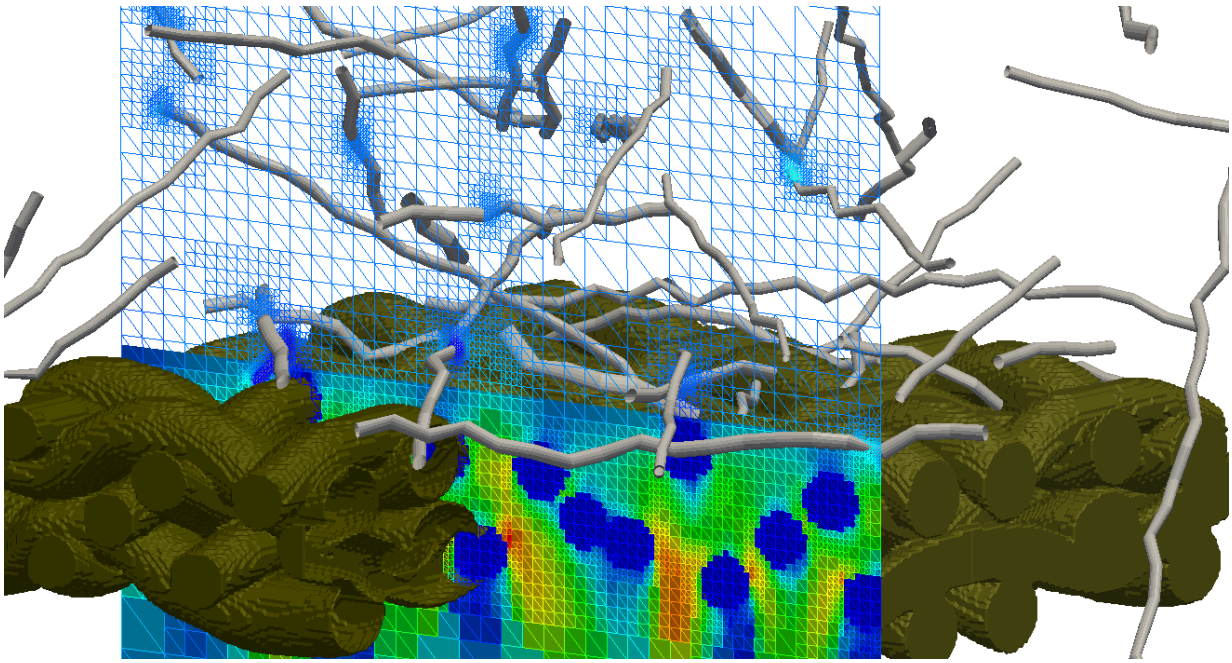
CHALMERS



Development of a nonlinear Finite Element beam model for dynamic contact problems applied to paper forming

Master's Thesis in Solid and Fluid Mechanics

ERIK SVENNING

Fraunhofer Chalmers Centre
Department of Computational Engineering and Design
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2011
Master's Thesis 2011:05

MASTER'S THESIS 2011:05

Development of a nonlinear Finite Element beam model for dynamic contact problems applied to paper forming

Master's Thesis in Solid and Fluid Mechanics

ERIK SVENNING

Fraunhofer Chalmers Centre
Department of Computational Engineering and Design
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2011

Development of a nonlinear Finite Element beam model for dynamic contact problems applied to paper forming

Master's Thesis 2011:05
ISSN 1652-8557
Fraunhofer Chalmers Centre
Department of Computational Engineering and Design
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: + 46 (0)31-772 1000

Cover:
Paper fibers falling down onto a forming fabric. The wireframe shows the grid refinements around the fibers and the forming fabric. The slice is colored by the fluid velocity. Forming fabric geometry courtesy of Albany International.

Chalmers Reproservice
Göteborg, Sweden 2011

Finite element computations of the dynamic impact and contact of interacting bodies are notoriously difficult.

F. Cirak and M. West, 2005

Development of a nonlinear Finite Element beam model for dynamic contact problems applied to paper forming
Master's Thesis in Solid and Fluid Mechanics
ERIK SVENNING
Fraunhofer Chalmers Centre
Department of Computational Engineering and Design
Chalmers University of Technology

Abstract

A Finite Element model for simulation of paper forming has been developed and validated. Paper forming is the first step in the paper machine where a fiber suspension leaves the headbox and flows through a forming fabric. The fibers land on the fabric and start to form the fiber web. Understanding this process is important for the development of better paper products, because the orientation and distribution of the individual fibers during this step have a large influence on the final quality. Simulation of paper forming offers great challenges since it involves structures with large displacements and large rotations, flow with complex boundaries, fluid-structure interaction with strong coupling and dynamic collisions.

The fiber model, which is based on a dynamic co-rotational formulation of the Euler-Bernoulli beam equation, accounts for geometric nonlinearities under the assumption of small strains. Two contact models have been implemented, a penalty method and the impulse based method Decomposition Contact Response. These models can handle fiber-fiber collisions as well as collisions between fibers and the forming fabric, which may have arbitrary geometry. Friction is included in the models and elastic/inelastic collisions are accounted for with the coefficient of restitution. The fiber model was implemented in C++ and the nonlinear system of equations was solved with Newton's method. The flow around the fibers was simulated with the CFD software IBOFlow developed at FCC. IBOFlow is based on a finite volume discretization on a Cartesian octree grid that can be dynamically refined and coarsened. The flow around the moving fibers is resolved and the Hybrid Immersed Boundary Method is used to model the presence of fibers in the flow.

Extensive validation of the implementation has been performed against several demanding test cases from the literature. These cases include static instability with postbuckling, large amplitude oscillation of slender structures and dynamic impacts. Large effort was dedicated to making the code robust and efficient.

The code was used to study two fluid-structure interaction problems. First, a single fiber oscillating in a cross flow was studied and the numerical results were compared to an analytical solution obtained from a Fourier series expansion of the Euler-Bernoulli beam equation. Paper forming with two forming fabrics of different geometry was also studied. A qualitative comparison of the resulting distribution and orientation of paper fibers was made.

Keywords: paper forming, fluid-structure interaction, FEM, contact modeling, Immersed Boundary Methods

Contents

Abstract I

Contents III

Preface V

Notations VI

1 Introduction 1
- 1.1 Purpose . . . 1
- 1.2 Limitations . . . 2
- 1.3 Approach . . . 2
- 1.4 Review of beam models . . . 2
- 1.5 Review of contact models . . . 4
  - 1.5.1 Penalty methods . . . 4
  - 1.5.2 Lagrange multipliers . . . 6
  - 1.5.3 Impulse based methods . . . 7

2 Theory 8
- 2.1 Fluid model . . . 8
  - 2.1.1 Fluid-structure coupling . . . 9
  - 2.1.2 Simplified alternative for fluid-structure coupling . . . 11
  - 2.1.3 Treatment of complex boundaries . . . 11
  - 2.1.4 CFD solver used in the present work . . . 11
- 2.2 Fiber model . . . 12
  - 2.2.1 Geometry of a fiber . . . 12
  - 2.2.2 Small rotation FE formulation . . . 15
  - 2.2.3 Mathematics of finite rotations . . . 17
  - 2.2.4 Computation of element deformations . . . 18
  - 2.2.5 Large rotation Finite Element formulation . . . 19
  - 2.2.6 Dynamic FE formulation . . . 21
- 2.3 Contact detection . . . 22
  - 2.3.1 Spatial search - identifying segments that are close to each other . . . 23
  - 2.3.2 Distance between a segment and a fixed point . . . 23
  - 2.3.3 Distance between two segments . . . 26
- 2.4 Contact models . . . 28
  - 2.4.1 A penalty method suitable for paper forming . . . 28
  - 2.4.2 Decomposition Contact Response . . . 29
    - 2.4.2.1 DCR for fiber-fiber contact . . . 31
    - 2.4.2.2 DCR for fiber-fabric contact . . . 32
- 2.5 Measures of fiber web properties . . . 32
  - 2.5.1 Mass distribution (Grammage) . . . 32
  - 2.5.2 Fiber directions (anisotropy) . . . 33

3 Numerical results 34
- 3.1 Validation of FE solver . . . 34
  - 3.1.1 Validation of static problems . . . 34
    - 3.1.1.1 Hockling of a cable . . . 34
    - 3.1.1.2 Lateral buckling of a hinged frame . . . 37

3.1.2 Validation of dynamic problems . . . 40
3.1.2.1 Free vibration of right-angle cantilever beam . . . 40
3.1.2.2 Vibration of a cantilever beam with twisted cross section . 44
3.1.3 Validation of problems involving contact . . . 49
3.1.3.1 Collision of two rods . . . 49
3.1.3.2 Impact between spinning rod and table . . . 51
3.2 Fluid-Structure Interaction without contact . . . 54
3.2.1 Oscillating cantilever beam in cross flow . . . 54
3.3 Simulations of paper forming . . . 60
3.3.1 Problem description . . . 60
3.3.2 Laydown simulation . . . 63
3.3.3 Qualitative comparison of two forming fabrics . . . 67

4 Summary and Conclusions **72**

5 Further work **73**

A Formulas for three-dimensional rotations **76**

B Formulas for the static co-rotational formulation **76**
B.1 $\underline{\underline{\Gamma}}$ . . . 76
B.2 $\underline{\underline{P}}$ . . . 78
B.3 $\underline{\underline{F}}$ . . . 78

C Fourier series expansion of an oscillating cantilever beam in a fluid **78**

D Jacobian of inertia force **81**

E Shortest unsigned distance between two segments **84**

Preface

This thesis is submitted in partial fulfillment of the requirements for the MSc degree from Chalmers University of Technology. The work was carried out during the period from January 2011 to June 2011 at Fraunhofer Chalmers Centre (FCC). Professor Kenneth Runesson at the Department of Applied Mechanics, Division of Material & Computational Mechanics was examiner and Björn Andersson at FCC was supervisor.

Acknowledgements

This work was part of the ISOP project with industrial partners Albany International, Eka Chemicals, Stora Enso and Tetra Pak. It was supported in part by the Swedish Foundation for Strategic Research (SSF) through the Gothenburg Mathematical Modeling Centre (GMMC).

I wish to thank the colleagues at FCC for the great atmosphere. I especially want to thank my supervisor Björn Andersson for encouraging support, fruitful discussions and great help with programming. I would also like to thank my examiner Professor Kenneth Runesson. A special thanks to Dr. Andreas Mark for sharing his expertise in CFD, for teaching me high performance computing and for always keeping his door open. I would like to thank Assoc. Professor Fredrik Edelvik for his faith in me and for giving me the possibility to work with this interesting project. Thanks to Dr. Robert Bohlin for help with quaternions and to Assoc. Professor Mats Jirstrand for help with Mathematica. My fellow student Anton Berce is greatfully acknowledged for sharing his knowledge about paper machines and for the good discussions about work and life.

Finally, I would like to thank Annie for her love and support.

Göteborg June 2011
Erik Svenning

Nomenclature

δu	Infinitesimal change in u
δ_{Dirac}	Dirac delta function
δ_{ij}	Kronecker delta
λ	Lagrangian multiplier
μ_{fr}	Coefficient of friction
ν	Poisson's ratio
$\overline{\boldsymbol{u}}$	An overbar indicates the deformational part of a quantity.
ρ	Density
$\boldsymbol{\omega}$	Spin axis
$\boldsymbol{\theta}$	Angle
$\boldsymbol{A}$	Angular acceleration
$\boldsymbol{e}_x$, $\boldsymbol{e}_y$, $\boldsymbol{e}_z$	Base vectors in the current configuration.
$\boldsymbol{f}$	Force
$\boldsymbol{n}$	Unit normal
$\boldsymbol{p}_e$	Subscript e indicates a variable associated with the end point of an element.
$\boldsymbol{p}_s$	Subscript s indicates a variable associated with the start point of an element.
$\boldsymbol{p}_{loc}$	Subscript loc indicates a variable expressed in local coordinates.
$\boldsymbol{q}$	Vector that rotates with the major axis of the cross section.
$\boldsymbol{t}$	Unit tangent
$\boldsymbol{u}^{el}$	The superscript el indicates that the variable is associated with an element.
$\boldsymbol{v}_{rel}$	Relative velocity
$\boldsymbol{w}$	Angular velocity
$\boldsymbol{\Omega}$	Spin tensor
$\mathbf{E}$	Change of coordinates matrix. The base vectors in the current configuration are the columns of $\mathbf{E}$.
$\mathbf{I}_\rho$	Inertia tensor
$\mathbf{T}$	Rotation tensor
$\mathbf{T}^g$	Subscript or superscript g denotes a variable expressed in global coordinates.
$\underline{\underline{\Gamma}}$	Matrix that relates the rotation of the element frame to the displacements and rotations at the element nodes.

$\underline{\underline{B}}$ Matrix containing the derivatives of the base functions.

$\underline{\underline{F}}$ Matrix used in the computation of the geometric stiffness. $\underline{\underline{F}}$ is a function of the internal force vector only.

$\underline{\underline{H}}$ Matrix that transforms the tangent stiffness and internal force vector from angles to spin variables.

$\underline{\underline{K}}$ Tangent stiffness matrix

$\underline{\underline{M}}$ Mass matrix

$\underline{\underline{P}}$ Projector matrix

$\underline{\Psi}$ Rotational pseudovector

$\underline{p}$ Momentum

φ Twisting angle of undeformed cross section

c_d Drag coefficient

E Young's modulus

e_{cor} Coefficient of restitution

f_n Normal force

g Gap function

i Index: $i = 1, 2, 3$; Einstein's summation convention is used where summation is implied over repeated indices unless otherwise stated.

L Length of undeformed segment.

l Length of deformed segment.

M Bending moment

R_a Radius of major axis of the elliptical cross section

R_b Radius of minor axis of the elliptical cross section

$S\,()$ Spin operator

V Potential energy

1 Introduction

Paper forming is the first step in the paper machine where a fiber suspension leaves the headbox and flows through a forming fabric. The fibers land on the fabric and start to form the fiber web. Understanding this process is important for the development of better paper products, because the orientation and distribution of the individual fibers during this step have a large influence on the final paper quality.

Simulation of this process offers huge challenges since it involves transient fluid flow with many immersed solid objects subjected to large displacements. The problem involves fluid structure interaction with strong coupling: the immersed objects are forced to follow the fluid, but the fluid is also strongly influenced by the immersed objects. The effect of strong coupling can be described by throwing a ball in air and water. The trajectory of a ball thrown in air can be described relatively accurate even without considering the surrounding fluid, the coupling is quite weak and the weak coupling makes the problem easier. The trajectory of a ball thrown in water is much more difficult to predict, the water will have a very strong influence on the motion of the ball. The difficulty of the problem is influenced by how heavy the ball is compared to the fluid. The ball is heavy compared to the air, which makes the case of a ball in air easy. When compared to water, the weight of the ball is relatively low. As a result, the motion of the ball is influenced more by the fluid forces than by its own inertia. This effect makes the formulation of the fluid-structure coupling important and the problem becomes difficult. The problem becomes very difficult when the object has the same density as the fluid. When this occurs, the object is said to be buoyant. Paper fibers in water are buoyant.

Strong coupling is not the only challenge in simulations of paper forming, contact phenomena play an important role when the fibers fall down onto the forming fabric. Contact forces with friction keep the fibers from falling off the forming fabric and beeing carried away by the flow. The number of contacts increases as more and more fibers lay down on top of the fiber web. Since the fiber web is kept together by contact forces, very robust modeling of contacts is required.

Understanding the phenomena governing paper forming requires DNS simulations where the coupling between the fluid and the solid objects is handled properly and the flow around every fiber is resolved. Simulations of fibers described in the literature use drag correlations instead of resolving the flow around the individual fibers and model the fibers as chains of spheres or rigid rods instead of employing rigorous beam models that are normally used to study slender structures. Therefore, an implementation of more accurate methods is necessary.

1.1 Purpose

The goal of the project is to develop a model that can be used to study initial paper forming. To achieve this, a nonlinear Finite Element beam model with collisions is required. Therefore, the goal of this thesis consists of two parts:

- Implement and validate a Finite Element code for simulation of paper fibers. The model must be rigorous and derived from fundamental laws of continuum mechanics. The code must be fast and robust. In order to ensure that the implementation is correct and physically sound, the code should be validated against demanding structural dynamics problems described in the literature.
- Show that the code can be used to simulate paper forming. Especially, it must be shown that the code can handle a large number of interacting fibers without stability problems.

1.2 Limitations

The project is mainly restricted in the following directions:

- Linear elastic material is assumed, other constitutive models are not considered.
- Initial simulations of paper forming are performed in order to demonstrate the capabilities of the code. However, there are not enough computational resources available to perform a full set of simulations of paper forming.
- Two contact models are implemented and compared. It is not the purpose of this project to implement all possible classes of contact algorithms.
- It is not the purpose of this project to investigate the effects of turbulence on the process.

1.3 Approach

FCC's in-house CFD code IBOFlow [16] (Immersed Boundary Octree Flow solver) is used to simulate the fluid flow around the paper fibers. Geometrical descriptions of forming fabrics and paper fibers are generated with GeoDict [10]. A Finite Element fiber model based on a co-rotational formulation of the Euler-Bernoulli beam equation is implemented. Two contact models are also implemented and included in the Finite Element model and demanding test problems from the literature are chosen for validation of the fiber code. The FE code is coupled with the CFD code, resulting in a simulation software capable of handling fluid-structure interaction with strong coupling and dynamic impacts between the solid objects. The Immersed Boundary Method is used to resolve the flow around the fibers, while the forming fabrics are described numerically with voxelizations.

A simple drag correlation with one-way coupling was also implemented. The results obtained from these simplified simulations should be interpreted carefully, but this option gives the possibility to show that the FE code can handle a large number of fibers without stability problems. It is also shown that relevant postprocessing data can be extracted from the simulations and a qualitative comparison between two forming fabrics is made.

The code is written in C++ and generic high performance libraries are used for distance searches, matrix computations and solution of the large and sparse system of equations. The open source program Paraview is used for 3D visualizations and Matlab is used to draw 2D plots.

1.4 Review of beam models

Beams are slender objects characterized by the fact that one dimension is much larger than the other two dimensions. Many researches have derived equations governing the dynamic motion of such objects. Two widely used examples are the Euler-Bernoulli beam theory and the Timoshenko beam theory [12]. In the classical, small displacement formulation, the Euler-Bernoulli beam equation includes the effect of translational inertia, but it neglects the effect of rotational inertia. Bending is included in the strain energy, but shearing is not. The Timoshenko beam equation includes rotational inertia as well as shearing in the cross section. It does, however, assume small rotations and small strains. Both the Euler-Bernoulli and the Timoshenko beam theories are good approximations for slender beams oscillating at low frequencies, but the Timoshenko beam theory is a better approximation for non-slender beams and high frequencies. In simulations of paper forming, the paper fibers will be transported by the fluid flow field and may be subjected to large displacements

as well as large (finite) rotations. The Euler-Bernoulli and Timoshenko beam theories as presented in [12] do not allow finite rotations and therefore these models can not be used without modification. Han et al. [12] also discuss the Rayleigh beam theory and the Shear beam theory. These models suffer from the same fundamental weakness, they do not allow finite rotations.

Ibrahimbegović and Mikdad [17] derived a beam model based on the Reissner beam theory. The model is geometrically exact and it is capable of handling finite strains as well as finite rotations. Different ways of parametrizing large rotations were discussed.

Simo and Vu-Quoc [29] derived a fully nonlinear, geometrically exact beam model. The model is capable of handling finite strains as well as finite rotations. Several numerical examples were given and the computational aspects of the implementation were discussed.

Nour-Omid and Rankin [25] derived a co-rotational formulation which can be used to extend a linear Finite Element model so that finite rotations are allowed. The fundamental idea of the co-rotational approach is that the linear equations are formulated in a co-rotational frame which moves with the element. In this way, finite rotations can be allowed even if the original Finite Element model assumes small rotations. The consistent linearization of the co-rotational formulation results in a projector matrix, which is used to modify the linear element model. The model proposed in [25] is derived for static problems.

Crisfield, Galvanetto and Jelenić [5] studied the dynamics of co-rotational beams subjected to finite rotations. The weak form of the inertia terms was derived and different time stepping schemes were discussed.

Several authors interested in the motion of fibers in a fluid flow have modeled fibers as a chain of spheres or rods. One example is the model proposed by Lindström and Uesaka [18, 19, 20], where fibers are treated as a chain of rigid rods connected with springs. The derivation of the model proposed in [19] starts with Newton's second law for linear and angular momentum. The fibers are considered to be inextensible, but the inextensibility constraint is neither enforced with Lagrangian multipliers nor with an axial stiffness in the setting of a penalty method. Instead, a constraint on the velocity in the joints between the segments is proposed. However, no discussion is given on how the resulting constraint force is computed or how it enters the equation of motion.

The equations in the models proposed by Crisfield et al. [5], Ibrahimbegović and Mikdad [17], Nour-Omid and Rankin [25] and Simo and Vu-Quoc [29] are solved with Newton's method. Therefore second order convergence of the iterations is obtained with these models, so that an accurate solution is obtained after just a few iterations. Furthermore, Newton's method is very robust, so that these models will converge even in cases where strong nonlinearities occur. It is unclear if Lindström and Uesaka [19] use Newton's method.

A beam model suitable for simulation of paper fibers must allow finite rotations. The classical theories described in [12] do not fulfill this criterion without modification. It is desirable to use a fiber model based on a beam theory. Therefore, three possible choices remain: the finite strain Reissner beam model proposed by Ibrahimbegović and Mikdad [17], the finite strain beam model proposed by Simo and Vu-Quoc [29] and the finite rotation co-rotational formulation proposed by Nour-Omid and Rankin [25] with inertia terms computed according to Crisfield et al. [5]. Paper fibers falling down onto a forming fabric will be subjected to large rotations, but the strain will be small or moderate. Therefore, the extra complexity in [17] and [29] needed to allow finite strains is not necessary for the purpose of the present work.

The models in [17], [25] and [29] are all physically sound models that would be suitable for simulations of paper forming. The co-rotational approach described by Nour-Omid and Rankin [25] with inertia accounted for as described by Crisfield et al. [5] is the simplest

model that fulfills the requirements for simulation of paper fibers. Therefore, this model is chosen for the implementation in the present work.

1.5 Review of contact models

The contact models described in the literature can roughly be divided into three classes of algorithms [13]: penalty methods, methods based on Lagrangian multipliers and impulse-based methods. All three classes of contact models suffer from different kinds of numerical problems. There is no perfect contact model that works well for all types of problems, the choice of contact model will depend on the properties of the problem studied and the desired level of detail of the simulation. This section gives a short overview of the different types of contact models available.

In the following, the overlap is characterized by the gap function g, which measures the penetration in the normal direction of the contacting surfaces.

1.5.1 Penalty methods

Penalty methods allow the elements to overlap and add a repulsive force which increases with increasing overlap. Explicitly adding a contact force which depends on the magnitude of the overlap is the fundamental idea of this algorithm and this idea is used in all different formulations of penalty methods. The difference between different penalty methods is how this repulsive force is computed. Wriggers [32] added a quadratic term to the energy potential, resulting in a penalty force that varies linearly with the overlap. This approach is energy conserving, all collisions are considered to be elastic. Crowe, Sommerfeld and Tsuji [6] use a penalty force based on Hertzian contact theory for spheres of equal size, where the normal force is given by:

$$f_n = -f_0 \left(-g\left(\boldsymbol{x}\right)\right)^{1.5} - \eta \boldsymbol{v}_{rel} \cdot \boldsymbol{n} \tag{1.1}$$

Here, f_0 is the normal stiffness and the Hertzian theory is extended with dissipation characterized by a damping coefficent η. This approach allows modeling of collisions that are not completely elastic, but the relation between η and the coefficient of restitution e_{cor} is not trivial. A relation between η and e_{cor} for a sphere is given in [6]. However, it that relation η depends on the mass of the sphere as well as the overlap. Since η depends on the configuration of the system, it can no longer be interpreted as a constant material parameter.

Harmon [13] proposed several extensions to penalty methods in his PhD thesis. One such modification is the introduction of penalty layers. The gap function g is usually defined in such a way that $g = 0$ when the distance between the impacting elements is zero and $g < 0$ when the elements are overlapping. It is possible to shift the gap function so that $g = 0$ when the distance between the elements is small but not zero. The gap function then becomes negative when the distance is exactly zero. The idea of a shifted gap function is visualized in figure (1.1). In this way, a penalty force can be applied when the elements are close rather than actually overlapping. Harmon [13] suggests using this approach for cases where the geometrical effects of overlap are difficult to repair. Such a case occurs when the geometry of the problem is complicated. In this case, it is fairly easy to determine the distance from a point to a surface, but it could be difficult to determine if the point is outside the geometry or if it is inside the geometry so that overlap has occured. If this is a problem, a penalty layer can be applied to ensure that overlap never occurs in the simulation.

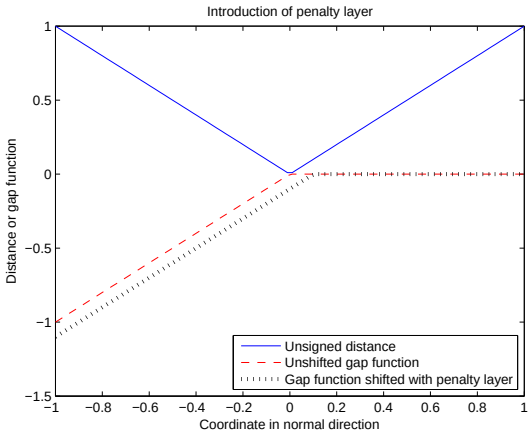


Figure 1.1: *Introduction of penalty layer. If no penalty layer is used, the gap function is zero when the elements are separated and it becomes negative when the elements overlap. The gap function can be shifted by introducing a penalty layer. The gap function then becomes negative already when the elements are close, but have not yet started to overlap.*

Harmon [13] also proposed that inelastic collisions could be accounted for by introducing the coefficient of restitution e_{cor} directly in the energy potential:

$$V(g) = \frac{1}{2} c(v_n) \cdot f_0 \cdot [g(\underline{x})]^2$$
$$c(v_n) = \begin{cases} 1 & \text{if } v_n \leq 0 \\ e_{cor} & \text{otherwise} \end{cases} \tag{1.2}$$

Here v_n is the relative velocity in the direction of the contact normal, g is the gap function and $\underline{x}$ is the vector of variables in the configuration space. Equation (1.2) states that an inelastic or partly inelastic collision can be modeled by applying a lower force during decompression than during compression: the force is reduced by a factor e_{cor} during the decompression phase compared to the compression phase. Therefore, the equation gives a direct correlation between the coefficient of restitution and the repulsive force. A possible drawback of this approach is that the force will be discontinuous in time at the turning point where v_n changes sign.

Many different formulas for the penalty force have been proposed in the literature. A linear force and a force based on Hertzian contact theory have already been mentioned. de la Fuente and Felippa [7] used a bell shaped penalty function while several authors use a penalty function that increases exponentially with the overlap. Regardless of which penalty function is chosen, the contact stiffness f_0 will always have to be chosen and this is associated with two fundamental problems which are characteristic for penalty methods [13]:

- Choosing f_0 too high will make the resulting system of equations stiff and require very small timesteps.
- Choosing f_0 too low will allow the elements to pass through each other without stopping the collision.

Many applications of practical interest involve friction. Coulomb friction is often assumed and the main problem is then to separate the cases of sliding and sticking. A possibility to avoid treating the cases of sliding and sticking separately is to use a regularization of Coulomb's law as discussed by Wriggers [32]. According to Coulombs law, the friction force is a discontinuous function of the relative velocity. To use a discontinuous function is

impractical in the setting of penalty methods. Therefore, the aim of the regularization is to approximate the discontinuous Coulomb law with a differentiable function that is as similar to the Coulomb law as possible. This is achieved by constructing a function that depends on a parameter ε in such a way that $\varepsilon \to 0$ reduces to Coulomb's law exactly while a finite but small ε gives a smooth approximation to Coulomb's law. This idea is illustrated in figure (1.2). The regularization can be constructed in different ways. Wriggers [32] suggests formulations based on a square root function, a function including the hyperbolic tangent or a piecewise polynomial function.

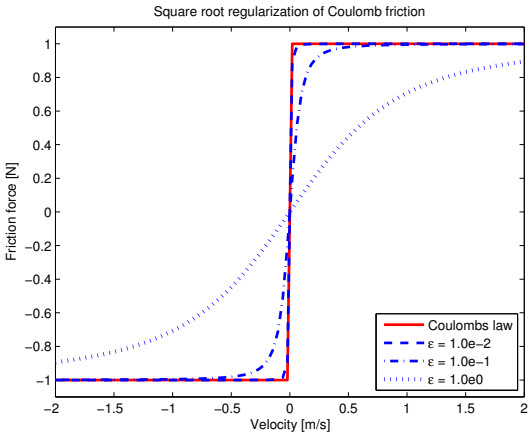


Figure 1.2: *Square root regularization of Coulomb friction. A small value of ε gives higher accuracy but may require shorter time steps.*

1.5.2 Lagrange multipliers

A contact problem can be considered as a set of constraints that have to be satisfied. There are two types of constraints [13]:

- Bilateral (two-sided) constraints: a function is constrained to have an exact value
- Unilateral (one-sided) constraints: a function can take values in an allowed configuration space, but motion out of this space is prohibited

An example of a bilateral constraint is that the length of an inextensible bar is constant. A ball bouncing on the floor is an example of a unilateral constraint: the ball may move up from the floor, but it may not move downwards and penetrate into the floor. Contact constraints, which are unilateral constraints, can be resolved with the method of Lagrange multipliers.

A contact constraint can be imposed on the configuration by adding a term to the potential energy [32]:

$$V_c^{LM} = \int_{\Gamma_c} (\lambda_n g_n + \boldsymbol{\lambda}_t \cdot \boldsymbol{g}_t)\, dA \tag{1.3}$$

Here λ_n and g_n are the Lagrange multiplier and gap function corresponding to the normal direction. $\boldsymbol{\lambda}_t$ and $\boldsymbol{g}_t$ are the Lagrange multiplier and gap function corresponding to the tangential direction. λ_n corresponds to the contact pressure in the normal direction. If no adhesion occurs between the contacting surfaces, the contact pressure p_n must be negative or zero. If the contacting surfaces are not allowed to penetrate into each other, the gap function g_n must not be negative. Furthermore, if the gap function is postivite, then the

contact pressure must be zero and if the contact pressure is not zero, then the gap function must be zero. Therefore, the contact pressure and normal gap function are subjected to the Kuhn-Tucker-Karush conditions [32]:

$$g_n \geq 0 \quad p_n \leq 0 \quad g_n p_n = 0 \quad \text{on} \quad \Gamma_c \tag{1.4}$$

In the equation above, Γ_c is the part of the boundary subjected to contact.

Coulomb's law can be used in the tangential direction and slip-stick behavior can be enforced exactly without regularization. In order to do this, the cases of slip and stick must be treated separately.

Many variants of the Lagrange multiplier method exist. The constraint equations can be discretized either implicitly, enforcing the constraints at the end of the time step, or explicitly, enforcing the constraints at the start of the time step [13]. Furthermore, perturbed Lagrange formulations exist, where a Lagrange multiplier method is mixed with a penalty method [32].

The Lagrange multiplier method can be used for dynamic problems with perfectly elastic collisions [3]. For such cases, the constraint equations can be formulated by requiring that the overlap is zero at the end of the time step.

In order to model inelastic collisions with Lagrange multipliers, it would not suffice to add a constraint on the displacement. In principle, a constraint could instead be added on the velocity to ensure that the pre- and postcollisional velocities are related by the coefficient of restitution. However, if a constraint on the velocity is imposed, the impenetrability constraint on the displacement would have to be sacrificed.

1.5.3 Impulse based methods

A collision between two solid objects results in a high contact force during a short time interval. Instead of resolving the high force and the short time interval, the contact forces can be treated as instantaneous forces, i.e. impulses [13]. Impulse based methods use this approach to predict a change in momentum due to the collision instead of predicting a change in acceleration.

One example is the Hard sphere model discussed in [6]. In that model, instantaneous contact between two spheres is considered. Elastic and inelastic collisions are accounted for with the coefficient of restitution e_{cor}.
There are two possible choices for the time stepping strategy in impulse based methods:

- Simulate until a collision occurs, resolve the collision and then simulate until the next collision occurs. An advantage of this approach is that geometry overlap will never occur, because the simulation is halted just before impact and the collision is resolved before proceeding. A disadvantage is that there are cases when this algorithm will not be able to reach the end time of the simulation t_{end} in a finite number of time steps because an infinite number of collisions occur before reaching that time [2]. An example of this phenomenon is a ball with $0 < e_{cor} < 1$ bouncing on the floor. At each bounce, a fraction of the kinetic energy in the ball will be restituted and the time to the next impact will be smaller than the time between the previous two impacts. As a result, the ball will hit the floor an infinite number of times before coming to rest. Chatterjee and Ruina [2] suggest that this problem could be handled with extrapolation or truncation of the motion.

- Predictor-corrector steps: first take a time step with a predefined time step size, then resolve all the collisions that have occured during that time step. When all collisions have been resolved, a new time step can be taken. If this approach is

chosen, some overlap must be allowed and this overlap must be handled in a robust way. The main advantage of this approach is that a fixed time step can be used and therefore the number of time steps needed to simulate a given physical time interval will be known a-priori. An example of this approach is the Decomposition Contact Response (DCR) algorithm proposed by Cirak and West [4]. This algorithm relies on a decomposition of the momentum $\underline{p} = \underline{\underline{M}} \cdot \underline{v}$ into a normal and a tangential part. The normal direction is defined as the gradient of the constraint function: $\underline{n} = \nabla g$. The impulse in the normal direction is found by projecting $\underline{p}$ onto $\underline{n} = \nabla g$ and from this the tangential impulse can be computed. When the impulse has been decomposed into its normal and tangential parts, the coefficient of restitution e_{cor} can be used to update the normal impulse and Coulomb's law can be used to update the tangential impulse. This algorithm can handle the different cases of sliding and sticking without difficulty.

2 Theory

2.1 Fluid model

The motion of an incompressible viscous fluid is governed by the Navier-Stokes equations:

$$\frac{\partial u_j}{\partial x_j} = 0 \tag{2.1}$$

$$\rho_f \frac{\partial}{\partial t}(u_i) + \rho_f u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial u_i}{\partial x_j}\right) + \rho_g g_i \tag{2.2}$$

In the equation above, ρ_f is the fluid density, u_i is the velocity in coordinate direction i, p is the pressure, μ is the fluid viscosity and g_i is the gravity. (2.1) is the continuity equation and (2.2) gives the momentum equations in the three coordinate direction. The four equations in (2.1) and (2.2) can be used to solve for the four unknowns: the three components of the velocity and the pressure. The Navier-Stokes equations are nonlinear and coupled, note that each velocity component is present in all four equations. Therefore, the equations in (2.1) and (2.2) can not be used one by one to solve for one variable at a time. The equations can be solved simultaneously, or they can be rewritten to enable solution of one equation at a time. The nonlinearity in the equations could be handled by solving them with e.g. Newton's method or fix-point iterations. The widely used CFD codes use fix-point iterations and solve the equations in a segregated way, thus solving for one variable at a time. Two problems must be handled to solve the equations in a segregated way. First, the coupling of the equations must be handled sufficiently well, so that the iterative solution converges. The second problem is that the momentum equations (2.2) offer equations for the velocity components, but the continuity equation (2.1) can not be used to solve for the pressure without modification. A popular way to solve (2.1) and (2.2) in a segregated way is to use the SIMPLE method and rewrite the continuity equation as an equation for pressure correction. The SIMPLE method can be summarized as follows [21]:

1. Solve the momentum equations (2.2) with a guessed pressure.

2. Solve the pressure correction equation.

3. Update the pressure with the computed pressure correction. Correct the velocity so that continuity holds.

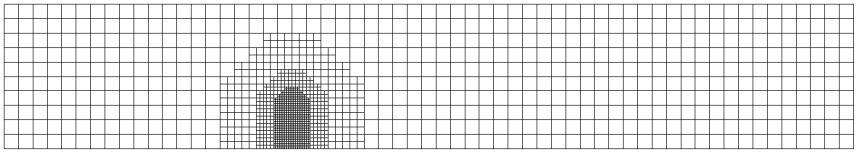


Figure 2.1: *Domain discretized with the finite volume method.*

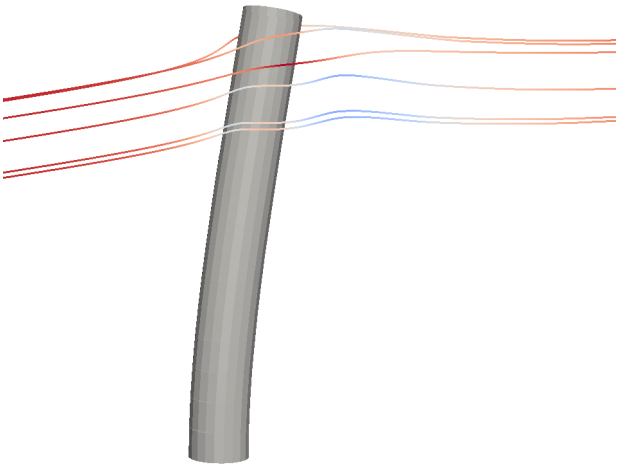


Figure 2.2: *A fiber immersed in a fluid: the forces from the fluid cause the fiber to deflect and the presence of the fiber disturbs the fluid. The streamlines are colored by the fluid velocity.*

4. Use the new pressure as initial guess for the new iteration and return to 1.

5. Terminate when converged.

The Finite Volume method guarantees global as well as local conservation. It is therefore often used to discretize the Navier-Stokes equations. The discretization is carried out by dividing the domain into small control volumes. The discrete equations for each control volume are then established by integrating the Navier-Stokes equations over the control volume and using the Divergence theorem to rewrite volume integrals of divergence as surface integrals. As a result, the discrete equations describe a balance of fluxes over the faces of the control volume. The discretization is illustrated in figure (2.1), which shows a domain divided into small cells. Two options are available for storage of the unknown variables: staggered and co-located grid arrangement. A staggered grid means that velocities are stored at the cell faces and the pressure is stored at the cell centers. A co-located arrangement implies that all variables are stored at the cell centers. A co-located grid is easier to construct, but it has the drawback that pressure oscillations may occur. The pressure oscillations can be suppressed with Rhie-Chow interpolation.

Immersed objects, such as e.g. fibers, may be present in the flow. The fluid will exert a force on the immersed objects and the immersed objects will disturb the fluid. The result is a mutual coupling between immersed objects and fluid as shown in figure (2.2). The coupling between immersed objects and fluid requires special attention. In the present work, the coupling is resolved with the Immersed Boundary Method, as described below.

2.1.1 Fluid-structure coupling

The Immersed Boundary Method [21, 22] constrains the fluid velocity to follow the surface velocity of the immersed object. This constraint is enforced without the need of a body fitted mesh as shown in figure (2.3), which enables fast and efficient mesh generation.

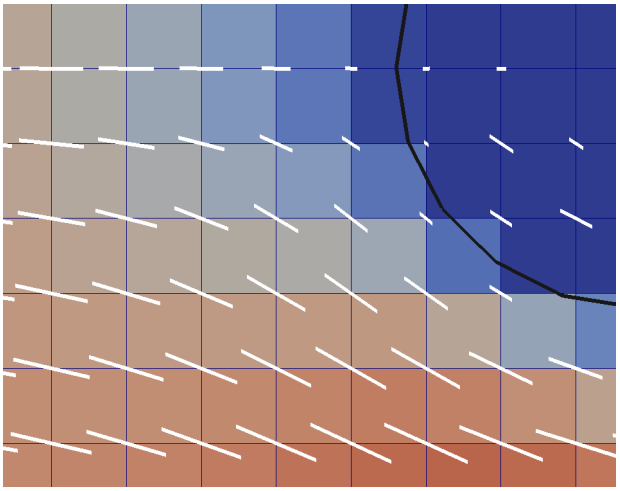


Figure 2.3: *Flow field close to an Immersed Boundary (black line). The flow field (white lines) follows the surface of the immersed object without need of a body fitted mesh. The cells are colored by the fluid velocity: blue denotes low velocity and red denotes high velocity.*

Application of an Immersed Boundary method requires classification of the cells in the grid. The cells are divided into the following groups [22]:

- *Fluid cells*: Cells located outside the immersed object, far away from the object.
- *Internal cells*: Cells located inside the immersed object, sufficiently far from the surface of the object.
- *Mirroring cells*: Cells located inside the immersed object, close to the surface of the object.
- *Extrapolation cells*: Cells located outside the immersed object, close to the surface of the object.

