at the obstacle boundary.
For the implementation this will handled in the same way like a half space obstacle for this node.
To compute this tangential direction on the obstacle boundary see F as a 'normal' function from $\mathbb{R}^2$ to the $\mathbb{R}^1$.
Then the gradient in the point $x + \alpha u$ points to inner normal direction of the obstacle boundary and the needed tangential direction s is an orthogonal vector to the gradient:

$$n = \nabla F(x + \alpha u) \qquad\qquad s = \begin{bmatrix} -n_2 \\ n_1 \end{bmatrix}$$

So all values for the projector are computed and can used in the solver cycle. For some examples see appendix A.2.

6 Obstacle description by spline curves

6.1 Motivation for spline curves

Implicit functions are easy to handle, but not very realistic for practical purposes. A better choice is, the use of spline curves.

Definition 2 (Splinecurve). *Let* $\{(x_1^1, x_2^1), (x_1^2, x_2^2) \dots (x_1^n, x_2^n)\}$ *be a set of given controlpoints. Further some boundary conditions in the points* (x_1^1, x_2^1) *and* (x_1^n, x_2^n) *are fixed. With this values and an arbitrary parameter* t *two cubic splines* $S_1(t)$ *and* $S_2(t)$ *are fixed.*
So the mapping

$$\Gamma : [0, 1] \to \mathbb{R}^2 \quad \textit{with} \quad \Gamma(t) := \begin{bmatrix} S_1(t) \\ S_2(t) \end{bmatrix} \tag{5}$$

defines a splinecurve in the plane.

So you can define a tool-contour by some control-points and conditions at the endpoints. As a first example we approximate the unit circle.

6.2 Example for a splinecurve

For the approximation of the unit circle we fix 5 control points

$$\{(-1, 0), (0, 1), (1, 0), (0, -1), (-1, 0)\}$$

and a parameter $t \in [0, 1]$ with $t = i \cdot \frac{1}{5}$ corresponds to x^i.
Further we set the derivatives in the startpoint and endpoint, which yields to

$$\begin{bmatrix} \dot{S}_1(0) \\ \dot{S}_2(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 2\pi \end{bmatrix} = \begin{bmatrix} \dot{S}_1(1) \\ \dot{S}_2(1) \end{bmatrix} \tag{6}$$

Remark: This values of the derivatives come from the wellknown parametrization

$$\begin{aligned} x_1(t) &= cos(2\pi t) \\ x_2(t) &= sin(2\pi t) \end{aligned}$$

with

$$\begin{aligned} \dot{x}_1(t) &= -2\pi sin(2\pi t) \\ \dot{x}_2(t) &= 2\pi cos(2\pi t) \end{aligned}$$

So the values of first derivatives in the boundarypoints must set to

$$\begin{aligned} \dot{x}_1(0) &= 0 \\ \dot{x}_2(0) &= 2\pi \\ \dot{x}_1(1) &= 0 \\ \dot{x}_2(1) &= 2\pi \end{aligned}$$

For plots of the approximation and the pointwise error see figures 9 and 10.

