



Diplomová práce

Bc. Jakub Špatka

21.10. 2020

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Špatka** Jméno: **Jakub** Osobní číslo: **459637**
Fakulta/ústav: **Fakulta strojní**
Zadávací katedra/ústav: **Ústav technické matematiky**
Studijní program: **Strojní inženýrství**
Studijní obor: **Matematické modelování v technice**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Simulace interakce kapalina - pevná fáze - kontakt při procesu válcování

Název diplomové práce anglicky:

Computational Fluid-Structure-Interaction in Lubricated Temper Rolling

Pokyny pro vypracování:

Diplomová práce je zaměřena na rozšíření výpočetního modulu pevné fáze v rámci projektu numerické simulace procesu válcování. Cíle práce jsou následující

1. rešerše problematiky interakce kapaliny a pevné fáze s možným kontaktem (FSCI) se zaměřením na teorii tření
2. přidání modelu tření do existujícího solveru FEAFA (RWTH Aachen)
3. modelový výpočet a interpretace výsledků

Seznam doporučené literatury:

[1] Hilger D., Hosters N., Key F., Elgeti S. and Behr M.: A novel approach to fluid-structure interaction simulations involving large translation and contact, preprint submitted to IGAA2018 Conference Proceedings (2019)

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. Ing. Jan Halama, Ph.D., ústav technické matematiky FS

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

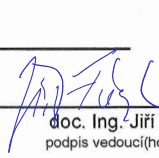
Datum zadání diplomové práce: **20.10.2020**

Termín odevzdání diplomové práce: **22.01.2021**

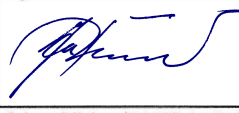
Platnost zadání diplomové práce: _____



doc. Ing. Jan Halama, Ph.D.
podpis vedoucí(ho) práce



doc. Ing. Jiří Fůrst, Ph.D.
podpis vedoucí(ho) ústavu/katedry



prof. Ing. Michael Valášek, DrSc.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

30.10.2020
Datum převzetí zadání

Jakub Špatka
Podpis studenta

Anotační list

Jméno autora	Bc. Jakub Špatka
Název práce	Simulace interakce kapalina - pevná fáze - kontakt při procesu válcování
Anglický název	Computational Fluid-Structure-Interaction in Lubricated Temper Rolling
Rok	2020
Obor	Matematické modelování v technice
Ústav	Ústav technické matematiky
Vedoucí práce na RWTH Aachen	Thomas Spenke, M. Sc.
Vedoucí práce na ČVUT FS	doc. Ing. Jan Halama, Ph.D.
Bibliografické údaje	Počet stran 59 Počet obrázků 52 Počet tabulek 6
Klíčová slova	Numerická simulace, MKP, NURBS, CFD, CSD, CCM, kontaktní mechanika s třením
Key words	Numerical simulation, FEM, NURBS, CFD, CSD, CCM, frictional contact

Anotace

Cílem této diplomové práce je implementace třecího modelu do software frameworku vyvíjeného ústavem CATS na RWTH Aachen. Tato práce vzniká jako spolupráce ČVUT FS a RWTH Aachen CATS během studijního zahraničního pobytu v rámci programu Erasmus+.

Abstract

This thesis has been designated to the implementation of frictional module to the already existing software framework of CATS institute at RWTH. This thesis is a product of cooperation between institute of Mechanical Engineering at CTU and CATS institute at RWTH during the study abroad exchange program Erasmus+.

**Diese Arbeit wurde vorgelegt am
Lehrstuhl für computergestützte Analyse technischer Systeme**

Simulation von Fluid-Struktur-Kontakt-Interaktion im geschmierten Nachwalzen
Computational Fluid-Structure-Interaction in Lubricated Temper Rolling

Masterarbeit
Master Thesis

von / presented by
Bc. Jakub Spatka

1. Prüfer/in / Evaluated by Prof. Marek Behr, Ph.D.
2. Prüfer/in / Evaluated by Dr.-Ing. Norbert Hosters
Betreuer/in Supervisor Thomas Spenke, M. Sc.

Aachen, October 21, 2020

I hereby declare that I wrote this thesis on my own and without the use of any other than the cited sources and tools and all explanations that I copied directly or in their sense are marked as such.