The velocity in the internal cells is set to match the velocity of the immersed object in that point. A Dirichlet boundary condition is used to enforce this constraint. For mirroring cells, an exterior normal point $\boldsymbol{p}_e$ is defined as [22]:

$$\boldsymbol{p}_e = \boldsymbol{p}_{mi} + 2.0\,(\boldsymbol{p}_{ib} - \boldsymbol{p}_{mi}) \tag{2.3}$$

In the equation above, $\boldsymbol{p}_{mi}$ is the center of the mirroring cell and $\boldsymbol{p}_{ib}$ is the closest point on the IB. For extrapolation cells, an exterior point is defined as:

$$\boldsymbol{p}_e = \boldsymbol{p}_{ib} + 2.0\,(\boldsymbol{p}_{ex} - \boldsymbol{p}_{ib}) \tag{2.4}$$

In the equation above, $\boldsymbol{p}_{ex}$ is the cell center of the extrapolation cell. The boundary condition for a mirroring cell is:

$$\frac{\boldsymbol{u}_{mi} + \boldsymbol{u}_e}{2} = \boldsymbol{u}_{ib} \tag{2.5}$$

The boundary condition for an extrapolation cell takes the following form:

$$\frac{\boldsymbol{u}_{ib} + \boldsymbol{u}_e}{2} = \boldsymbol{u}_{ex} \tag{2.6}$$

The velocity in the point $\boldsymbol{p}_e$ is interpolated and inserted into the Immersed Boundary condition. As a result, (2.5) and (2.6) become implicit boundary conditions which can be added to the matrix in the discretized equations. This results in a fictitious fluid velocity field inside the immersed object. Mass conservation is ensured by excluding the fictitious velocity field in the discretized equations. The result is a robust method that is second order accurate in space.

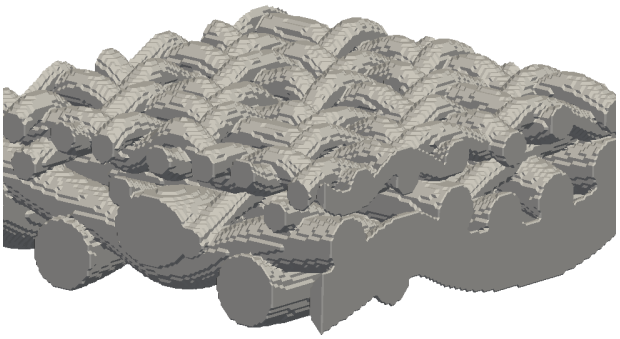


Figure 2.4: *Geometry of a forming fabric.*

2.1.2 Simplified alternative for fluid-structure coupling

Sometimes it is desirable to get approximate results fast. To offer this possibility, a simplified one-way coupling using a drag correlation was implemented. This was done by first simulating a steady-state solution of the fluid and then tracking the fibers using the drag correlation for long cylinders given in [23]. It is emphasized that this is an additional feature for rough estimates, the obtained results should be interpreted carefully.

2.1.3 Treatment of complex boundaries

Forming fabrics with complex geometries are studied in the present work, one of the forming fabrics is shown in figure (2.4). The forming fabric shown in figure (2.4) would be difficult to describe analytically due to its complexity, so a numerical description of the geometry is needed. In the present study, a voxelization has been used to describe the forming fabrics. With this method, the forming fabric is defined by a cloud of points that lie on the surface of the forming fabric. Each point, or voxel, has a radius so that the cloud of points and their radii cover the surface of the forming fabric. This is illustrated in figure (2.5). The voxelizations were generated from CAD data with GeoDict [10].

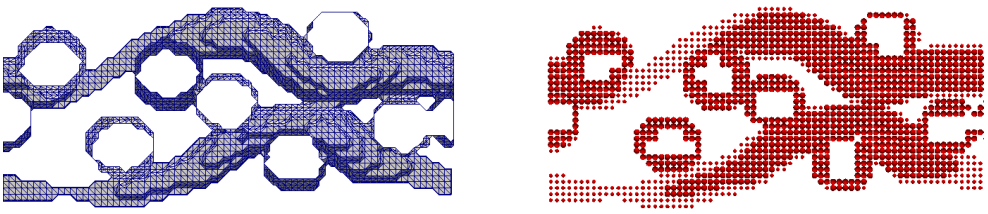


Figure 2.5: *Numerical treatment of complex forming fabrics: CAD geometry (left) and the corresponding voxelization (right). Forming fabric geometry courtesy of Albany International.*

2.1.4 CFD solver used in the present work

FCC's in-house CFD solver IBOFlow [16] is used to predict the behavior of the water surrounding the paper fibers. IBOFlow is a highly efficient code capable of performing transient DNS simulations with complex and moving boundaries. It relies on a finite volume discretization of the incompressible Navier-Stokes equations and the SIMPLEC method is used to handle the pressure-velocity coupling. All variables are stored in a co-located arrangement and Rhie-Chow interpolation is employed to prevent pressure oscillations. The solver uses a Cartesian octree grid to efficiently handle moving boundaries and automatic moving refinements. IBOFlow offers two possibilities to account for the presence of fibers

in the flow: the Mirroring Immersed Boundary method [21] and the Hybrid Immersed Boundary method [22]. These methods are capable of fully resolving the flow around each individual fiber and constraining the fluid velocity to follow the fiber velocity with second order accuracy. The resolved velocity field allows computation of the fluid force acting on a fiber segment by integrating the traction vector over the surface of the fiber segment.

2.2 Fiber model

The Finite Element (FE) model of a fiber is built up in several steps. First, the geometry of a fiber is described and definitions are introduced. Then the static FE model of a segment which is only subjected to small rotations is formulated. This is a classical problem which can be found in many standard books on Finite Elements, see e.g. [15, 26]. However, here we have the additional complication of a varying cross section. When the linear FE model is established, the extensions necessary for finite rotations are described. This is achieved by using the method of co-rotational frames. Finally, inertia terms are added to the static model so that transient simulations can be performed. Newmark's interpolation is used to interpolate velocities and accelerations in time. Hilber's α-method [14] is used to introduce numerical dissipation without sacrificing the second order accuracy.

The ultimate goal of the discussion in this section is to obtain a tangent stiffness matrix and a residual force vector that can be used to perform a transient simulation with Newton's method. The internal force vector $\underline{f}_{int}$ and the inertia force vector $\underline{f}_{m}$ can be computed as described below. Fluid forces, contact forces and possibly gravity are added to the external force vector $\underline{f}_{ext}$. When the solution has been found, the inertia force and internal force should be balanced by the external force. Therefore, the residual is computed according to:

$$\underline{res} = \underline{f}_{ext} - \underline{f}_{int} - \underline{f}_{m} \tag{2.7}$$

In the same way, the tangent stiffness matrix is composed of the Jacobian of the internal force, the Jacobian of the inertia force and the Jacobian of the external forces:

$$\underline{\underline{K}} = \underline{\underline{K}}_{int} + \underline{\underline{K}}_{m} + \underline{\underline{K}}_{ext} \tag{2.8}$$

When the tangent stiffness matrix and the residual have been computed, these can be used to take an iteration and find the increment $\Delta\underline{x}$:

$$\Delta\underline{x} = -\underline{\underline{K}}^{-1} \cdot \underline{res} \tag{2.9}$$

The increment vector $\Delta\underline{x}$ contains the displacement increments $\Delta\boldsymbol{u}$ and the rotation (spin) increments $\Delta\boldsymbol{\omega}$. The nodal coordinates are updated with the displacement increments according to:

$$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k + \Delta\boldsymbol{u} \tag{2.10}$$

The spin increment $\Delta\boldsymbol{\omega}$ is used to update the rotation matrix of the node as described later. When the displacements and rotations have been updated, the velocities and accelerations are updated with Newmark's method as described in [5].

2.2.1 Geometry of a fiber

A fiber is discretized by dividing it into several elements. Each element is associated with two nodes, a node at the start point and a node at the end point. Figure (2.6) shows the

centerline of a fiber with nodes and elements highlighted. The fiber is assumed to have an elliptical cross section with major radius R_a and minor radius R_b. The cross section may be hollow with interior major radius $R_{a,int}$ and interior minor radius $R_{b,int}$. The major axis of the ellipse does not have to be aligned with the local $\boldsymbol{e}_y$-axis. The angle between the major axis and the local $\boldsymbol{e}_y$-axis in the undeformed configuration is φ. A typical fiber cross section with R_a, R_b and φ highlighted is shown in figure (2.7). The major radius R_a, the minor radius R_b and the angle φ are not constant, they may vary linearly along the axis of the element.

Geometric nonlinearities will be accounted for in the present work and therefore a local coordinate system is associated with every element. The base vectors of this local frame are defined as follows: The local $\boldsymbol{e}_x$ vector lies along the line joining the start point and the end point of the element. If the start point is denoted by $\boldsymbol{p}_s$ and the end point is denoted by $\boldsymbol{p}_e$, then $\boldsymbol{e}_x$ is computed as:

$$\boldsymbol{e}_x = \frac{\boldsymbol{p}_e - \boldsymbol{p}_s}{|\boldsymbol{p}_e - \boldsymbol{p}_s|} \tag{2.11}$$

The local $\boldsymbol{e}_y$-axis is aligned with the major axis in the start point when the element is undeformed. When the element is deformed, $\boldsymbol{e}_y$ and $\boldsymbol{e}_z$ are defined as proposed by Nour-Omid and Rankin [25]: Let $\boldsymbol{q}$ be a vector in the direction of the major axis in the start point. $\boldsymbol{q}$ is rigidly attached to the start point and rotates with the start point. $\boldsymbol{e}_y$ and $\boldsymbol{e}_z$ are computed from $\boldsymbol{e}_x$ and $\boldsymbol{q}$:

$$\boldsymbol{e}_z = \frac{\boldsymbol{e}_x \times \boldsymbol{q}}{|\boldsymbol{e}_x \times \boldsymbol{q}|} \tag{2.12}$$

$$\boldsymbol{e}_y = \boldsymbol{e}_z \times \boldsymbol{e}_x \tag{2.13}$$

In this way, an orthogonal frame is constructed from the location of the nodes and the direction of the major axis of the cross section.

Every node has 6 degrees of freedom: translation in 3 directions and rotation about 3 axes. Hence a beam element has 12 degrees of freedom (dofs):

- 3 coordinates describing the location of the start point
- 3 angles describing rotation about the coordinate axes in the start point
- 3 coordinates describing the location of the end point
- 3 angles describing rotation about the coordinate axes in the end point

Figure (2.8) shows a typical fiber consisting of several elements.

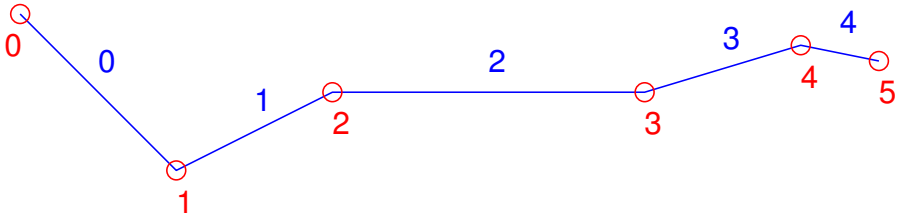


Figure 2.6: *Centerline of a discretized fiber. The blue lines are the elements and the blue numbers above the centerline show the element numbers. The red circles show the nodes and the red numbers below the centerline show the node numbers.*

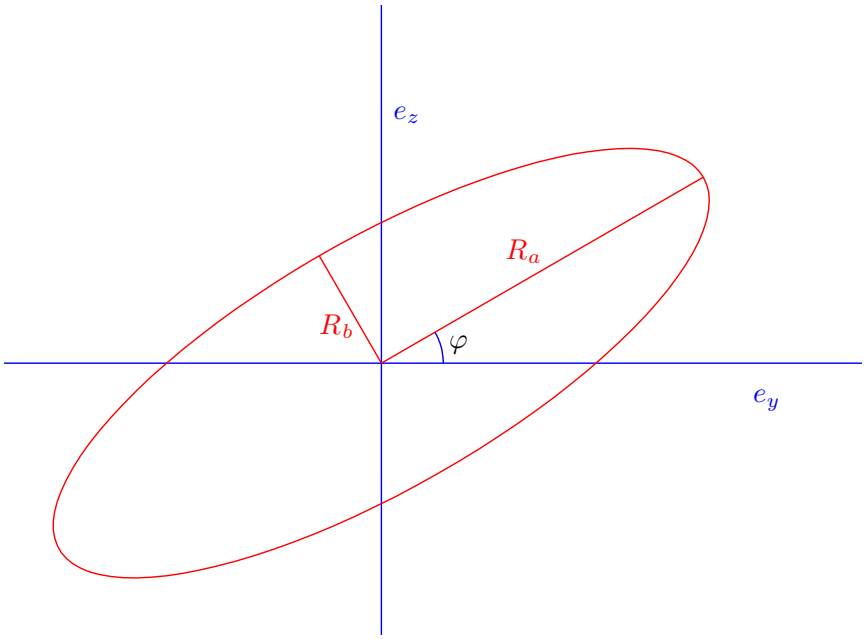


Figure 2.7: *Cross section of a fiber. The major radius R_a and the minor radius R_b are shown with red color. The local base vectors $\mathbf{e}_y$ and $\mathbf{e}_z$ are shown with blue color. The major axis of the ellipse is not aligned with the local $\mathbf{e}_y$-axis, the angle between them is φ.*

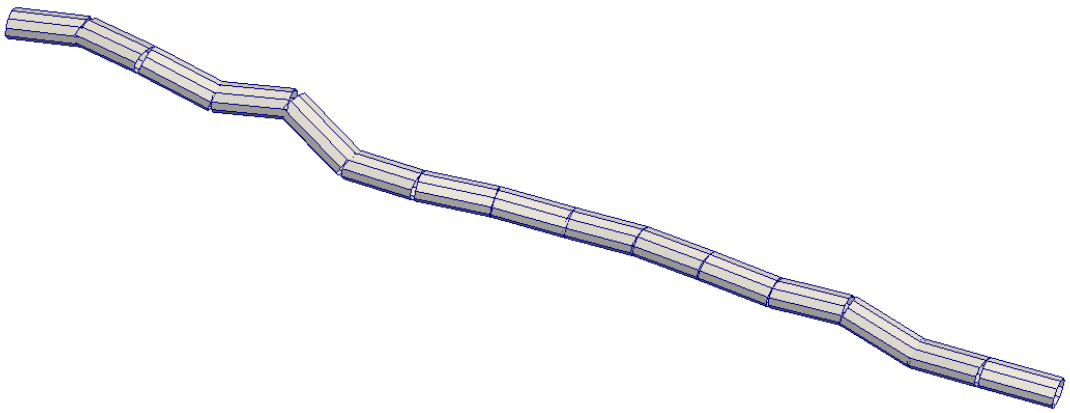


Figure 2.8: *A typical fiber divided into several elements.*

2.2.2 Small rotation FE formulation

A deformable fiber segment can be subjected to stretching, twisting and bending. The bending can occur in the local $\boldsymbol{e}_y$ direction as well as in the local $\boldsymbol{e}_z$ direction. If the cross section of the segment is symmetric about the $\boldsymbol{e}_y$-axis and the $\boldsymbol{e}_z$-axis, only plane bending will occur. This means that a force in the $\boldsymbol{e}_y$ direction gives rise to a deflection in the $\boldsymbol{e}_y$ direction, but no deflection occurs in the $\boldsymbol{e}_z$ direction. (The same holds for the $\boldsymbol{e}_z$ direction.) In the problem studied in this thesis, the cross section of the segment will not necessarily be symmetric about the $\boldsymbol{e}_y$-axis and the $\boldsymbol{e}_z$-axis. Therefore, skew bending may occur. This means that a force in the $\boldsymbol{e}_y$ direction may give rise to a deflection in the $\boldsymbol{e}_z$ direction. In summary, the deformation of a fiber segment is composed of:

- Bending of an Euler-Bernoulli beam in two dimensions (the local $\boldsymbol{e}_y$ and $\boldsymbol{e}_z$ directions)
- Twisting of the segment around its own axis
- Stretching of the segment in the axial direction

For a detailed discussion on beam theory in the context of small deformation elasticity and Finite Elements, see a good standard text book on Finite Elements, e.g. [15, 26]. The discrete Finite Element equations of a linear elastic, static beam element are outlined in the following.

The Euler-Bernoulli beam equation for the case of static bending can be written as:

$$\frac{\partial^2}{\partial x^2}\left(M_\alpha\right) = q_\alpha,\ \ \alpha = y, z \tag{2.14}$$

Twisted cross sections are considered and therefore skew bending must be included. Therefore, the bending moment M_α will consist of two parts: plane bending and skew bending. Multiplying (2.14) by a test function, using Galerkin's method and integrating by parts over the length of the element gives the bending terms of the stiffness matrix:

$$\underline{\underline{K}}_\alpha^{Bend} \cdot \underline{u} = \int_0^L \underline{\underline{B}}^T \left(M_\alpha^{plane} + M_\alpha^{skew}\right) dx \tag{2.15}$$

Here, the matrix $\underline{\underline{B}}$ contains the second derivatives of the test functions and the partial integration was used to move two derivatives from the bending moment to the test function. For a linear elastic material, the bending moment around the axis α can be computed as:

$$M_{\alpha\alpha} = EI_\alpha \frac{\partial^2 w_\alpha}{\partial x^2},\ \ M_{\alpha\beta} = ED_{\alpha\beta} \frac{\partial^2 w_\beta}{\partial x^2},\ \ \alpha = y, z,\ \ \beta = y, z \tag{2.16}$$

By introducing base functions for the deflection, the different components of the stiffness matrix can be identified. Bending can occur in y- and z-direction. Plane bending and skew bending in both directions must be considered. Therefore, the bending stiffness terms can be divided into:

- Deflection in y-direction as a result of force in the y-direction (plane bending): $K_{el,beam}^{yy} = \int_0^L \left(N''_{y,beam}\right)^T EI_z\left(x\right) N''_{y,beam} dx$
- Deflection in y-direction as a result of force in the z-direction (skew bending): $K_{el,beam}^{yz} = \int_0^L \left(N''_{y,beam}\right)^T ED_{yz}\left(x\right) N''_{z,beam} dx$

- Deflection in z-direction as a result of force in the y-direction (skew bending):
  $K^{zy}_{el,beam} = \int_0^L \left(N''_{z,beam}\right)^T ED_{zy}\left(x\right) N''_{y,beam} dx$

- Deflection in z-direction as a result of force in the z-direction (plane bending):
  $K^{zz}_{el,beam} = \int_0^L \left(N''_{z,beam}\right)^T EI_y\left(x\right) N''_{z,beam} dx$

Here N'' denotes the second derivative of the base function N with respect to x.

The stiffness terms corresponding to twisting of the element around the x-axis can be computed with the same approach:

$$K^{twist}_{el} = \int_0^L \left(N'_{twist}\right)^T KG\left(x\right) N'_{twist} dx \tag{2.17}$$

The stiffness terms corresponding to stretching in the axial direction become:

$$K^{axial}_{el} = \int_0^L \left(N'_{axial}\right)^T EA\left(x\right) N'_{axial} dx \tag{2.18}$$

As mentioned previously, each node has 6 degrees of freedom: 3 deflections $u_x,\ u_y,\ u_z$ and 3 rotations $\theta_x,\ \theta_y,\ \theta_z$. Thus, a fiber segment has 12 degrees of freedom. We would like to write the force equilibrium as:

$$\underline{f}_{el} = \begin{bmatrix} R^{el}_{sx} \\ R^{el}_{sy} \\ R^{el}_{sz} \\ M^{el}_{sx} \\ M^{el}_{sy} \\ M^{el}_{sz} \\ R^{el}_{ex} \\ R^{el}_{ey} \\ R^{el}_{ez} \\ M^{el}_{ex} \\ M^{el}_{ey} \\ M^{el}_{ez} \end{bmatrix} = \underline{\underline{K}}_{el} \cdot \begin{bmatrix} u^{el}_{sx} \\ u^{el}_{sy} \\ u^{el}_{sz} \\ \theta^{el}_{sx} \\ \theta^{el}_{sy} \\ \theta^{el}_{sz} \\ u^{el}_{ex} \\ u^{el}_{ey} \\ u^{el}_{ez} \\ \theta^{el}_{ex} \\ \theta^{el}_{ey} \\ \theta^{el}_{ez} \end{bmatrix} = \underline{\underline{K}}_{el} \cdot \underline{u}_{el} \tag{2.19}$$

To achieve this, the elements of the stiffness matrices from bending, twisting and elongation are gathered in the following way:

$$\underline{\underline{K}}_{el} = \begin{bmatrix} K^a_{el11} & 0 & 0 & 0 & 0 & 0 & K^a_{el12} & 0 & 0 & 0 & 0 & 0 \\ 0 & K^{yy,b}_{el11} & K^{yz,b}_{el11} & 0 & K^{yz,b}_{el12} & K^{yy,b}_{el12} & 0 & K^{yy,b}_{el13} & K^{yz,b}_{el13} & 0 & K^{yz,b}_{el14} & K^{yy,b}_{el14} \\ 0 & K^{zy,b}_{el11} & K^{zz,b}_{el11} & 0 & K^{zz,b}_{el12} & K^{zy,b}_{el12} & 0 & K^{zy,b}_{el13} & K^{zz,b}_{el13} & 0 & K^{zz,b}_{el14} & K^{zy,b}_{el14} \\ 0 & 0 & 0 & K^t_{el11} & 0 & 0 & 0 & 0 & 0 & K^t_{el12} & 0 & 0 \\ 0 & K^{zy,b}_{el21} & K^{zz,b}_{el21} & 0 & K^{zz,b}_{el22} & K^{zy,b}_{el22} & 0 & K^{zy,b}_{el23} & K^{zz,b}_{el23} & 0 & K^{zz,b}_{el24} & K^{zy,b}_{el24} \\ 0 & K^{yy,b}_{el21} & K^{yz,b}_{el21} & 0 & K^{yz,b}_{el22} & K^{yy,b}_{el22} & 0 & K^{yy,b}_{el23} & K^{yz,b}_{el23} & 0 & K^{yz,b}_{el24} & K^{yy,b}_{el24} \\ K^a_{el21} & 0 & 0 & 0 & 0 & 0 & K^a_{el22} & 0 & 0 & 0 & 0 & 0 \\ 0 & K^{yy,b}_{el31} & K^{yz,b}_{el31} & 0 & K^{yz,b}_{el32} & K^{yy,b}_{el32} & 0 & K^{yy,b}_{el33} & K^{yz,b}_{el33} & 0 & K^{yz,b}_{el34} & K^{yy,b}_{el34} \\ 0 & K^{zy,b}_{el31} & K^{zz,b}_{el31} & 0 & K^{zz,b}_{el32} & K^{zy,b}_{el32} & 0 & K^{zy,b}_{el33} & K^{zz,b}_{el33} & 0 & K^{zz,b}_{el34} & K^{zy,b}_{el34} \\ 0 & 0 & 0 & K^t_{el21} & 0 & 0 & 0 & 0 & 0 & K^t_{el22} & 0 & 0 \\ 0 & K^{zy,b}_{el41} & K^{zz,b}_{el41} & 0 & K^{zz,b}_{el42} & K^{zy,b}_{el42} & 0 & K^{zy,b}_{el43} & K^{zz,b}_{el43} & 0 & K^{zz,b}_{el44} & K^{zy,b}_{el44} \\ 0 & K^{yy,b}_{el41} & K^{yz,b}_{el41} & 0 & K^{yz,b}_{el42} & K^{yy,b}_{el42} & 0 & K^{yy,b}_{el43} & K^{yz,b}_{el43} & 0 & K^{yz,b}_{el44} & K^{yy,b}_{el44} \end{bmatrix} \tag{2.20}$$

Third order Hermite polynomials are used for the discretization of the bending terms while linear base functions are used for the twisting and the elongation. The integrals are evaluated with Gaussian quadrature.

With (2.19) and (2.20) we have a linear element stiffness matrix and corresponding force vector. These can be computed for all elements and assembled to a large and sparse linear system of equations of the form:

$$\underline{\underline{K}} \cdot \underline{u} = \underline{f} \tag{2.21}$$

This equation can be solved for $\underline{u}$. For a discussion on the topic of assembling the element matrices to a global matrix, see e.g. [15].

2.2.3 Mathematics of finite rotations

In linear theory, angles are assumed to be small so that the local coordinate system of each element is constant. In the present work, angles can not be assumed to be small and therefore the local element frame will not be constant. This fact necessitates the treatment of rotating coordinate systems. Therefore, the elementary mathematics of finite rotations is reviewed before proceeding with the formulation of nonlinear beam elements.

A coordinates system in space can be described with 3 base vectors: $\boldsymbol{e}_x$, $\boldsymbol{e}_y$ and $\boldsymbol{e}_z$. These are the base vectors of the local frame, expressed in the global coordinate system. With these base vectors, a second order tensor $\mathbf{E}$ that transforms a vector from the local coordinate system to the global coordinate system can be formed. This tensor has the local base vectors expressed in the global frame as columns:

$$\mathbf{E} = \begin{bmatrix} \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \end{bmatrix} \tag{2.22}$$

A first order tensor $\boldsymbol{v}$ is transformed from local to global coordinates according to:

$$\boldsymbol{v}_{glob} = \mathbf{E} \cdot \boldsymbol{v}_{loc} \tag{2.23}$$

A second order tensor $\mathbf{A}$ is transformed from local to global coordinates according to:

$$\mathbf{A}_{glob} = \mathbf{E} \cdot \mathbf{A}_{loc} \cdot \mathbf{E}^T \tag{2.24}$$

As noted in [5], $\mathbf{E}$ is an orthogonal tensor and therefore has the following properties:

$$\mathbf{E}^{-1} = \mathbf{E}^T; \; det\left(\mathbf{E}\right) = 1 \tag{2.25}$$

An orthogonal second order tensor $\mathbf{T}$ can also represent a rotation. In such a case, there is a relation between the spin axis $\boldsymbol{\omega}$ (a vector), the spin tensor $\boldsymbol{\Omega}$ (a skew symmetric tensor) and the rotation tensor $\mathbf{T}$ (an orthogonal tensor). The relation between $\boldsymbol{\omega}$ and $\boldsymbol{\Omega}$ is defined by the spin operation *S()* [25]:

$$\boldsymbol{\Omega} = S\left(\boldsymbol{\omega}\right) = \boldsymbol{\omega}\times = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{2.26}$$

As noted in [25], *S()* is related to the cross product through ($\boldsymbol{\omega}$ and $\boldsymbol{r}$ are vectors):

$$S\left(\boldsymbol{\omega}\right) \cdot \boldsymbol{r} = \boldsymbol{\omega} \times \boldsymbol{r} = -\boldsymbol{r} \times \boldsymbol{\omega} = -S\left(\boldsymbol{r}\right) \cdot \boldsymbol{\omega} \tag{2.27}$$

The function *axial()* reverses the effect of spin:

$$axial\left(\boldsymbol{\Omega}\right) = axial\left(\begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}\right) = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \boldsymbol{\omega} \tag{2.28}$$

The relation between the spin tensor $\boldsymbol{\Omega}$ and the corresponding rotation tensor $\mathbf{T}$ is given by the matrix exponent:

$$\mathbf{T} = exp\left(\boldsymbol{\Omega}\right) \tag{2.29}$$

The matrix logarithm reverses the effect of the matrix exponent:

$$\mathbf{\Omega} = log\left(\mathbf{T}\right) \tag{2.30}$$

Explicit expressions for the matrix exponential and the matrix logarithm are given in appendix A.

The above formulas can be used to update the rotations in the nodes. If a spin increment $\Delta\boldsymbol{\omega}$ has been computed, the rotation matrix $\mathbf{T}$ can be updated according to [25]:

$$\mathbf{T}^{n+1} = exp\left(S\left(\Delta\boldsymbol{\omega}\right)\right) \cdot \mathbf{T}^{n} \tag{2.31}$$

Here $n+1$ denotes the new iteration step and n denotes the old iteration step. In equation (2.31), the rotation tensor is updated with a spin increment $\Delta\boldsymbol{\omega}$. The linear FE formulation presented in the previous section has the rotation angles as unknows. Therefore, it is necessary to perform a change of variables from angles to spin variables. This can be done by studying the variation of the angles at a node:

$$\delta\boldsymbol{\theta} = \frac{\partial\boldsymbol{\theta}}{\partial\boldsymbol{\omega}} \cdot \delta\boldsymbol{\omega} = \mathbf{\Lambda} \cdot \delta\boldsymbol{\omega} \tag{2.32}$$

The tensor $\mathbf{\Lambda}$ in the equation above was derived in [25]. An expression for $\mathbf{\Lambda}$ is given in appendix A.

2.2.4 Computation of element deformations

Measures of the deformation of an element are necessary to evaluate the internal force vector. To identify the deformation of an element, consider a fiber segment subjected to an arbitrary motion expressed in the local coordinate system. The segment has 12 dofs with corresponding displacements and rotations:

$$\underline{u}^{el} = \left[\, u^{el}_{s,x}\ u^{el}_{s,y}\ u^{el}_{s,z}\ \theta^{el}_{s,x}\ \theta^{el}_{s,y}\ \theta^{el}_{s,z} u^{el}_{e,x}\ u^{el}_{e,y}\ u^{el}_{e,z}\ \theta^{el}_{e,x}\ \theta^{el}_{e,y}\ \theta^{el}_{e,z}\,\right] \tag{2.33}$$

The local coordinate system is rigidly attached to the first node of the segment, so the deformational displaclements in the first node are identically zero:

$$u^{el}_{s,x} = u^{el}_{s,y} = u^{el}_{s,z} = 0 \tag{2.34}$$

The $\boldsymbol{e}_x$-axis of the local coordinate system is always aligned with the centerline of the segment. Therefore, the deformational displacement in the second node can be expressed as:

$$u^{el}_{e,x} = l - L \tag{2.35}$$

$$u^{el}_{e,y} = u^{el}_{e,z} = 0 \tag{2.36}$$

Here, l is the length of the deformed segment and L is the initial length of the segment. In summary, the deformation of an element (with rigid body motion filtered out) in local coordinates can be expressed as:

$$\overline{\underline{u}}^{el} = \left[\, 0\ 0\ 0\ \theta^{el}_{s,x}\ \theta^{el}_{s,y}\ \theta^{el}_{s,z}(l-L)\ 0\ 0\ \theta^{el}_{e,x}\ \theta^{el}_{e,y}\ \theta^{el}_{e,z}\,\right] \tag{2.37}$$

To calculate the length of a segment is trivial, but calculation of the deformational rotations in (2.37) requires further attention. Let $\mathbf{E}$ denote the transformation matrix from the local frame to the global frame in the current configuration, as described in the previous section. Furthermore, let $\mathbf{E}_0$ denote the change of variables from the local frame to the global frame

in the initial configuration. Also, let $\mathbf{T}$ be a rotation matrix associated with each node describing how a vector attached to that node has rotated. Then, the rotation matrix corresponding to the rotational deformation can be computed according to equation (14) in [25]:

$$\overline{\mathbf{T}}_e = \mathbf{E}^T \cdot \mathbf{T}^g \cdot \mathbf{E}_0 \tag{2.38}$$

In equation (2.38), the notation used in [25] has been adopted, so that a superposed bar denotes the deformational part of a quatitiy. The superscript g denotes quantities expressed in global coordinates. It should be noted that the expression in (2.38) is not the only way to compute the rotational deformation. See for example eqn (22) in [5] for an alternative expression. Eqn (2.38) gives a rotation matrix describing the rotational deformation. The small deformation formulation requires the deformational rotation angles. Therefore, these angles must be extracted from $\overline{\mathbf{T}}_e$. This is done by using the matrix logarithm described in the previous section:

$$\overline{\boldsymbol{\theta}}_e = axial(log(\overline{\mathbf{T}}_e)) \tag{2.39}$$

2.2.5 Large rotation Finite Element formulation

This section presents the static terms in the nonlinear FE formulation used in the present thesis. The co-rotational (CR) formulation proposed by Nour-Omid and Rankin [24],[25] is used. The essence of the co-rotational technique is that small strains are assumed, but the small strain finite element formulation is expressed in a coordinate system which is attached to each element and moves with the element. In this way, the small strain formulation developed previously can be extended to include large rotation effects.

The linear formulation developed in the previous section is expressed with angles as rotational parameters. As noted previously, it is convenient to use spin variables to update the rotation tensors and therefore a change of variables must be performed. Following the work in [25], this change of variables transforms the small rotation internal force vector according to:

$$\underline{\overline{f}}_a^{el}\left(\overline{\boldsymbol{u}}^{el}, \overline{\boldsymbol{\omega}}^{el}\right) = \underline{\underline{H}}\left(\overline{\boldsymbol{\theta}}_a^{el}\right) \cdot \underline{\widetilde{f}}_a^{el}\left(\overline{\boldsymbol{u}}^{el}, \overline{\boldsymbol{\theta}}^{el}\right), \; a = s, e \tag{2.40}$$

Here, $\underline{\underline{H}}$ is given by:

$$\underline{\underline{H}}\left(\overline{\boldsymbol{\theta}}_a^{el}\right) = \begin{bmatrix} \underline{\underline{I}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \boldsymbol{\Lambda}\left(\overline{\boldsymbol{\theta}}_a^{el}\right) \end{bmatrix} = \begin{bmatrix} \underline{\underline{I}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \frac{\partial \overline{\boldsymbol{\theta}}_a^{el}}{\partial \overline{\boldsymbol{\omega}}_a^{el}} \end{bmatrix}, \; a = s, e \tag{2.41}$$

$\boldsymbol{\Lambda}$ in equation (2.41) is given in appendix A. Now consider the internal force vector $\underline{f}$, which in contrast to $\underline{\overline{f}}$ and $\underline{\widetilde{f}}$ properly accounts for finite rotations. $\underline{f}$ is the derivative of the strain energy Φ with respect to the nodal coordinates and rotations. Therefore, $\underline{f}$ can be expressed as:

$$f_i = \frac{\partial \phi}{\partial d_j} = \{\text{Chain rule}\} = \frac{\partial \overline{d}_j}{\partial d_i}\frac{\partial \phi}{\partial \overline{d}_j} = P_{ij}^T \overline{f}_j \tag{2.42}$$

Here d_i is a total displacement or a rotation in the node and $\overline{d}_i$ is the corresponding deformational part of the displacement or rotation. The employment of the chain rule in eqn (2.42) reveals that the internal force vector can be written as a contraction of a matrix $\underline{\underline{P}}$ and the linear, small deformation force vector $\underline{\overline{f}}$.

With the definition above, the matrix $\underline{\underline{P}}$ is computed as:

$$P_{ij} = \frac{\partial \overline{d}_i}{\partial d_j} \tag{2.43}$$

Therefore, a $6 \cdot 6$ block of the matrix $\underline{\underline{P}}$, associated with nodes a and b is computed as:

$$\underline{\underline{P}}_{ab} = \begin{bmatrix} \frac{\partial \overline{\boldsymbol{u}}_a^{el}}{\partial \boldsymbol{u}_b^{el}} & \frac{\partial \overline{\boldsymbol{u}}_a^{el}}{\partial \boldsymbol{\omega}_b^{el}} \\ \\ \frac{\partial \overline{\boldsymbol{\omega}}_a^{el}}{\partial \boldsymbol{u}_b^{el}} & \frac{\partial \overline{\boldsymbol{\omega}}_a^{el}}{\partial \boldsymbol{\omega}_b^{el}} \end{bmatrix} \; ; \; a = s, e; \; b = s, e \tag{2.44}$$

In (2.44), $\boldsymbol{\omega}_a^{el}$ is the spin variable associated with node a, expressed in local coordinates. Transforming (2.42) to global coordinates gives the following expression for the internal force vector:

$$\underline{f}^g = \underline{\underline{G}} \cdot \underline{\underline{P}}^T \cdot \underline{\overline{f}}^e \tag{2.45}$$

Here $\underline{\underline{G}}$ is the matrix that transforms a $12 \cdot 1$ vector from local to global coordinates:

$$\underline{\underline{G}} = \begin{bmatrix} \mathbf{E} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E} \end{bmatrix} \tag{2.46}$$

When the global equations of motion are solved, (2.45) enters the residual. To solve the resulting system of equations with Newton's method, the variation of (2.45) must be computed. Following the work in [25], this is computed as:

$$\delta \underline{f}^g = \underline{\underline{G}} \cdot \underline{\underline{P}}^T \cdot \delta \underline{\overline{f}}^{el} + \underline{\underline{G}} \cdot \delta \underline{\underline{P}}^T \cdot \underline{\overline{f}}^{el} + \delta \underline{\underline{G}} \cdot P^T \cdot \underline{\overline{f}}^{el} \tag{2.47}$$

The three terms in (2.47) are derived in [25]. Somewhat lengthy calculations give the following expressions:

$$\begin{aligned} \underline{\underline{G}} \cdot \underline{\underline{P}}^T \cdot \delta \underline{\overline{f}}^{el} &= \underline{\underline{G}} \cdot \underline{\underline{P}}^T \cdot \underline{\underline{K}}^{el} \left(\overline{\boldsymbol{u}}^{el}, \overline{\boldsymbol{\omega}}^{el} \right) \cdot \underline{\underline{P}} \cdot \underline{\underline{G}}^T \cdot \delta \underline{d}^g \\ G \cdot \delta P^T \cdot \underline{\overline{f}}^{el} = (...) &= -\underline{\underline{G}} \cdot \underline{\underline{\Gamma}} \cdot \underline{\underline{\widetilde{F}}}^T \cdot \underline{\underline{P}} \cdot \underline{\underline{G}}^T \cdot \delta \underline{d}^g, \; \underline{\underline{\widetilde{F}}} = \begin{bmatrix} S\left(\boldsymbol{f}_{int,a}^{el}\right) \\ \mathbf{0} \end{bmatrix} \\ \delta \underline{\underline{G}} \cdot \underline{\underline{P}}^T \cdot \underline{\overline{f}}^{el} = (...) &= -\underline{\underline{G}} \cdot \underline{\underline{\overline{F}}} \cdot \underline{\underline{\Gamma}}^T \cdot \underline{\underline{G}}^T \cdot \delta \underline{d}^g, \; \underline{\underline{\overline{F}}} = \begin{bmatrix} S\left(\boldsymbol{f}_{int,a}^{el}\right) \\ S\left(\boldsymbol{m}_{int,a}^{el}\right) \end{bmatrix} \end{aligned} \tag{2.48}$$

In (2.48), $\underline{\underline{\Gamma}}$ is the derivative of the spin of the local element frame with respect to the nodal degrees of freedom. For a beam element with two nodes it is given by:

$$\underline{\underline{\Gamma}}^T = \begin{bmatrix} \frac{\partial \boldsymbol{\omega}_E}{\partial \boldsymbol{u}_s} & \frac{\partial \boldsymbol{\omega}_E}{\partial \boldsymbol{\omega}_s} & \frac{\partial \boldsymbol{\omega}_E}{\partial \boldsymbol{u}_e} & \frac{\partial \boldsymbol{\omega}_E}{\partial \boldsymbol{\omega}_e} \end{bmatrix} \tag{2.49}$$

$\underline{\underline{\overline{F}}}$ and $\underline{\underline{\widetilde{F}}}$ in (2.48) are matrices which depend on the internal force vector, which has here been partitioned as:

$$\underline{f}_a^{el} = \begin{bmatrix} \boldsymbol{f}_{int,a}^{el} \\ \boldsymbol{m}_{int,a}^{el} \end{bmatrix} \tag{2.50}$$

In the equation above, $\boldsymbol{f}_{int,a}^{el}$ is the internal element force associated with node a and $\boldsymbol{m}_{int,a}^{el}$ is the internal element moment associated with node a.

An explicit epression for $\underline{\underline{F}}$ is given in appendix B. $\underline{\underline{\Gamma}}$ depends on the definition of the local frame. An expression for $\underline{\underline{\Gamma}}$ with the choice of coordinate system used in the present work is given in appendix B. Inserting the expressions in (2.48) into (2.47) gives the following expression for the tangent stiffness matrix in local coordinates:

$$\underline{\underline{K}}^{el} = \underline{\underline{P}}^T \cdot \underline{\underline{\overline{K}}}^{el} \cdot \underline{\underline{P}} - \underline{\underline{\Gamma}} \cdot \underline{\underline{\widetilde{F}}}^T \cdot \underline{\underline{P}} - \underline{\underline{\overline{F}}} \cdot \underline{\underline{\Gamma}}^T \tag{2.51}$$

It should be noted that the second and third terms in (2.51) are unsymmetric. For reasons of computational efficiency, it is desirable to have a symmetric tangent stiffness matrix.