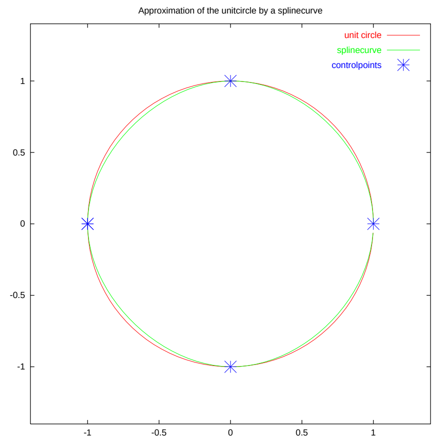


Figure 9: Approximation of the unitcircle

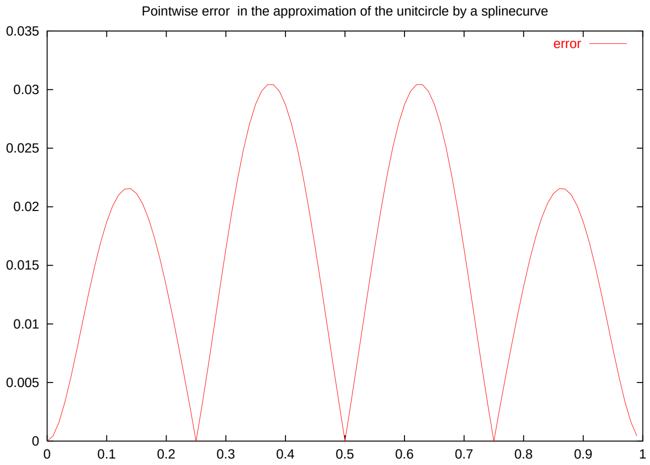


Figure 10: Error of the approximation

6.3 Spline curves as obstacles

For the usage of this curves as obstacles a penetration check, like in the other variants of the obstacle descriptions is to do.
Unfortunatly this test can not be performed in such an easy manner as in the case of implicit functions, but if the test is passed, the projection point is computed too.

6.3.1 The penetration test

To perform the test, define a (nonlinear) mapping $\mathbb{R}^2 \longrightarrow \mathbb{R}^2$ by

$$\sigma(\alpha, t) := \begin{bmatrix} (x_1 + \alpha u_1) - S_1(t) \\ (x_2 + \alpha u_2) - S_2(t) \end{bmatrix} \tag{7}$$

with node-coordinates $x = [x_1, x_2]^T$, displacement $u = [u_1, u_2]^T$ and spline-curve $S(t) = [S_1(t), S_2(t)]^T$.
So the intersection point of the obstacle with the displacement-direction of the node x is computeable as a zero-problem of σ.

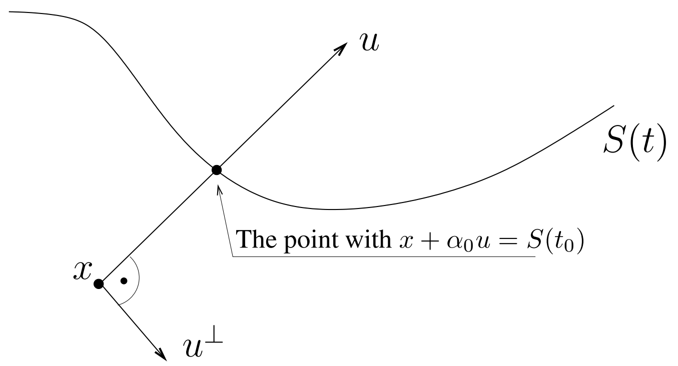


Figure 11: Penetration test and computation of the projection point

With the solution $[\alpha_0, t_0]$ where $\sigma(\alpha_0, t_0) = 0$ the decision if the node penetrates can be done in the following way.

- For $t_0 < 0$ or $t_0 > 1$ there is no intersection between the line $x + \alpha u$ and the node does not penetrate. In this case is nothing to do, the value of α_0 must not be considered.
- For $\alpha_0 > 1$ the node is outside the obstacle and it is not neccesary to correct the node.
- For $\alpha_0 < 1$ the node penetrates the obstacle, the displacement u has to be corrected to $u := x + \alpha_0 u$ and the projector will switched on. For the normal and tangential direction at the projection point to define the projector the (already computed) derivatives of the splinecurve are useable. ($s = \left[\dot{S}_1(t_0), \dot{S}_2(t_0)\right]^T$)

6.3.2 Changing the zero-problem for $\sigma(\alpha, t)$ to a zero-problem of $\sigma(t)$

It is possible to solve this (nonlinear) problem for the two unknowns t and α by an preiteration with some steps of a gradient-method and then start a Newton-iteration.
But with some assumptions to the projection point it can transformed into an (nonlinear) $1D$-Problem.