Contents

Glossary	I
Acronyms	V
List of Figures	VI
List of Tables	VIII
1 Introduction	1
2 Theory	3
2.1 Fluid-Structure-Contact Interaction	3
2.1.1 Computational Fluid Dynamics	3
2.1.2 Computational Solid Mechanics	4
2.1.3 Coupling Conditions	6
2.2 Partitioned Algorithms	8
2.2.1 Solution Approaches for FSCI	8
2.2.2 Transfer of Loads	8
2.2.3 Dirichlet-Neumann Coupling Scheme	12
2.2.4 Added-Mass Effect	13
2.3 Isogeometrical Analysis	14
2.4 Computational Contact Mechanics	17
2.4.1 Contact Detection	18
3 Solution of the Contact Problem	22
3.1 Variational Formulation for the Frictional Contact	24
3.2 Implementation	25
3.2.1 Normal Contact	27
3.2.2 Tangential Contact	28
3.2.3 Algorithm for the Frictional Contact	33
4 Software Environment	34
4.1 Flow Solver XNS	34
4.2 Structural Solver FEAFa	34
4.3 Coupling Module ACM	34
4.4 Splinelib Library	35
5 Numerical Results	36
5.1 Sliding Rectangular	36
5.1.1 Problem Description	36
5.1.2 Results	38

5.2	2D Shallow Ironing Problem with Friction - Benchmark Computation . .	38
5.2.1	Problem Description	38
5.2.2	Results	41
5.3	Sensitivity Analysis	45
5.3.1	Influence of the Penalty Parameters	46
5.3.2	The Influence of the Ironing Depth	46
5.3.3	Influence of the Time-Step	48
5.4	3D simulation of Shallow Ironing	50
5.4.1	Problem Setup	50
5.4.2	Results	51
6	Conclusion	58
	Acknowledgments	59
	References	60

Glossary

Notation Symbols Description

Computational Fluid Dynamics

Ω_F	fluid domain
\boldsymbol{v}	fluid velocity
\boldsymbol{v}_t	fluid acceleration
ν	kinematic viscosity
p	kinematic pressure
ρ	density
\boldsymbol{b}	body forces
Γ_F^D	Dirichlet boundary
\boldsymbol{v}_D	prescribed fluid velocity on the Dirichlet boundary
Γ_F^N	Neumann boundary
\boldsymbol{n}	outer normal of the Neumann boundary
\boldsymbol{v}_0	initial flow field

Computational Solid Mechanics

Ω_s	structural domain, current state
Ω_s^0	reference structural domain
\boldsymbol{X}	position vector in the reference domain
\boldsymbol{x}	position vector in the current domain
\boldsymbol{u}	displacement
\boldsymbol{u}_{tt}	acceleration
ρ	density of the structure
$\boldsymbol{\sigma}$	Cauchy stress tensor
\boldsymbol{b}	resultant body forces
\boldsymbol{F}	deformation gradient
\boldsymbol{S}	2nd Piola-Kirchhoff stress tensor
\boldsymbol{E}	Green-Lagrange strain tensor
\boldsymbol{C}	material matrix
Γ_D^0	Dirichlet boundary
$\tilde{\boldsymbol{u}}$	prescribed displacement on the Dirichlet boundary

Γ_N^0	Neumann boundary
\mathbf{n}	outer normal of the Neumann boundary
$\bar{\mathbf{t}}_0$	prescribed traction on the Neumann boundary
\mathbf{u}_0	initial displacement
$\delta \mathbf{u}$	virtual displacement
\mathbf{I}	identity matrix
\mathbf{U}	displacement field
\mathbf{M}	mass matrix
$\mathbf{K}(\mathbf{U})$	stiffness matrix
\mathbf{F}	right hand side vector

Coupling Conditions

\mathbf{u}_F	prescribed displacement on the fluid side
\mathbf{u}_S	prescribed displacement on the solid side
$\dot{\mathbf{u}}_F$	prescribed velocity on the fluid side
$\dot{\mathbf{u}}_S$	prescribed velocity on the solid side
Γ_{FS}	fluid structure interaction boundary
$\boldsymbol{\sigma}_F$	prescribed stress on the fluid side
$\boldsymbol{\sigma}_S$	prescribed stress on the solid side