In [25] it is proven that the tangent stiffness in (2.51) can be symmetrized in such a way that second order convergence of the Newton iterations is preserved. This symmetrization takes the following form:

$$\underline{\underline{K}}^{el}_{symm} = \underline{\underline{P}}^T \cdot \underline{\underline{\overline{K}}}^{el} \cdot \underline{\underline{P}} - \underline{\underline{\Gamma}} \cdot \underline{\underline{F}}^T - \underline{\underline{P}}^T \cdot \underline{\underline{F}} \cdot \underline{\underline{\Gamma}}^T, \; \underline{\underline{F}} = \begin{bmatrix} S\left(\boldsymbol{f}^{el}_{int,s}\right) \\ S\left(\frac{1}{2}\boldsymbol{m}^{el}_{int,s}\right) \\ S\left(\boldsymbol{f}^{el}_{int,e}\right) \\ S\left(\frac{1}{2}\boldsymbol{m}^{el}_{int,e}\right) \end{bmatrix} \tag{2.52}$$

The matrix $\underline{\underline{\overline{K}}}^{el}$ in (2.52) is the small deformation tangent stiffness expressed in spin variables. Its $6 \cdot 6$ blocks associated with nodes a and b are given by:

$$\underline{\underline{\overline{K}}}^{el}_{ab}\left(\overline{\boldsymbol{u}}^{el}, \overline{\boldsymbol{\omega}}^{el}\right) = \underline{\underline{H}}^T_a \cdot \underline{\underline{\widetilde{K}}}^{el}_{ab}\left(\overline{\boldsymbol{u}}^{el}, \overline{\boldsymbol{\theta}}^{el}\right) \cdot \underline{\underline{H}}_b + \delta_{ab} \frac{\partial \underline{\underline{H}}^T_a}{\partial \overline{\boldsymbol{\omega}}^{el}_b} \cdot \underline{f}^{el}_a \tag{2.53}$$

The equations (2.45) and (2.52) give expressions for the internal force vector and the corresponding tangent stiffness, which can be used directly to solve quasi static problems such as the quasi static examples in the validation section. As noted in [25], the derivative of $\underline{\underline{H}}$ in (2.53) is negligible for a sufficiently fine grid.

2.2.6 Dynamic FE formulation

The equations (2.45) and (2.52) give the local internal force vector and tangent stiffness for the quasistatic case. A simulation of paper fibers will be highly transient, especially when collisions are present. Therefore, inertia effects must be added to the governing equations. The strong form of the equations of motion are given by equation (27) and (28) in [5]:

$$\begin{aligned} \frac{\partial \boldsymbol{k}}{\partial t} &= \boldsymbol{f}_{ext} + \frac{\partial}{\partial x}\left(\mathbf{E} \cdot \boldsymbol{f}_{int}\right) \\ \frac{\partial \boldsymbol{\pi}}{\partial t} &= \boldsymbol{m}_{ext} + \frac{\partial \boldsymbol{r}}{\partial x} \times \left(\mathbf{E} \cdot \boldsymbol{f}_{int}\right) + \frac{\partial}{\partial x}\left(\mathbf{E} \cdot \boldsymbol{m}_{int}\right) \end{aligned} \tag{2.54}$$

Following the notation in [5], $\boldsymbol{k}$ is the linear momentum and $\boldsymbol{\pi}$ is the angular momentum. $\boldsymbol{f}_{ext}$ is the externally applied body force and $\boldsymbol{m}_{ext}$ is the externally applied body moment. $\mathbf{E}$ is the transformation from local to global coordinates, $\boldsymbol{f}_{int}$ is the internal force in a point and $\boldsymbol{m}_{int}$ is the internal moment in a point. Since this section deals with the inertia terms, the expressions for $\frac{\partial \boldsymbol{k}}{\partial t}$ and $\frac{\partial \boldsymbol{\pi}}{\partial t}$ are considered. Equations (29) and (30) in [5] give the follwing expressions for $\boldsymbol{k}$ and $\boldsymbol{\pi}$ for a beam cross section:

$$\boldsymbol{k} = \rho A \frac{\partial \boldsymbol{u}}{\partial t} \tag{2.55}$$

$$\boldsymbol{\pi} = \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} \tag{2.56}$$

Note that no assumptions on A and $\mathbf{J}$ have been introduced in (2.55) and (2.56), so the geometry of the cross section may vary over the length of the segment. This is in contrast to the formulation in [5], where the cross section is assumed to be constant. In the present work, the geometry of the cross section will be allowed to vary throughout the derivation of the inertia terms.

First consider the linear momentum. Take the time derivative of (2.55) (where $\boldsymbol{u}$ is given in global coordinates):

$$\frac{\partial \boldsymbol{k}}{\partial t} = \rho A \frac{\partial^2 \boldsymbol{u}}{\partial t^2} \tag{2.57}$$

Now consider the angular momentum. Take the time derivative of (2.56):

$$\begin{aligned}\dot{\boldsymbol{\pi}} &= \dot{\mathbf{E}} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \dot{\mathbf{E}}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}} = \\ &= S(\boldsymbol{w}) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot (S(\boldsymbol{w}) \cdot \mathbf{E})^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}} = \\ &= S(\boldsymbol{w}) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}}\end{aligned} \tag{2.58}$$

To get the weak form of the inertia force, multiply (2.58) by a test function N and integrate over the length of the beam element:

$$\underline{f}_m^g = \int_0^l \begin{bmatrix} N_1 \dot{\boldsymbol{k}} \\ N_1 \dot{\boldsymbol{\pi}} \\ N_2 \dot{\boldsymbol{k}} \\ N_2 \dot{\boldsymbol{\pi}} \end{bmatrix} dx = \int_0^l \begin{bmatrix} N_1 \rho A \ddot{\boldsymbol{u}} \\ N_1 \left(S(\boldsymbol{w}) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}} \right) \\ N_2 \rho A \ddot{\boldsymbol{u}} \\ N_2 \left(S(\boldsymbol{w}) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}} \right) \end{bmatrix} dx \tag{2.59}$$

Here N_1 and N_2 are the base functions. Galerkin's method is used, so the functions used to interpolate nodal quantities are the same as the test functions. Therefore, the quantities that may vary with the axial coordinate are interpolated according to:

$$\begin{aligned}\ddot{\boldsymbol{u}} &= N_1 \ddot{\boldsymbol{u}}_1 + N_2 \ddot{\boldsymbol{u}}_2 \\ \boldsymbol{w} &= N_1 \boldsymbol{w}_1 + N_2 \boldsymbol{w}_2 \\ \dot{\boldsymbol{w}} &= N_1 \dot{\boldsymbol{w}}_1 + N_2 \dot{\boldsymbol{w}}_2\end{aligned} \tag{2.60}$$

Linear base functions are used to interpolate the inertia terms, as suggested in [5]. The information given above is sufficient to evaluate the element inertia force numerically with Gaussian quadrature. In order to solve the resulting system of equations with Newton's method, it is necessary to calculate the variation of (2.59). Taking this variation will require taking the variation of the local frame. In this thesis, the local frame is defined as suggested by Nour-Omid and Rankin [25]. This definition is different than the definition of the element frame used in [5]. Therefore, the derivation of the Jacobian presented in [5] can not be used. A derivation of the Jacobian of the inertia force with the coordinate system used in the present work is given in appendix D.

2.3 Contact detection

Before any contact model can be applied, the fibers in contact must be identified. More precisely, the point of contact and the overlap must be determined. For circular cylinders with constant radius, an explicit formula for the contact point can easily be found. For segments with elliptic cross section with varying radii, the author has not succeeded to find explicit formulas for the shortest distance between two segments. Therefore, the shortest distance and the corresponding collision normal must be computed iteratively. To iteratively locate contact points becomes computationally expensive and therefore it is necessary to divide the contact detection into two step. First a search is performed based only on the nodal coordinates in order to find segments that are so close to each other that there is a possibility that contact might occur. Then the more expensive exact contact search is applied only to those segments that have been found to be so close to each other that contact could occur.

The section is divided into three parts. First, the algorithm used for locating segments that are close each other is described. Then, an iterative scheme for finding the shortest distance between a segment and a fixed, exterior point is derived. Finally, the shortest distance between two segments is studied.

2.3.1 Spatial search - identifying segments that are close to each other

The problem of identifying collision candidates can be solved in different ways. The easiest way is to loop over all elements and check every element against all other elements. If the number of elements is N, the computational cost for this algorithm will scale like $O(N^2)$. For problems where N is large, the computational cost for contact detection with this approach becomes unacceptably high. As noted in [31], the computational time spent in contact search may be well over 80% of the total CPU time for a typical problem with 1000 elements. Therefore an algorithm which scales better than $O(N^2)$ must be employed.

One good way to overcome this problem is to construct a binary search tree and use the tree structure to search for contact pairs. The computational cost for creating a binary search tree is $O(N\ log\ N)$ and the cost of searching is $O(N\ log\ N)$ [32]. Therefore the total cost will also scale like $O(N\ log\ N)$ which is much better than $O(N^2)$. In the present work, a kd-tree is used for the contact search. A kd-tree is a binary tree with nodes representing points in space. The points are inserted into the tree in such a way that every node splits the space along one of the coordinate axes. Implementing a kd-tree was not a part of the present work, an implementation already available was used.

2.3.2 Distance between a segment and a fixed point

Consider a segment with geometry as described previously and a point $\boldsymbol{q}$ with coordinates (q_x, q_y, q_z). A point on the surface of the segment is described by the parameters s and t. s describes the axial distance along the centerline, where $s = 0$ corresponds to the start point and $s = 1$ corresponds to the end point. t is a circumferential parameter and $t = 0$ corresponds to a point on the major axis of the studied cross section. For increasing t, the surface point rotates counter-clockwise around the ellipse and for $t = 2\pi$ the point has completed one full revolution. The definitions of s and t are illustrated in figure (2.9). The radii along the major and minor principal axes vary linearly with s. The parameter t is defined such that a point $\boldsymbol{p}$ on the surface is given in local coordinates as:

$$\boldsymbol{p}_{loc} = \{Ls,\ (R_{ae}s + R_{as}(1-s))\cos(t),\ (R_{be}s + R_{bs}(1-s))\sin(t)\} \tag{2.61}$$

Here R_{as} is the major radius in the start point, R_{ae} is the major radius in the end point, R_{bs} is the minor radius in the start point and R_{be} is the minor radius in the end point. It should be noted that that the parameter t is not equal to the angle between a vector in the radial direction and the local y-axis. To save computer power, it is assumed that the segment is not twisted around the centerline.

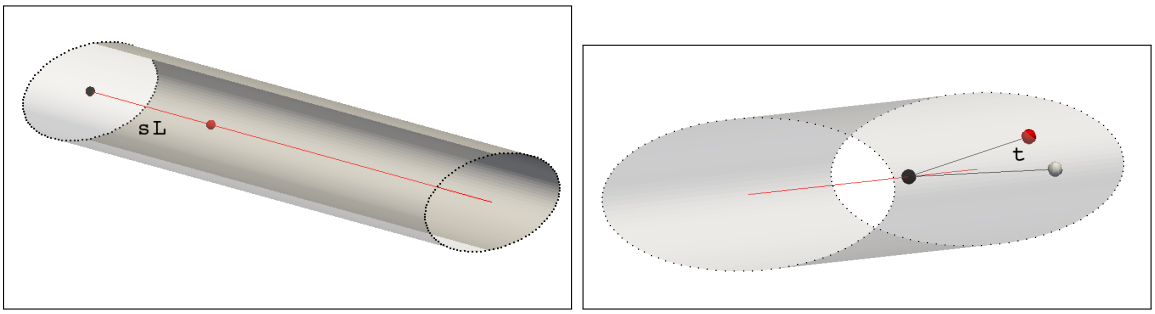


Figure 2.9: *Definition of the surface parameters s (left) and t (right). The red point on the centerline in the left figure corresponds to s: the distance between this point and the black start point is $s \cdot L$. The red surface point in the right figure corresponds to the angle parameter t.*

Let $\boldsymbol{p}_s$ denote the coordinates of the start point of the element. Furthermore, let $\mathbf{E}$ denote the change of basis matrix from local to global coordinates. Then the surface point

can be expressed in global coordinates as:

$$\boldsymbol{p} = \boldsymbol{p}_s + \mathbf{E} \cdot \boldsymbol{p}_{loc} \tag{2.62}$$

Let the exterior point be $\boldsymbol{q}$. Then a vector from the segment surface to the exterior point can be calculated as:

$$\boldsymbol{d} = \boldsymbol{q} - \boldsymbol{p} = \boldsymbol{q} - \boldsymbol{p}_s - \mathbf{E} \cdot \boldsymbol{p}_{loc} \tag{2.63}$$

The square of the distance between the surface and the exterior point is:

$$d^2 = \boldsymbol{d} \cdot \boldsymbol{d} \tag{2.64}$$

The parameters (s, t) corresponding to the shortest distance between the segment surface and the exterior point can be found by solving the following minimization problem:

$$\text{Find } (s, t) \text{ such that } d(s, t) \rightarrow \min \tag{2.65}$$

To find the parameters corresponding to the minimum of a distance which is always positive, it is sufficient to find the minimum of the square of the distance. Therefore, the minimization problem in (2.65) can be reformulated as:

$$\text{Find } (s, t) \text{ such that } \left(d(s, t)\right)^2 \rightarrow \min \tag{2.66}$$

The function to be minimized in (2.66) has critical points when the gradient is zero:

$$\frac{\partial}{\partial x_\alpha}\left(d^2\right) = 0, \; x_\alpha = (s, t) \tag{2.67}$$

The equation in (2.67) can be solved with Newton's method. To recast the problem into the framework of Newton's method in several variables, (2.67) is considered as a residual which should be forced to zero:

$$res_\alpha = \frac{\partial}{\partial x_\alpha}\left(d^2\right), \; x_\alpha = (s, t) \tag{2.68}$$

Performing the differentiation in (2.68) gives the residual as:

$$\begin{aligned} res_1 &= \frac{\partial d^2}{\partial s} = -2 d_i E_{ij} \frac{\partial p_j^{loc}}{\partial s} \\ res_2 &= \frac{\partial d^2}{\partial t} = -2 d_i E_{ij} \frac{\partial p_j^{loc}}{\partial t} \end{aligned} \tag{2.69}$$

To perform Newton iterations, the Jacobian $\underline{\underline{K}}$ of the residual in (2.69) must be computed. The Jacobian is calculated as:

$$K_{\alpha\beta} = \frac{\partial}{\partial x_\beta}\left(res_\alpha\right), \; x_\alpha = s, t, \; x_\beta = s, t \tag{2.70}$$

Since the Jacobian of the residual is obtained by differentiating d^2 twice, d^2 can be consiered as a potential for the problem to be solved. Since the Jacobian of the residual (the Hessian

of d^2) is derived from a potential, it will be symmetric. Performing the differentiation in (2.70) gives the following expressions for the Jacobian:

$$\begin{aligned}
K_{11} &= \frac{\partial^2 d^2}{\partial s \partial s} = \frac{\partial}{\partial s}\left(-2d_i E_{ij}\frac{\partial p_j^{loc}}{\partial s}\right) = 2E_{ik}\frac{\partial p_k^{loc}}{\partial s}E_{ij}\frac{\partial p_j^{loc}}{\partial s} \\
K_{12} = K_{21} &= \frac{\partial^2 d^2}{\partial t \partial s} = \frac{\partial}{\partial t}\left(-2d_i E_{ij}\frac{\partial p_j^{loc}}{\partial s}\right) = 2E_{ik}\frac{\partial p_k^{loc}}{\partial t}E_{ij}\frac{\partial p_j^{loc}}{\partial s} - 2d_i E_{ij}\frac{\partial^2 p_j^{loc}}{\partial t \partial s} \\
K_{22} &= \frac{\partial^2 d^2}{\partial t \partial t} = \frac{\partial}{\partial t}\left(-2d_i E_{ij}\frac{\partial p_j^{loc}}{\partial t}\right) = 2E_{ik}\frac{\partial p_k^{loc}}{\partial t}E_{ij}\frac{\partial p_j^{loc}}{\partial t} - 2d_i E_{ij}\frac{\partial^2 p_j^{loc}}{\partial t \partial t}
\end{aligned} \tag{2.71}$$

The derivatives of the surface point coordinates in the local frame needed in (2.69) and (2.71) are:

$$\begin{aligned}
\frac{\partial \boldsymbol{p}_{loc}}{\partial s} &= [L,\ (R_{ae} - R_{as})\cos(t),\ (R_{be} - R_{bs})\sin(t)] \\
\frac{\partial \boldsymbol{p}_{loc}}{\partial t} &= [0,\ \sin(t)(-(R_{ae}s + R_{as}(1-s))),\ \cos(t)(R_{be}s + R_{bS}(1-s))] \\
\frac{\partial^2 \boldsymbol{p}_{loc}}{\partial s \partial s} &= [0,\ 0,\ 0] \\
\frac{\partial^2 \boldsymbol{p}_{loc}}{\partial t \partial s} &= [0,\ -(R_{ae} - R_{as})\sin(t),\ (R_{be} - R_{bs})\cos(t)] \\
\frac{\partial^2 \boldsymbol{p}_{loc}}{\partial t \partial t} &= [0,\ \cos(t)(-(R_{ae}s + R_{as}(1-s))),\ \sin(t)(-(R_{be}s + R_{bs}(1-s)))]
\end{aligned}$$

With expressions for the residual and the Jacobian available, Newton iterations can be performed to get successively better approximations of the closest point on the segment surface. For each iteration, perform the following:

- $\Delta \underline{x} = -\underline{\underline{K}}^{-1} \cdot \underline{res}$
- $\underline{x} = \underline{x} + \Delta \underline{x}$
- Terminate when converged

The procedure outlined above solves the unconstrained optimization problem of finding the point on an elliptic surface which is closest to a given exterior point. It does not account for the constraint that the segment has finite length, so that the parameter s is constrained: $s \in [0, 1]$. If $s < 0$, the closest point on the surface is on the edge at $s = 0$ and if $s > 1$, the closest point is on the edge at $s = 1$. If this occurs, s should be set to 0 or 1, depending on which edge has been encountered, and iterations should be performed to find the t corresponding to the shortest distance. To achieve this, the Jacobian derived previously can be reduced so that the iterations do not change the value of s:

$$res_1 = 0 \tag{2.72}$$
$$K_{11} = 1 \tag{2.73}$$
$$K_{12} = K_{21} = 0 \tag{2.74}$$

By reducing the Jacobian in this way, the case where a point on the ellipse surface is closest to the exterior point and the case where a point on the edge is closest to the exterior point can be handled in the same iteration loop.

There is one more special case that might occur. Consider the case when the external point is to the left of the left edge or to the right of the right edge and the point is close to the centerline of the segment. In such a case, it might happen that the distance to one of the flat end surfaces of the segment is closer than the distance to the elliptic surface. This case must be treated separately. Furthermore, if the external point lies within the volume bounded by the elliptic surface and the flat end surfaces (i.e. inside the segment), the distance is considered to be negative. By adding a sign to the distance and creating a signed distance, the signed distance can be used to determine if penetration of the segement surface occurs. The proposed procedure for finding the closest distance between the segment surface and an external point, while accounting for the special cases that may occur, can be summarized as follows:

1. Assume that the closest point is somewhere on the elliptic surface. Guess an initial value for $s \in [0, 1]$ and an initial value for $t \in [0, 2\pi]$.

2. Use (2.69) and (2.71) to get successively better estimates for s and t. If s goes outside the range $[0, 1]$, then the closest point is on the edge. If this occurs, set $s = 0$ or $s = 1$ depending on which edge is closest. Continue iterations with reduced Jacobian.

3. - If $s \in]0, 1[$ and the exterior point is located outside the segment, then the closest point is on the ellipse surface and given by (s, t). The distance is positive.
   - If $s \in]0, 1[$ and the exterior point is located inside the segment, then the closest point is on the ellipse surface and given by (s, t). In this case, the distance is considered to be negative.
   - If $s \leq 0$ or $s \geq 1$ and the radial distance from the segment centerline to the exterior point is greater than the radial distance from the centerline to the ellipse surface, then the closest point lies on the edge. This point is given by the iteratively found t. s is zero or unity depending on which edge the point is located on.
   - If $s \leq 0$ or $s \geq 1$ and the radial distance from the segment centerline to the exterior point is smaller than the radial distance from the centerline to the ellipse surface, then the closest point lies on the flat end surface of the segment. This point is found by taking the vector from the start point to the exterior point and projecting it onto the plane which has the centerline as normal direction (i.e. the plane spanned by the major and minor axis in the start point).

2.3.3 Distance between two segments

As noted previously, the shortest distance between two segments with constant, cylindrical cross section can be found analytically. For the case with varying, elliptical cross section, the shortest distance must be found iteratively. The case with cylindrical cross section is treated first, since it can be used as an initial guess for the iterative solution. It is also interesting to highlight the extreme increase in complexity when going from circular cross sections to elliptic cross sections.

Consider two segments with constant circular cross section. Segment I has start point $\boldsymbol{p}_{Is}$ and end point $\boldsymbol{p}_{Ie}$ while segment II has start point $\boldsymbol{p}_{IIs}$ and end point $\boldsymbol{p}_{IIe}$. If the segments are parallel, then any point on one segment can be the closest point. If the segments are not parallel, a vector that is perpendicular to both centerlines can be computed

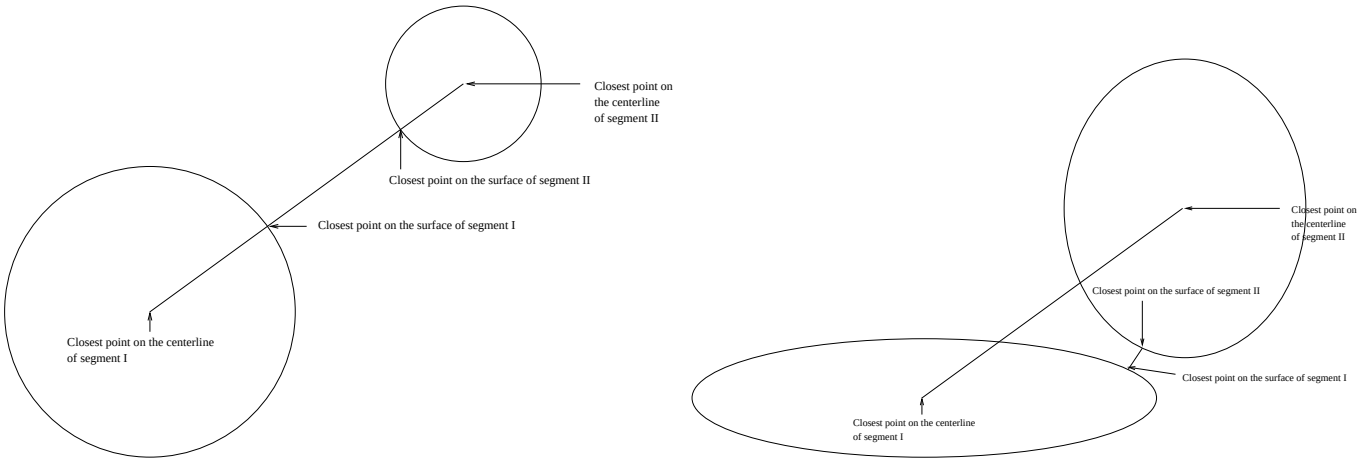


Figure 2.10: *Closest points on the surfaces of two segments with circular cross section (left) and elliptic cross section (right). When both cross sections are circular, the closest surface points will be on the line connecting the closest centerline points. This simple relationship does not hold for elliptic cross sections.*

as:

$$\boldsymbol{e}_{xI} = \frac{\boldsymbol{p}_{Ie} - \boldsymbol{p}_{Is}}{|\boldsymbol{p}_{Ie} - \boldsymbol{p}_{Is}|} \tag{2.75}$$

$$\boldsymbol{e}_{xII} = \frac{\boldsymbol{p}_{IIe} - \boldsymbol{p}_{IIs}}{|\boldsymbol{p}_{IIe} - \boldsymbol{p}_{IIs}|} \tag{2.76}$$

$$\boldsymbol{n} = \frac{\boldsymbol{e}_{xI} \times \boldsymbol{e}_{xII}}{|\boldsymbol{e}_{xI} \times \boldsymbol{e}_{xII}|} \tag{2.77}$$

A vector from the first segment to the second segment is the vector between the start points:

$$\boldsymbol{s} = \boldsymbol{p}_{IIs} - \boldsymbol{p}_{Is} \tag{2.78}$$

The shortest distance between the centerlines is the projection of this vector onto the unit normal:

$$d_{cl} = \boldsymbol{s} \cdot \boldsymbol{n} \tag{2.79}$$

If the segments have radii r_I and r_{II}, the shortest distance between the surfaces is:

$$d_{surf} = d_{cl} - r_I - r_{II} \tag{2.80}$$

The distance given in (2.80) is valid if the closest point is on the circular surface and not on the edge. Therefore, it must be checked if the closest point is outside the edge on any of the segments.

The case with constant circular cross sections is simple: short, explicit formulas can easily be derived. What makes this case simple is that the vector through the closest points on the centerlines also goes through the closest points on the segment surfaces. When the closest points on the centerlines have been identified, the closest points on the surfaces follow trivially by stepping in the direction of the vector between the centerline points. For segments with elliptical cross section, this is not the case. The closest points on the surfaces will in the general case not be located on the line connecting the closest centerline points. This is illustrated in figure (2.10). This phenomenon makes the case with elliptical cross sections much more difficult, since studying the centerlines is not sufficient to determine the shortest distance between the surfaces.

A procedure for computing the shortest (unsigned) distance between two segments is given in appendix E. The derivation follows the same ideas as for the case when the distance between a segment and a point is sought.

2.4 Contact models

This section describes the contact models implemented in the present work. Two contact algorithms have been implemented: a penalty method and the DCR method proposed in [4].

2.4.1 A penalty method suitable for paper forming

A penalty method suitable for paper forming has been constructed from the variants of penalty methods described previously. As discussed earlier, several possible choices exist for the penalty force in the normal direction. This work is based on a normal force of the following form:

$$f_n\left(g, v_n\right) = \begin{cases} K \cdot \left(\frac{-g}{d_{ref}}\right)^{\zeta} & \text{if } v_n \leq 0 \\ e_{cor} \cdot K \cdot \left(\frac{-g}{d_{ref}}\right)^{\zeta} & \text{otherwise} \end{cases} \tag{2.81}$$

The above expression allows modeling of elastic as well as inelastic collisions by introducing the coefficient of restitution e_{cor} directly into the expression for the force as proposed in [13]. The parameters d_{ref} and ζ give the possibility to control how rapidly the function increases with increasing overlap. Note that equation (2.81) reduces to familiar cases for special choices of parameters: setting $d_{ref} = 1$ and $\zeta = 1.5$ gives Hertz contact theory while $\zeta = 1$ gives a linear relationship between force and overlap.

In principle, there are two possible ways to handle friction: to handle the cases of slip and stick separately or to try to find a model that can handle both cases simultaneously. To handle the cases of sliding and sticking separately in a penalty method is impractical. The other possibility is to regularize the friction force to give a smooth and differentiable transition between sticking and sliding. With this approach, the possibility to model perfect sticking is sacrificed in favour of a smooth expression for the friction force. In the present work, a square root regularization proposed by Wriggers [32] is used:

$$\begin{aligned} f_t\left(f_n, v_t\right) &= -\mu \cdot \chi\left(v_t\right) \cdot \left|\boldsymbol{f}_n\right| \\ \chi\left(v_t\right) &= \frac{v_t}{\sqrt{v_t^2 + \varepsilon^2}} \end{aligned} \tag{2.82}$$

In (2.82), ε is a small number which controls how accurate (and how stiff) the regularization is. A small ε gives better agreement with Coulomb's friction law but also gives stronger nonlinearities so that smaller time steps may have to be taken. A larger value of ε gives a smoother transition with less accuracy. This behavior is shown in figure (2.11).

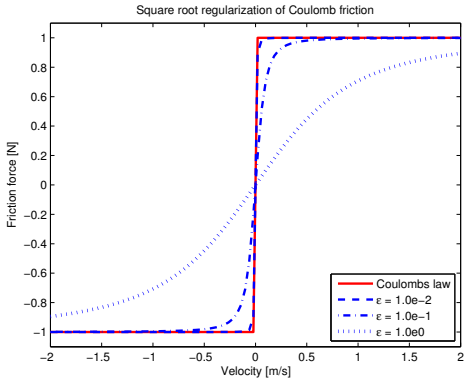


Figure 2.11: *Square root regularization of Coulomb friction. A small value of ε gives higher accuracy but may require shorter time steps.*

When the magnitude of the normal force and the tangential force have been computed, the total contact force is computed as:

$$\boldsymbol{f} = f_n \boldsymbol{n} + f_t \boldsymbol{t} \tag{2.83}$$

Here $\boldsymbol{n}$ is the contact normal direction and $\boldsymbol{t}$ is the tangential sliding direction. $\boldsymbol{n}$ is computed as ($\boldsymbol{p}_I$ and $\boldsymbol{p}_{II}$ are the closest points on segment I and II):

$$\boldsymbol{n} = \frac{\boldsymbol{p}_{II} - \boldsymbol{p}_I}{|\boldsymbol{p}_{II} - \boldsymbol{p}_I|} \text{ if } |\boldsymbol{p}_{II} - \boldsymbol{p}_I| \neq 0 \tag{2.84}$$

$$\boldsymbol{n} = \frac{\frac{\partial \boldsymbol{p}_I}{\partial s} \times \frac{\partial \boldsymbol{p}_I}{\partial t}}{\left| \frac{\partial \boldsymbol{p}_I}{\partial s} \times \frac{\partial \boldsymbol{p}_I}{\partial t} \right|} \text{ otherwise} \tag{2.85}$$

i.e. the normal is taken as the closest point on surface II minus the closest point on surface I if these points do not coincide. If the closest points do coincide, the collision normal is instead taken as the normal direction of surface I.

The tangential sliding direction is computed as:

$$\boldsymbol{t} = \frac{\boldsymbol{v}_{tang}}{|\boldsymbol{v}_{tang}|} \text{ if } |\boldsymbol{v}_{tang}| \neq 0 \tag{2.86}$$

$$\boldsymbol{t} = \boldsymbol{0} \text{ otherwise} \tag{2.87}$$

Note that $\boldsymbol{t}$ could be set to any value if $|\boldsymbol{v}_{tang}| = 0$, because in this case the friction force will be zero any way.

The contact force is a function of the overlap and the relative velocity, which in turn are functions of the nodal positions at the current time step. As a result, the contact forces will change during the iterations of a single time step. Therefore, only adding the contact forces to the right hand side without changing the tangent stiffness matrix will destroy the second order convergence of the Newton iterations. In order to retain second order convergence, the Jacobian of the contact must be computed and added to the tangent stiffness matrix. As mentioned previously, the author has not succeeded to find an explicit expression for the contact point. Therefore it is not possible to compute the Jacobian of the contact force analytically and two options remain:

- Compute the Jacobian of the contact force numerically. This is always possible, but it comes at an increased computational cost.
- Perform the simulation without using a Jacobian for the contact force and accept that the convergence will not be second order.

Both options have been tried in the present work. It was found that the loss of convergence was small for the cases of interest here and therefore most of the simulations were performed without using a Jacobian for the contact force.

2.4.2 Decomposition Contact Response

The algorithm proposed by Cirak and West [4] has been implemented in the present work. Next, the algorithm given in [4] is described in greater detail than in the introduction. Then, modifications are proposed to make the algorithm suitable for simulations of paper forming.

The DCR method is an algorithm of predictor-corrector type: first a predictor time step is taken as if no contacts occured, then interpenetrations are removed and velocities are

corrected. In [4], the signed volume of intersection is used as contact constraint function. However, as noted in [4], any properly defined constraint function such as the gap function can be used. The DCR method relies on a decomposition of the momentum of two bodies in contact, which can be expressed as:

$$\underline{p} = \underline{\underline{M}} \cdot \underline{v} \tag{2.88}$$

where $\underline{\underline{M}}$ is the mass matrix and $\underline{v}$ is the velocity vector. The constraint function g has the following properties:

- $g = 0$ corresponds to a configuration where contact just occurs, but no interpenetration occurs

- ∇g gives the contact (non-unit) normal

First, the momentum vector is decomposed into a normal and a tangential part:

$$\underline{p} = \underline{p}_{norm} + \underline{p}_{tang} \tag{2.89}$$

The normal component is taken as the projection of $\underline{p}$ onto ∇g. This definition gives the following condition on $\underline{p}_{norm}$ (eqn (17) in [4]):

$$(\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot \left(\underline{p} - \underline{p}_{norm}\right) = 0 \tag{2.90}$$

Since $\underline{p}_{norm}$ is defined to be in the direction of ∇g, it can be written as:

$$\underline{p}_{norm} = a_1 \nabla g \tag{2.91}$$

for some scalar a_1. Inserting (2.91) in (2.90) gives:

$$(\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot (\nabla g)\, a_1 = (\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p} \tag{2.92}$$

$$\Rightarrow a_1 = \left[(\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot \nabla g\right]^{-1} \cdot (\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p} \tag{2.93}$$

$$\Rightarrow \underline{p}_{norm} = a_1 \nabla g = \left[(\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p}\right] \left[(\nabla g)^T \cdot \underline{\underline{M}}^{-1} \cdot \nabla g\right]^{-1} \nabla g \tag{2.94}$$

When the normal part of the impulse has been extracted, the normal contact can be modeled by introducing the coefficient of restitution e_{cor}. Let $\underline{p}^-$ denote the momentum prior to collision and let $\underline{p}^+$ denote the momentum after collision. Then the momentum in the normal direction can be expressed as:

$$\underline{p}^+_{norm} = (-e_{cor})\, \underline{p}^-_{norm} \tag{2.95}$$

The normal impulse transferred during the collision is:

$$\underline{I}_{norm} = -\left(1 + e_{cor}\right) \underline{p}^-_{norm} \tag{2.96}$$

To include friction, the part of the impulse that causes relative motion between the contacting bodies must be separated from the rigid body motion. This is achieved by introducing a vector $\underline{h}$ between the contact points of the two bodies:

$$\underline{h} = \underline{x}_L - \underline{x}_R \tag{2.97}$$

The rigid body velocity $\underline{\underline{M}}^{-1} \cdot \underline{p}_{fix}$ keeps $\underline{h}$ unchanged:

$$(\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot \underline{p}_{fix} = \underline{0} \tag{2.98}$$

(2.98) can be used to find a condition for $\underline{p}_{nonfix}$:

$$(\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot \left(\underline{p} - \underline{p}_{nonfix}\right) = \underline{0} \tag{2.99}$$

$$\Rightarrow (\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \underline{p} - (\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \underline{p}_{nonfix} = \underline{0} \tag{2.100}$$

$\underline{p}_{nonfix}$ must be in the direction of $\nabla \underline{h}$ and therefore it can be written as:

$$\underline{p}_{nonfix} = (\nabla \underline{h})^T \cdot \underline{c}_2 \tag{2.101}$$

for some vector $\underline{c}_2$. This vector can be computed by inserting (2.101) in (2.100):

$$(\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \underline{p} = (\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot (\nabla \underline{h})^T \cdot \underline{c}_2 \tag{2.102}$$

$$\Rightarrow \underline{c}_2 = \left[(\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot (\nabla \underline{h})^T\right]^{-1} \cdot (\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot \underline{p} \tag{2.103}$$

$$\Rightarrow \underline{p}_{nonfix} = (\nabla \underline{h})^T \cdot \left[(\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot (\nabla \underline{h})^T\right]^{-1} \cdot (\nabla \underline{h}) \cdot \underline{\underline{M}}^{-1} \cdot \underline{p} \tag{2.104}$$

When $\underline{p}_{nonfix}$ and $\underline{p}_{norm}$ are known, the sliding part of the impulse, responsible for friction, can be computed as:

$$\underline{p}_{slide} = \underline{p}_{nonfix} - \underline{p}_{norm} \tag{2.105}$$

When friction is present, the momentum jump equation for a contact is written as:

$$\underline{p}^+ = \underline{p}^- + \underline{I}_{norm} + \underline{I}_{slide} \tag{2.106}$$

The impulse in the sliding direction is given by equation (45) in [4]:

$$\underline{I}_{slide} = \begin{cases} -\underline{p}_{slide} & \text{if } \underline{p}_{slide}^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p}_{slide} < \mu \underline{p}_{norm}^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p}_{norm} \\ -\mu \left(\frac{\underline{p}_{norm}^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p}_{norm}}{\underline{p}_{slide}^T \cdot \underline{\underline{M}}^{-1} \cdot \underline{p}_{slide}} \right) \underline{p}_{slide} & \text{otherwise} \end{cases} \tag{2.107}$$

In the equation above, the first case corresponds to sticking, while the second case corresponds to sliding. Note that this model is capable of modeling perfect stick, in contrast to the penalty model developed in the previous section, where sticking could only be fulfilled approximately.

2.4.2.1 DCR for fiber-fiber contact

The DCR algorithm can be summarized as follows:

1. Update nodal positions and velocities without considering collisions
2. Update velocities by using the momentum jump equation
3. Remove penetrations by using closest point projection

Step 1 above does not need any further clarification. In step 2, the nodal velocities should be updated by using (2.96), (2.106) and (2.107). These formulas require the mass matrix $\underline{\underline{M}}$, the gradient of the gap function g and the gradient of the vector $\underline{h}$. To be consistent, the same mass matrix as used when assembling the inertia stiffness should be used. However, to reduce the computational cost and simplify the calculations, Cirak and West [4] suggest using a lumped mass matrix instead when solving the momentum jump equation.

The gradient of g is also required. Since no closed form expression for g has been found, the derivative of g must be computed numerically. Since a contact involves two segments and each segment has 12 degrees of freedom, each contact involves 24 dofs and g will have 24 entries:

$$\nabla g = \left[\frac{\partial g}{\partial \boldsymbol{u}_{aI}} \; \frac{\partial g}{\partial \boldsymbol{\theta}_{aI}} \; \frac{\partial g}{\partial \boldsymbol{u}_{bI}} \; \frac{\partial g}{\partial \boldsymbol{\theta}_{bI}} \; \frac{\partial g}{\partial \boldsymbol{u}_{aII}} \; \frac{\partial g}{\partial \boldsymbol{\theta}_{aII}} \; \frac{\partial g}{\partial \boldsymbol{u}_{bII}} \; \frac{\partial g}{\partial \boldsymbol{\theta}_{bII}} \right] \tag{2.108}$$

Hence, ∇g will have dimension $24 \cdot 1$. The gradient of h, which is evaluated in the same way, will have dimension $24 \cdot 3$.

Step 3 above corrects the nodal positions by moving interpenetrating segments in such a way that the surfaces exactly touch each other. Let $\boldsymbol{n}$ denote the contact normal, directed from segment I to segment II as described previously. Note that $\boldsymbol{n}$ is the physical contact normal in R^3, so this $\boldsymbol{n}$ is not equal to ∇g, which is in R^{24}. Since the gap function gives the penetration distance in the normal direction, the closest point projection can be achieved by moving the nodal coordinates of segment I a distance $\frac{g}{2}$ in the direction of $-\boldsymbol{n}$ and moving the nodal coordinates of segment II a distance $\frac{g}{2}$ in the direction of $\boldsymbol{n}$.

2.4.2.2 DCR for fiber-fabric contact

In the present work, the forming fabric is considered to be rigid. The fabric is described by a cloud of points defining the surface of the fabric. Contact between the fibers and the fabric is accounted for by computing the distance between the fiber nodes and the fabric points. If this distance is smaller than a chosen penalty layer thickness, the collision is resolved by applying the DCR method to the contacting node. In this case the lumped mass matrix of the contacting node is used instead of using the mass matrix of the whole segment.

2.5 Measures of fiber web properties

Some measures that can be used to characterize a fiber web need to be defined. Two such measures are proposed below.

2.5.1 Mass distribution (Grammage)

It is interesting to study how the paper fibers are distributed in the plane of the fiber web. For this purpose, an area mass distribution can be computed [19]. Consider a fiber web in the xy-plane so that the thickness of the web is in the z-direction. A rectangular grid is defined in the xy-plane with N_x elements in the x-direction and N_y elements in the y-direction, see figure (2.12). All fiber segments within a specified bounding box are projected onto the xy-plane and all segments corresponding to a given 2D-grid element are identified. The mass of all segments corresponding to a 2D-grid element are added to that 2D-grid element. In this way, a certain mass of paper is associated with each 2D-grid element. Dividing this mass by the area of the 2D-grid element gives a mass per unit area. This mass per unit area can be used to study how the distribution of fibers varies in the fiber web.

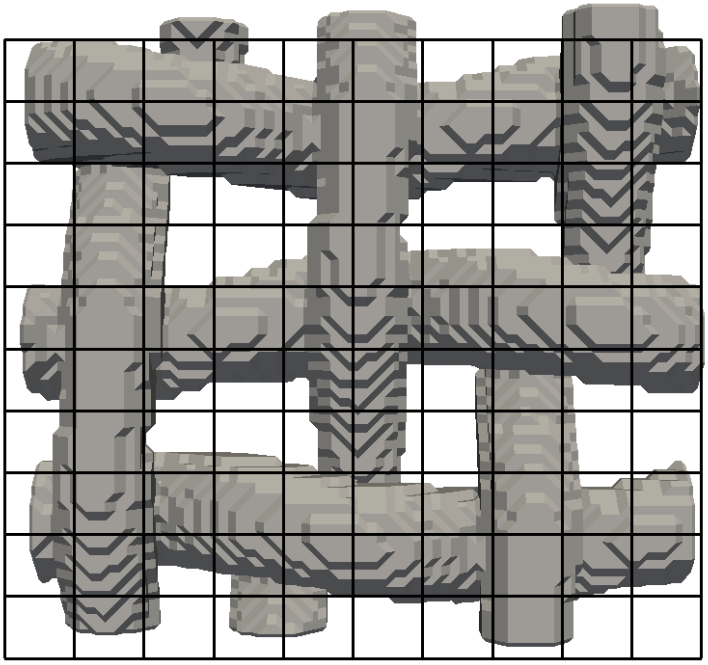


Figure 2.12: *Domain with forming fabric divided into two-dimensional elements for postprocessing. Forming fabric geometry courtesy of Albany International.*

2.5.2 Fiber directions (anisotropy)

An important property of the fiber web is the orientation of fibers. The direction of the fibers could be fully randomized, or the fibers could be aligned in some direction. A simple measure is to define a 2D-grid as described above and compute the direction vector $\boldsymbol{e}_x^i$ of the centerline of each fiber segment i. When directions are studied, the vector $\boldsymbol{e}_x^i$ should be equivalent to the vector $-\boldsymbol{e}_x^i$. Therefore, a direction vector rotated to the right half plane is defined:

$$\boldsymbol{a}^i = \begin{cases} \boldsymbol{e}_x^i & \text{if} \quad e_{x,1}^i \geq 0 \\ -\boldsymbol{e}_x^i & \text{if} \quad e_{x,1}^i < 0 \end{cases} \tag{2.109}$$

When the direction vector $\boldsymbol{a}$ has been computed, the angle between this vector and the centerline can be used as a measure of the orientation of the fibers segment. When the angle of each segment has been computed and assigned to a 2D-grid cell, the mean value and standard deviation of the angle in each cell can be computed.