Starting with equation (7) the α can be isolated by multiplying the equation with the (nonzero) displacement u

$$\begin{aligned}
\alpha u &= S(t) - x && (8)\\
\alpha\langle u, u\rangle &= \langle S(t) - x, u\rangle && (9)\\
\alpha &= \frac{\langle S(t) - x, u\rangle}{\langle u, u\rangle} && (10)\\
& && (11)
\end{aligned}$$

By insertion of this α in (7) $\sigma(\alpha, t)$ becomes to $\sigma(t)$

$$\sigma(t) := x + \frac{\langle S(t) - x, u\rangle}{\langle u, u\rangle} \cdot u - S(t) = 0 \tag{12}$$

which is equivalent to

$$\sigma(t) = x + \underbrace{\frac{\langle S(t), u\rangle}{\langle u, u\rangle} \cdot u}_{P_u S(t)} - \underbrace{\frac{\langle x, u\rangle}{\langle u, u\rangle} \cdot u}_{P_u x} - S(t) = 0 \tag{13}$$

where $P_u S(t)$ and $P_u x$ are orthogonal projections of $S(t)$ and x to the displacement u.
In other words σ can be written as

$$\begin{aligned}
\sigma(t) &= (I - P_u)x - (I - P_u)S(t) = 0 && (14)\\
\Leftrightarrow &\ (I - P_u)(x - S(t)) = 0 && (15)
\end{aligned}$$

With the orthogonal splitting of $\mathbb{R}^2$ in u and $u^\perp$ and multiplication of (15) with $u^\perp$ this leads to

$$\begin{aligned}
\langle (I - P_u)(x - S(t)), u^\perp\rangle &= 0 && (16)\\
\langle (x - S(t)), u^\perp\rangle - \underbrace{\langle P_u(x - S(t)), u^\perp\rangle}_{=0 \text{ bec. } \langle P_u z, u^\perp\rangle = 0 \quad \forall z} &= 0 && (17)\\
\langle x - S(t), u^\perp\rangle &= 0 && (18)
\end{aligned}$$

which is a scalar equation for t.
After computing t_0 with $\langle x - S(t_0), u^\perp\rangle = 0$ the corresponding α_0 follows from (10).

6.3.3 Solving the zero-problem for $\sigma(t)$

For having good starting values for a Newton-iteration to solve (18) it is useful to start with a C^0 - interpolation of the splinecurve (i.e. piecewise linear approximation of the curve, see figure 12) and test for every segment from the controlpoint $\left[x_1^i, x_2^i\right]^T$ to $\left[x_1^{i+1}, x_2^{i+1}\right]^T$ ($i = 1 \cdots n-1$) if there is a intersection point of the lines $x + \alpha u$ and the segment.

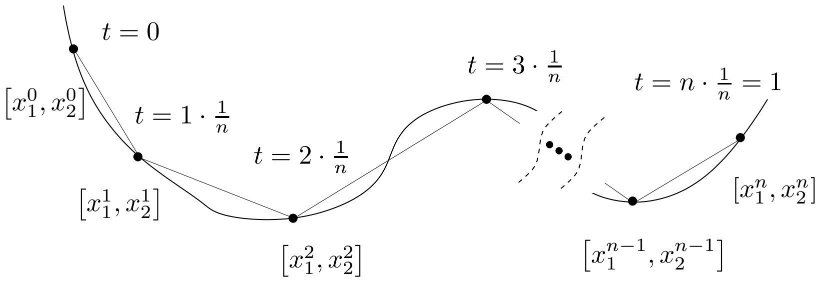


Figure 12: C^0 Approximation of the splinecurve

For every segment i this can be done by enforcing

$$\begin{bmatrix} x_1^i \\ x_2^i \end{bmatrix} + \lambda \begin{bmatrix} x_1^{i+1} - x_1^i \\ x_2^{i+1} - x_2^i \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \alpha \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{19}$$

which is a linear system for α and λ

$$\begin{bmatrix} u_1 & -(x_1^{i+1} - x_1^i) \\ u_2 & -(x_2^{i+1} - x_2^i) \end{bmatrix} \begin{bmatrix} \alpha \\ \lambda \end{bmatrix} = \begin{bmatrix} x_1^i - x_1 \\ x_2^i - x_2 \end{bmatrix} \tag{20}$$

such that

$$\begin{aligned} \delta_m &:= u_2(x_1^{i+1} - x_1^i) - u_1(x_2^{i+1} - x_2^i) && (21) \\ \alpha &= \delta_m^{-1} \left(-(x_2^{i+1} - x_2^i)(x_1^i - x_1) + (x_1^{i+1} - x_1^i)(x_2^i - x_2) \right) && (22) \\ \lambda &= \delta_m^{-1} \left(-u_2(x_1^i - x_1) + u_1(x_2^i - x_2) \right) && (23) \end{aligned}$$