Partitioned Algorithms

\mathbf{t}	tangential vector
\mathbf{x}_F	projecting point coordinate
\mathbf{x}_S	projected point coordinate
t^n, t^{n+1}	time level n, n+1
D^n, D^{n+1}	solid deformation at time level n, n+1
ϵ	tolerance criterion
$F(\bullet)$	fluid solver acting on \bullet
$S(\bullet)$	solid solver acting on \bullet
m^F	mass of the fluid
m^S	mass of the solid
μ_i	eigenvalues of a system
ρ^F	fluid density
ρ^S	density of the solid
C	setup dependent constant

Isogeometrical Analysis

C^n	function smoothness of n-th order
B_I	set of control points
$N_{I,p}(\xi)$	basis function of order p at parametric coordinate ξ
C	1D NURBS curve
S	2D NURBS curve
Θ^n	knot vector of dimension n

Computational Contact Mechanics

ρ^*	closest point on the master surface
\mathbf{r}_s	slave node
ρ	point coordinate at the master surface
Γ_m	master boundary surface
g_N	normal gap
t_N	magnitude of normal traction

Solution of Contact Problem

$\Pi, \Pi_{int}, \Pi_C, \Pi_{ext}$	total, internal, contact and external potential energy
$\delta\Pi, \delta\Pi_{int}, \delta\Pi_C, \delta\Pi_{ext}$	variation of total, internal, contact and external potential energy
V	function space of admissible mechanical deformations
$\delta\varphi$	admissible mechanical deformation
Π_N, Π_T	normal, tangential contact potential
σ_{ij}	stress tensor
$\delta\epsilon_{ij}$	virtual strain
δu_i	admissible virtual displacement
t_i^N	normal contact traction
ϵ_N	normal penalty parameter
ϵ_T	tangential penalty parameter
\mathbf{F}	body forces
$\mathbf{f}, \mathbf{f}_{int}, \mathbf{f}_C, \mathbf{f}_{ext}$	total, internal, contact and external force
\mathbf{N}	basis function
$\mathbf{t}_C, \mathbf{t}_N, \mathbf{t}_T$	contact, normal, tangential traction vector
\mathbf{u}_s^e	element displacement in slave domain
\mathbf{u}_m^e	element displacement in master domain
Γ_C^2	candidate contact surface

Normal contact

\mathbf{n}_p	normal at projection point
d_N	negative of the normal gap
\mathbf{N}_s	basis functions at slave elements
$c_p^{\alpha\beta}$	transformation matrix
\mathbf{a}_\bullet^p	tangent vector at projection point along \bullet direction
$\hat{\mathbf{a}}_\bullet^p$	unit tangent vector at projection point along \bullet direction
κ_\bullet	curvature along \bullet direction

Tangential contact

f_{slip}	slip criterion function
μ	friction coefficient
p	magnitude of normal traction
$\dot{\mathbf{g}}_T$	relative sliding velocity
\mathbf{n}_T	unit vector in tangential direction
$\mathbf{t}_{T,n+1}^{trial}$	trial tangential traction - elastic predictor
ξ_p	parametric coordinate of projection point
ξ_{sl}	parametric coordinate from last time step of final sliding point
\mathcal{D}	dissipation
\mathcal{L}	Lie derivative
\mathbf{a}_p^\bullet	contra-variant tangent vector at projection point along \bullet direction

Results

dt	time-step
\mathcal{H}	Heaviside function

Acronyms

ACM	Aeroelastic Coupling Module
AME	Added Mass Effect
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CCM	Computational Contact Mechanics
CFD	Computational Fluid Dynamics
CSM	Computational Solid Mechanics
FEA	Finite-Element Analysis
FEAFA	Finite Element Analysis For Aeroelasticity
FSCI	Fluid-Structure-Contact Interaction
FSI	Fluid-Structure Interaction
IGA	Isogeometrical Analysis
NURBS	Non-Uniform Rotational B-Splines