3 Numerical results

3.1 Validation of FE solver

Before proceeding with simulations of paper forming, the different models implemented in the present work must be validated. If the fiber model does not give accurate results for cases without fluid coupling, it can not be expected that the fiber model will give accurate results when a fluid is present. Therefore, the fiber model is first validated for a number of cases where no fluid is involved. First two static validation cases are studied and compared to results from the literature. The purpose of this part is to show that the code is capable of handling complex structural behavior including large rotations. Then two cases involving large three dimensional motion are studied and it is shown that the solver can handle dynamical problems with large rotations. Finally, two contact problems are studied. In the first contact problem, a collision between two bars is studied and it is shown that the displacement history agrees with results from the literature. The second contact problem involves inelastic contact with friction between a beam and a rigid table.

3.1.1 Validation of static problems

3.1.1.1 Hockling of a cable

In the first static test case, the cable segment shown in figure (3.1) is twisted around its own axis. This is a linear problem for small twisting angles, but here the cable is twisted so much that its capacity to store energy in torsion is exceeded. The result is that the cable becomes unstable and buckles with a complex three-dimensional response. The problem has previously been studied by Nour-Omid and Rankin [25] and later by Gruttmann, Sauer and Wagner [11]. Nour-Omid and Rankin analyzed the problem with their co-rotational beam model while Gruttmann et al. used a model with Timoshenko beam kinematics.

The cable is modeled with 20 cubic beam elements of equal size. The right end is fixed, while the left end is allowed to twist around the x-axis and translate along the x-axis. Displacement control is used to control the simulation: the left end of the cable is twisted one revolution around the x-axis. The cable has the following properties:

- Total length: $L = 240.0\ mm$
- Polar moment: $I_x = 2.16\ mm^4$
- Area moments: $I_y = I_z = 0.0833\ mm^4$
- Young's modulus: $E = 71240\ N/mm^2$
- Poisson's ratio: $\nu = 0.31$
- Shear modulus: $G = 27190\ N/mm^2$

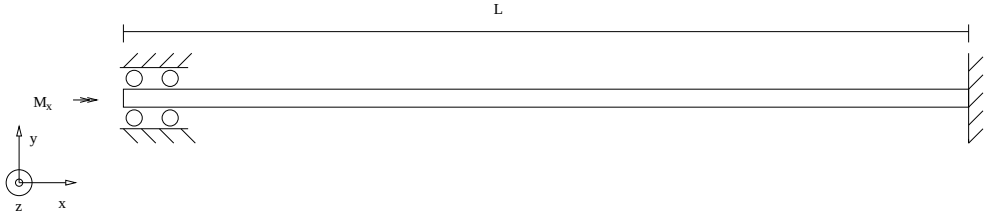


Figure 3.1: *Geometry of the cable considered in the first static test case.*

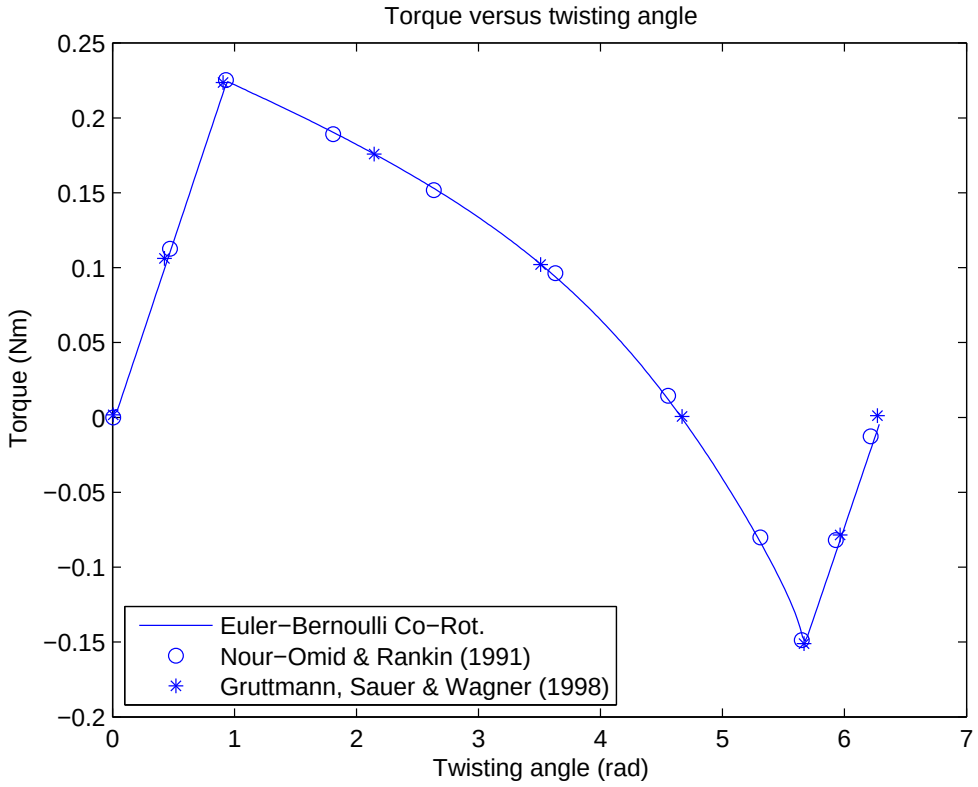


Figure 3.2: *Reaction torque as a function of twisting angle.*

During the first steps of the simulation, the cable shows a linear response and twists around its own axis without any deflection in the y- or z-direction. A critical point is reached when the twisting angle is roughly 1 radian. The cable is unstable at this point and bifurcation to a secondary path is possible. A small perturbation was added to the system by twisting the left end 10^{-4} radians around the y-axis. This small perturbation was enough to initiate buckling, which results in large deflection in the y- and z-direction. The applied torque decreases during this process and the cable deforms into a helical shape when the left end of the cable is pulled to the right. The left end continues to move to the right, which results in a gradual change of the shape of the cable from a helical form to a circle. Figure (3.3) shows the shape of the cable at different stages of this deformation process. The circular configuration is the second critical point of the cable. From this point, with negative torque, the cable untwists into a circular configuration free of torsion. This is exactly the behavior predicted by Nour-Omid and Rankin [25]. Figure (3.2) shows the reaction torque as a function of the twisting angle. The values predicted by Nour-Omid and Rankin as well as Gruttmann, Sauer and Wagner are also shown for comparison. The agreement is excellent.

This problem is a good test of the models capability of handling geometrical nonlinearities. Especially, this example shows that the model is capable of handling coupling between bending and torsion.

300 time steps were used in the simulation, where the left end of the cable was twisted $\frac{2\pi}{300}$ radians each time step. The simulation time necessary for this problem was 1.9 s on a single thread of an Intel Core 2 Quad @ 2.40 GHz with 8 GB of RAM.

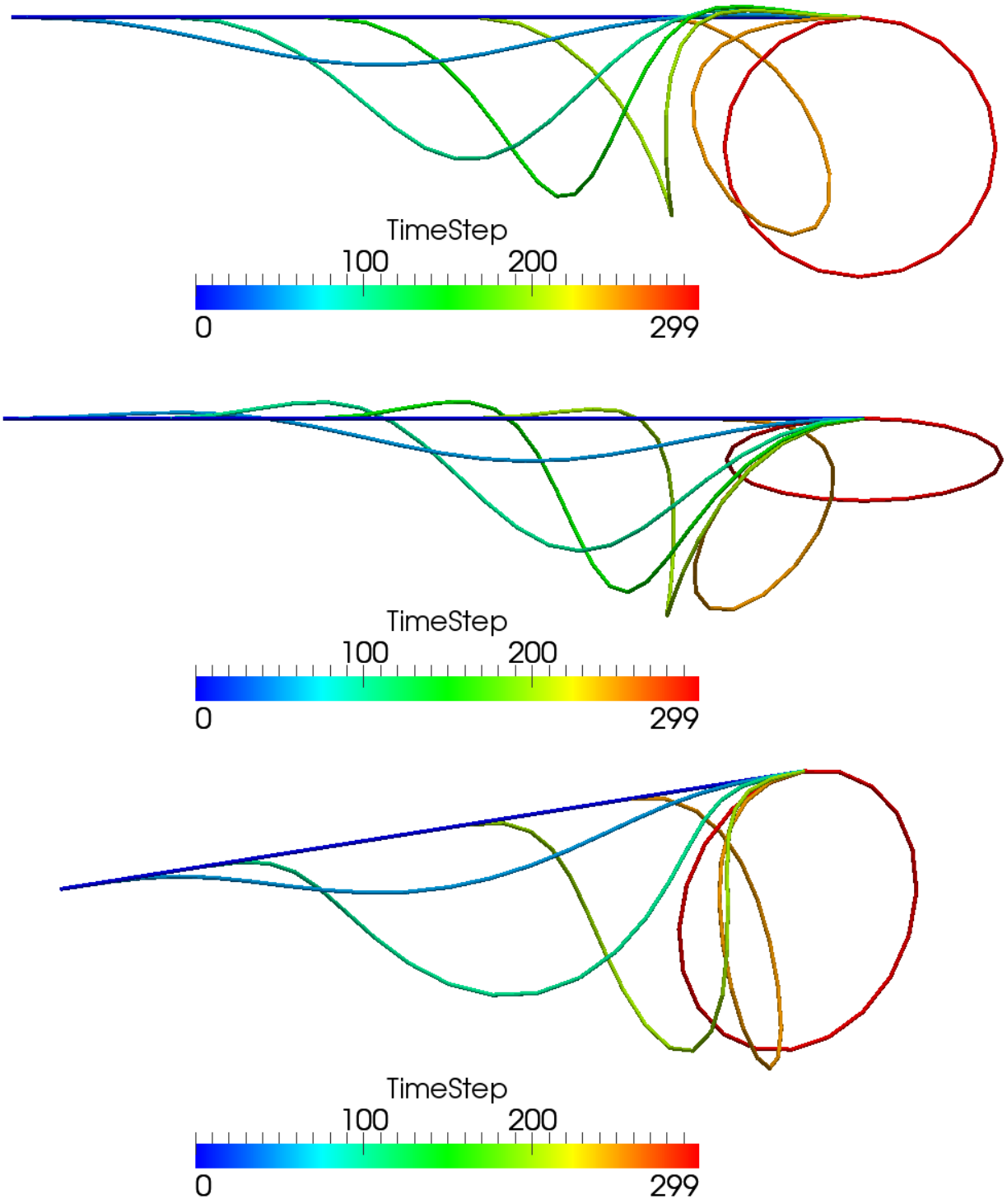


Figure 3.3: *Deformed shapes of a hockling cable: view in the positive y-direction (top), view in the negative z-direction (center) and perspective view (bottom).*

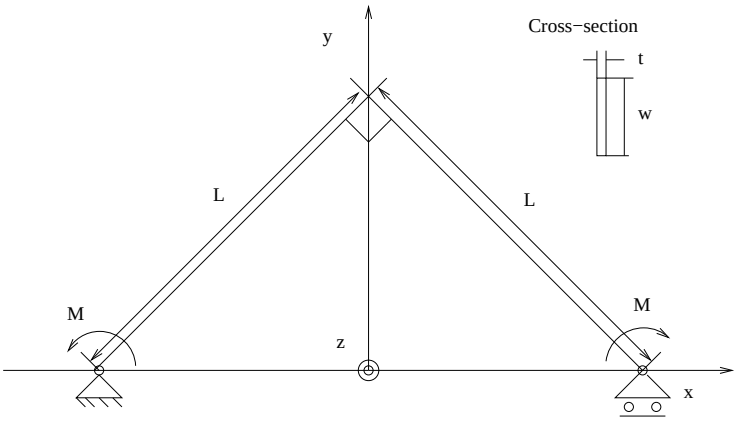


Figure 3.4: *Geometry of the hinged frame studied in the second static test case.*

3.1.1.2 Lateral buckling of a hinged frame

The second static test case is concerned with lateral buckling of a hinged, right angled frame. This problem was introduced by Argyris et al. [9] and has later been analyzed by several authors. Nour-Omid and Rankin [25] studied the problem with their co-rotational beam model, which was also implemented in the present work, while Simo and Vu-Qouc [28] applied their finite strain beam model. Gruttman, Sauer and Wagner [11] used a beam model with Timoshenko kinematics to analyze the problem.

The geometry of the frame is shown in figure (3.4). As can be seen in the figure, the structure is symmetric about the y-z plane. This symmetry can be exploited by only modelling the right half of the frame and prescribing symmetry about the y-z plane as a boundary condition for the node in the elbow. The support is allowed to translate in the x-direction and rotate about the z-axis. The half frame is modelled with ten equal EB-beam elements with cubic base functions. Nour-Omid and Rankin also used ten EB-beam elements, while Simo and Vu-Quoc used ten beam elements with quadratic base functions. Gruttmann et al. discretized the structure with 10 three-noded beam elements. In the present work, displacement control was used to control the simulation and a small perturbation load was added to initiate buckling.

A bending moment about the z-axis is applied at the support as indicated in figure (3.4). This bending moment is applied in such a way that the support is twisted one revolution about the z-axis. The initial response is linear, but a critical point is reached when the bending moment is $M \approx 0.62\ Nm$. At this point, the frame buckles with a large displacement in the out-of-plane direction. The frame rotates two full revolutions out of plane as the support is twisted one revolution about the z-axis. As noted is [11], the arc length method must be employed to follow the second revolution. The arc length method has not been implemented in the present work and therefore only the first revolution could be simulated. The reaction moment as a function of the twisting angle is shown in figure (3.6) and the trajectory traced by the apex of the frame in the y-z-plane is shown in figure (3.7). The agreement with [25] and [11] is very good. It is interesting to note that the reference solutions are based on different strain measures: the present work and [25] assume small strain, while [28] use the deformation gradient and [11] use the Green-Lagrange strain.

Figure (3.8) shows the shape of the centerline of the frame as well as the deformed frame surface at different stages of the deformation process. Note the large three-dimensional rotations and the extreme aspect ratio of the cross section, which makes this problem very challenging. The simulation time necessary for this problem was 2.0 s on a single thread of an Intel Core 2 Quad @ 2.40 GHz with 8 GB of RAM.

Figure 3.5: *Perspective view of the geometry of the hinged frame studied in the second static test case. Note the extreme aspect ratio of the cross section.*

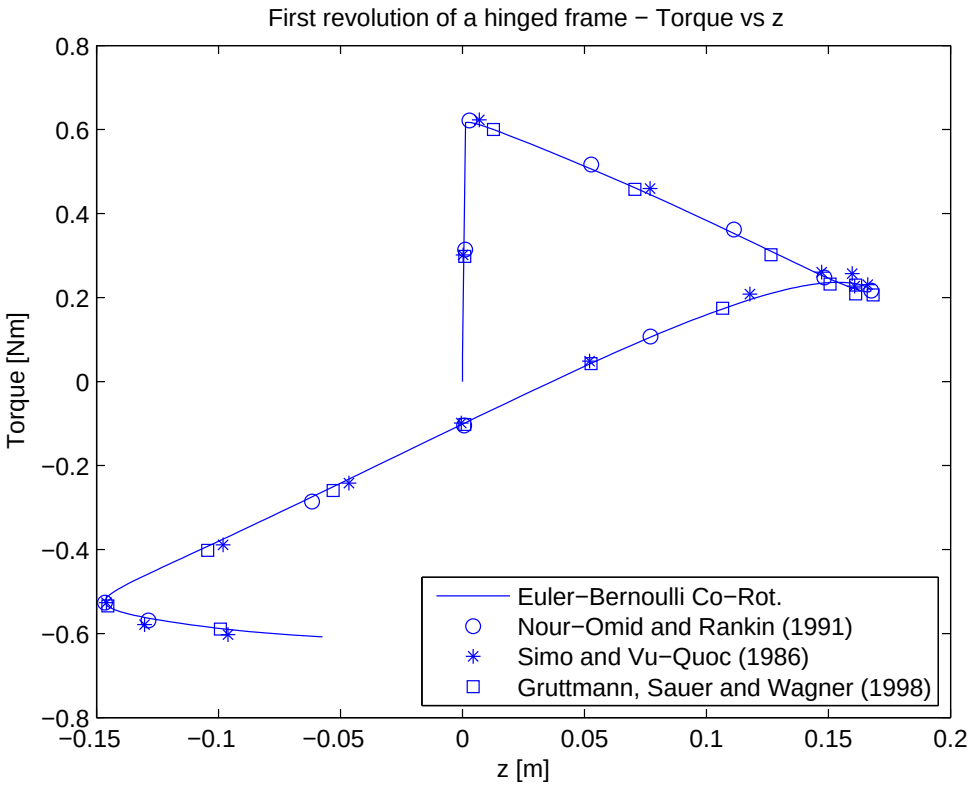


Figure 3.6: *Reaction torque as a function of twisting angle.*

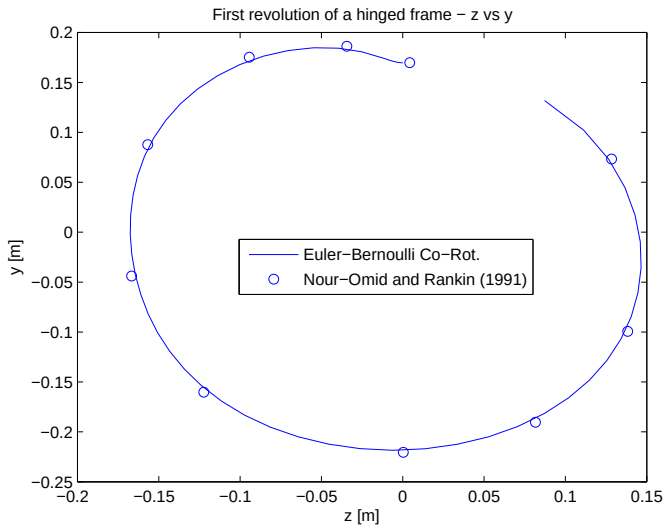


Figure 3.7: *Trajectory of the frame apex in the y-z-plane.*

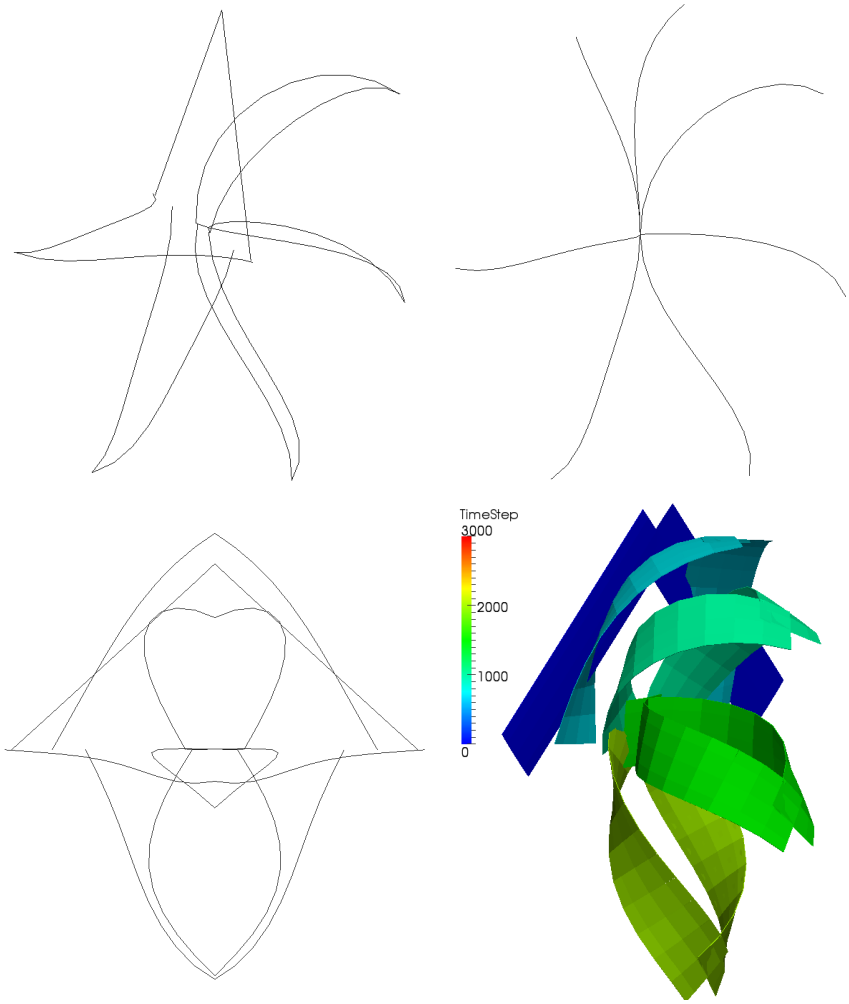


Figure 3.8: *Deformed shapes of the frame: perspective view (top left), view in negative x-direction (top right), view in negative z-direction (bottom left) and perspective view of the deformed frame surface (bottom right). The "cut" in the frame surface at the elbow is an artifact from the visualization routine.*

3.1.2 Validation of dynamic problems

3.1.2.1 Free vibration of right-angle cantilever beam

The first dynamic test case hase been studied by several authors, among others Simo and Vu-Quoc [29] and later Ibrahimbegović and Mikdad [17]. In this problem, the L-shaped cantilever beam shown in figure (3.9) is studied. A point force is applied in the out-of-plane direction at the elbow. Figure (3.9) shows the time history of the force, which has a maximum value of 50 N and is applied for two seconds. The geometry of the beam is defined in figure (3.9) and the properties of the cross section are:

- $GA_y = GA_z = EA = 10^6$
- $EI_y = EI_z = GJ = 10^3$
- $A\rho = 1$
- $I\rho_x = 20$
- $I\rho_y = I\rho_z = 10$

It should be noted that in the present work, a model based on the Euler-Bernoulli beam theory was used, which implicitly assumes $GA_y = GA_z = \infty$. As will be seen, this assumption does not have a noticeable effect on the results for this test case. The reason for this is most likely that the structure studied in this problem is indeed very slender and therefore the Euler-Bernoulli assumption is justified.

The beam is initially at rest and starts to deflect when the external force is applied. After the force has been removed, the structure performs free vibrations with large magnitude. As noted in [17], the amplitude of the vibration is of the same order of magnitude as the length of the structure. Therefore, this problem is a good test of the models capability of handling transient problems involving large rotations and displacements.

Figure (3.10) shows the z-displacement of the elbow as a function of time, while figure (3.11) shows the z-displacement of the tip. Values predicted by Ibrahimbegović and Mikdad [17] as well as by Simo and Vu-Quoc [29] are also shown for comparison. Note that the agreement is good even though different beam models were used in the reference solutions [17] and [29]. Ibrahimbegović and Mikdad used the Reissner beam theory while Simo and Vu-Quoc used a geometrically exact finite strain rod model. The results shown in figure (3.10) and (3.11) were obtained with the same time step size as used by Simo and Vu-Quoc: $\Delta t = 0.25\ s$, but 20 elements were used for the spatial resolution. The solution presented by Simo and Vu-Quoc was obtained with ten elements with quadratic base functions, and in that study it was not investigated whether the solution obtained was grid-independent or not. This lack of grid-independency is probably the reason for the discrepancy observed for the deflection of the elbow at $t \approx 17\ s$. Here, the result from the present study agrees well with the results predicted by Ibrahimbegović and Mikdad while slightly different results are given by Simo and Vu-Quoc.

Figure (3.12) shows the temporal convergence of the solution and figure (3.13) shows the spatial convergence. An element size of $h = 2\ m$ corresponding to a mesh consisting of ten elements gives a fairly grid independent solution. However, note that $\Delta t = 0.25\ s$ does not give a solution independent of the time step size. With a time step size of $\Delta t = 0.25\ s$, the displacement of the tip shows a valley in the curve at $t \approx 24\ s$. This valley is levelled out when the time step is decreased.

Figure (3.14) shows the centerline of the deformed beam at different instants in time and the surface of the deformed beam is shown in figure (3.15). The simulation time necessary for this problem, with 40 elements and 480 time steps, was 3.9 s on a single thread of an Intel Core 2 Quad @ 2.40 GHz with 8 GB of RAM.

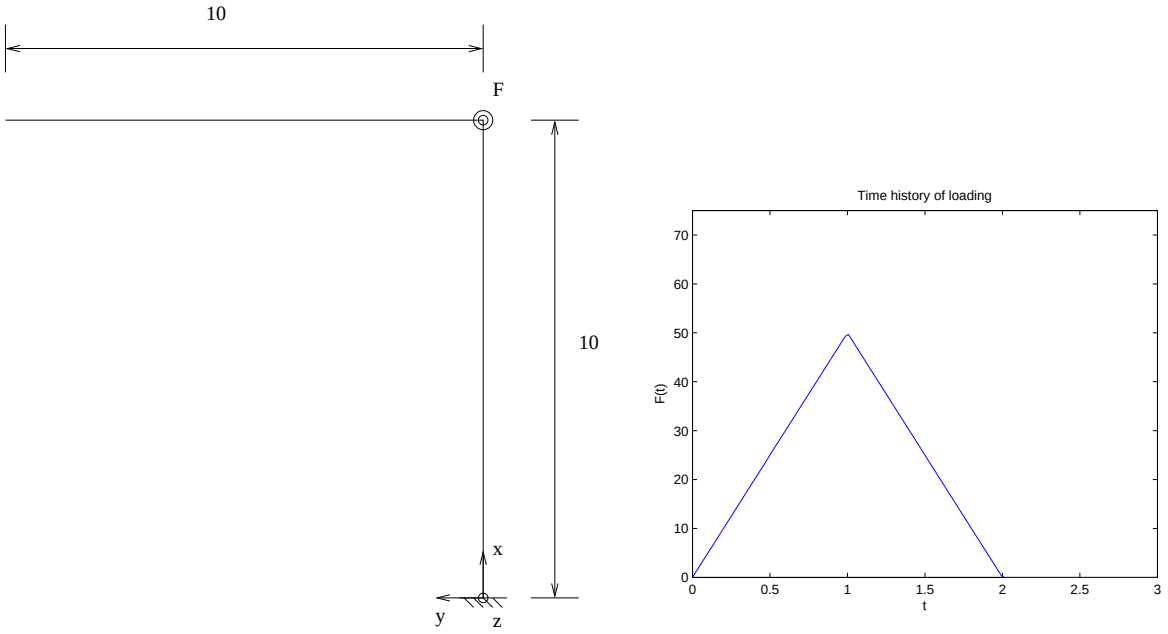


Figure 3.9: *Geometry of the L-shaped beam considered in the first dynamic test case (left) and the corresponding loading history (right).*

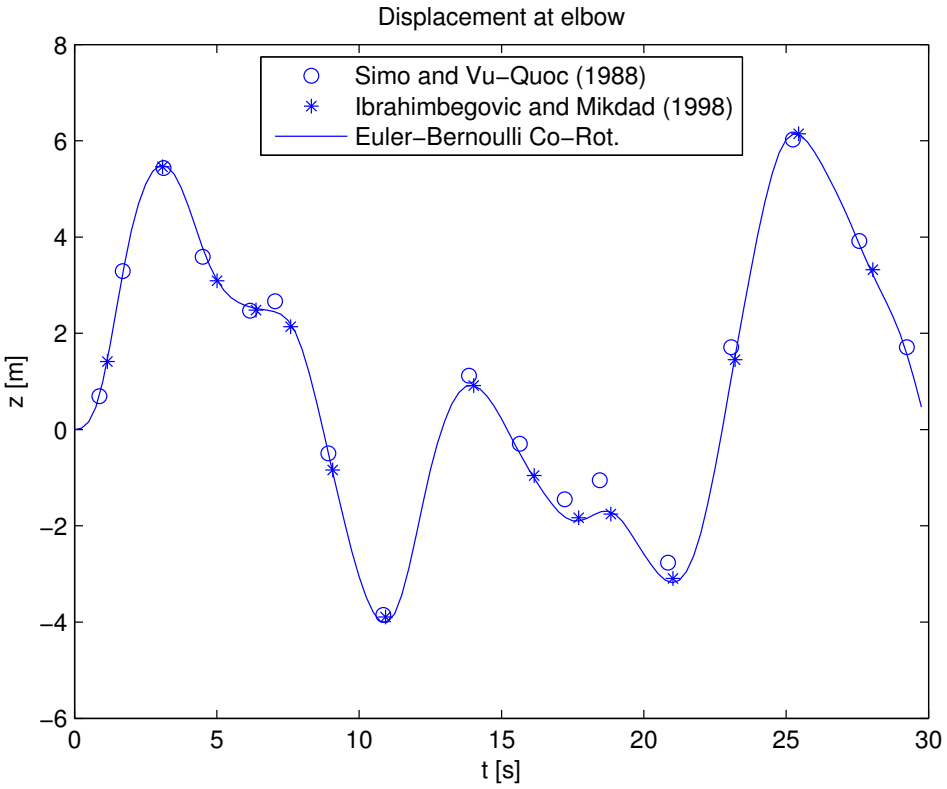


Figure 3.10: *Out-of-plane displacement of the elbow.*

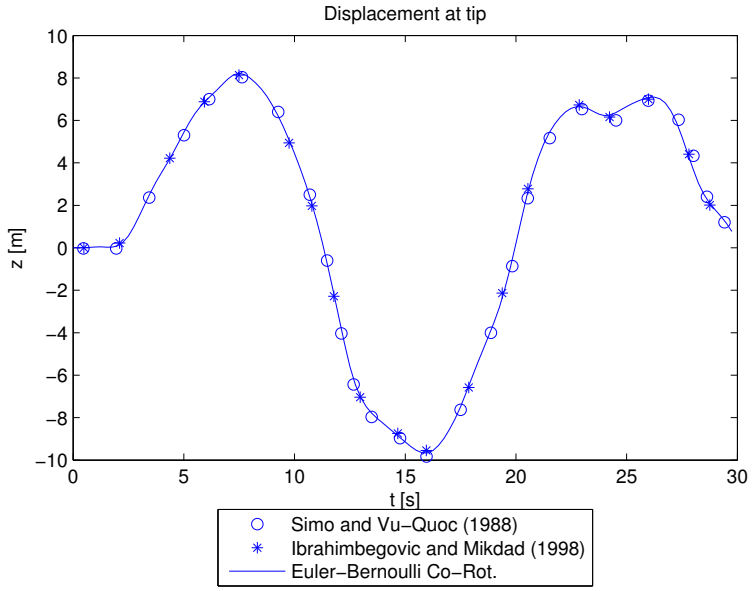


Figure 3.11: *Out-of-plane displacement of the tip.*

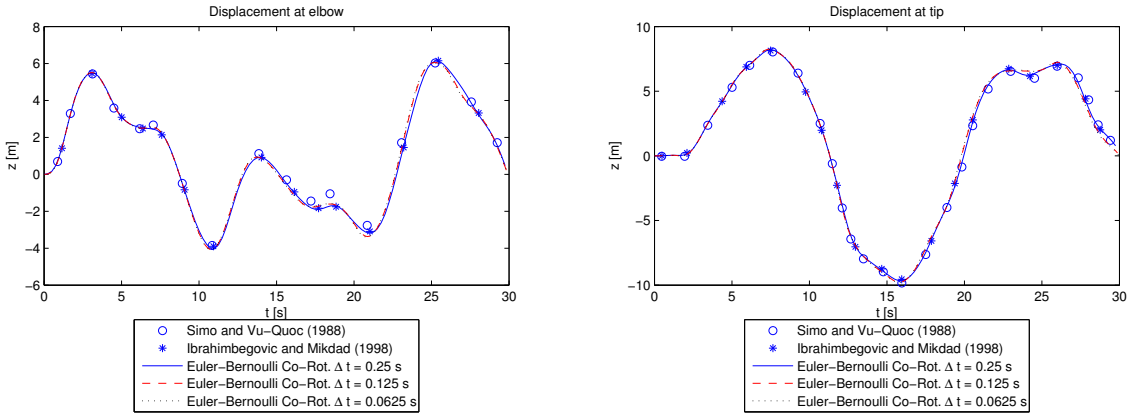


Figure 3.12: *Temporal convergence of the out-of-plane displacement of the elbow (left) and the tip (right).*

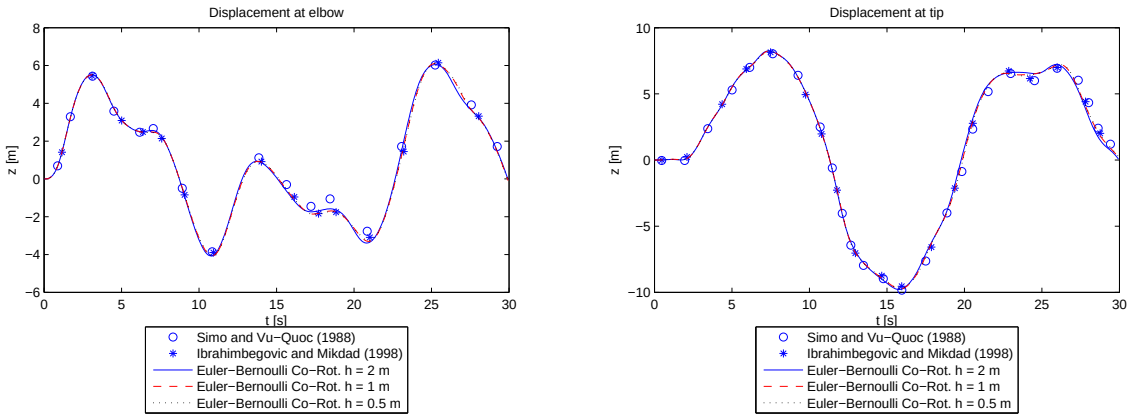


Figure 3.13: *Spatial convergence of the out-of-plane displacement of the elbow (left) and the tip (right).*

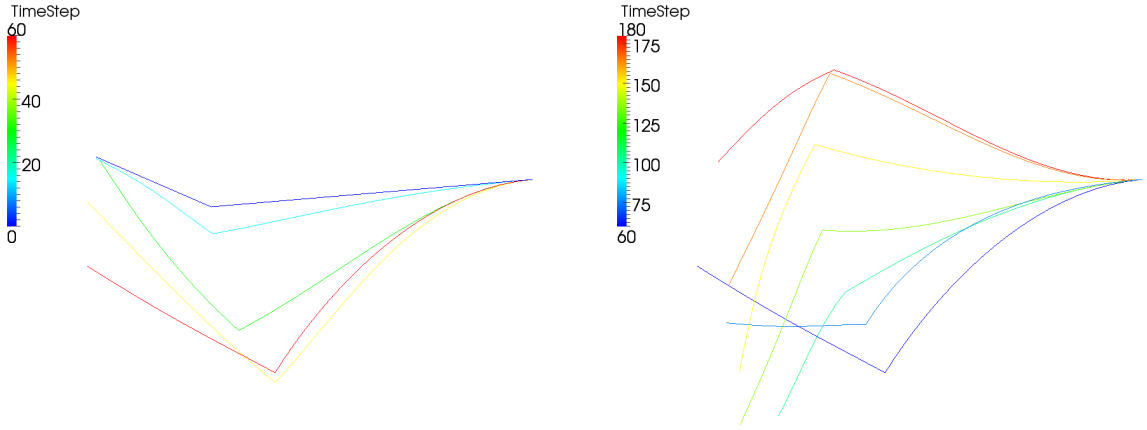


Figure 3.14: *Centerline of the deformed beam at different instants in time.*

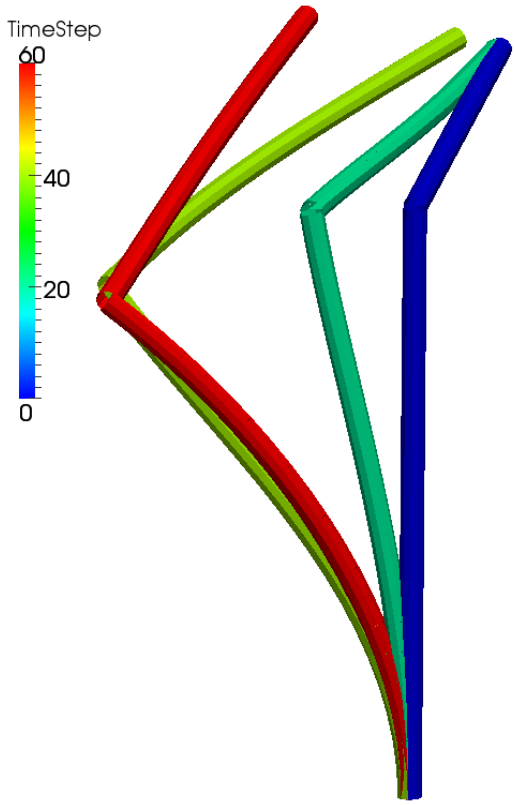


Figure 3.15: *Surface of the deformed beam at different instants in time.*

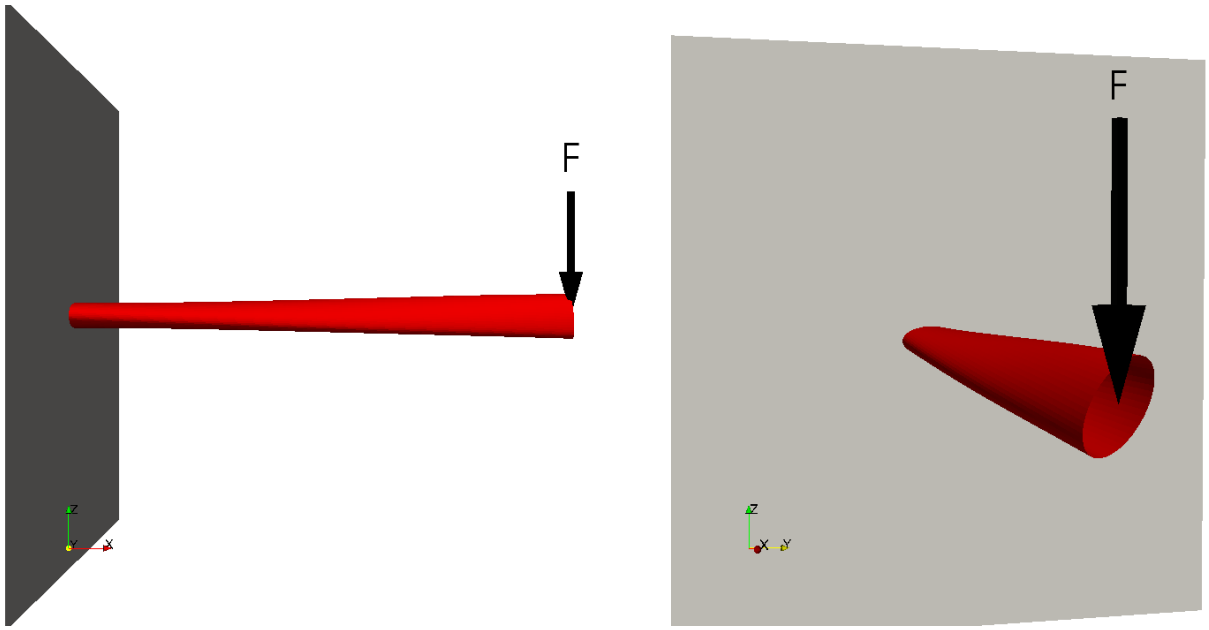


Figure 3.16: *Cantilever beam with twisted cross section.*

3.1.2.2 Vibration of a cantilever beam with twisted cross section

All the numerical examples studied above have a constant cross section. Therefore, it is necessary to validate the model for a case with varying cross section. For this purpose, the cantilever beam shown in figure (3.16) is studied. The beam is clamped at the left end, while a constant force in the negative z-direction is applied at the right end. The beam has an elliptical cross section and the major axis of the ellipse is aligned with the y-axis at $x = 0$. For $0 < x \leq L$, the major axis of the ellipse is not aligned with the y-axis: the angle between the global y-axis and the major axis of the ellipse is φ, see figure (3.17). $\varphi = 0$ at $x = 0$ and $\varphi = 60^o$ at $x = L$. The properties of the beam are as follows:

- Young's modulus: $E = 210\ GPa$
- Poisson's ratio: $\nu = 0.3$
- Length: $L = 0.400\ m$
- Major axis: $a = 0.020\ m$
- Minor axis: $b = 0.010\ m$
- Twisting angle: $\varphi = \left(\frac{x}{L}\right) \cdot \varphi_{tot},\ \varphi_{tot} = 60^o$

At $t = 0$, a constant load of $F_z = -50 \cdot 10^3\ N$ is applied at the right end. This load is applied during the whole simulation time of $t_{tot} = 30\ ms$. As a result, the beam oscillates with large amplitude as shown in figure (3.23).