For $\lambda \in [0, 1]$ and $\alpha \leq 1$ a penetration is possible. Then a Newton-iteration for solving (18) will start with the starting value

$$t = \frac{i + \lambda}{n} \tag{24}$$

because the solution λ from the local view to a single interval must be mapped back to the whole C^0- spline.

After convergence of this iteration the parameters t and α are computed for the real C^2-spline and the penetration test is done.

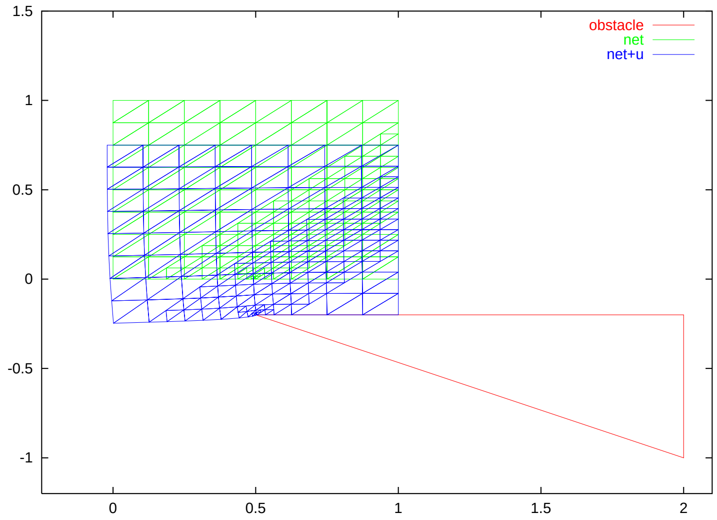
The node penetrates the obstacle if $\alpha \leq 1$. In this case the point P and the tangential direction s on the obstacle in this point P is to compute by an evaluation of $[S_1(t), S_2(t)]^T$ and $\left[\dot{S}_1(t), \dot{S}_2(t)\right]^T$.

With this values a selective projector which handles a slip-boundary-condition for the node $[x, y]^T$ will switched on and the same procedure is to do for all other contact-suspect nodes.

A Computed examples

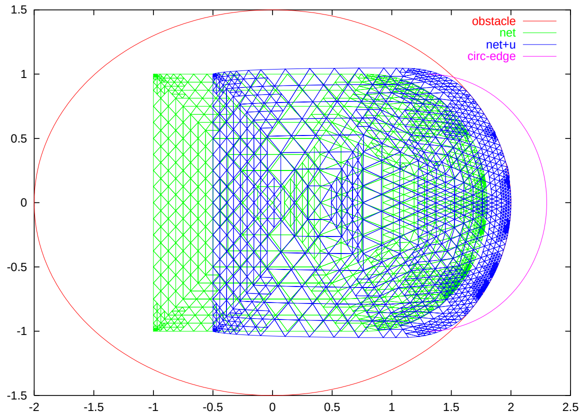
A.1 Planar obstacles

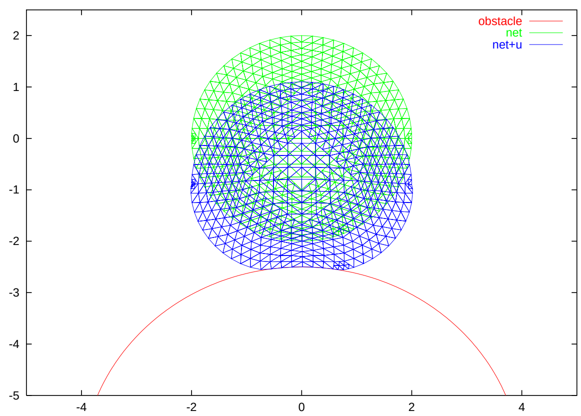
As an example for the computation of problem with a finite planar obstacle a unitsquare is moved by a given displacement against the obstacle.