List of Figures

1	The General Setup of FSCI Problem	3
2	The Space-Time Slab for DSD/SST Formulation.	4
3	Current and Previous Configuration.	5
4	Coupling Conditions.	7
5	Example of Matching and Non-matching Interfaces.	9
6	Non-matching Interfaces - Nearest Neighbour Method.	10
7	Point Projection.	10
8	Finite Interpolation Method - Load Transfer.	11
9	Weakly - Strongly Coupled Scheme.	11
10	Scheme of the Dirichlet-Neumann Coupling.	12
11	Simplified Illustration of the Principle of AME.	14
12	Anatomy of the General Engineering Analysis Process	15
13	Comparison of FE and IGA Spatial Discretization.	15
14	Illustration of Scattered Gauss Point over the Slave Contact Surface.	18
15	Closest Point Detection.	19
16	Projection Point - Errors.	20
17	Axis-aligned Bounding Box Method.	21
18	Contact Detection - Drawbacks.	21
19	Physical Interpretation of Penalty Method.	25
20	Interpretation of Slip Criterion Function.	29
21	Sliding Rectangular - Problem Setup.	36
22	Sliding Rectangular - Phases.	37
23	Sliding Rectangular - Horizontal and Vertical Contact Forces.	39
24	Sliding Rectangular - Influence of the External Load on the Number of Newton-Raphson Iterations.	39
25	Sliding Rectangular - Displacement Over Time-Steps.	40
26	Sliding Rectangular - Visualization of the Displacement.	40
27	2D Shallow Ironing Problem - Setup.	41
28	2D Shallow Ironing Problem - $t = 1$ s	42
29	2D Shallow Ironing Problem - $t = 1.2$ s	42
30	2D Shallow Ironing Problem - $t = 1.6$ s	42
31	2D Shallow Ironing Problem - $t = 2$ s	43
32	2D Shallow Ironing Problem - Stress Field from [14].	43
33	2D Shallow Ironing Problem - Horizontal and Vertical Contact Forces.	44
34	2D Shallow Ironing Problem - Horizontal and Vertical Contact Forces from the Benchmark Computation.	44
35	2D Shallow Ironing Problem - Number of Newton-Raphson Iterations throughout the Computation.	45

36	Sensitivity Analysis - Influence of Lower Penalty Parameters - Horizontal and Vertical Forces.	47
37	Sensitivity Analysis - Influence of the Lower Penalty Parameters $t = 2$ s.	47
38	Sensitivity Analysis - Influence of Lower Penalty Parameters on Number of Newton-Raphson Iterations.	48
39	Influence of the Ironing Depth - Components of the Contact Force.	49
40	Influence of the Ironing Depth - Newton-Raphson Iterations.	49
41	3D Computation - Setup	51
42	3D Ironing - Contact Force.	52
43	3D ironing - Newton-Raphson Iterations.	52
44	3D Ironing - Force in z-direction $t = 0$ s.	53
45	3D Ironing - Force in z-direction $t = 0.25$ s.	53
46	3D Ironing - Force in z-direction $t = 0.5$ s.	54
47	3D Ironing - Force in z-direction $t = 0.75$ s.	54
48	3D Ironing - Force in z-direction $t = 1$ s.	55
49	3D Ironing - Force in z-direction $t = 1.1$ s.	55
50	3D Ironing - Force in z-direction $t = 1$ s.	56
51	3D Ironing - Force in z-direction $t = 1.3$ s.	56
52	3D Ironing - Force in z-direction $t = 1.35$ s.	57

List of Tables

1	Differences Between IG and FE Analysis.	17
2	Common Features Shared by IG and FE Analysis.	17
3	Sliding Rectangular - Problem Setup.	37
4	Benchmark 2D Shallow Ironing Problem with Friction - Setup Parameters.	40
5	Influence of the Time-Step Value	50
6	3D computation - Setup Parameters.	50

List of Algorithms

1	Frictional Contact Workflow - Commented	33
---	---	----

1 Introduction

Engineering world is getting more and more sophisticated every day. The trends change almost on a daily basis and production tries to keep up. One of the many moving forces is the rise and ever growing applicability of **Computer Aided Engineering** (CAE). Mathematical modelling is omnipresent: From engineering world - car industry, production engineering, aeronautics, through electrical engineering, physics all the way to medicine.