For comparison, the same case was also simulated with the open source FE code Calculix [8]. The simulation was performed with the beam elements available in Calculix. These elements are implemented in such a way, that after they have been defined as beam elements, they are converted to three-dimensional 20-node brick elements in the simulation. Geometric non-linearities were included and the α-method with $\alpha = -0.05$ was used for the time stepping.

Figure (3.18) shows the tip displacement in x-, y- and z-direction predicted by Calculix and by the code developed in the present work. The agreement is very good. Note that

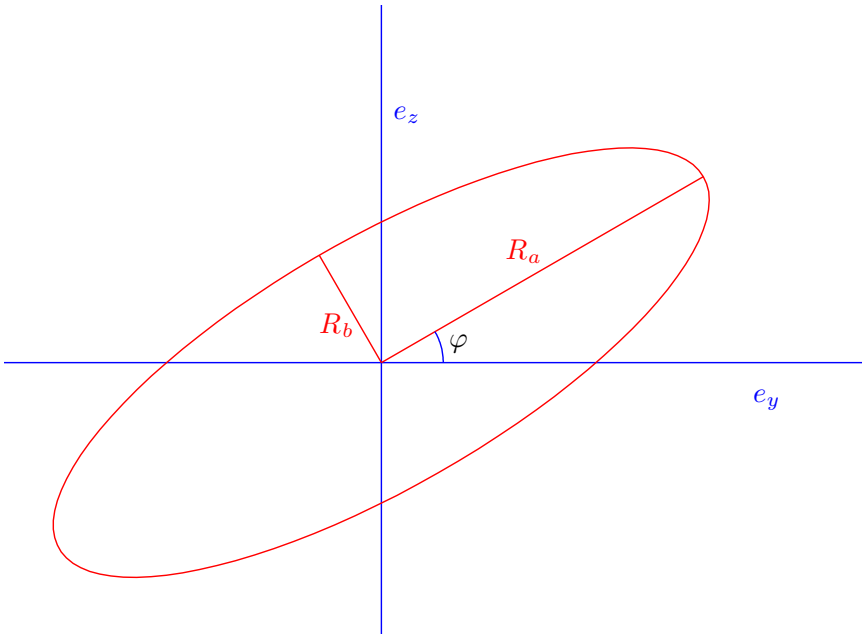


Figure 3.17: *Cross section of a fiber. The major radius R_a and the minor radius R_b are shown with red color. The local base vectors $\mathbf{e}_y$ and $\mathbf{e}_z$ are shown with blue color. The major axis of the ellipse is not aligned with the local $\mathbf{e}_y$-axis, the angle between them is φ.*

due to the twisted cross section, the force in z-direction gives rise to deflection in the y-direction.

The temporal and spatial convergence of the Calculix simulation are shown in figures (3.19) and (3.20). The temporal and spatial convergence of the code developed in the present work are shown in figures (3.21) and (3.22). It is interesting to note that the code developed in the present work performs better than Calculix on coarse grids. The reason for this is most likely that the code developed in the present work allows the cross section to vary over the element, while the simulation in Calculix assumed the cross section to be constant over an element. The simulation time necessary for the case with 80 elements and 30000 time steps was 4 m 5 s on a single thread of an Intel Core 2 Quad @ 2.40 GHz with 8 GB of RAM. The simulation time necessary for the same case with Calculix was 78 m 53 s on a single thread of an Intel Core i5 Duo @ 3.20 GHz with 8 GB of RAM.

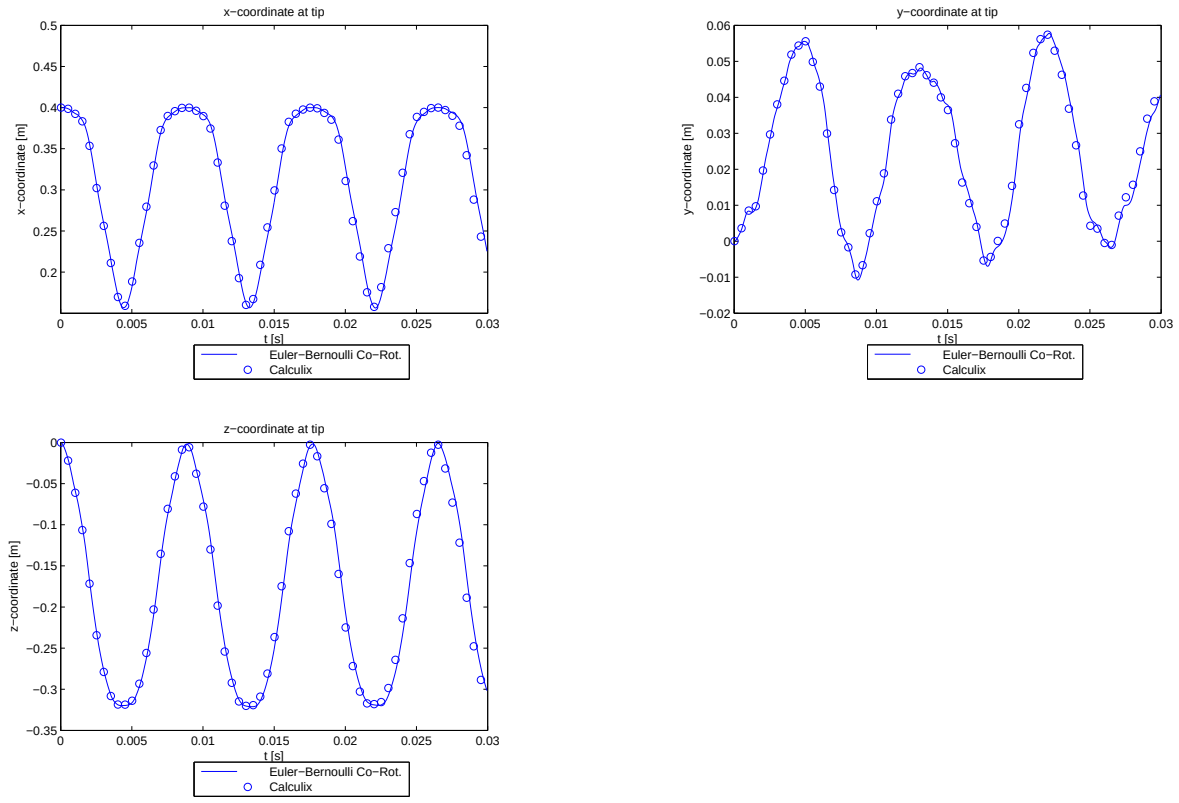


Figure 3.18: *Displacement versus time for a cantilever beam with twisted cross section. Comparison between Calculix and the code developed in the present work.*

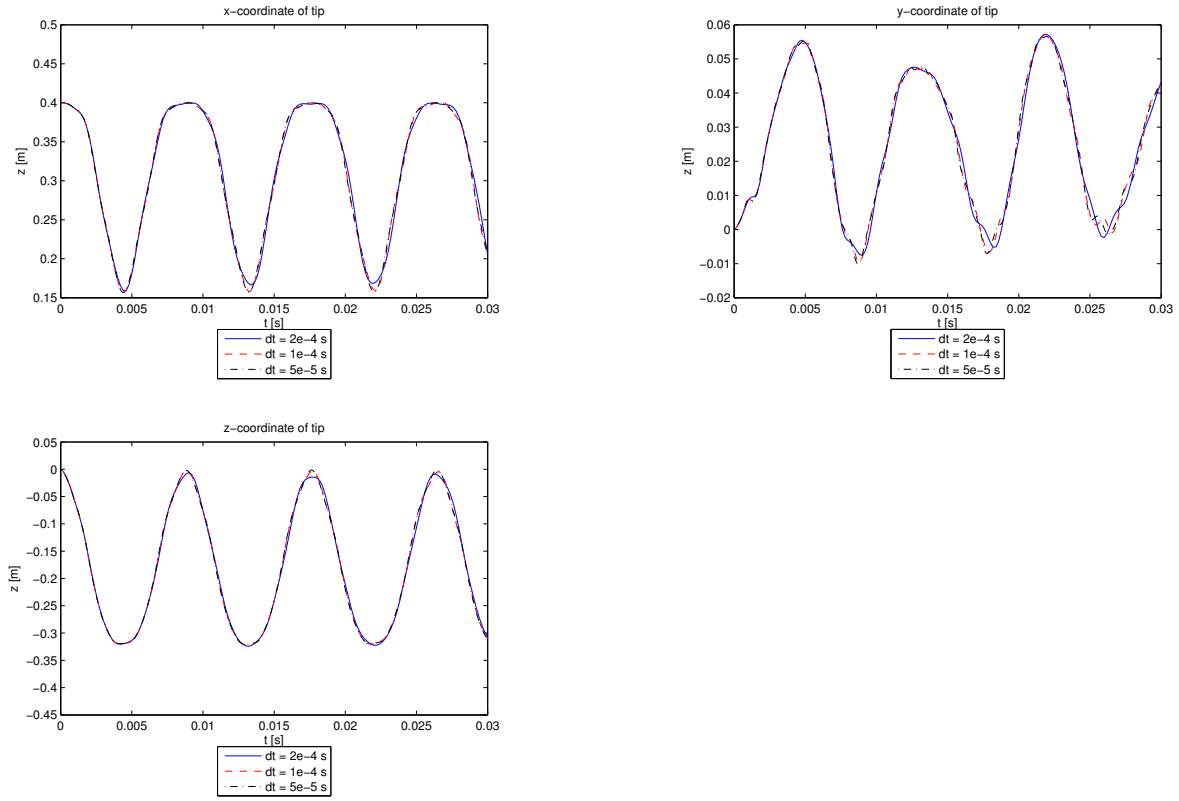


Figure 3.19: *Temporal convergence in Calculix.*

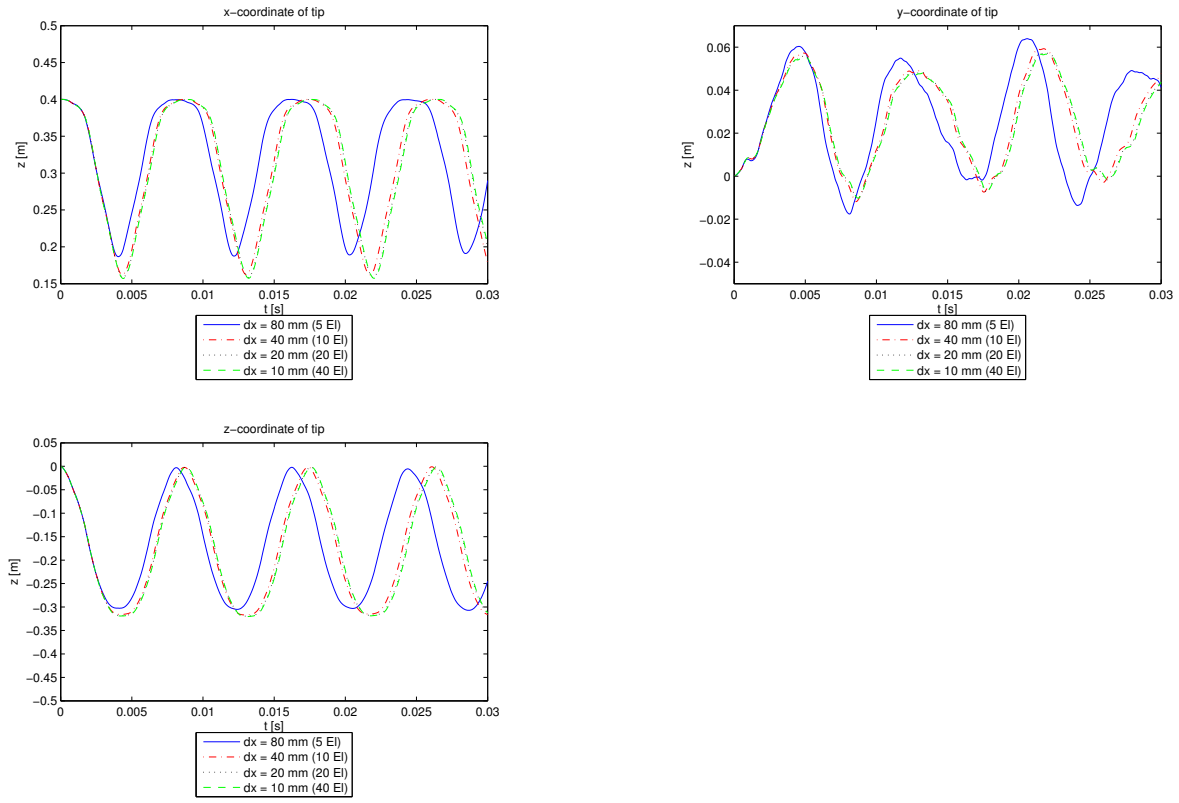


Figure 3.20: *Spatial convergence in Calculix.*

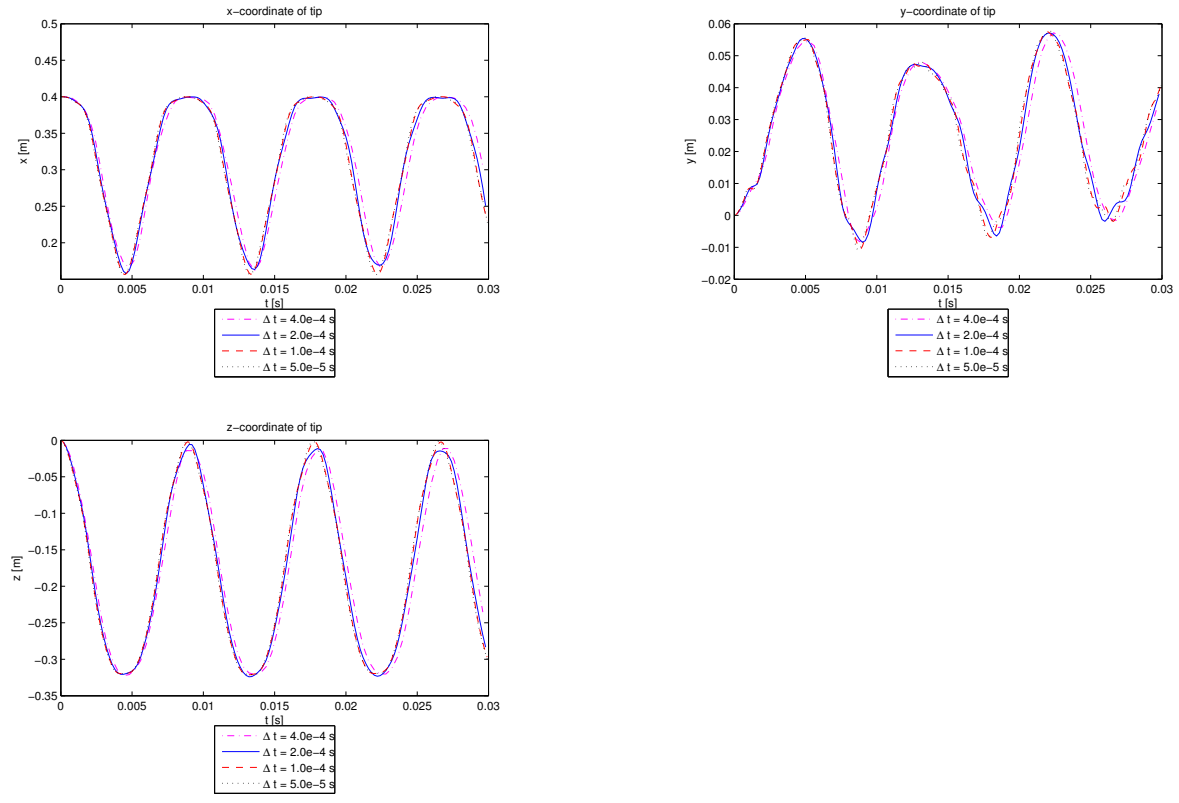


Figure 3.21: *Temporal convergence of the code developed in the present work.*

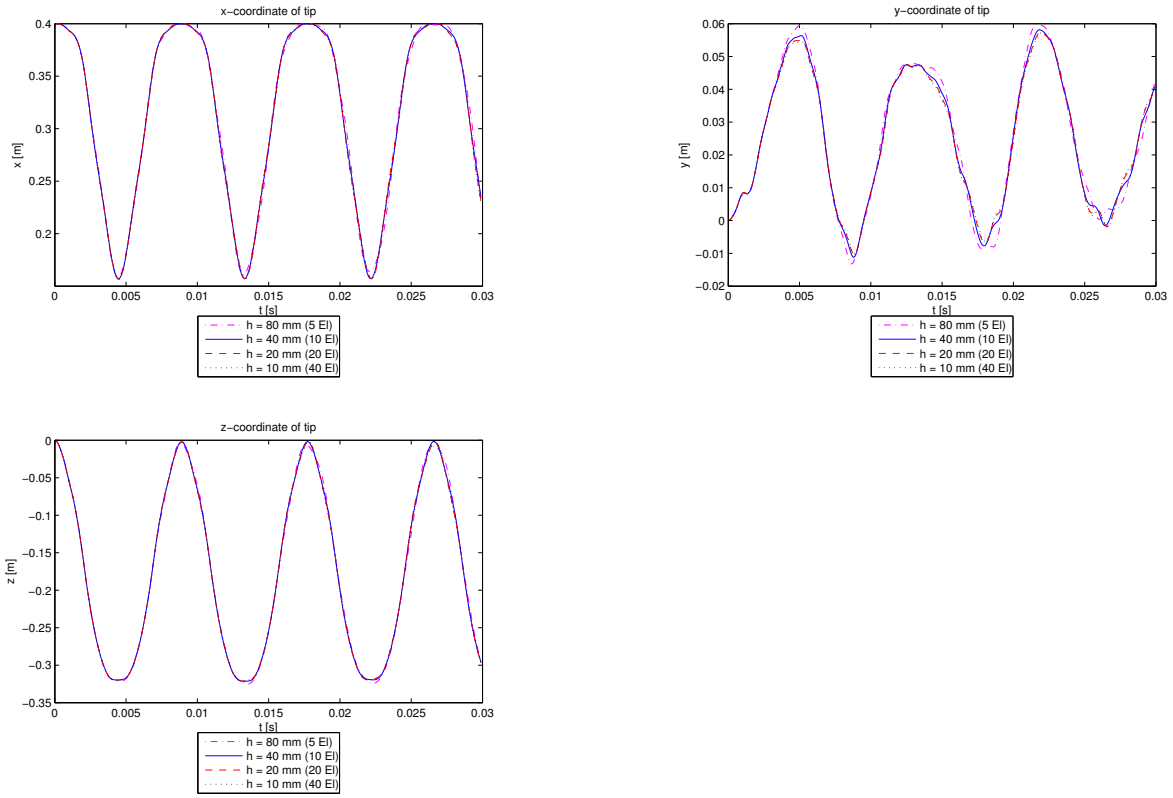


Figure 3.22: *Spatial convergence of the code developed in the present work.*

Figure 3.23: *Deformed shapes of cantilever beam with twisted elliptical cross section.*

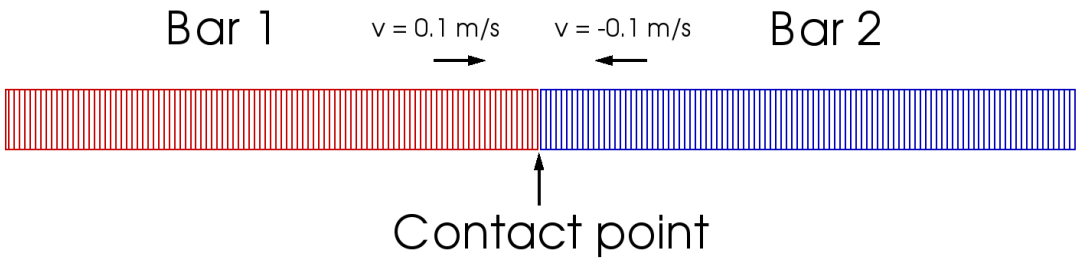


Figure 3.24: *Geometry of two impacting rods.*

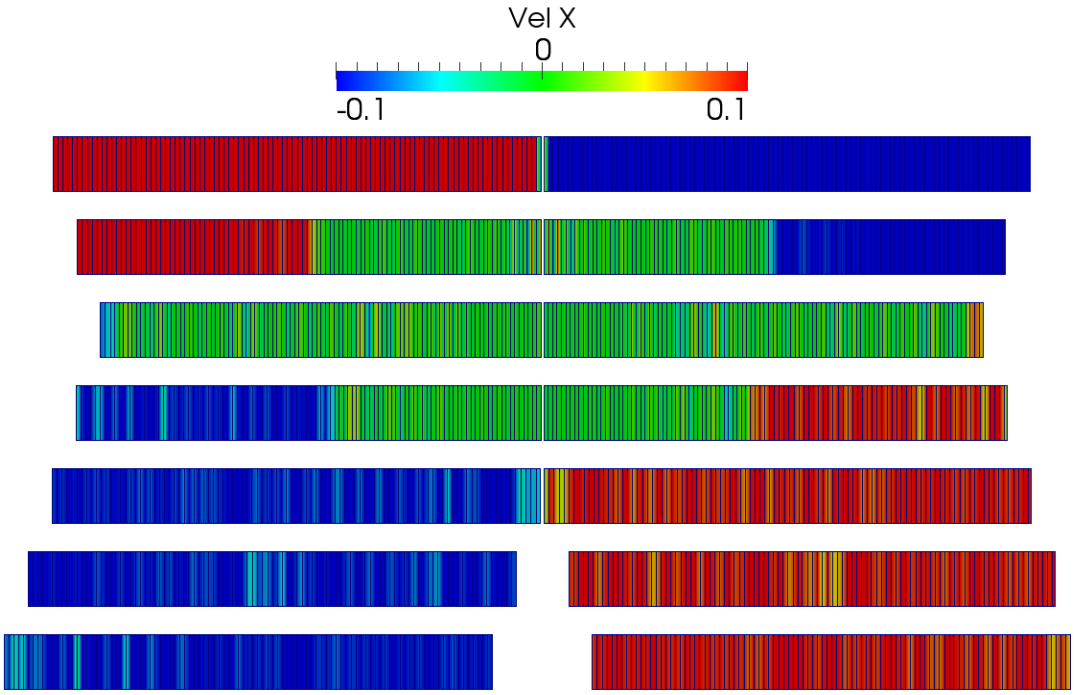


Figure 3.25: *Deformed shapes of two impacting rods.*

3.1.3 Validation of problems involving contact

3.1.3.1 Collision of two rods

In this example, which was previously studied by Cirak and West [4], one-dimensional impact of two rods is considered. The geometry of the two identical rods, meshed with 100 elements each, is shown in figure (3.24). Rod 1 has an initial velocity of $v_1 = 0.1\ m/s$ to the right and rod 2 has an initial velocity of $v_2 = 0.1\ m/s$ to the left. The rods are located next to each other so that impact occurs at $t = 0\ s$. The dimensions and material properties of the rods are as follows:

- Length $L = 10\ m$
- Young's modulus $E = 1\ Pa$
- Density $\rho = 1\ kg/m^3$
- Cross section area $A = 1\ m^2$
- Time step $\Delta t = 0.01\ s$

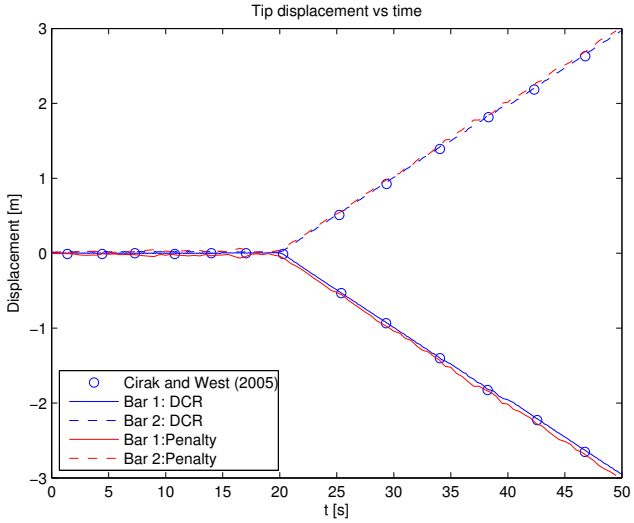


Figure 3.26: *Displacement of the impacting tips of the two rods.*

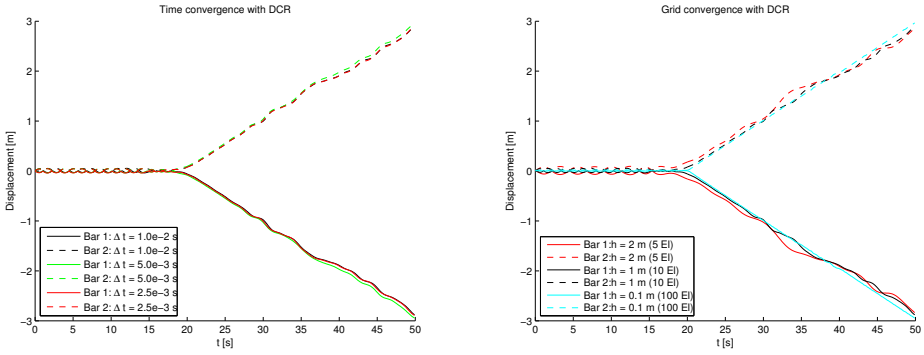


Figure 3.27: *Temporal (left) and spatial convergence (right) of impact simulation.*

Simulations were performed with the DCR method and the penalty method described in the theory section. Figure (3.25) shows the deformed shape of the rods at different instants in time. Substantial deformation occurs due to the low value of E and some high frequency oscillations can be seen. Figure (3.27) shows the displacement of the impact points of the two bars. The agreement with the reference solution in [4] is very good for the DCR method as well as for the penalty method implemented in the present work. The temporal and spatial convergence of the simulation are shown in figure (3.27). It can be concluded that the time step size of $\Delta t = 0.01\ s$ used by Cirak and West is sufficient to resolve the displacement of the points in contact. Furthermore, it can be noted that a coarser grid than 100 elements per bar is sufficient to capture the overall motion.

3.1.3.2 Impact between spinning rod and table

The last validation case is a spinning rod falling onto a table. This problem, which involves fricton and inelastic contact, was previously studied by Stewart and Trinkle [30]. The rod shown in figure (3.28) is released from a height of 1 m with an angle of 30^o to the horizontal. Initially, the center of mass of the rod has no translational velocity, but the initial angular velocity is 4 rad/s about the center of mass. Gravity causes the rod to fall downwards and impact the table located at $y = 0$. The table is rigid and the coefficient of friction for the contact between the rod and the table is $\mu_{fr} = 0.6$. The impacts are considered to be inelastic with $e_{cor} = 0$. The length of the rod excluding the rounded ends is $l = 0.5\ m$ and it has a radius of $r = 0.05\ m$. The mass of the rod is 1 kg and its moment of inertia with respect to the center of gravity is $J = 0.002\ kgm^2$. Stewart and Trinkle studied a rigid rod. The code developed in the present work deals with elastic problems and can not simulate perfectly rigid objects. Instead, the rod was meshed with one element and the material parameters of steel were used ($E = 210\ GPa,\ \nu = 0.3$). In this way, the rod will behave as almost rigid in the simulations.

The initial angular velocity causes the rod to rotate as it falls towards the table. It impacts the table with one of its ends and continues to rotate towards the table with its other end. Eventually the other end hits the table and the rod comes to rest. Figure (3.29) shows the centerline of the rod at different instants in time as it falls towards the table. Note that the rod has a radius of $r = 0.05\ m$ and therefore the centerline of the rod stops at a distance of 0.05 m from the table.

The problem has been simulated with the penalty method and the DCR method implemented in the present work. When the penalty method was used, the penalty parameter was set to $K = 1.0 \cdot 10^4\ N$. The reference length was set to $d_{ref} = 5 \cdot 10^{-2}\ m$ and the exponent was set to $\zeta = 1.0$ resulting in a linear penalty force. Figure (3.30) shows a comparison with the results given by Stewart and Trinkle [30]. The agreement between the penalty method and the reference solution is good. Note that this good agreement was achieved even though a regularization of Coulomb's law was used in this simulation. The agreement between the DCR method and the reference solution is fair, but som discrepancy can be seen between $t = 0.4\ s$ and $t = 0.6\ s$. Especially, note that the DCR method predicts a slightly too high vertical velocity after the first impact at $t = 0.4\ s$. The reason for this could be the explicit character of the DCR method: the closest point projection leads to an increase in internal energy, allowing for a light bounce-off when this internal energy is released as kinetic energy.

The temporal convergence of the simulations performed is shown in figure (3.31) for the penalty method and in figure (3.32) for the DCR method. A time step of $\Delta t = 62.5\ \mu s$ gives a reasonable resolution for both methods. This can be compared to the results given in [30], where a time step of 5 ms gave a solution independent of the time step length. The reason for the difference in necessary time step length is most likely that [30] used a rigid body code to study a rigid body problem, while the present work used an elasticity code to study the same rigid body problem.

The simulation time for this problem with a time step of $\Delta t = 31.25\ \mu s$ was 13.7 s with the penalty method. A time step of $\Delta t = 31.25\ \mu s$ corresponds to 32000 time steps and the simulation was performed on a single thread of an Intel Core 2 Quad @ 2.40 GHz with 8 GB of RAM. The simulation time needed by the impulse based method was 16.5 s with the same settings and the same computer.

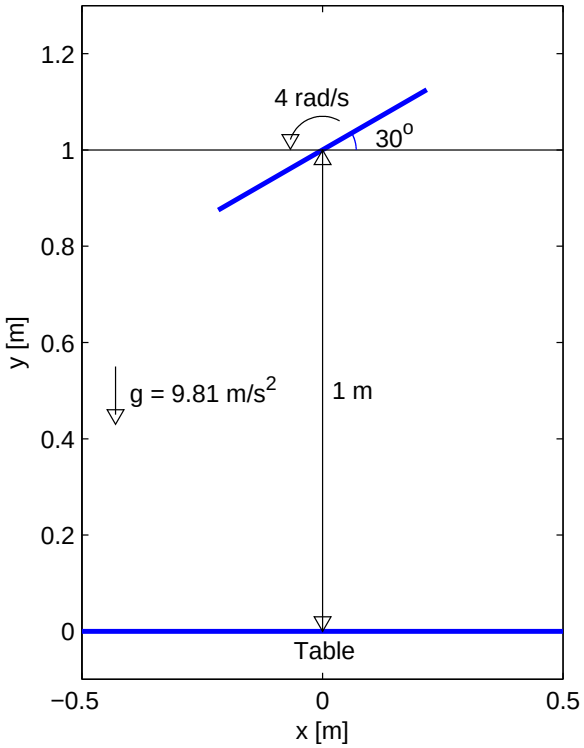


Figure 3.28: *Geometry of a rod impacting a table.*

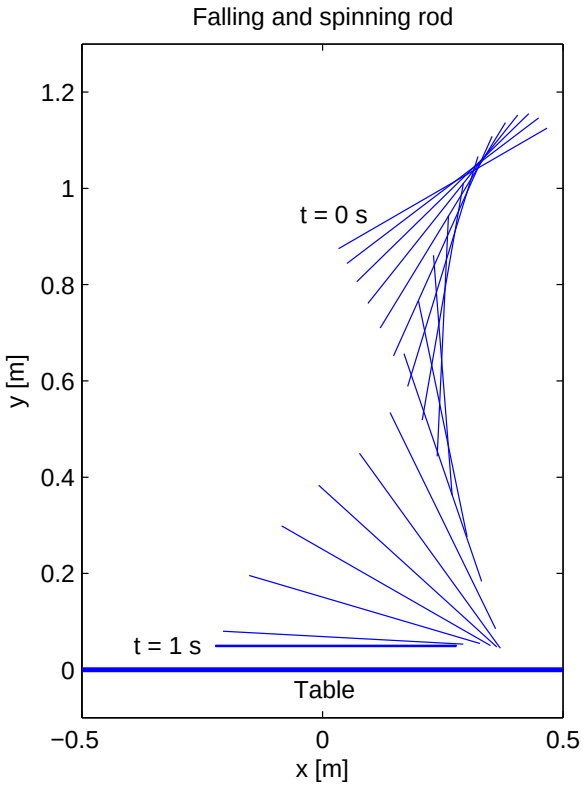


Figure 3.29: *A falling rod impacts a table. The figure shows the centerline of the rod.*

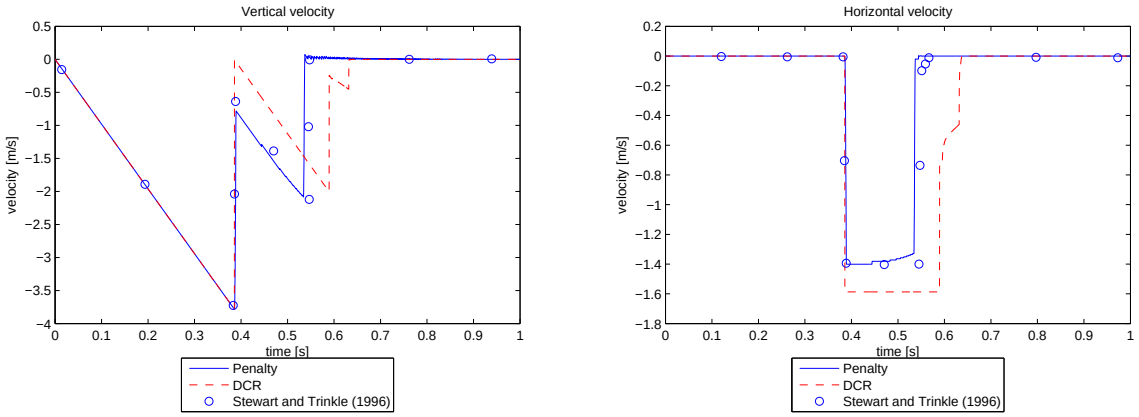


Figure 3.30: *Vertical (left) and horizontal (right) velocity of a rod impacting a table.*

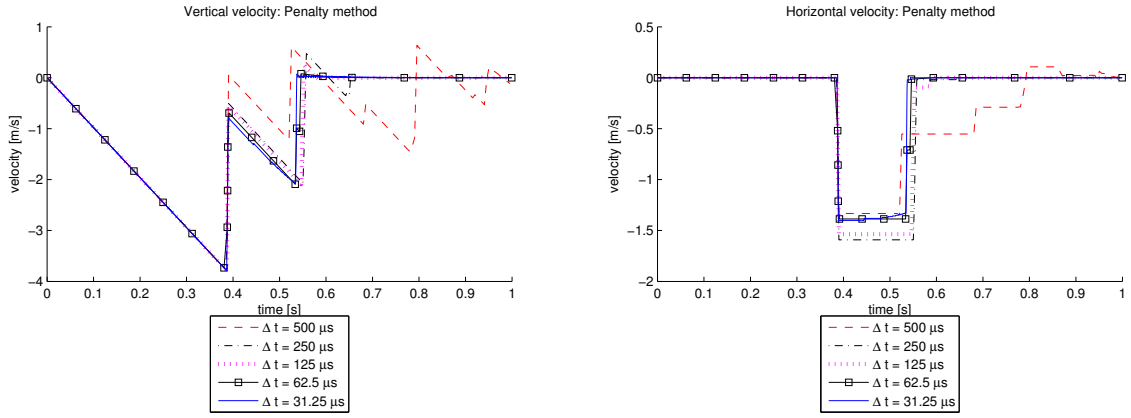


Figure 3.31: *Temporal convergence of vertical (left) and horizontal (right) velocity of a rod impacting a table. The simulations were performed with the penalty method.*

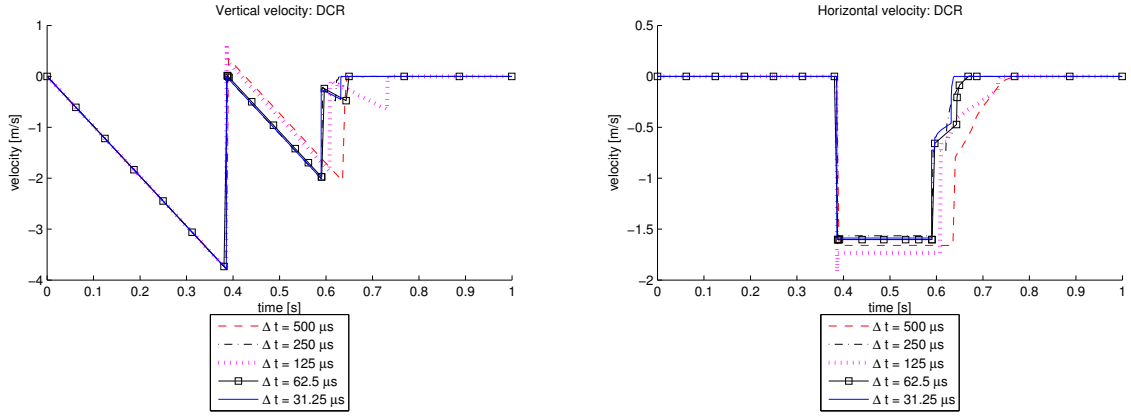


Figure 3.32: *Temporal convergence of vertical (left) and horizontal (right) velocity of a rod impacting a table. The simulations were performed with the DCR method.*

3.2 Fluid-Structure Interaction without contact

3.2.1 Oscillating cantilever beam in cross flow

The first FSI problem investigates the interaction between a fluid flow and a slender structure. An elastic beam is clamped at the wall in a domain filled with Newtonian, incompressible fluid. The beam is initially at rest, but the forces from the fluid cause the beam to deflect and start oscillating. These oscillations are gradually damped out by the fluid and a wake develops due to the presence of the beam. The problem has been studied with three methods: a DNS simulation with two-way coupling, a simulation with one-way coupling and a Fourier series expansion.

Figure (3.34) shows a perspective view of the fluid domain considered in this problem. The domain is hexahedral in shape and cubic cells are used for the discretization. z^+- and y^--views of the domain are shown in figure (3.33). The beam is surrounded by water with viscosity $\mu_f = 1.0 \cdot 10^{-3}\ Pas$ and density $\rho_f = 1000\ kg/m^3$. The fluid domain is bounded by its lower and upper corner:

- Domain lower corner: $(\ 0,\ 0,\ 0\ )\ mm$
- Domain upper corner: $(\ 0.5, 3.0, 1.0\ )\ mm$

A uniform grid is used for the initial discretization of the fluid domain. When the uniform base grid has been built, the desired number of refinements is added around the beam. Each refinement halves the cell size next to the beam and the refinement length was set to $0.2\ mm$, which corresponds to 10 times the diameter of the beam. The cubes in the base grid have a side length of $h_0 = 5.0 \cdot 10^{-5}\ m$. The bottom plot in figure (3.34) shows a slice through the fluid domain with a number of refinements around the beam.