A.2 Obstacles, described by implicit functions

In this an example for implicit functions as obstacle boundary the 2 cases of convex and concave obstacles are shown





In the example with the body inside the ellipsoidal obstacle the right boundary of the body is a circle edge, painted too for showing how deep the body would penetrate without processing the obstacle.

A.3 Obstacles, described by spline-curves

In the last part you see some pictures of computed examples with obstacles , described by spline curves.
Like in the other cases the body is moved by a fixed displacement on the upper boundary against the obstacle.

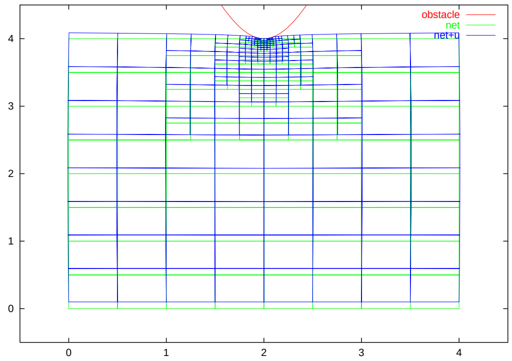


Figure 13: Obstacle description with a spline curve

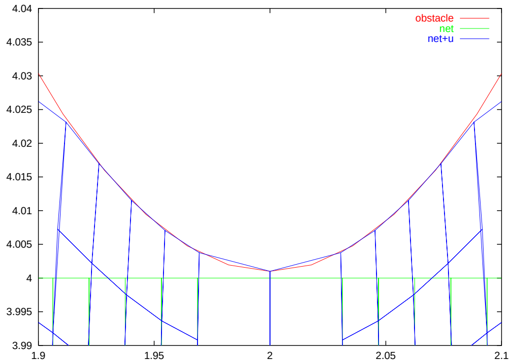


Figure 14: Zoom

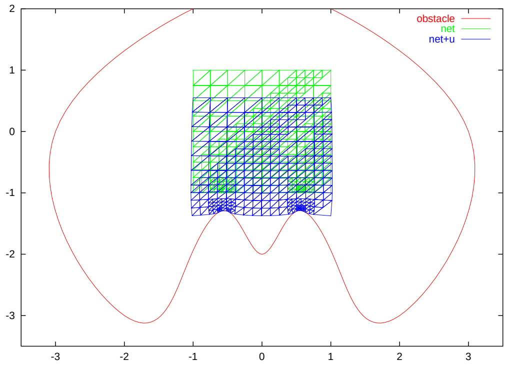


Figure 15: Obstacle description with a spline curve

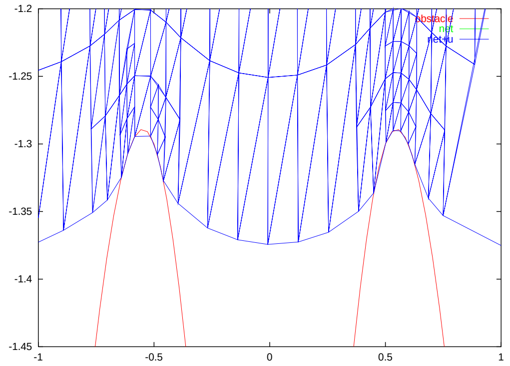


Figure 16: Zoom

References

[KO88] N. Kikuchi and J. T. Oden. *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods*. siam Philadelphia, 1988.

[Mey99] A. Meyer. *Projected PCGM for Handling Hanging Nodes in Adaptive Finite Element Procedures*. Preprint SFB393 99-25 TU Chemnitz, 1999.

[Mey01] A. Meyer. *Programmer's Manual for Adaptive Finite Element Code SPC-PM 2Ad*. Preprint SFB393 01-18 TU Chemnitz, 2001.

[Wri02] P. Wriggers. *Computational Contact Mechanics*. John Wiley & Sons West Sussex, 2002.