Mathematical modelling provides us with an insight on every scale, at any condition imaginable, at any time of the process. The scope of the computation ranges from particle physics, description of the Brownian motion, through direct numerical simulation of turbulent flow, up to the formation of clouds and weather forecasting. For each of these, CAE uses a mathematical description that, with respect to the more or less simplifying assumptions, offers the user a more or less accurate prediction of the studied phenomenon. In contrast to experiments, mathematical models are much less constrained with respect to the applied condition such as pressure, temperature, velocities or densities. Through the mathematical lens we can predict a temperature at places where the direct measurement is impossible due to the economical or technological difficulties, or we can test our designs in pressures or velocities that are only possible in outer space.

While in some industries mathematical modelling creates completely new technologies, in others it helps to optimize processes, that are known to the human kind for many decades, in some cases even for centuries. Perfect example would be metal shaping with rolling mills.

First sketch of a metal rolling mill is believed to come from Leonardo da Vinci around 1485. The evolution from then made a huge leap forward in terms of variability of mechanisms, ever-growing range of metals suitable for this shaping process or in terms of automation of this technology.

This thesis is meant as a part of bigger project, which aims to truthfully model *temper rolling*. From engineering point of view, temper rolling is a cold rolling method, which is mainly used to improve the flatness, minimize stretching or impart specific surface texture, depending on the demands of the end user. During this process, the thickness of the processed material usually decreases by 0.1 - 5%. To prevent the metal from sticking to the rollers of the rolling mill, lubricant is used. This lubricant is usually a mixture of water (around 95%) and oil.

From a mathematical point of view, this process represents a rather challenging problem extending over many separate fields of research. In fact, the mathematical modelling theory involved comes from three separate fields: 1) **Computational Fluid Dynamics** (CFD) models the inflow of the lubricant over the rollers as well as the outflow

from the working area. 2) **Fluid Structure Interaction** (FSI) techniques are utilized in case a fluid volume gets captured between the rollers and the metal sheet. In such case, the biggest challenge is to describe the sudden peaks in pressure on the roller as well as the metal sheet caused by the incompressible lubricant. The main purpose of temper rolling is the change of the metal surface and thickness. 3) This interaction is described by the mathematical theory for the **Computational Contact Mechanics** (CCM). The main objective is to capture the interaction of two (or more) solids and their subsequent deformation, which is described through plasticity models of each body, e.g., in case of this thesis, we assume the rollers to be rigid. This specific combination of fields is called **Fluid-Structure-Contact Interaction** (FSCI).

This thesis is designated to CCM, in particular, to the modelling of **frictional contact**. Nevertheless, in order to sustain the extensiveness of FSCI problems, basic theory behind each of the discussed topics will be provided. Namely, governing equations for CFD are introduced in Section 2.1.1. General solid mechanics for large deformations will be discussed in Section 2.1.2. Since **Fluid-Structure Interaction** (FSI) problematic is part of the whole FSCI concept, some basic theory on partitioned algorithms and their coupling is provided in Section 2.2. **Isogeometrical Analysis** (IGA) has been adopted as a spatial discretization for the solid body and therefore its basics are given in Section 2.3 along with some tips for additional reading. In Section 2.4 we discuss the overall idea and challenges of CCM. However, the main focus of this thesis is described in detail in Section 3. Section 4 gives a brief overview of the modules of the computational framework used and enhanced throughout this work. In the end of this thesis, in Section 5, we will demonstrate the implemented code and discuss its performance as well as possible improvements.

2 Theory

2.1 Fluid-Structure-Contact Interaction

Fluid-Structure-Contact Interaction (FSCI) is generally coping with 2 different computational fields. That is a *fluid domain* Ω_F on the one hand and a *structural domain* Ω_S on the other. However, unlike in **Fluid-Structure Interaction (FSI)** problems, we also need to take care of two different contact interfaces - *Fluid-Structure interface* Γ_{FS} and *Contact (Structure-Structure) interface* Γ_C , which will be discussed in 2.4.

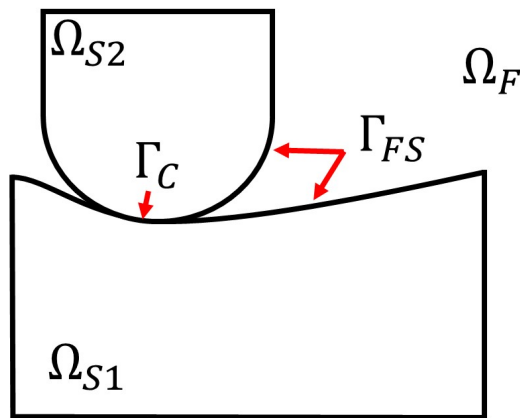


Figure 1: The General Setup of a Fluid-Structure-Interaction Problem.