The beam is made of a linear elastic material with Young's modulus $E = 1.0 \cdot 10^7\ Pa$ and Poisson's ratio $\nu_s = 0.3$. The density of the beam material is $\rho_s = 5000\ kg/m^3$. The beam has a circular cross section and its geometry is defined by the start point, the end point and the radius:

- Start point: $(\ 0,\ 1.0,\ 0.5\ )\ mm$
- End point: $(\ 0.15,\ 1.0,\ 0.5\ )\ mm$
- Radius: $r = 0.01\ mm$

15 elements are used for the discretization of the beam. The time step length was set to $\Delta t = 0.3125\ \mu s$, resulting in a CFL number of roughly 0.5 on the finest grid. The end time of the simulation was set to $t_{end} = 1\ ms$ and the inlet velocity was set to one meter per second: $v_{in} = (0,\ 1,\ 0)\ m/s$. The fluid boundary conditions are illustrated in figure (3.33). In summary, the following boundary conditions are imposed on the fluid:

- x_{bottom} : wall (no slip)
- x_{top} : symmetry
- y_{bottom} : inlet
- y_{top} : outlet
- z_{bottom} : symmetry
- z_{top} : symmetry

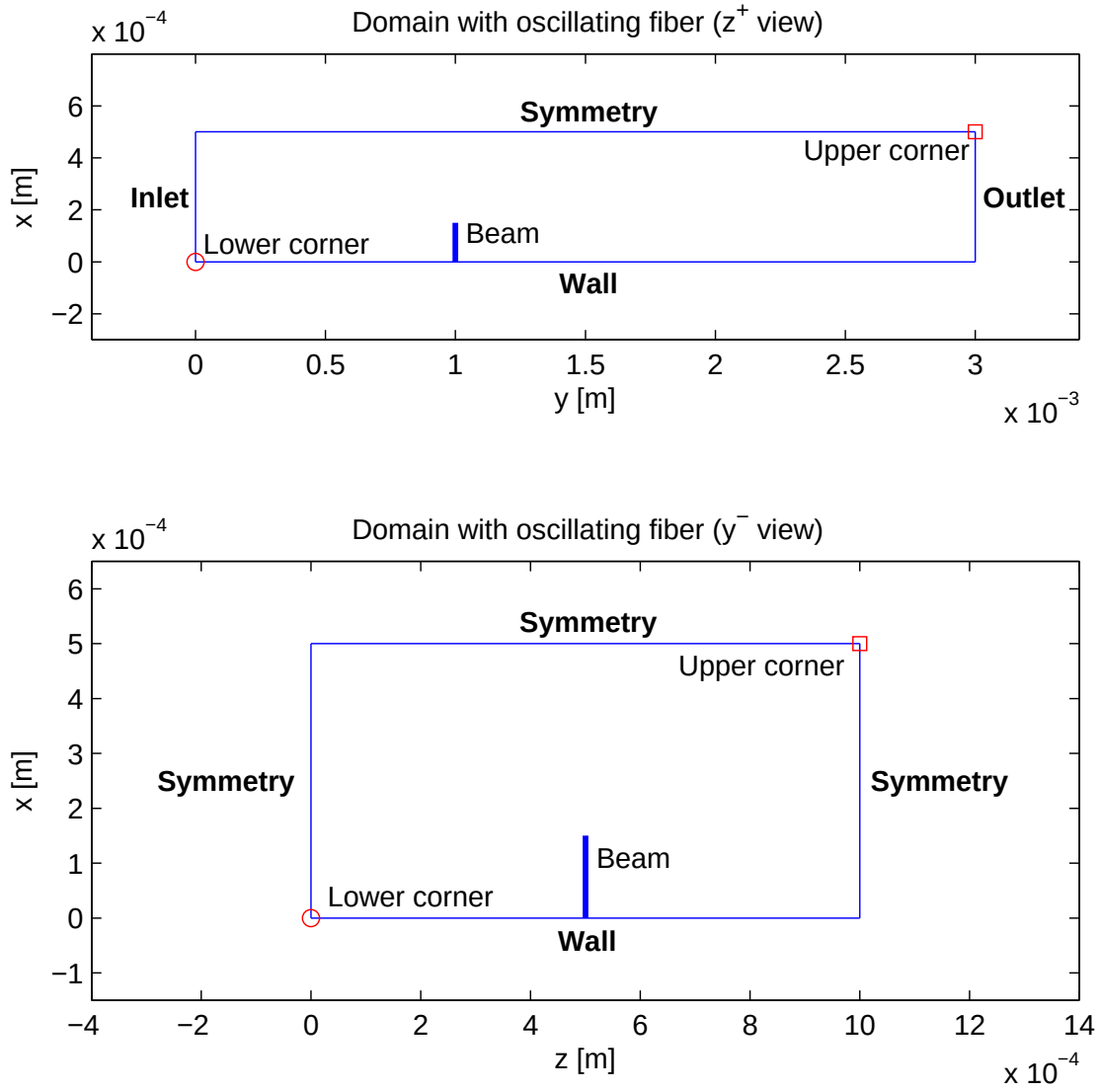


Figure 3.33: *Fluid domain seen in z^+-direction and y^--direction.*

Simulations were performed with several different grids and the deflection of the beam tip was monitored. The beam was kept fixed for the first $1.0 \cdot 10^{-4}$ s in all the simulations in order to allow the boundary layer around the beam to develop. Five different grids were used, ranging from 3 refinements and $2.8 \cdot 10^4$ cells to 6 grid refinements and $2.0 \cdot 10^5$ cells. These grids are described in table (3.1). The first order upwind scheme was used for the convective terms and the implicit Euler scheme was used for the temporal discretization in the fluid simulation. DNS simulations were performed, hence no turbulence model was used. Hilber's α-method with $\alpha = -0.05$ was used for the temporal discretization of the Finite Element equations describing the motion of the beam.

Figure (3.35) shows the velocity field in a slice through the center of the domain. The deflection of the beam as well as the gradual development of the wake can be seen. Note the effect on the wake when the beam springs back at $t = 330$ μs: the motion of the beam gives a thicker wake at the tip of the beam than at the center of the beam.

The deflection of the beam tip as a function of time predicted with the different grids is shown in figure (3.36). The results predicted with 3 and 4 refinements, corresponding to approximately 3 and 6 grid cells over the diameter of the beam, are clearly not grid independent. The differences between the finer grids are much smaller. 5 refinements,

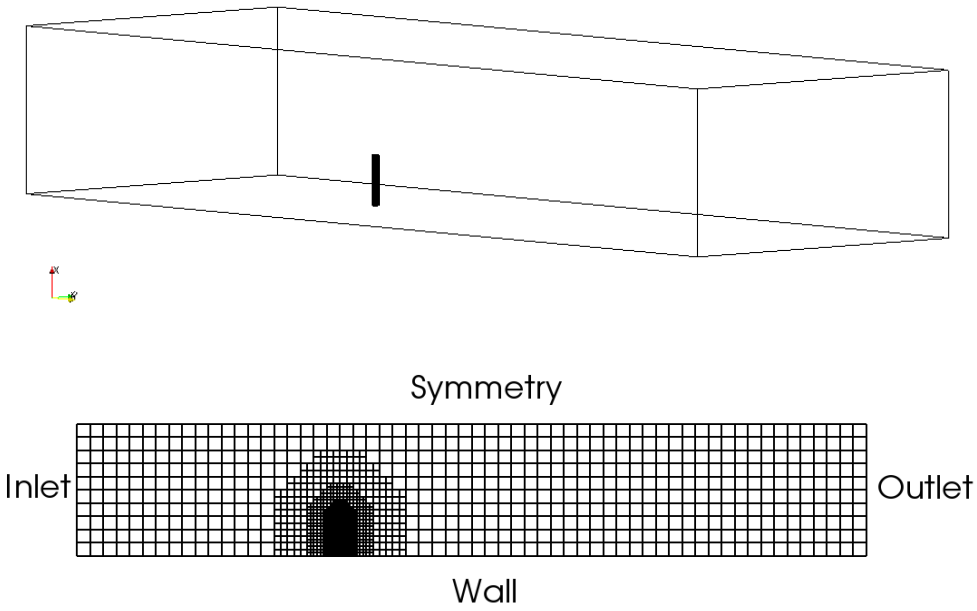


Figure 3.34: *Top: perspective view of fluid domain and beam. Bottom: Fluid grid with refinements seen in the z^+-direction.*

Table 3.1: *Fluid grids used in convergence study.*

Number of refinements	Number of cells	Min. cell size [m]
3	28212	$6.25 \cdot 10^{-6}$
4	42660	$3.125 \cdot 10^{-6}$
5	67356	$1.5625 \cdot 10^{-6}$
6	203156	$781.25 \cdot 10^{-9}$

corresponding to 13 cells over the diameter, gives a reasonably converged solution.

Figure (3.37) shows the temporal convergence for the case with 6 refinements. It can be concluded that the time step of $\Delta t = 0.3125$ μs used in the spatial convergence study is sufficiently grid independent. Even a time step of $\Delta t = 0.625$ μs corresponding to a CFL number of unity is sufficient. Therefore, the time step is in this case restricted by the CFL number rather than the need to resolve high frequency oscillations. The flow is almost uniform, which results in a quite uniform force on the beam. The effect of the uniform load is that almost all the energy transfered from the fluid to the beam enters the first eigenmode: oscillations of higher frequency are not visible to the naked eye in figure (3.36). The system response is therefore dominated by the lowest eigenfrequency.

The time history of the total force on the beam is shown in figure (3.38). The pressure part and the viscous part of the force are also given. A high force is predicted at the start of the simulation because the boundary layer around the beam has not developed at this stage. The beam experiences a rapidly decreasing force during the first time steps of the simulation, but the decrease starts to level out very soon. A new rapid decrease is noted at $t = 1 \cdot 10^{-4}$ s when the initially fixed beam is released. Two distinct oscillations in the force can be seen as the the beam oscillates, but the low frequency oscillations die out after $t \approx 6 \cdot 10^{-4}$ s and the force approaches the steady state value asymptotically.

The oscillation of a beam in a fluid was also studied with modal analysis. A Fourier series expansion of the Euler-Bernoulli beam equation was computed and the influence of

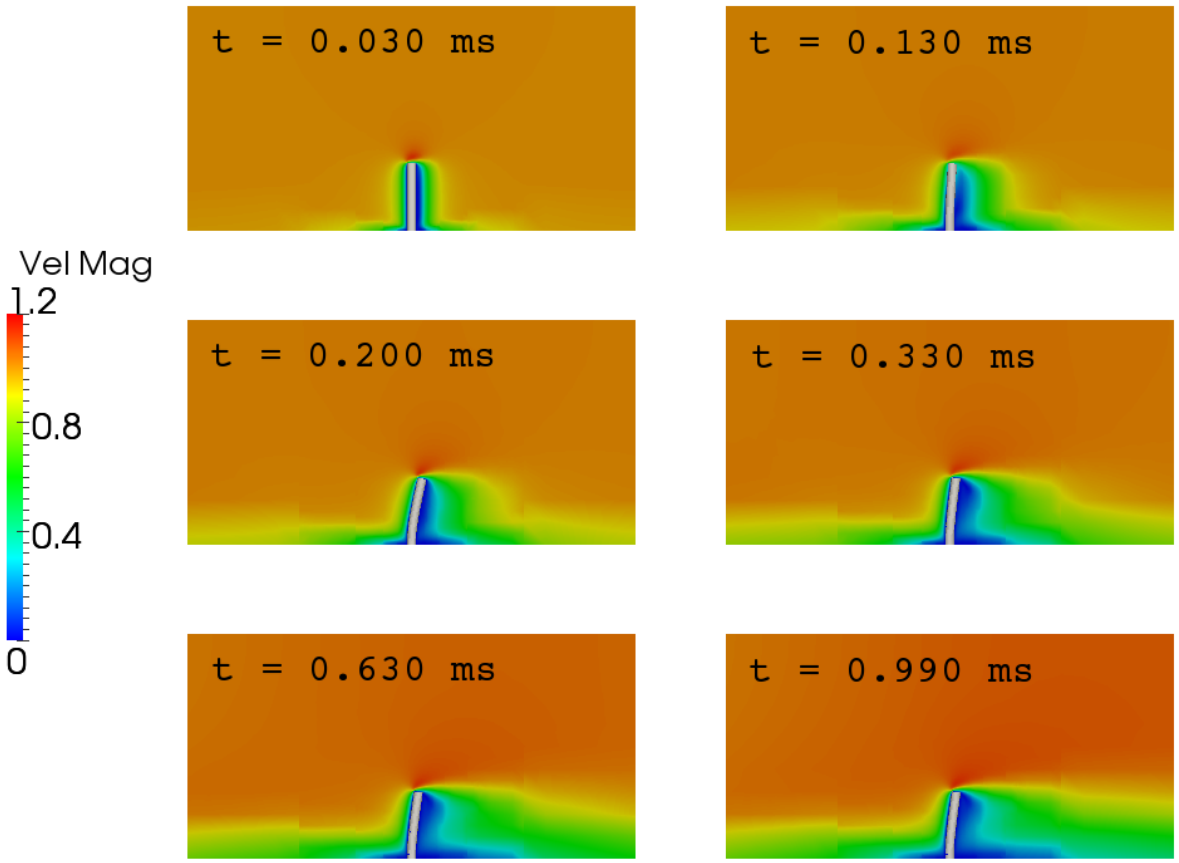


Figure 3.35: *Flow field around an oscillating beam.*

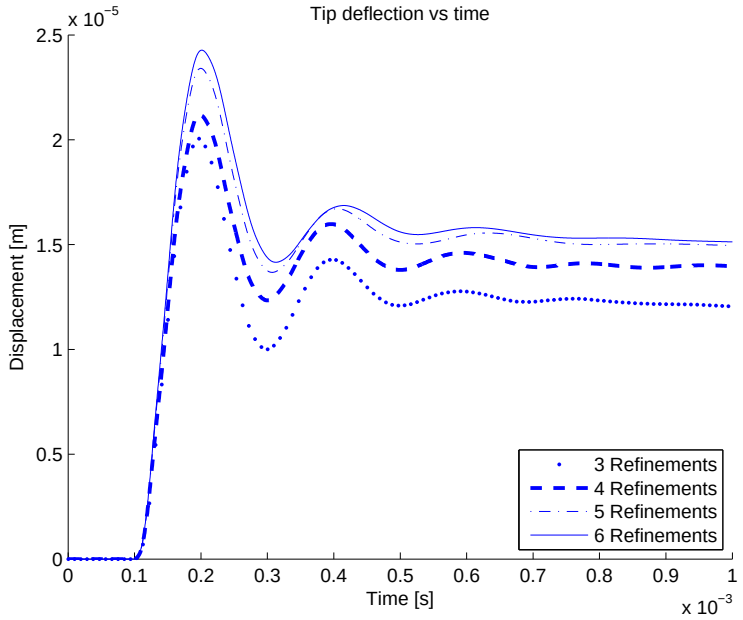


Figure 3.36: *Tip deflection in y-direction of a beam oscillating in a fluid.*

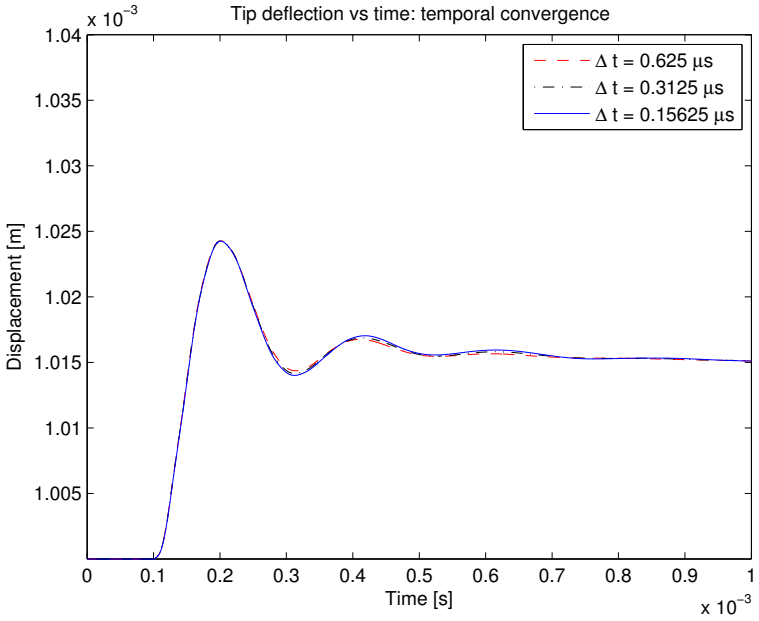


Figure 3.37: *Temporal convergence of a beam oscillating in a fluid.*

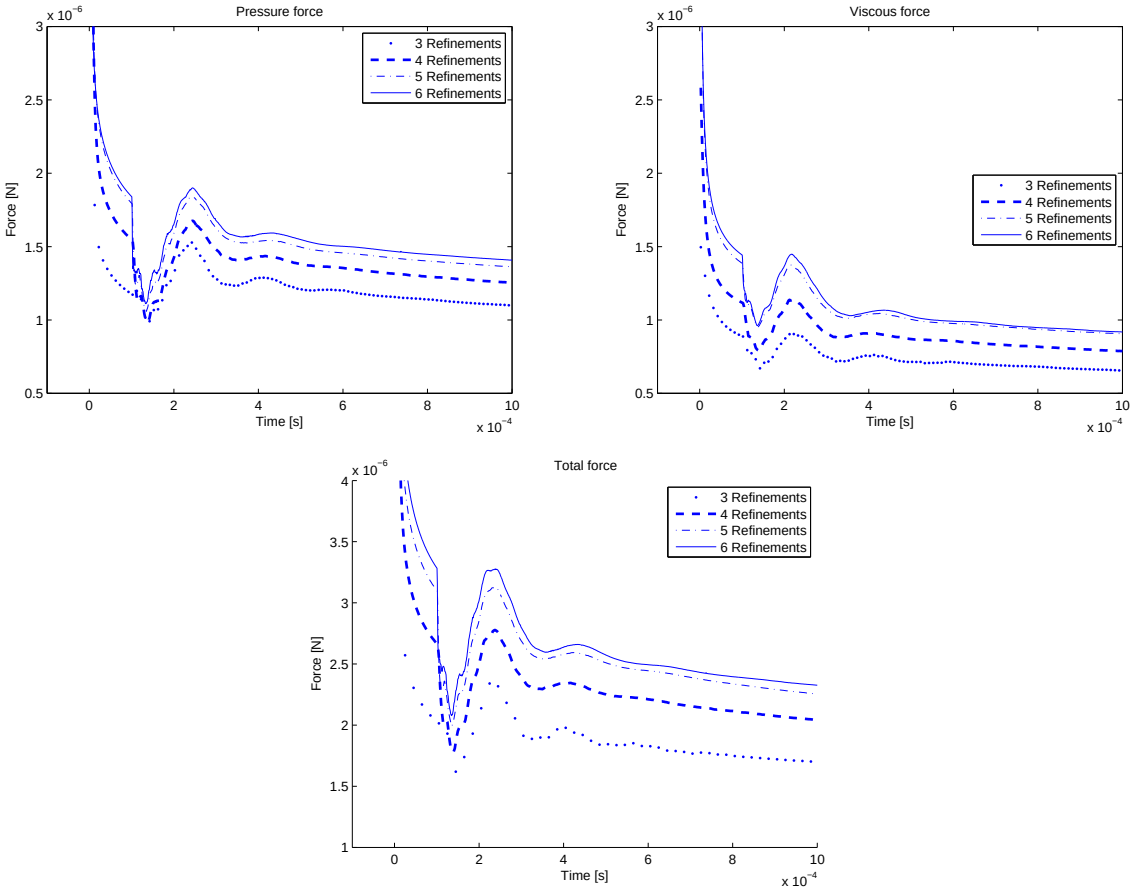


Figure 3.38: *Pressure force, viscous force and total force on a beam oscillating in a fluid.*

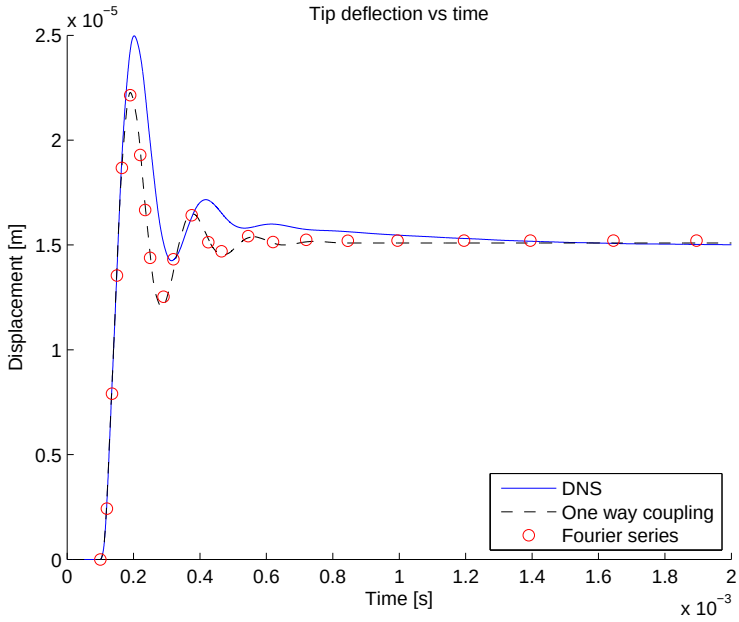


Figure 3.39: *Tip deflection in y-direction of a beam oscillating in a fluid: comparison of results obtained with DNS, simulation with one way coupling and Fourier series analysis.*

the surrounding fluid was included as a distributed load under the assumption of one-way coupling. The flow field was assumed to be uniform and a drag correlation for long cylinders was used to estimate the fluid force. The derivation of the Fourier series expansion is given in appendix C.

Figure (3.39) shows a comparison between the results obtained with the Fourier series analysis described above, the DNS simulation and the simulation with one-way coupling. The solution obtained from the Fourier series expansion is nearly identical to the simulation with one way coupling. This is expected, because both of these computations use the same assumption for the fluid force: one way coupling is used and it is assumed that the drag force is the only important force. The small difference between the Fourier series solution and the one-way coupling simulation can be attributed to the fact that the Fourier series expansion assumed a uniform flow field while the one-way coupling simulation sampled the fluid velocity from the simulated flow field. This simulated flow field is not perfectly uniform due to the effect of the wall, even though the boundary layer is thin compared to the length of the beam. The DNS simulation predicts an oscillation with larger amplitude, especially during the first period. The reason for this is most likely that the DNS simulation accounts for the history force and the added mass force, which were neglected in the simplified analysis. Furthermore, the DNS simulation predicts a slightly larger period time due to the strong coupling with the fluid. It is interesting to note that all three methods approach the same steady state value. This is also expected, because the simplified analysis included the stationary force but neglected transient forces. The differences between the DNS simulation and the simplified analysis are noticeable, but the error decreases when steady state is approached. This observation suggests that the DNS simulation could be used to improve the drag correlation. Since the discrepancies occur during the transient phase, the drag correlation could be extended by adding a transient term which depends on the relative acceleration. The DNS simulation could then be used to determine the coefficient in this term.

3.3 Simulations of paper forming

This section investigates the possibilities to simulate paper forming with the code developed in the present work. First, the problem setup is defined. Then, a simulation of paper forming with 50 fibers is described. Finally, the robustness of the fiber model is demonstrated by simulating a larger number of fibers under the simplification of one-way coupling.

3.3.1 Problem description

The simulation starts with a cloud of fibers randomly distributed in the fluid domain. As the simulation evolves, the fibers follow the fluid and fall down on a forming fabric in the lower part of the domain and start to form a fiber web. The progress of this process is shown in figure (3.40), where fibers suspended in the fluid gradually fall down on a forming fabric. Note the large number of contacts involved when the fibers pile up on the fabric. The load on the fibers in the web increases as more and more fibers lay down on top. This is stressful for the FE solver and puts a very severe test on the implementation of the contact model: any weakness in the implementation will cause rapid divergence.

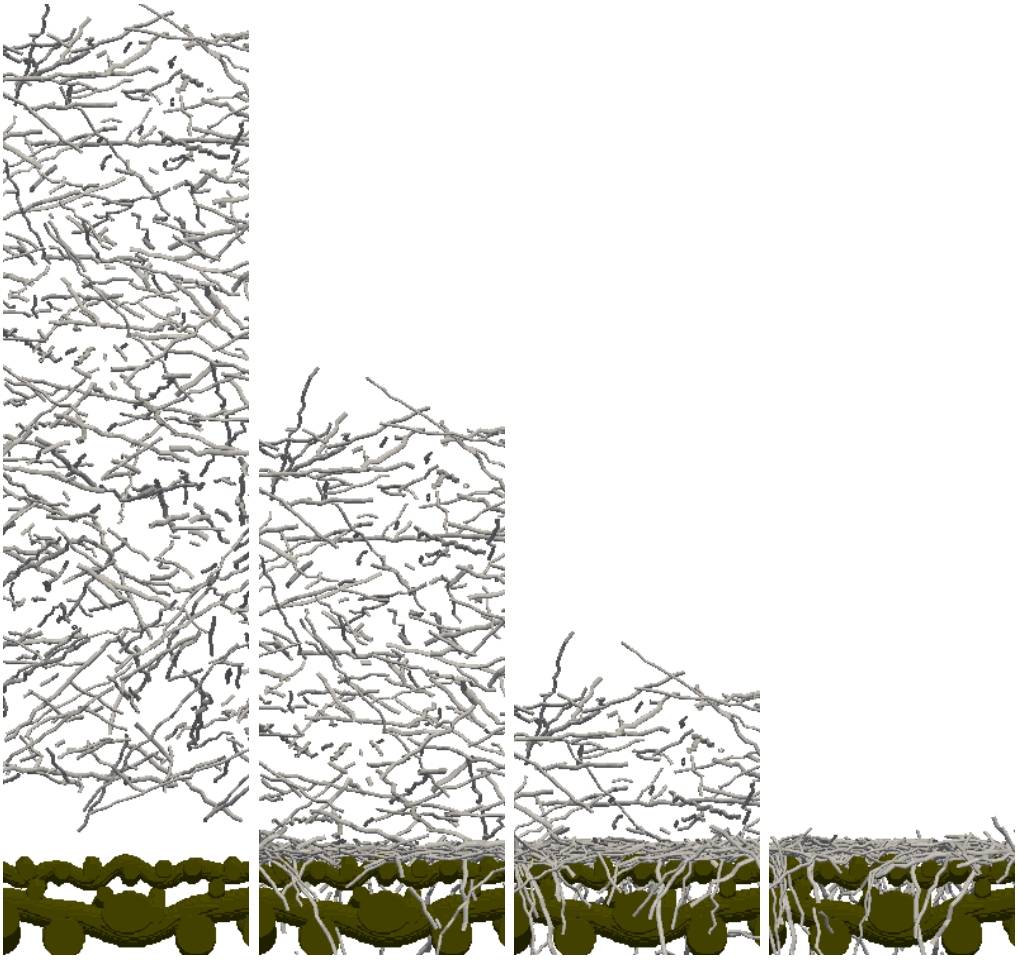


Figure 3.40: *A cloud of fibers falls down onto a forming fabric. Forming fabric geometry courtesy of Albany International.*

Two different forming fabrics are studied in this project. Figure (3.41) shows the geometry of the two forming fabrics, in the following called fabric A and fabric B. Note the geometric complexity of both forming fabrics. As can be seen in figure (3.41), the representation of

fabric B has been constructed by mirroring a smaller part of the true geometry. The geometrical description of fabric B used in this study is therefore an approximation.

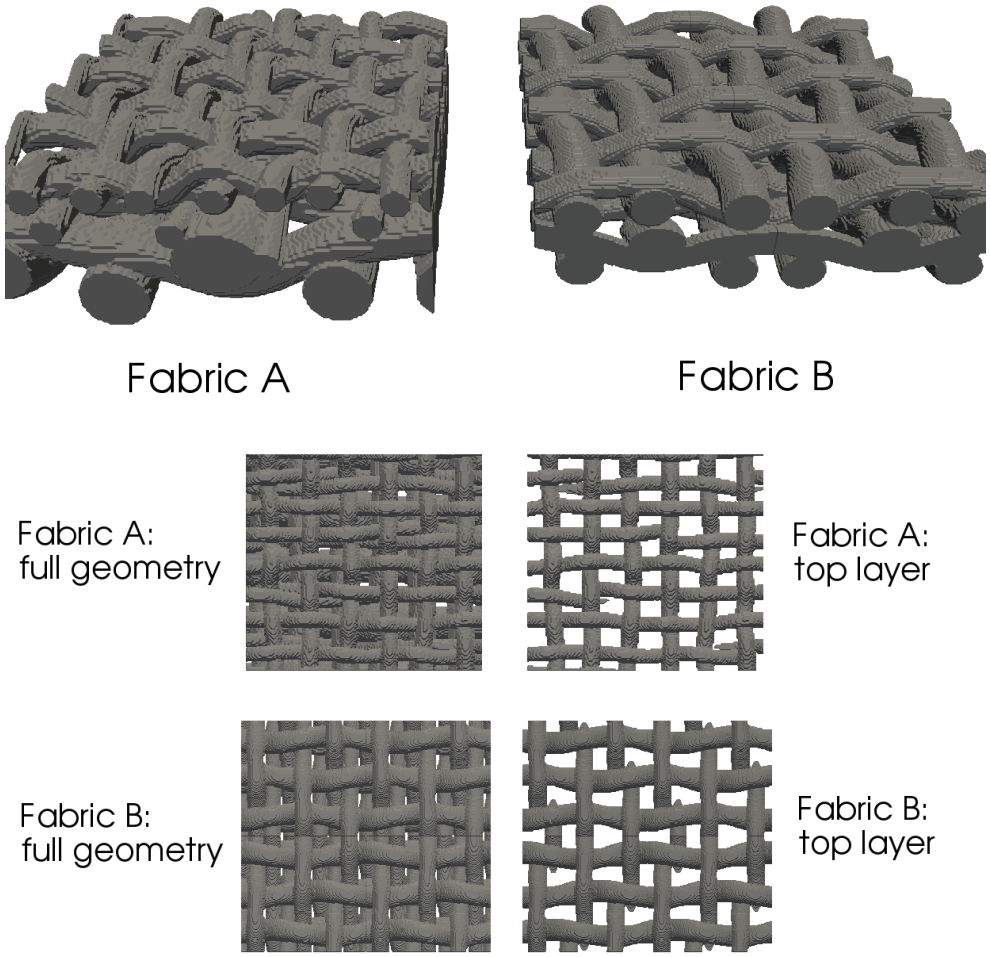


Figure 3.41: *Geometry of forming fabrics used in the simulations: perspective view (top) and top view (bottom). As can be seen, the representation of fabric B has been constructed by mirroring a smaller part of the true geometry. The geometrical description of fabric B used in this study is therefore an approximation. Forming fabric geometry courtesy of Albany International.*

The simulation domain is filled with water with viscosity $\mu_f = 1.0 \cdot 10^{-3}\ Pas$ and density $\rho_f = 1000\ kg/m^3$. The domain is hexahedral and it is bounded by its lower and upper corner:

- Lower corner $(0, 0, 0)\ m$
- Upper corner $(2.15 \cdot 10^{-3}, 1.91 \cdot 10^{-3}, z_{uc})\ m$

z_{uc} is the height of the domain, which is varied depending on the number of fibers used in the simulation. If many fibers are simulated, a higher domain is necessary. The forming fabric is located in the lower part of the domain as shown in figure (3.42). Figure (3.43) shows slices through the domain with the fluid boundary conditions indicated. The water with the suspended fibers flows through the domain from top to bottom. In summary, the boundary conditions on the fluid are:

- x_{bottom} : inlet
- x_{top} : inlet

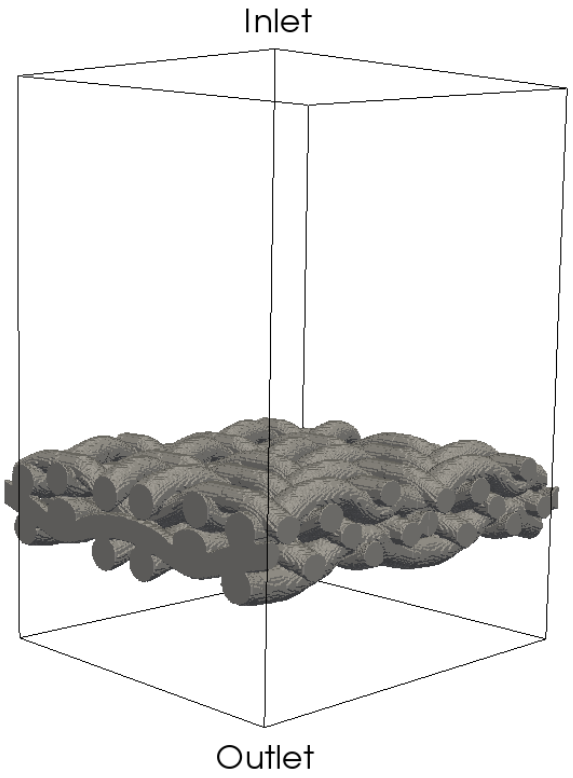


Figure 3.42: *Forming fabric and simulation domain. Forming fabric geometry courtesy of Albany International.*

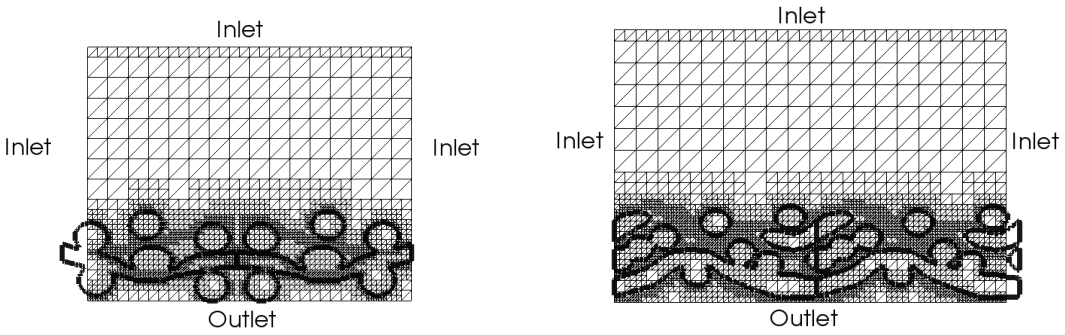


Figure 3.43: *Fabric and simulation domain.*

- y_{bottom} : inlet
- y_{top} : inlet
- z_{bottom} : outlet
- z_{top} : inlet

The inlet velocity was set to $v_{in} = (0, 0, -1)\ m/s$.

The geometrical description of the paper fibers was generated with GeoDict. To perform a simulation, the mechanical properties of the fibers need to be defined. Material data for for dry, pressed sheets of paper are relatively easy to find in the literature. However, material data for wet individual fibers, which have not been pressed, is much more difficult to measure and difficult to find in the literature. In the present work, buoyant fibers with a density of $\rho_s = 1000\ kg/m^3$ were studied. A Young's modulus of $E = 5.0 \cdot 10^7\ Pa$ and

a Poisson's ratio of $\nu_s = 0.3$ were assumed. The coefficient of friction was set to $\mu_{fr} = 1$ and the collisions were modeled as inelastic with $e_{cor} = 0$.

The first order upwind scheme was used for the convective terms and the implicit Euler scheme was used for the temporal discretization of the Navier-Stokes equations. Hilber's alpha method with $\alpha = -0.3$ was used for the temporal discretization of the FE equations.

3.3.2 Laydown simulation

50 fibers falling down onto forming fabric B were simulated. The height of the domain was set to $z_{uc} = 3\ mm$. The time step length was set to $\Delta t = 0.25\ \mu s$. The penalty method was used for contact between fibers and the DCR method was used for contact between fibers and forming fabric. Figure (3.44) shows the grid seen in the y-direction. Note the adaptive refinements around the fibers as well as around the forming fabric. The grid consists of roughly 700000 cells and 8000 time steps were taken. The simulation time necessary for this problem was 5 days on two threads of an Intel Core 2 Quad @ 2.40 GHz with 8 GB of RAM.

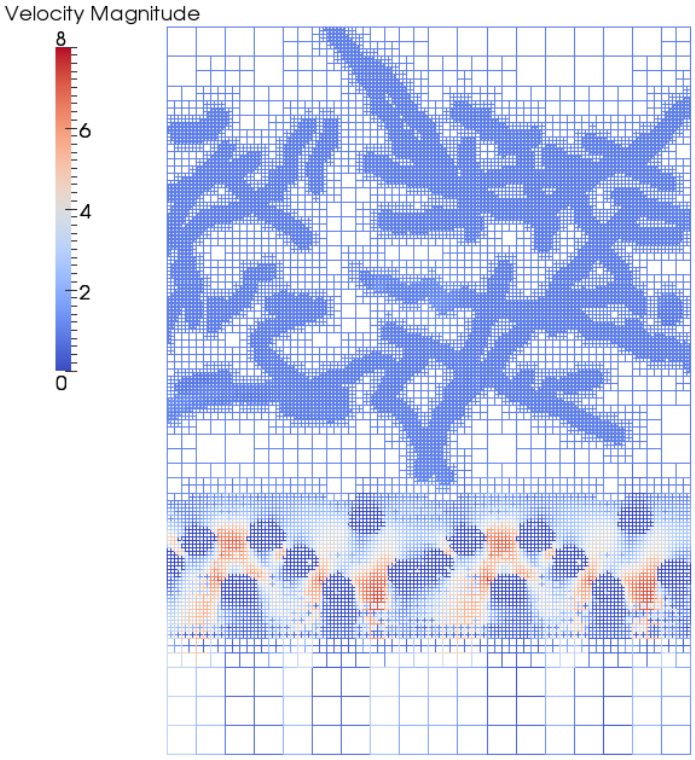


Figure 3.44: *Grid refinements around fibers and forming fabric. The grid is colored by the fluid velocity.*

Figure (3.45) shows how the fibers fall down on the forming fabric. The streamlines are colored by the fluid velocity and the forming fabric is shown partly transparent to reveal the flow pattern through the holes of the forming fabric. The flow is quite uniform far away from the forming fabric, but highly complex close to the forming fabric. The fluid is accelerated from the free stream velocity of 1 m/s to approximately 8 m/s when passing through the holes of the forming fabric. As a result, the fibers falling down on the forming fabric are subjected to both collision forces and a nonuniform flow field. However, fibers flowing freely in the domain far away from the forming fabric follow the fluid because the fibers are buoyant. As a result, the fibers have the same velocity as the water before they come close to the forming fabric, but fibers impacting with the forming fabric experience

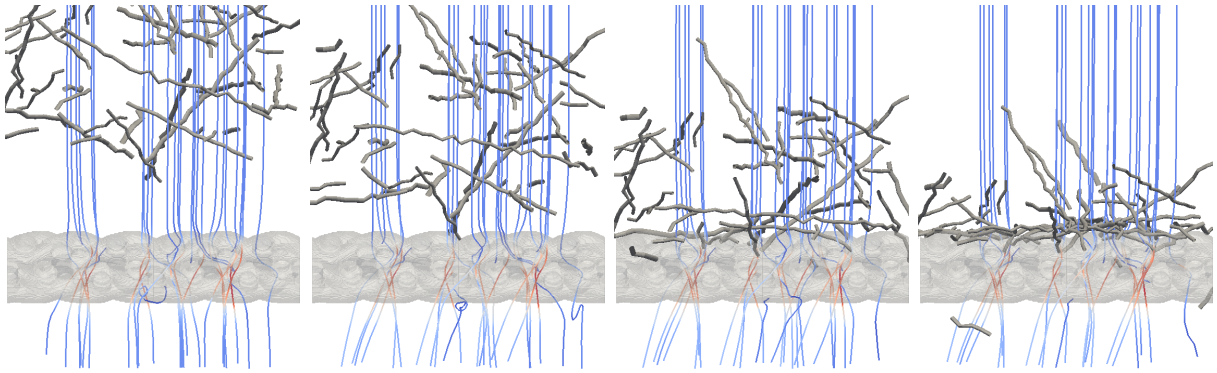


Figure 3.45: *50 fibers falling down on a forming fabric. The streamlines are colored by the fluid velocity.*

a rapid change in velocity due to the collision. The contact forces from the forming fabric prevent the fibers from following the water downwards through the domain. Therefore, the fibers will disturb the fluid more when they are resting on the forming fabric, thus getting support from the forming fabric. This phenomenon is shown in figure (3.46), which shows the velocity in a slice through the domain. The thick white circles show the contours of the forming fabric and the thin white ellipses show a slice through the surface of the fibers. The fibers in the left plot in (3.46) are located in the free stream quite far away from the forming fabric. They therefore follow the fluid, the velocity is uniform close to the fibers. In the plot to the right in (3.46), several fibers rest on the forming fabric. The wakes visible below these fibers reveal that the fibers disturb the fluid. Figure (3.47) shows the region close the the forming fabric zoomed in. The effect of the fibers on the flow field is clearly to be seen even though the grid used is quite coarse.

As described above, the fibers have a strong influence on the flow field close to the forming fabric, but fibers far away from the forming fabric follow the fluid without disturbing the flow. This observation suggests a possible simplification of the model to improve the efficiency without loosing too much accuracy. The Immersed Boundary Method and a drag correlation could be combined: the flow around fibers close to the forming fabric could be resolved with the Immersed Boundary Method, but a drag correlation could be used for fibers far away from the forming fabric. In this way, the flow around fibers which disturb the flow will be resolved, but the flow around fibers that do not disturb the flow will be neglected. This approach has not been implemented in the present work, but it would be a very interesting extension in a future project.

The moving adaptive grid refinements avaiable in IBOFlow are essential for the simulation of this problem. Figure (3.48) shows how the grid refinements follow the fibers as they flow through the water. Note the structure of the refinements where each refinement halves the size of the fluid cells. This type of refinement prevents deterioration of the mesh quality: since cubic cells are always used and the refinements are always done by cutting the cell size in half, cells with extreme aspect ratio will never occur in the grid.

The quality of a fiber web can be evaluated by studying variations in grammage and fiber direction. To get a statistically meaningful measure of these properties, the fiber web studied must contain sufficiently many fibers. 50 fibers are not enough for a meaningful analysis of grammage and fiber directions. The postprocessing capabilities of the code are therefore demonstrated in the next section instead.

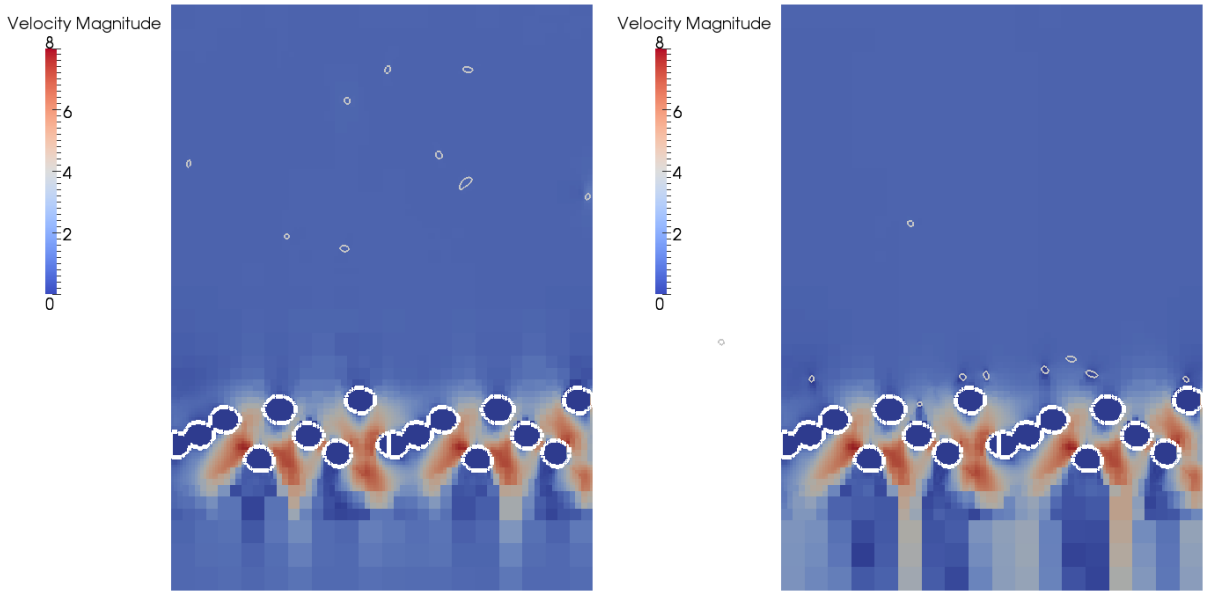


Figure 3.46: *50 fibers falling down on a forming fabric: slice colored by the fluid velocity.*

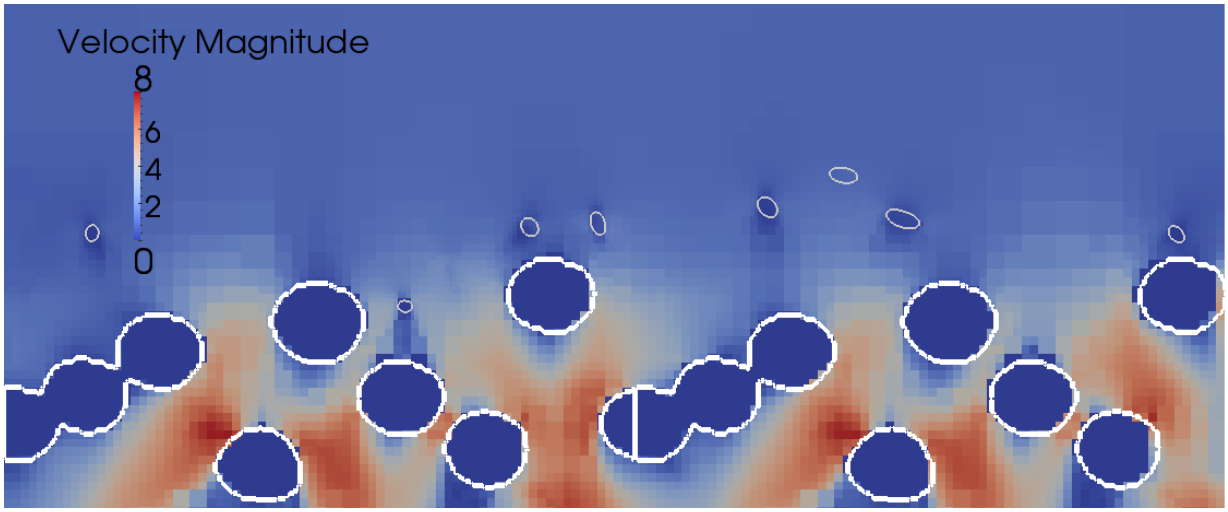


Figure 3.47: *50 fibers falling down on a forming fabric: forming fabric region zoomed in.*

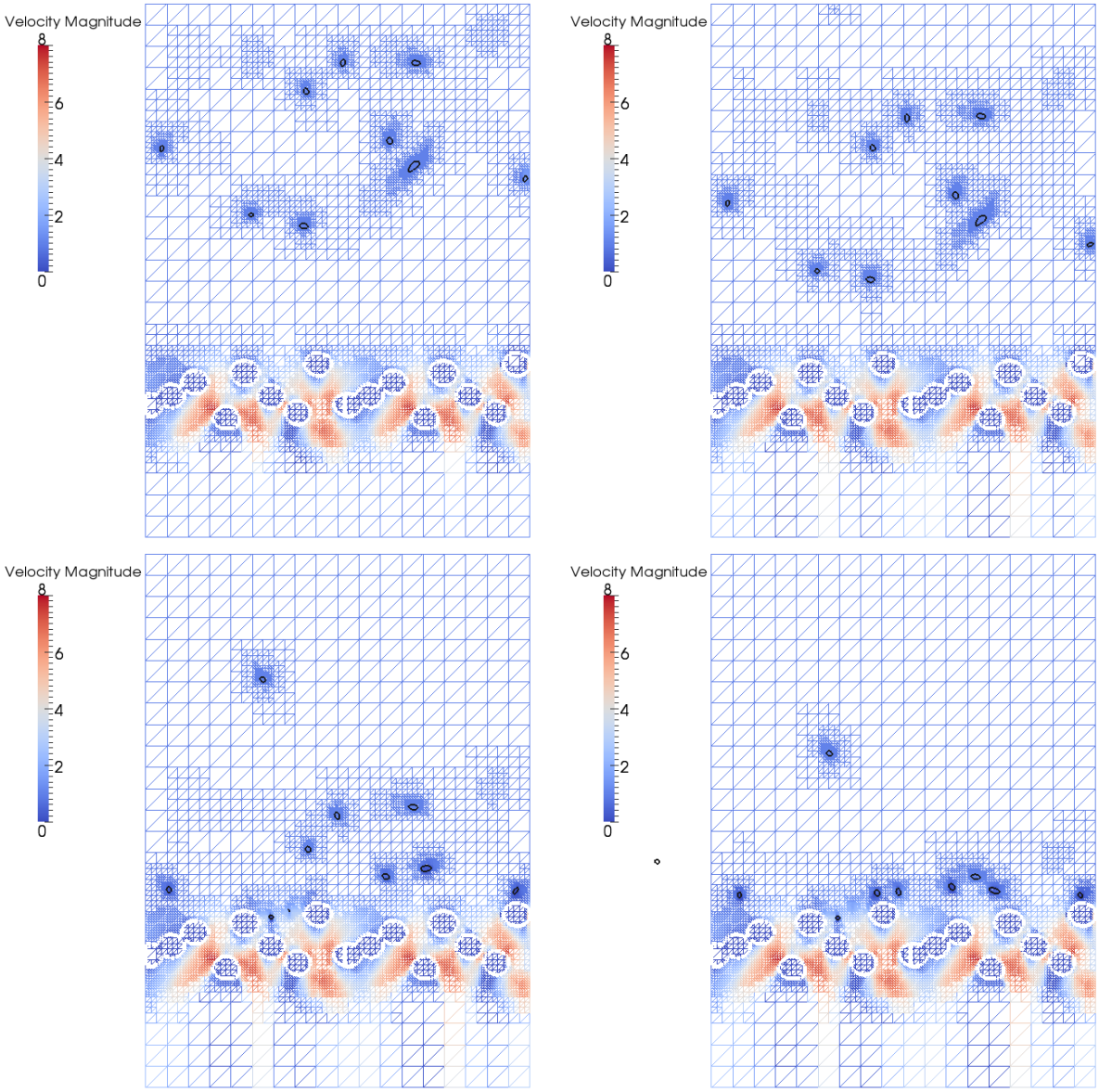


Figure 3.48: *50 fibers falling down on a forming fabric. The adaptive grid refinements move with the fibers as they flow through the water. The grid is colored by the fluid velocity.*

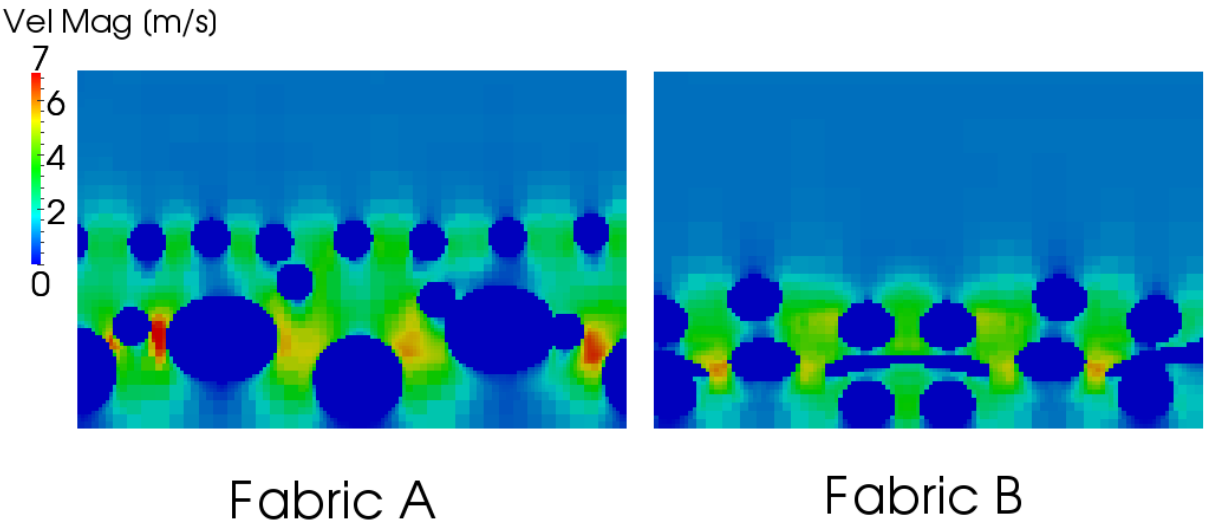


Figure 3.49: *Velocity field in a plane through fabric A (left) and fabric B (right)*

3.3.3 Qualitative comparison of two forming fabrics

A qualitative comparison of two different forming fabrics has been performed. The simulations were performed with one-way coupling and roughly 400 fibers were distributed in the domain as shown in figure (3.40). The larger number of fibers is stressful for the FE solver and the simulations are therefore a good test of the robustness of the code. A steady state solution of the flow through the fabric was simulated and the fibers were tracked in the stationary fluid. The fluid mesh used in this case consists of approximately $5.0 \cdot 10^5$ cells. Figure (3.49) shows the flow field through the two forming fabrics and the corresponding pressure drops are shown in figure (3.50). The pressure drop over forming fabric A is higher than the pressure drop over forming fabric B because forming fabric A has a thicker, more complex geometry with narrower channels. In this simulation, the velocity at the inlet was prescribed. An alternative would be to prescribe the pressure drop. This would probably result in larger differences between the velocity fields through the two forming fabrics. The velocity through fabric B would then be higher than the velocity through fabric A due to the larger resistance in fabric A. These velocity differences could result in differences in the formed fiber webs. Such effects will not be captured in this simulation and will be investigated in the future.

The resulting fiber webs built up during the simulations are shown in figure (3.51). The fiber webs formed on the two fabrics look relatively similar in the top figure in (3.51), but the bottom figure in (3.51) reveals that the fibers follow the topology of the forming fabric: the fibers are sucked into the holes of the forming fabric. This phenomenon is particularly pronounced for fabric B, which has larger holes. Figure (3.52) shows the same fiber webs seen from above. This simulation example shows that the code can handle forming of larger fiber webs, no stability problems occur even though the fiber web is kept together only by contact forces and fluid forces.

An important measure of the quality of a paper is how evenly the fibers are distributed. The distribution of fibers can be studied with the code developed in the present work by computing the grammage as described in the theory section. Figure (3.53) shows plots of the grammage for fabric A and B. The mass distribution is relatively unsmooth on both forming fabrics and it is difficult to draw conclusions based on figure (3.53). The reason for the unsmooth plots is probably that 400 fibers are not enough to form a continuous

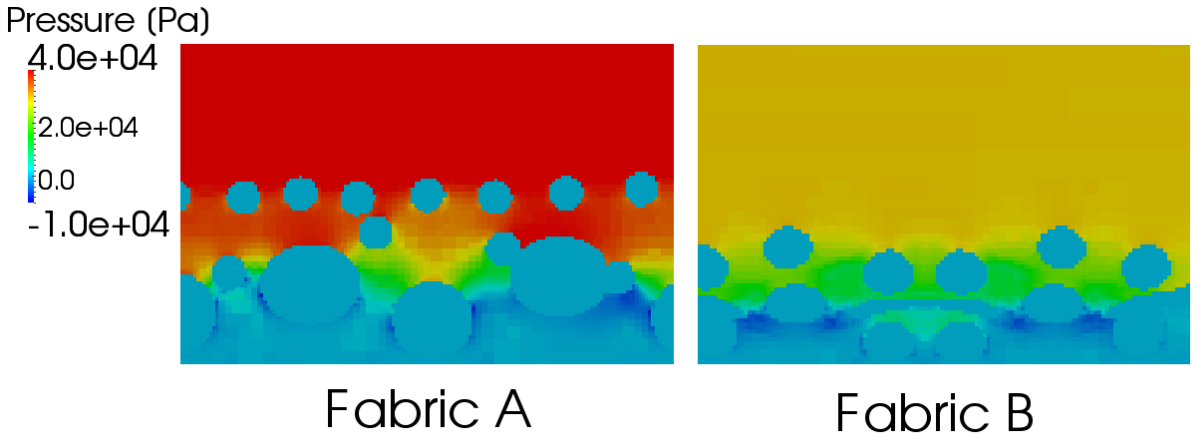


Figure 3.50: *Pressure drop over fabric A (left) and fabric B (right)*

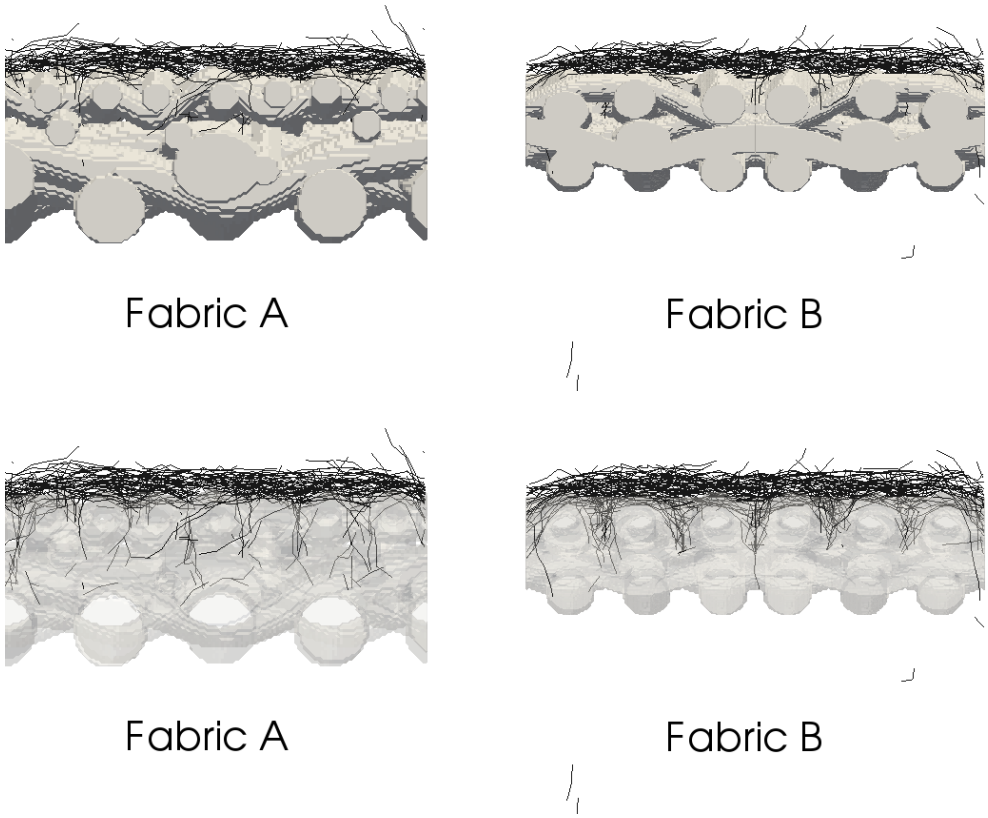


Figure 3.51: *Fiber web built up on two different forming fabrics. The lower figure, where the forming fabric is transparent, reveals that the fiber web follows the topography of the forming fabric: the fibers flow into the holes of the forming fabric. This phenomenon is particularly pronounced for fabric B, which has a simpler geometry. Forming fabric geometry courtesy of Albany International.*

fiber web with homogeneous thickness. This effect can also be seen in figure (3.52): the forming fabric is not completely covered by fibers. Therefore, more fibers should be used in the simulation in order to get a good average.

The hypothesis that the unsmooth distribution seen in figure (3.53) is caused by too few fibers in the simulation has been investigated. For this purpose, a simulation with 1300 fibers falling down onto Fabric B was performed. Figure (3.54) shows the distribution of fibers on forming fabric B when 400 and 1300 fibers were used in the simulation. The

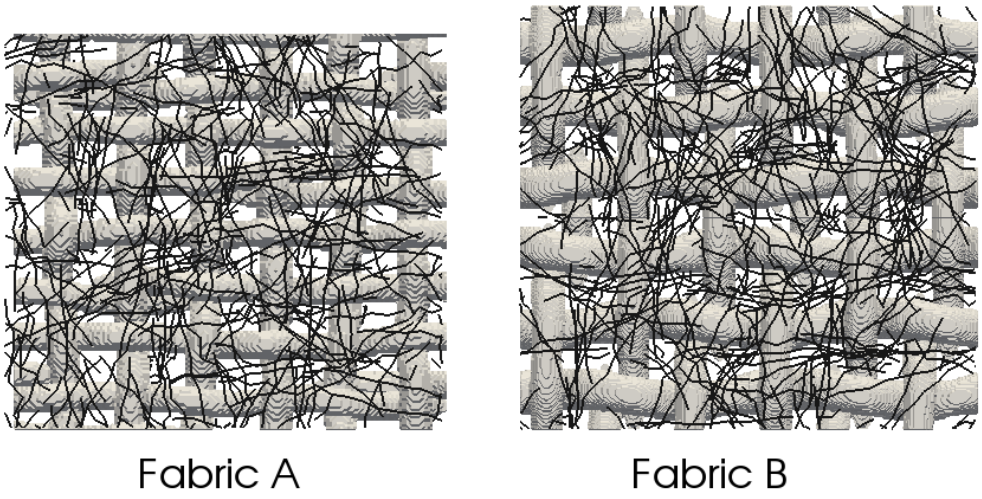


Figure 3.52: *Top view of fiber webs built up on two different forming fabrics. Forming fabric geometry courtesy of Albany International.*

simulation with 1300 fibers gives a much smoother distribution, confirming that more fibers are needed to get meaningful results from the postprocessing.

Figure (3.55) shows the fiber web formed during this simulation and the fiber web formed during the simulation with 400 fibers. As can be seen on the structure of the fiber web, the fibers are sucked into the holes of the fabric. The simulation with 1300 fibers reflects this observation in the variation of the grammage: the grammage has peeks above the holes. Comparing with the plot to the left in figure (3.55) reveals that this effect is less pronounced in the simulation with 400 fibers. The trend of fibers being sucked into the holes of the fabric can be seen when looking at a web with 400 fibers. However, more fibers are needed to see a clear impact on the variation of grammage. This comparison of fiber webs of different size shows that care must be exercised when studying properties of the fiber web: a fiber web with a moderate number of fibers can reveal interesting trends if it is inspected visually, but a large number of fibers is necessary to get meaningful statistical measures.

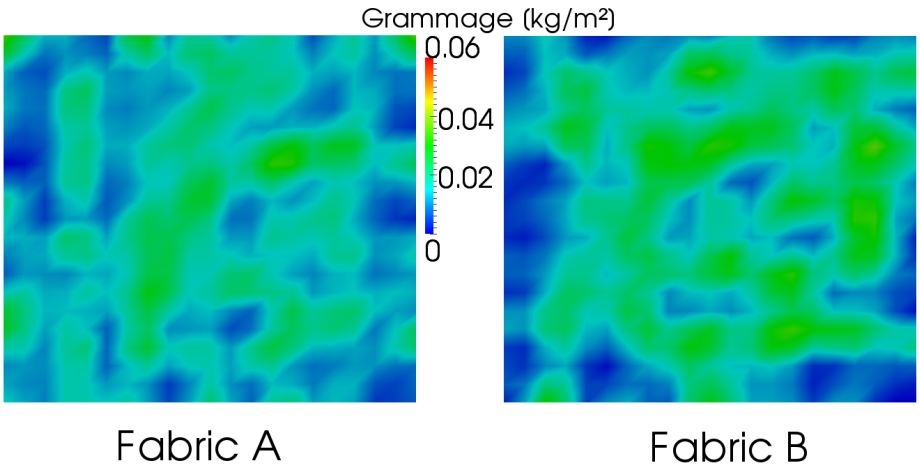


Figure 3.53: *Variation of grammage in the fiber web predicted with 400 fibers.*

Since 400 fibers were found to be insufficient, a simulation with a larger number of fibers falling down onto forming fabric A was also performed. A comparison between the fiber

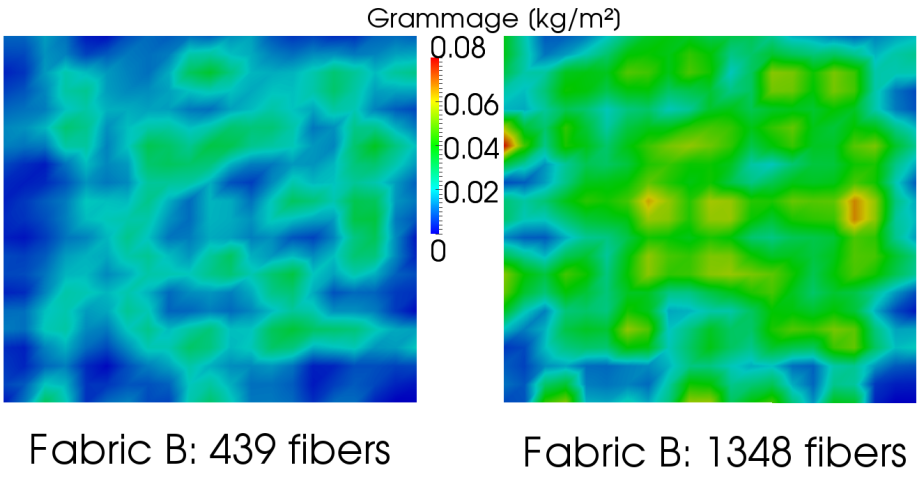


Figure 3.54: *Variation of grammage in the fiber web predicted with 439 fibers and 1348 fibers.*

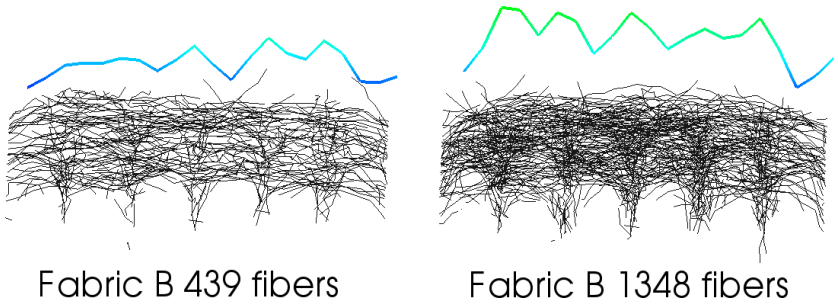


Figure 3.55: *Grammage in a slice through a fiber web simulated with 439 fibers (left) and 1348 fibers (right). Blue indicates low grammage and green indicates high grammage.*

webs built up on the two forming fabrics is shown in figure (3.56). The mass distribution and the mean angle are shown in figure (3.57). These simulations predict smoother mass distributions than the simulations with 400 fibers. The mean angle is close to 45 degrees ($\approx 0.8\ rad$) on both forming fabrics. Since the angle may vary from 0 to 90 degrees, a mean value of 45 degrees indicates that the fibers have not been aligned in a particular direction. One reason for the randomized fiber orientation is probably that the simulations were performed without horizontal velocity. Adding a horizontal velocity component would probably induce anisotropy. Furthermore, the orientation of fibers in a real paper machine will be highly anisotropic when the suspension leaves the headbox. The anisotropy from the headbox was not accounted for in this simulation, instead a fully randomized fiber orientation was assumed.

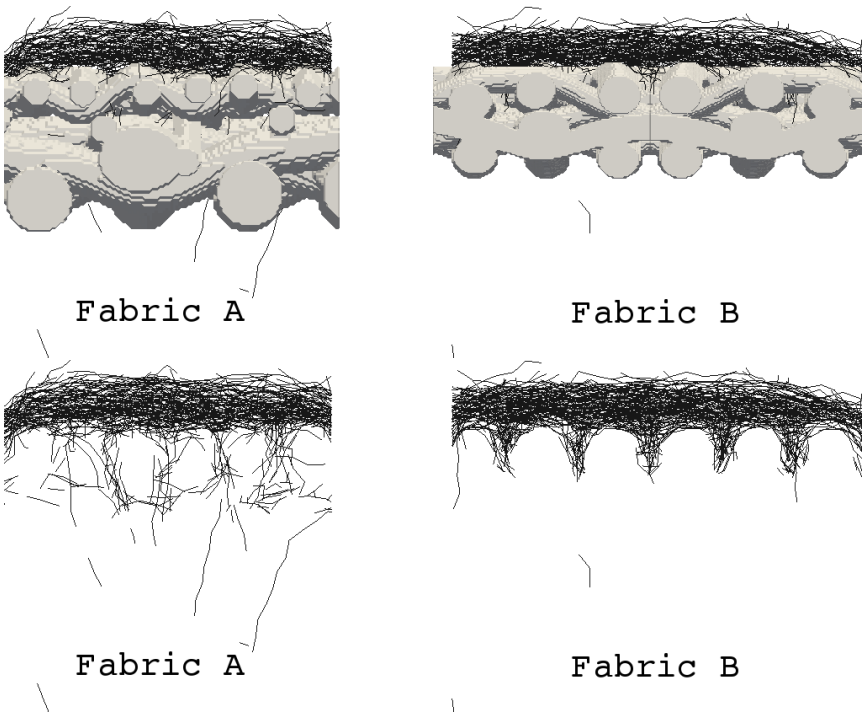


Figure 3.56: *Fiber web built up on forming fabric A (left) and forming fabric B (right).*

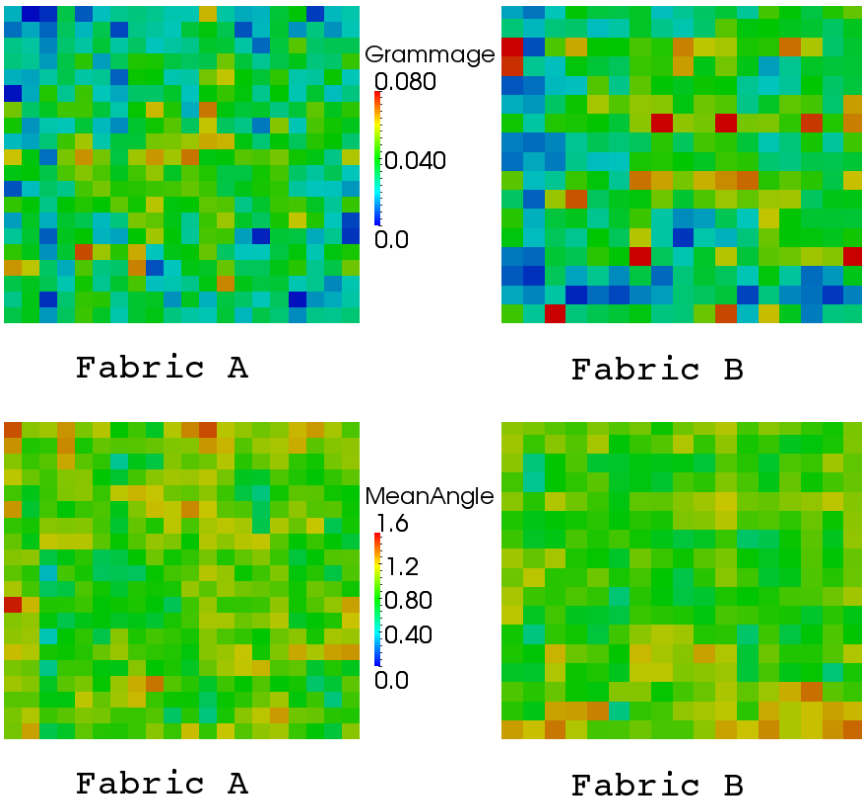


Figure 3.57: *Mass distribution (top) and mean angle with the x-axis (bottom) of the fiber webs built up on the two forming fabrics.*

4 Summary and Conclusions

A Finite Element beam model with geometric nonlinearities and contacts with friction has been implemented. The implementation has been validated against several demanding problems described in the literature. It can be concluded that the code performs very well for the test cases considered, even when compared to other beam models that account for finite strains and shearing of the cross sections. It has been shown that the code can handle postbuckling, dynamic problems with finite rotations and contacts with friction. Newton's method is used to solve the nonlinear system of equations, which results in a very robust code.

Contact detection for elements with elliptical cross section has been studied. Collision candidates were identified with a kd-tree and the shortest distance between two elements was computed with an iterative algorithm. It was concluded that it is difficult and computationally expensive to determine the overlap of two elements, but the distance between two non-overlapping elements can be found at a reasonable cost. Based on this observation, a small penalty layer was added in the contact models, so that collisions are resolved when the fibers are very close, but not actually touching. This approach allows the contacts to be resolved much faster.

Two contact models have been implemented: a penalty method and an impulse based method. Inelastic or partly inelastic contacts are modeled with the coefficient of restitution and friction is modeled with Coulomb's law. A regularization of Coulomb's law is used in the penalty method. Both models give acceptable results for the test problems and both models are robust enough to simulate paper forming. The penalty method seems to give slightly more accurate results for the test problems, but the impulse based method has the benefit of being able to handle perfect stick. The impulse based method as well as the penalty method can be adapted to run in parallel. Both contact models are therefore valuable.

The FE code has been coupled with the CFD code IBOFlow with the Immersed Boundary Method. This method is implicit, second order accurate and very robust. A beam oscillating in a cross flow has been studied with DNS, a drag correlation and a Fourier series approximation. It was found that DNS is necessary to fully capture the transient response, but the simplified Fourier series analysis and the drag correlation give the same steady state response. The DNS simulation could be used to improve the drag model so that the transient response can be captured. This could be done by adding a transient term to the drag correlation and using the DNS simulation to determine the coefficient of this term. This approach offers a possibility to use DNS simulations to improve the accuracy of simpler models.

The code has been used to simulate paper forming. The robustness of the implementation was very satisfactory for this extremely difficult problem. The large displacements of the immersed objects are handled without difficulty and the complex geometry of the forming fabric has no negative effects on the fluid mesh quality. The moving adaptive grid refinement technique available in IBOFlow is essential for the treatment of complex moving boundaries. The cubic cells in combination with the octree structure used to represent the fluid grid ensures that the mesh quality never deteriorates. The fast and robust FE code handles large rotations without difficulty. The collisions do not cause convergence problems if suitable time step lengths and simulation settings are chosen. However, short time steps are necessary to resolve inelastic collisions with friction. The Immersed Boundary Method is robust, buoyant fibers can be simulated without stability problems.

A qualitative comparison of two forming fabrics was performed with a larger number of fibers. The simulations were performed under the assumption of one-way coupling. It

was shown that the FE code is fast and robust enough to handle more than 1000 fibers. It was also demonstrated that relevant output data such as grammage and fiber orientations can be extracted from the simulations. It was found difficult to draw conclusions about variations in grammage if too few fibers are used in the simulation. If too few fibers are used, they do not form a continuous fiber web and then it is not meaningful to talk about a continuous variation of grammage. Even though problems were encountered with the statistical measures, interesting trends could be identified by inspecting the formed fiber webs. It was found that the fibers tend to flow into the holes of the forming fabric and here a difference could be seen between the forming fabrics.

5 Further work

Even though the code developed in the present work is highly efficient, the computational cost of the simulations is a problem. Small cells are needed to resolve the flow around the fibers and a huge amount of time steps is required to perform a full laydown simulation. The simulation time is dominated by the fluid solver, so a speedup of the simulations could be gained by increasing the speed of the CFD solver. In the future, IBOFlow will be parallelized so that simulations can be performed on clusters. This will allow much larger problems to be studied. The contact detection routines will also be adapted to run in parallel.

The FE code developed in the present work has been validated against several demanding problems described in the literature. The fluid-structure interaction has been examined by comparison with simpler, but well established models. Test cases involving fluid-structure interaction with strong coupling and large deformation of structures are difficult to find in the literature. It would be very interesting to set up an experiment with a demanding fluid-structure interaction problem and compare with simulation results from FCC's code. One example of such a problem could be a dambreak where an elastic beam is hit by the collapsing water column. Then the problem would include a combination of FSI and VOF, making the problem highly interesting on its own.

It is possible to combine the Immersed Boundary Method and a drag correlation. This can be done by using the IBM in regions where strong interaction between the fibers and the fluid is expected, but using a drag correlation in regions where the fibers are expected to follow the fluid without disturbing the fluid. This approach would allow the interaction between fluid and fibers close to the forming fabric to be captured, but the interaction between fibers and fluid far away from the forming fabric could be neglected. The number of cells needed for a simulation would then decrease dramatically.

The present work has described initial development of models that can be used to study the behavior of fibers suspended in a fluid. The project will continue and more detailed simulations will be performed. The models will be refined by taking the effect of fillers and fines into account. The present work has demonstrated that the code can be used to study the effect of different forming fabrics or different boundary conditions. It would be very interesting to perform parametric studies with the aim of optimizing the process.

The effect of chemicals on the interaction between fibers and forming fabric has not been studied in the present work. Chemicals play an important role in the process of paper forming, so good models are needed for the interaction between chemicals and fibers. Including fillers and fines in the simulation and modeling the effect of chemicals on fillers, fines and fibers would improve the accuracy of the simulations.

References

[1] A. Boström. *Fundamental structural dynamics*. Department of Mechanics, Chalmers University of Technology, 2001.

[2] A. Chatterjee and A. Ruina. A new algebraic rigid body collision law based on impulse space considerations. *Journal of Applied Mechanics*, 65:939–951, 1998.

[3] A. B. Chaudhary and K. J. Bathe. A solution method for static and dynamic analysis of three-dimensional contact problems with friction. *Computers and Structures*, 6:855–873, 1986.

[4] F. Cirak and M. West. Decomposition contact response (dcr) for explicit finite element dynamics. *International Journal for Numerical Methods in Engineering*, 64:1078–1110, 2005.

[5] M.A. Crisfield, U. Galvanetto, and G. Jelenić. Dynamics of 3-d co-rotational beams. *Computational Mechanics*, 20:507–519, 1997.

[6] C. Crowe, M. Sommerfeld, and Y. Tsuji. *Multiphase flows with droplets and particles*. CRC Press, 1998.

[7] H. M. de la Fuente and C.A. Felippa. Ephemeral penalty functions for contact-impact dynamics. *Finite Elements in Analysis and Design*, 9:177–191, 1991.

[8] G. Dhondt and K. Wittig. Calculix, a free software three-dimensional structural finite element program. http://www.Calculix.de.

[9] J.H. Argyris et al. Finite element method - the natural approach. *Comput. Methods in Appl. Mech. Engrg.*, 17/18:1–106, 1979.

[10] GeoDict. http://www.geodict.com.

[11] F. Gruttmann, R. Sauer, and W. Wagner. A geometrical nonlinear eccentric 3d-beam element with arbitrary cross-sections. *Comput. Methods in Appl. Mech. Engrg.*, 160:383–400, 1998.

[12] S. M. Han, H. Benaroya, and T. Wei. Dynamics of transversely vibrating beams using four engineering theories. *Journal of Sound and Vibration*, 225:935–988, 1999.

[13] D. Harmon. *Robust, Efficient and Accurate Contact Algorithms*. PhD thesis, Columbia university, 2010.

[14] H. M. Hilber, T. J. R. Hughes, and R. L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering and Structural Dynamics*, 5:283–292, 1977.

[15] T. J. R. Hughes. *The Finite Element Method - Linear Static and Dynamic Finite Element Analysis*. Dover, New York, 1987.

[16] IBOFlow. http://www.fcc.chalmers.se/comp/products/iboflow/.

[17] A. Ibrahimbegović and M.A. Mikdad. Finite rotations in dynamics of beams and implicit time-stepping schemes. *International Journal for Numerical Methods in Engineering*, 41:781–814, 1998.

[18] S. B. Lindström and T. Uesaka. Simulation of the motion of flexible fibers in viscous fluid flow. *Physics of Fluids*, 19:113307, 2007.

[19] S. B. Lindström and T. Uesaka. Particle-level simulation of forming of the fiber network in papermaking. *International Journal of Engineering Science*, 46:858–876, 2008.

[20] S. B. Lindström and T. Uesaka. Simulation of semidilute suspensions of non-brownian fibers in shear flow. *The Journal of Chemical Physics*, 128:024901, 2008.

[21] A. Mark. *The Mirroring Immersed Boundary Method - Modeling Fluids with Moving and Interacting Bodies*. PhD thesis, Chalmers University of Technology, 2008.

[22] A. Mark, R. Rundqvist, and F. Edelvik. Comparison between different immersed boundary conditions for simulation of complex fluid flows. *International Conference on Multiphase Flow*, 2010.

[23] E. E. Michaelides. *Particles, bubbles and drops*. World Scientific Publishing, 2006. ISBN: 981-256-647-3.

[24] B. Nour-Omid and C.C. Rankin. The use of projectors to improve finite element performance. *Computers and Structures*, 30:257–267, 1988.

[25] B. Nour-Omid and C.C. Rankin. Finite rotation analysis and consistent linearization using projectors. *Computer Methods in Applied Mechanics and Engineering*, 93:353–384, 1991.

[26] N. Ottosen and H. Peterson. *Introduction to the Finite Element Method*. Prentice Hall, 1992.

[27] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7:155–162, 1964.

[28] J.C. Simo and L. Vu-Quoc. A three-dimensional finite-strain rod model. part ii: Computational aspects. *Computer Methods in Applied Mechanics and Engineering*, 58:79–116, 1986.

[29] J.C. Simo and L. Vu-Quoc. On the dynamics in space of rods undergoing large motions - a geometrically exact approach. *Computer Methods in Applied Mechanics and Engineering*, 66:125–161, 1988.

[30] D.E. Stewart and J.C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering*, 39:2673–2691, 1996.

[31] J.R. Williams and R. O'Connor. Discrete element simulation and the contact problem. *Archives of Computational Methods in Engineering*, 6:279–304, 1999.

[32] P. Wriggers. *Computational Contact Mechanics*. Springer Berlin Heidelberg New York, 2006.

A Formulas for three-dimensional rotations

As noted in [25], the exponential $\mathbf{T}$ of a skew-symmetric $3 \cdot 3$-matrix can be computed with the formula first given by Rodrigues:

$$\boldsymbol{\Theta} = S(\boldsymbol{\theta}) \tag{A.1}$$

$$\theta = |\boldsymbol{\theta}| \tag{A.2}$$

$$\mathbf{T} = exp(\boldsymbol{\Theta}) = \mathbf{I} + \frac{\sin\theta}{\theta}\boldsymbol{\Theta} + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\Theta}^2 \tag{A.3}$$

The matrix logarithm, which is the inverse of the matrix exponential, can be computed as:

$$\tau = \frac{1}{2}\left|axial\left(\mathbf{T} - \mathbf{T}^T\right)\right| \tag{A.4}$$

$$\boldsymbol{\Theta} = log(\mathbf{T}) = \frac{\arcsin\tau}{2\tau} axial\left(\mathbf{T} - \mathbf{T}^T\right) \tag{A.5}$$

The transformation from angles to spin variables, $\boldsymbol{\Lambda}$, was derived in [25]. It can be computed as:

$$\boldsymbol{\Lambda} = \frac{\partial\boldsymbol{\theta}}{\partial\boldsymbol{\omega}} = \mathbf{I} - \frac{1}{2}\boldsymbol{\Theta} + \xi\boldsymbol{\Theta}^2 \tag{A.6}$$

$$\boldsymbol{\Theta} = S(\boldsymbol{\theta}),\ \theta = |\boldsymbol{\theta}| \tag{A.7}$$

$$\xi = \frac{2\sin\theta - \theta(1+\cos\theta)}{2\theta^2\sin\theta} \tag{A.8}$$

B Formulas for the static co-rotational formulation

B.1 $\underline{\underline{\Gamma}}$

The matrix $\underline{\underline{\Gamma}}$ is used when the internal force vector and tangent stiffness are calculated. It relates an infinitesimal motion of the element frame to an infinitesimal motion of the nodal degrees of freedom. $\underline{\underline{\Gamma}}$ is defined as follows:

$$\underline{\underline{\Gamma}}^T = \begin{bmatrix} \frac{\partial\boldsymbol{\omega}_E}{\partial\boldsymbol{x}_1} & \frac{\partial\boldsymbol{\omega}_E}{\partial\boldsymbol{\omega}_1} & \frac{\partial\boldsymbol{\omega}_E}{\partial\boldsymbol{x}_2} & \frac{\partial\boldsymbol{\omega}_E}{\partial\boldsymbol{\omega}_2} \end{bmatrix} \tag{B.1}$$

A complete derivation of $\underline{\underline{\Gamma}}$ is given in [24] for the choice of local coordinate system used in the present work. The main steps are recited here and the final result is given.

As can be seen in equation (B.1), $\boldsymbol{\Gamma}$ contains the derivative of the spin of the element frame with respect to all element degrees of freedom. The derivation of $\boldsymbol{\Gamma}$ requires taking the variation of the local frame, which is defined as follows: The first base vector is given by the centerline of the element:

$$\boldsymbol{e}_1 = \frac{\boldsymbol{x}_2^e - \boldsymbol{x}_1^e}{|\boldsymbol{x}_2^e - \boldsymbol{x}_1^e|} = \frac{\boldsymbol{x}_2^e - \boldsymbol{x}_1^e}{l} \tag{B.2}$$

To define the second and third base vectors, a vector $\boldsymbol{q}$ is introduced. This vector is initially aligned with the $\boldsymbol{e}_2$-axis in the start point and rotates with the start point when the element deforms:

$$\boldsymbol{q}_{loc} = \mathbf{E}^T \cdot \mathbf{T}_1 \cdot \mathbf{E}_0 \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{B.3}$$

Note that equation (B.3) gives $\boldsymbol{q}$ in local coordinates. The second and third base vectors are defined as:

$$\boldsymbol{e}_3 = \frac{\boldsymbol{e}_1 \times \boldsymbol{q}}{|\boldsymbol{e}_1 \times \boldsymbol{q}|} \tag{B.4}$$

$$\boldsymbol{e}_2 = \boldsymbol{e}_3 \times \boldsymbol{e}_1 \tag{B.5}$$

The variation of $\boldsymbol{\omega}_E$ can be related to the variation of the base vectors by taking the variation of the transformation matrix $\mathbf{E}$:

$$S\left(\delta\boldsymbol{\omega}_E\right) = \mathbf{E}^T \cdot \delta\mathbf{E} \tag{B.6}$$

Writing out the terms in (B.6) gives:

$$\begin{bmatrix} 0 & -\delta\omega_E^3 & \delta\omega_E^2 \\ \delta\omega_E^3 & 0 & -\delta\omega_E^1 \\ -\delta\omega_E^2 & \delta\omega_E^1 & 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{e}_1 \cdot \delta\boldsymbol{e}_1 & \boldsymbol{e}_1 \cdot \delta\boldsymbol{e}_2 & \boldsymbol{e}_1 \cdot \delta\boldsymbol{e}_3 \\ \boldsymbol{e}_2 \cdot \delta\boldsymbol{e}_1 & \boldsymbol{e}_2 \cdot \delta\boldsymbol{e}_2 & \boldsymbol{e}_2 \cdot \delta\boldsymbol{e}_3 \\ \boldsymbol{e}_3 \cdot \delta\boldsymbol{e}_1 & \boldsymbol{e}_3 \cdot \delta\boldsymbol{e}_2 & \boldsymbol{e}_3 \cdot \delta\boldsymbol{e}_3 \end{bmatrix} \tag{B.7}$$

Identifying terms in (B.7) gives the following expressions:

$$\begin{bmatrix} \delta\omega_E^1 \\ \delta\omega_E^2 \\ \delta\omega_E^3 \end{bmatrix} = \begin{bmatrix} -\boldsymbol{e}_2 \cdot \delta\boldsymbol{e}_3 \\ -\boldsymbol{e}_3 \cdot \delta\boldsymbol{e}_1 \\ \boldsymbol{e}_2 \cdot \delta\boldsymbol{e}_1 \end{bmatrix} \tag{B.8}$$

The expression in (B.8) is given as equation (51) in [24]. With the help of equation (B.8), the problem of computing the variation of the spin of the local frame has been reduced to taking the variation of the base vectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_3$. Taking the variation of $\boldsymbol{e}_1$ gives:

$$\delta\boldsymbol{e}_1 = \frac{1}{l}\left(\mathbf{I} - \boldsymbol{e}_1 \otimes \boldsymbol{e}_1\right) \cdot \left(\delta\boldsymbol{u}_2 - \delta\boldsymbol{u}_1\right) \tag{B.9}$$

$\delta\boldsymbol{e}_3$ is also required. From the definition in (B.4), it can be seen that $\boldsymbol{e}_3$ depends on $\boldsymbol{e}_1$ and $\boldsymbol{q}$. Therefore, the variation of $\boldsymbol{e}_3$ can be expressed as:

$$\delta\boldsymbol{e}_3 = \frac{\partial\boldsymbol{e}_3}{\partial\boldsymbol{q}} \cdot \delta\boldsymbol{q} + \frac{\partial\boldsymbol{e}_3}{\partial\boldsymbol{e}_1} \cdot \delta\boldsymbol{e}_1 \tag{B.10}$$

Evaluating the derivatives in (B.10) by using the definition (B.4) gives the variation of $\boldsymbol{e}_3$ as:

$$\delta\boldsymbol{e}_3 = \frac{1}{lq_2}\left[\boldsymbol{e}_2\left(q_1\left(\delta x_2^3 - \delta x_1^3\right) - l\delta q_3\right) - \boldsymbol{e}_1 q_2\left(\delta x_2^2 - \delta x_1^2\right)\right] \tag{B.11}$$

Inserting (B.9) and (B.11) into (B.8) gives the following expression for the variation of the spin of the local frame (equation (54) in [24], but note the spelling mistake on the index in [24]):

$$\delta\boldsymbol{\omega}_E = \begin{bmatrix} \frac{1}{lq2}\left(l\delta q_3 - q_1\left(\delta x_2^3 - \delta x_1^3\right)\right) \\ -\frac{1}{l}\left(\delta x_2^3 - \delta x_1^3\right) \\ \frac{1}{l}\left(\delta x_2^2 - \delta x_1^2\right) \end{bmatrix} \tag{B.12}$$

Taking the variation of $\boldsymbol{q}$ gives:

$$\delta\boldsymbol{q} = S\left(\delta\boldsymbol{\omega}_s\right) \cdot \boldsymbol{q} \tag{B.13}$$

Combining the above expressions gives the following expression for $\underline{\underline{\Gamma}}$:

$$\underline{\underline{\Gamma}}^T = \begin{bmatrix} 0 & 0 & \frac{\eta}{l} & 1 & (-\eta) & 0 & 0 & 0 & \left(-\frac{\eta}{l}\right) & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{l} & 0 & 0 & 0 & 0 & 0 & \left(-\frac{1}{l}\right) & 0 & 0 & 0 \\ 0 & \left(-\frac{1}{l}\right) & 0 & 0 & 0 & 0 & 0 & \left(\frac{1}{l}\right) & 0 & 0 & 0 & 0 \end{bmatrix} \tag{B.14}$$

$$\eta = \frac{q_1}{q_2} \tag{B.15}$$

B.2 $\underline{\underline{P}}$

According to eqn (33) in [25], the $6 \cdot 6$ blocks of the matrix $\underline{\underline{P}}$ can be computed as:

$$\underline{\underline{P}}_{ab} = \underline{\underline{I}}\delta_{ab} - \underline{\underline{\Psi}}_a \cdot \underline{\underline{\Gamma}}_b^T \tag{B.16}$$

By taking $\underline{\underline{\Gamma}}$ from (B.14) and $\underline{\underline{\Psi}}$ from equation (34) in [25], the projector matrix $\underline{\underline{P}}$ is found to be:

$$\underline{\underline{P}} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\eta/l & 0 & \eta & 0 & 0 & 0 & \eta/l & 0 & 0 & 0 \\
0 & 0 & -1/l & 0 & 1 & 0 & 0 & 0 & 1/l & 0 & 0 & 0 \\
0 & 1/l & 0 & 0 & 0 & 1 & 0 & -1/l & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\eta/l & -1 & \eta & 0 & 0 & 0 & \eta/l & 1 & 0 & 0 \\
0 & 0 & -1/l & 0 & 0 & 0 & 0 & 0 & 1/l & 0 & 1 & 0 \\
0 & 1/l & 0 & 0 & 0 & 0 & 0 & -1/l & 0 & 0 & 0 & 1
\end{bmatrix} \tag{B.17}$$

B.3 $\underline{\underline{F}}$

The matrix $\underline{\underline{F}}$ depends on the internal force vector of an element. Let $\boldsymbol{n}_1$ and $\boldsymbol{m}_1$ be the internal force and moment in the start point of the element. In the same way, let $\boldsymbol{n}_2$ and $\boldsymbol{m}_2$ be the internal force and moment in the end point. The matrix $\underline{\underline{F}}$ is defined in [25] as:

$$\underline{\underline{F}} = \begin{bmatrix}
S(\boldsymbol{n}_1) \\
\frac{1}{2}S(\boldsymbol{m}_1) \\
S(\boldsymbol{n}_2) \\
\frac{1}{2}S(\boldsymbol{m}_2)
\end{bmatrix} \tag{B.18}$$

Here, $S()$ denotes the spin operator described in the theory section.

C Fourier series expansion of an oscillating cantilever beam in a fluid

The force on a cylinder submerged in a fluid can be described by the drag force. Other forces, such as the history force, may also be important. However, only the drag force is considered in this simplified Fourier series expansion. For a given cylinder geometry, the drag coefficient c_d can be obtained from tabulated data or curve fitting formulas and the total drag force on the cylinder is computed as [6]:

$$F_{drag} = \frac{1}{2}\rho_F c_d A_{proj} \left|u_F - \dot{w}\right| \left(u_F - \dot{w}\right) \tag{C.1}$$

In the present work, values for c_d for flow around a long cylinder given by Clift et al. are used [23]:

$$\begin{aligned}
c_d &= \tfrac{9.689}{Re^{0.78}}\left(1 + 0.147 \cdot Re^{0.82}\right) \quad \text{if} \quad 0.1 < Re < 5 \\
c_d &= \tfrac{9.689}{Re^{0.78}}\left(1 + 0.227 \cdot Re^{0.5}\right) \quad \text{if} \quad 5 \leq Re < 40 \\
c_d &= \tfrac{9.689}{Re^{0.78}}\left(1 + 0.0838 \cdot Re^{0.82}\right) \quad \text{if} \quad 40 \leq Re < 400
\end{aligned} \tag{C.2}$$

The projected area of a circular beam is $A_{proj} = 2r \cdot L$, where r is the radius of the beam and L is the total beam length. The force per unit length on a thin slice of the beam is therefore:

$$q = \frac{1}{2}\rho_F c_d 2r \left| v_F - \dot{w} \right| \left(v_F - \dot{w} \right) \tag{C.3}$$

With the fluid properties and simulation settings used here, the boundary layer on the lower wall will be thin compared to the length of the beam. Therefore, the far-field fluid velocity is regarded as constant. As a result, the fluid force on the beam will only vary with the beam velocity in that point.

The Euler-Bernoulli beam equation can be written as [1]:

$$\alpha \ddot{w} = \frac{q}{EI} - w^{IV}, \alpha = \frac{A_{cs}\rho_s}{EI} \tag{C.4}$$

Here, a superscript with roman letters denotes derivative with respect to x, the coordinate in the axial direction of the beam. A_{cs} is the cross section area of the beam and w denotes the deflection of the beam. The problem can be studied with modal analysis by making the following ansatz for the deflection:

$$w\left(x,t\right) \approx \sum_{n=1}^{m} X_n\left(x\right) T_n\left(t\right) \tag{C.5}$$

In the equation above, X_n is a function of x only and T_n is a function of t only. Subscript n denotes functions associated with mode n and m is the total number of modes included in the approximation. The functions X_n must fulfill the boundary conditions. For Euler-Bernoulli beams, X_n can be expressed in Duncan functions [1]:

$$X_n = A_{1n}s_1\left(\mu_n x\right) + A_{2n}c_1\left(\mu_n x\right) + A_{3n}s_2\left(\mu_n x\right) + A_{4n}c_2\left(\mu_n x\right) \tag{C.6}$$

$$s_1\left(\xi\right) = \sin\xi + \sinh\xi \tag{C.7}$$

$$c_1\left(\xi\right) = \cos\xi + \cosh\xi \tag{C.8}$$

$$s_2\left(\xi\right) = -\sin\xi + \sinh\xi \tag{C.9}$$

$$c_2\left(\xi\right) = -\cos\xi + \cosh\xi \tag{C.10}$$

As noted in [1], the Duncan functions are related through:

$$c_2\left(\xi\right) = s_2^{'}\left(\xi\right) = c_1^{''}\left(\xi\right) = s_1^{'''}\left(\xi\right) \tag{C.11}$$

Furthermore, the Duncan functions are ideal for Dirichlet boundary conditions at $\xi = 0$ because they have the following convenient properties [1]:

$$s_1\left(0\right) = s_2\left(0\right) = c_2\left(0\right) = 0,\ c_1\left(0\right) = 2 \tag{C.12}$$

$$c_1^{'}\left(0\right) = s_2^{'}\left(0\right) = c_2^{'}\left(0\right) = 0,\ s_1^{'}\left(0\right) = 2 \tag{C.13}$$

$$s_1^{''}\left(0\right) = c_1^{''}\left(0\right) = s_2^{''}\left(0\right) = 0,\ c_2^{''}\left(0\right) = 2 \tag{C.14}$$

$$s_1^{'''}\left(0\right) = c_1^{'''}\left(0\right) = c_2^{'''}\left(0\right) = 0,\ s_2^{'''}\left(0\right) = 2 \tag{C.15}$$

The beam studied in this example is clamped at the left end and free at the right end. Therefore, the deflection and the angle are zero at $x = 0$. The bending moment and the transverse force are zero at the free end at $x = L$. These boundary conditions must be

fulfilled at all instants in time and therefore X_n must fulfill these conditions. The resulting conditions on X_n are:

$$X_n(0) = 0 \tag{C.16}$$
$$X_n'(0) = 0 \tag{C.17}$$
$$X_n''(\lambda_n L) = 0 \tag{C.18}$$
$$X_n'''(\lambda_n L) = 0 \tag{C.19}$$

By inserting these conditions into (C.6), the eigenvalue λ_n and three of the constants A_{1n}, A_{2n}, A_{3n} and A_{4n} can be obtained. The result is the following expression for X_n:

$$X_n(x) = A_{4n}\left[c_2(\lambda_n x) - \frac{c_1(\lambda_n L)}{s_1(\lambda_n L)} s_2(\lambda_n x)\right] \tag{C.20}$$

If the constant A_{4n} is included in the time function $T_n(t)$, the deflection can be expressed as:

$$w(x,t) = \sum_{n=1}^{\infty} T_n(t)\left[c_2(\lambda_n x) - \frac{c_1(\lambda_n L)}{s_1(\lambda_n L)} s_2(\lambda_n x)\right] \tag{C.21}$$

Insert (C.21) into the PDE in (C.4):

$$\sum_{n=1}^{\infty} T_n(t)\,\lambda_n^4\left[s_1(\lambda_n x) - \frac{c_1(\lambda_n L)}{s_1(\lambda_n L)} c_2(\lambda_n x)\right] = \tag{C.22}$$

$$= \sum_{n=1}^{\infty}\left(-\frac{A_{cs}\rho_s}{EI}\ddot{T}_n(t)\left[s_1(\lambda_n x) - \frac{c_1(\lambda_n L)}{s_1(\lambda_n L)} c_2(\lambda_n x)\right]\right) + \frac{q}{EI} \tag{C.23}$$

Define $\omega_n^2 = \lambda_n^4 \frac{EI}{A\rho_s}$ and $b_n = \frac{c_1(\lambda_n L)}{s_1(\lambda_n L)}$. Insert these definitions together with the expression for the distributed load in equation (C.23):

$$\sum_{n=1}^{\infty}\left[\omega_n^2 T_n + \ddot{T}_n\right]\left[s_1(\lambda_n x) - b_n c_2(\lambda_n x)\right] = -\frac{q_0}{A_{cs}\rho_s}\left|\dot{w}_{rel}\right|\dot{w}_{rel} \tag{C.24}$$

Expand the load in the same Fourier series as $X(x)$:

$$f = -\frac{q_0}{A\rho_s}\left|\dot{w} - v_f\right|\left(\dot{w} - v_f\right) \overset{!}{=} \sum_{n=1}^{\infty} a_n(t)\left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right] \tag{C.25}$$

To compute the coefficients a_n, start by multiplying (C.25) by the eigenmode shape $[c_2(\lambda_m x) - b_n s_2(\lambda_m x)]$ and integrate over the length of the beam:

$$\int_0^L f\cdot\left[c_2(\lambda_m x) - b_n s_2(\lambda_m x)\right]dx = \int_0^L \sum_{n=1}^{\infty} a_n(t)\left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right]\left[c_2(\lambda_m x) - b_m s_2(\lambda_m x)\right]dx =$$
$$= \sum_{n=1}^{\infty} a_n(t)\underbrace{\int_0^L \left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right]\left[c_2(\lambda_m x) - b_m s_2(\lambda_m x)\right]dx}_{I_1} \tag{C.26}$$

The integral I_1 in equation (C.26) was evaluated numerically. It was found that the orthogonality of the eigenmodes holds and the value of the integral was found to be:

$$I_1 = \int_0^L \left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right]\left[c_2(\lambda_m x) - b_m s_2(\lambda_m x)\right]dx = L\cdot\delta_{mn} \tag{C.27}$$

Here δ_{mn} denotes the Kronecker delta. Inserting (C.27) in (C.26) gives a_n as:

$$a_n(t) = \frac{1}{L}\int_0^L f \cdot \left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right] dx =$$
$$= \frac{1}{L}\int_0^L \frac{q_0}{A\rho_s}\left|\dot{w} - v_f\right|\left(\dot{w} - v_f\right)\left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right] dx \tag{C.28}$$

The integral in equation (C.28) above is a complicated function of x and would be difficult to integrate analytically. Especially, note that q_0 depends on c_d which in turn is a function of the Reynolds number Re. The Reynolds number is calculated from the local relative velocity, which varies with x, so that $q_0 = q_0(x)$. Therefore, the integral in (C.28) is integrated numerically with the Runge-Kutta 45 scheme. When a_n has been evaluated, the expanded expression for the load can be inserted in equation (C.24):

$$\sum_{n=1}^{\infty}\left[\ddot{T}_n + \omega_n^2 T_n\right]\left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right] = \sum_{n=1}^{\infty} a_n(t)\left[c_2(\lambda_n x) - b_n s_2(\lambda_n x)\right] \tag{C.29}$$

Equation (C.29) gives an ordinary differential equation for each T_n:

$$\ddot{T}_n + \omega_n^2 T_n = a_n \tag{C.30}$$

Equation (C.30) can be integrated in time with Newmark's method. Let k denote time step. The acceleration at the new time step is computed as:

$$\ddot{T}_n^{k+1} = \frac{1}{\beta\Delta t^2}\left[\left(T_n^{k+1} - T_n^k\right) - \Delta t\dot{T}_n^k - \Delta t^2(0.5-\beta)\ddot{T}_n^k\right] \tag{C.31}$$

Insert (C.31) in (C.30):

$$\frac{1}{\beta\Delta t^2}T_n^{k+1} + \underbrace{\frac{1}{\beta\Delta t^2}\left[-T_n^k - \Delta t\dot{T}_n^k - \Delta t^2(0.5-\beta)\ddot{T}_n^k\right]}_{c^k} + \omega_n^2 T_n^{k+1} = a_n^{k+1} \tag{C.32}$$

If semi-explicit integration is used, so that a_n is evaluated at the previous time step instead of at the current time step, T_n^{k+1} can be computed as:

$$T_n^{k+1} = \frac{a_n^k - c^k}{\left(\frac{1}{\beta\Delta t^2} + \omega_n^2\right)} \tag{C.33}$$

When T_n^{k+1} is known the velocity and acceleration are updated according to:

$$\dot{T}_n^{k+1} = \frac{\gamma}{\beta\Delta t}\left(T_n^{k+1} - T_n^k\right) + \left(1 - \frac{\gamma}{\beta}\right)\dot{T}_n^k + \Delta t\left(1 - \frac{\gamma}{2\beta}\right)\ddot{T}_n^k \tag{C.34}$$

$$\ddot{T}_n^{k+1} = \frac{1}{\beta\Delta t^2}\left[\left(T_n^{k+1} - T_n^k\right) - \Delta t\dot{T}_n^k - \Delta t^2(0.5-\beta)\ddot{T}_n^k\right] \tag{C.35}$$

D Jacobian of inertia force

To compute the Jacobian of the inertia force, first consider the variation of the translational terms:

$$\delta\left(N_a\rho A\ddot{\boldsymbol{u}}\right) = N_a\rho A\delta\ddot{\boldsymbol{u}} = N_a\rho A\frac{\partial\ddot{\boldsymbol{u}}}{\partial\boldsymbol{u}}\cdot\delta\boldsymbol{u} \tag{D.1}$$

The derivative of the nodal acceleration with respect to the nodal displacement must be determined to evaluate the expression above. In the present work, Newmark's interpolation is used to interpolate the nodal quantities in time. In the following, β and γ denote the Newmark interpolation parameters. The relation between the coordinate and acceleration at the new time step is given by equation (53) in [5]:

$$\ddot{\boldsymbol{u}}^{n+1} = \frac{1}{\beta \Delta t^2} \left[\left(\boldsymbol{u}^{n+1} - \boldsymbol{u}^n \right) - \Delta t \dot{\boldsymbol{u}}^n - \Delta t^2 \left(0.5 - \beta \right) \ddot{\boldsymbol{u}}^n \right] \tag{D.2}$$

Take the derivative of (D.2):

$$\frac{\partial \ddot{u}_i^{n+1}}{\partial u_j^{n+1}} = \frac{\delta_{ij}}{\beta \Delta t^2} \tag{D.3}$$

Insert (D.3) and (2.60) in (2.59):

$$\delta \left(N_a \rho A \ddot{\boldsymbol{u}} \right) = \frac{N_a \rho A}{\beta \Delta t^2} \delta \boldsymbol{u} = \frac{N_a N_1 \rho A}{\beta \Delta t^2} \delta \boldsymbol{u}_1 + \frac{N_a N_2 \rho A}{\beta \Delta t^2} \delta \boldsymbol{u}_2, \ a = 1, 2 \tag{D.4}$$

The formula above gives an explicit expression for the inertia stiffness terms corresponding to the translational degrees of freedom.

Next, consider the variation of the rotational terms, which is more involved. Taking the variation of the rotational terms in (2.59) gives:

$$\begin{aligned} \delta \left[N_a \left(S\left(\boldsymbol{w}\right) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w} + \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}} \right) \right] = \\ \underbrace{N_a \left(\delta S\left(\boldsymbol{w}\right) \right) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w}}_{c_1} + \underbrace{N_a S\left(\boldsymbol{w}\right) \cdot \left(\delta \mathbf{E} \right) \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \boldsymbol{w}}_{c_2} + \underbrace{N_a S\left(\boldsymbol{w}\right) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \left(\delta \mathbf{E}^T \right) \cdot \boldsymbol{w}}_{c_3} + \\ \underbrace{N_a S\left(\boldsymbol{w}\right) \cdot \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \delta \boldsymbol{w}}_{c_4} + \underbrace{N_a \left(\delta \mathbf{E} \right) \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \dot{\boldsymbol{w}}}_{c_5} + \underbrace{N_a \mathbf{E} \cdot \mathbf{J} \cdot \left(\delta \mathbf{E}^T \right) \cdot \dot{\boldsymbol{w}}}_{c_6} + \underbrace{N_a \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \delta \dot{\boldsymbol{w}}}_{c_7} \end{aligned} \tag{D.5}$$

To evaluate the first term above, first note that $\delta S\left(\boldsymbol{w}\right) = S\left(\delta \boldsymbol{w}\right)$. Then consider the Newmark interpolation for the angular velocity given by eqn (54) in [5]:

$$\boldsymbol{w}^{n+1} = \frac{\gamma}{\beta \Delta t} \left(\boldsymbol{\Psi}^{n+1} - \boldsymbol{\Psi}^n \right) + \left(1 - \frac{\gamma}{\beta} \right) \boldsymbol{w}^n + \Delta t \left(1 - \frac{\gamma}{2\beta} \right) \dot{\boldsymbol{w}}^n \tag{D.6}$$

$$\Rightarrow \frac{\partial w_i^{n+1}}{\partial \Psi_j^{n+1}} = \frac{\gamma}{\beta \Delta t} \delta_{ij} \Rightarrow \delta \boldsymbol{w} = \frac{\gamma}{\beta \Delta t} \delta \boldsymbol{\Psi} \tag{D.7}$$

Here $\boldsymbol{\Psi}$ is an additive angle parameter. The relation between this angle and the corresponding spin variable is given by:

$$\delta \boldsymbol{\Psi} = \mathbf{H}\left(\Delta \boldsymbol{\Psi} \right) \cdot \delta \boldsymbol{\omega} \tag{D.8}$$

Now use (D.7) and (D.8) to evaluate c_1 in (D.5):

$$c_1 = -\frac{\overline{N}_i \gamma}{\beta \Delta t} S\left(\mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \boldsymbol{w} \right) \cdot \mathbf{H} \cdot \delta \boldsymbol{\omega} \tag{D.9}$$

To evaluate c_2, the variation of the local element frame must be computed. To achieve this, first consider the variation of the local frame contracted with an arbitrary vector $\boldsymbol{z}$:

$$\delta \mathbf{E} \cdot \boldsymbol{z} = S\left(\delta \boldsymbol{\omega}_E^g \right) \cdot \mathbf{E} \cdot \boldsymbol{z} = -S\left(\mathbf{E} \cdot \boldsymbol{z} \right) \cdot \delta \boldsymbol{\omega}_E^g \tag{D.10}$$

Now consider the variation of the element frame in global coordinates with respect to the nodal degrees of freedom in global coordinates:

$$\delta\boldsymbol{\omega}_E^g = \mathbf{E}\cdot\delta\boldsymbol{\omega}_E^e = \mathbf{E}\cdot\frac{\partial\boldsymbol{\omega}_E^e}{\partial\boldsymbol{x}^e}\cdot\delta\boldsymbol{x}^e = \mathbf{E}\cdot\frac{\partial\boldsymbol{\omega}_E^e}{\partial\boldsymbol{x}^e}\cdot\mathbf{G}^T\cdot\delta\boldsymbol{x}^g, \tag{D.11}$$

$$\mathbf{G} = \begin{bmatrix} \mathbf{E} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{E} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E} \end{bmatrix}, \tag{D.12}$$

$$\delta\mathbf{x}^g = [\delta\mathbf{u}_s\ \delta\boldsymbol{\omega}_s\ \delta\mathbf{u}_e\ \delta\boldsymbol{\omega}_e] \tag{D.13}$$

By comparing (D.11) with [24] and [25], it is noted that:

$$\frac{\partial\boldsymbol{\omega}_E^e}{\partial\boldsymbol{x}^e} = \boldsymbol{\Gamma} \tag{D.14}$$

By using the explicit expression for $\boldsymbol{\Gamma}$ for a beam given in [24], $\delta\boldsymbol{\omega}_E^g$ can be expressed in known quantities:

$$\delta\boldsymbol{\omega}_E^g = \underbrace{\mathbf{E}\cdot\boldsymbol{\Gamma}\cdot\mathbf{G}^T}_{\boldsymbol{\Gamma}^g}\cdot\delta\boldsymbol{x}^g \tag{D.15}$$

Inserting (D.15) in (D.10) gives the variation of the local frame contracted with a vector $\boldsymbol{z}$:

$$\delta\mathbf{E}\cdot\boldsymbol{z} = -S\left(\mathbf{E}\cdot\boldsymbol{z}\right)\cdot\boldsymbol{\Gamma}^g\cdot\delta\boldsymbol{x}^g \tag{D.16}$$

Use the expression above to evaluate c_2:

$$c_2 = N_i S\left(\boldsymbol{w}\right)\cdot\left(\delta\mathbf{E}\right)\cdot\underbrace{\mathbf{J}\cdot\mathbf{E}^T\cdot\boldsymbol{w}}_{\boldsymbol{z}} = -N_i S\left(\boldsymbol{w}\right)\cdot S\left(\mathbf{E}\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot\boldsymbol{w}\right)\cdot\boldsymbol{\Gamma}^g\cdot\delta\mathbf{x}^g \tag{D.17}$$

To evaluate c_3, first consider the variation of the transpose of the local frame contracted with an arbitrary vector $\boldsymbol{z}$:

$$\begin{aligned}\left(\delta\mathbf{E}^T\right)\cdot\boldsymbol{z} = \left(S\left(\delta\boldsymbol{\omega}_E^g\right)\cdot\mathbf{E}\right)^T\cdot\boldsymbol{z} = \mathbf{E}^T\cdot S\left(\delta\boldsymbol{\omega}_E^g\right)^T\cdot\boldsymbol{z} = \\ -\mathbf{E}^T\cdot S\left(\delta\boldsymbol{\omega}_E^g\right)\cdot\boldsymbol{z} = \mathbf{E}^T\cdot S\left(\boldsymbol{z}\right)\cdot\delta\boldsymbol{\omega}_E^g = \mathbf{E}^T\cdot S\left(\boldsymbol{z}\right)\cdot\boldsymbol{\Gamma}^g\cdot\delta\boldsymbol{x}^g\end{aligned} \tag{D.18}$$

Inserting (D.18) into the expression for c_3 gives:

$$c_3 = N_i S\left(\boldsymbol{w}\right)\cdot\mathbf{E}\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot S\left(\boldsymbol{w}\right)\cdot\boldsymbol{\Gamma}^g\cdot\delta\boldsymbol{x}^g \tag{D.19}$$

c_4, c_5 and c_6 can be evaluated in the same way:

$$c_4 = N_i S\left(\boldsymbol{w}\right)\cdot\mathbf{E}\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot\delta\boldsymbol{w} = \frac{N_i\gamma}{\beta\Delta t}S\left(\boldsymbol{w}\right)\cdot\mathbf{E}\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot\mathbf{H}\cdot\delta\boldsymbol{\omega} \tag{D.20}$$

$$c_5 = N_i\left(\delta\mathbf{E}\right)\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot\dot{\boldsymbol{w}} = -N_i S\left(\mathbf{E}\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot\dot{\boldsymbol{w}}\right)\cdot\boldsymbol{\Gamma}^g\cdot\delta\boldsymbol{x}^g \tag{D.21}$$

$$c_6 = N_i\mathbf{E}\cdot\mathbf{J}\cdot\left(\delta\mathbf{E}^T\right)\cdot\dot{\boldsymbol{w}} = N_i\mathbf{E}\cdot\mathbf{J}\cdot\mathbf{E}^T\cdot S(\dot{\boldsymbol{w}})\cdot\boldsymbol{\Gamma}^g\cdot\delta\boldsymbol{x}^g \tag{D.22}$$

To evaluate c_7, the derivative of the angular acceleration with respect to the nodal degrees of freedom must be computed. This can be done by starting with the Newmark interpolation given by equation (55) in [5]:

$$\dot{\boldsymbol{w}}_{n+1} = \frac{1}{\beta\Delta t^2}\left(\boldsymbol{\Psi}_{n+1} - \boldsymbol{\Psi}_n - \Delta t\boldsymbol{w}_n - \Delta t^2\left(\frac{1}{2}-\beta\right)\dot{\boldsymbol{w}}_n\right)$$

$$\Rightarrow \frac{\partial\dot{\boldsymbol{w}}_{n+1}}{\partial\boldsymbol{\Psi}_{n+1}} = \frac{1}{\beta\Delta t^2}\delta_{ij} \tag{D.23}$$

$$\Rightarrow \delta\dot{\boldsymbol{w}} = \frac{1}{\beta\Delta t^2}\delta\boldsymbol{\Psi} \tag{D.24}$$

Insert (D.24) into the expression for c_7:

$$c_7 = \frac{N_i}{\beta \Delta t^2} \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \delta \mathbf{\Psi} = \frac{N_i}{\beta \Delta t^2} \mathbf{E} \cdot \mathbf{J} \cdot \mathbf{E}^T \cdot \mathbf{H} \cdot \delta \boldsymbol{\omega} \tag{D.25}$$

By inserting the above expressions into (D.5), the contribution to the tangent stiffness from the rotational inertia is obtained.

E Shortest unsigned distance between two segments

Consider a point $\boldsymbol{p}_I$ on the surface of segment I and a point $\boldsymbol{p}_{II}$ on the surface of segment II. Expressed in the local frame of the corresponding element, the coordinates of these points are:

$$\begin{aligned} \boldsymbol{p}_I^{loc} &= \{L_I s_I,\ (R_{ae,I} s_I + R_{as,I}(1 - s_I)) \cos(t_I),\ (R_{be,I} s_I + R_{bs,I}(1 - s_I)) \sin(t_I)\} \\ \boldsymbol{p}_{II}^{loc} &= \{L_{II} s_{II},\ (R_{ae,II} s_{II} + R_{as,II}(1 - s_{II})) \cos(t_{II}),\ (R_{be,II} s_{II} + R_{bs,II}(1 - s_{II})) \sin(t_{II})\} \end{aligned} \tag{E.1}$$

Let $\boldsymbol{p}_{Is}$ and $\boldsymbol{p}_{IIs}$ be the start point of segment I and II, respectively. Then, the coordinates of the surface points can be expressed in the global frame as:

$$\boldsymbol{p}_I^g = \mathbf{E}_I \cdot \boldsymbol{p}_I^{loc} + \boldsymbol{p}_{Is} \tag{E.2}$$

$$\boldsymbol{p}_{II}^g = \mathbf{E}_{II} \cdot \boldsymbol{p}_{II}^{loc} + \boldsymbol{p}_{IIs} \tag{E.3}$$

The change of basis matrices $\mathbf{E}_I$ and $\mathbf{E}_{II}$ are not a function of s or t and therefore they are constants. A vector $\boldsymbol{d}$ from a point on segment I to a point on segment II can be computed as:

$$\boldsymbol{d} = \boldsymbol{p}_{II}^g - \boldsymbol{p}_I^g = \mathbf{E}_{II} \cdot \boldsymbol{p}_{II}^{loc} - \mathbf{E}_I \cdot \boldsymbol{p}_I^{loc} + \boldsymbol{p}_{IIs} - \boldsymbol{p}_{Is} \tag{E.4}$$

The square of the distance between the two points is:

$$d^2 = \boldsymbol{d} \cdot \boldsymbol{d} \tag{E.5}$$

The distance d has extreme values when the square of the distance d^2 has extreme values. d^2 has extreme values when the gradient is equal to zero:

$$\frac{\partial d^2}{\partial x_\alpha} = 0,\ x_\alpha = \{s_I,\ t_I,\ s_{II},\ t_{II}\} \tag{E.6}$$

The system of equations in (E.6) can be solved with Newton's method. Recasting the problem into the framework of Newton's method, the residual becomes:

$$\begin{aligned} res_1 &= \frac{\partial d^2}{\partial s_I} = -2 d_i E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial s_I} \\ res_2 &= \frac{\partial d^2}{\partial t_I} = -2 d_i E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial t_I} \\ res_3 &= \frac{\partial d^2}{\partial s_{II}} = 2 d_i E_{ij}^{II} \frac{\partial p_j^{II,loc}}{\partial s_{II}} \\ res_4 &= \frac{\partial d^2}{\partial t_{II}} = 2 d_i E_{ij}^{II} \frac{\partial p_j^{II,loc}}{\partial t_{II}} \end{aligned} \tag{E.7}$$

The Jacobian corresponding to the residual in (E.7) is:

$$
\begin{aligned}
K_{11} &= \frac{\partial^2 d^2}{\partial s_I \partial s_I} = 2E_{ik}^I \frac{\partial p_k^{I,loc}}{\partial s_I} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial s_I} - 2d_i E_{ij}^I \frac{\partial^2 p_j^{I,loc}}{\partial s_I^2} \\
K_{12} &= \frac{\partial^2 d^2}{\partial t_I \partial s_I} = 2E_{ik}^I \frac{\partial p_k^{I,loc}}{\partial t_I} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial s_I} - 2d_i E_{ij}^I \frac{\partial^2 p_j^{I,loc}}{\partial t_I \partial s_I} \\
K_{13} &= \frac{\partial^2 d^2}{\partial s_{II} \partial s_I} = -2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial s_{II}} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial s_I} \\
K_{14} &= \frac{\partial^2 d^2}{\partial t_{II} \partial s_I} = -2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial t_{II}} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial s_I} \\
K_{21} &= K_{12} \\
K_{22} &= \frac{\partial^2 d^2}{\partial t_I \partial t_I} = \frac{\partial^2 d^2}{\partial t_I \partial s_I} = 2E_{ik}^I \frac{\partial p_k^{I,loc}}{\partial t_I} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial t_I} - 2d_i E_{ij}^I \frac{\partial^2 p_j^{I,loc}}{\partial t_I \partial t_I} \\
K_{23} &= \frac{\partial^2 d^2}{\partial s_{II} \partial t_I} = -2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial s_{II}} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial t_I} \\
K_{24} &= \frac{\partial^2 d^2}{\partial t_{II} \partial t_I} = -2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial t_{II}} E_{ij}^I \frac{\partial p_j^{I,loc}}{\partial t_I} \\
K_{31} &= K_{13} \\
K_{32} &= K_{23} \\
K_{33} &= \frac{\partial^2 d^2}{\partial s_{II} \partial s_{II}} = 2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial s_{II}} E_{ij}^{II} \frac{\partial p_j^{II,loc}}{\partial s_{II}} + 2d_i E_{ij}^{II} \frac{\partial^2 p_j^{II,loc}}{\partial s_{II}^2} \\
K_{34} &= \frac{\partial^2 d^2}{\partial t_{II} \partial s_{II}} = 2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial t_{II}} E_{ij}^{II} \frac{\partial p_j^{II,loc}}{\partial s_{II}} + 2d_i E_{ij}^{II} \frac{\partial^2 p_j^{II,loc}}{\partial t_{II} \partial s_{II}} \\
K_{41} &= K_{14} \\
K_{42} &= K_{24} \\
K_{43} &= K_{34} \\
K_{44} &= \frac{\partial^2 d^2}{\partial t_{II} \partial t_{II}} = 2E_{ik}^{II} \frac{\partial p_k^{II,loc}}{\partial t_{II}} E_{ij}^{II} \frac{\partial p_j^{II,loc}}{\partial t_{II}} + 2d_i E_{ij}^{II} \frac{\partial^2 p_j^{II,loc}}{\partial t_{II}^2}
\end{aligned}
\tag{E.8}
$$

The derivatives of the surface point coordinates in the local frame ($\frac{\partial p_k^{I,loc}}{\partial s_I}$ and so on) are computed in exactly the same way as for the case when the shortest distance between a segment and a point is sought. Note that $\boldsymbol{p}_I$ is independent of s_{II} and t_{II}. In the same way, $\boldsymbol{p}_{II}$ is independent of s_I and t_I. With the residual in (E.7) and the Jacobian in (E.8), Newton iterations can be performed to find the closest surface points on two segments. As for the case when the distance between a segment and a point is sought, the algorithm must account for the fact that the segments have finite length, so that only $s_I \in [0,1]$ and $s_{II} \in [0,1]$ is allowed. This can be achieved by reducing the Jacobian, thus constraining the solution to the closest segment edge:

- if $s_I < 0$, then set $s_I = 0$ and reduce the Jacobian and residual according to:
  1. $res_1 = 0$
  2. $K_{11} = 1$
  3. $K_{12} = K_{13} = K_{14} = K_{21} = K_{31} = K_{41} = 0$
- if $s_I > 1$, then set $s_I = 1$ and reduce the Jacobian and residual according to:

1. $res_1 = 0$
2. $K_{11} = 1$
3. $K_{12} = K_{13} = K_{14} = K_{21} = K_{31} = K_{41} = 0$

- if $s_{II} < 0$, then set $s_{II} = 0$ and reduce the Jacobian and residual according to:

  1. $res_3 = 0$
  2. $K_{33} = 1$
  3. $K_{31} = K_{32} = K_{34} = K_{13} = K_{23} = K_{43} = 0$

- if $s_{II} > 1$, then set $s_{II} = 1$ and reduce the Jacobian and residual according to:

  1. $res_3 = 0$
  2. $K_{33} = 1$
  3. $K_{31} = K_{32} = K_{34} = K_{13} = K_{23} = K_{43} = 0$

The procedure outlined above can be used to find the shortest unsigned distance between two segments. It can, however, not be used to determine the overlap (signed distance) in the case of overlapping segments. If the segments are overlapping, the algorithm will return the shortest unsigned distance, which is zero. This fact must be taken into account in the contact algorithm, e.g. by employing a penalty layer so that overlap never occurs. If the overlap must be computed, a possible solution is to start with the algorithm above and first locate the values of s and t corresponding to zero distance. When the Newton iterations get stuck on this point, one can proceed with an optimization algorithm that is not based on derivatives, e.g. Powell's method [27]. The s and t found with Newton's method could then be used as initial guess for Powell's method.