2.1.1 Computational Fluid Dynamics

Computational Fluid Dynamics in FSCI, and within our computations, is governed by Navier-Stokes equations for incompressible Newtonian fluid. Their strong form for spatial domain Ω_F , its Lipschitz continuous boundary Γ_F . The system is closed by a set of initial and boundary conditions reads [8]

$$\mathbf{v}_t - \nu \nabla^2 \mathbf{v} + (\mathbf{v} \cdot \nabla) \mathbf{v} + \nabla p = \mathbf{b} \quad \text{in } \Omega_F \times (0, T), \quad (2.1a)$$

$$\nabla \cdot \mathbf{v} = 0 \quad \text{in } \Omega_F \times (0, T), \quad (2.1b)$$

$$\mathbf{v} = \mathbf{v}_D \quad \text{on } \Gamma_F^D \times (0, T), \quad (2.1c)$$

$$-p \mathbf{n} + \nu (\mathbf{n} \cdot \nabla) \mathbf{v} = \mathbf{q} \quad \text{on } \Gamma_F^N \times (0, T), \quad (2.1d)$$

$$\mathbf{v}(\mathbf{x}, 0) = \mathbf{v}_0(\mathbf{x}) \quad \text{in } \Omega_F. \quad (2.1e)$$

Therein, the kinematic viscosity $\nu \left[\frac{m}{s^2} \right]$ is a fluid parameter, while the velocity field is denoted as \mathbf{v} . \mathbf{b} stand for the body forces. The kinematic pressure $p = \frac{\tilde{p}}{\rho}$ are the unknowns. Γ_F^D and Γ_F^N denote segments of the domain boundary on which we have

imposed Dirichlet and Neumann boundary conditions respectively, such as the velocity on Dirichlet boundary v_D . \mathbf{n} stands for the outer normal of the domain boundary Γ and \mathbf{q} is the prescribed traction. Equation (2.1e) prescribes the initial velocity field at time zero \mathbf{v}_0 .

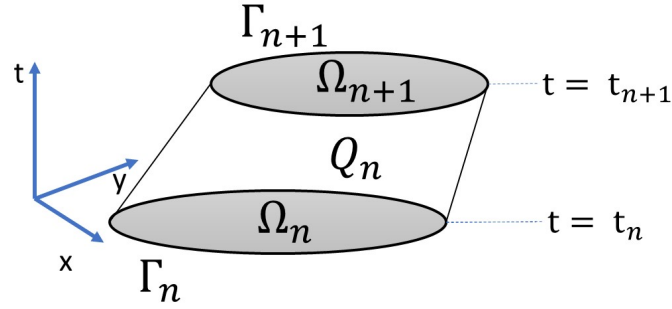


Figure 2: The Space-Time Slab for DSD/SST Formulation.

Figure 2 shows the temporal slab (t^n, t^{n+1}) . Q_n represents the space-time domain at time $t = n$, while Ω_n and Ω_{n+1} represent the spatial domain and Γ_n and Γ_{n+1} stand for its boundary at distinct time-levels.

2.1.2 Computational Solid Mechanics

The main goal of *Computational Solid Mechanics* (CSM) is to predict the deformation of a solid body caused by external loads. We typically obtain the solution in terms of displacements, which is defined as a difference in coordinates between the previous and deformed (current) configuration, $\mathbf{u} = \mathbf{x} - \mathbf{X}$.

Structural problems are governed by the *equation of motion*, which describes the balance of inner and outer stresses and has the following form

$$\rho \frac{D^2 \mathbf{u}}{Dt^2} = \text{div}(\boldsymbol{\sigma}) + \rho \mathbf{b}. \quad (2.2)$$

In Equation (2.2), we denote the material density as ρ , the Cauchy stresses as $\boldsymbol{\sigma}$, and external resultant body forces, e.g., gravity, acting on the surface as \mathbf{b} .

To solve Equation (2.2), we need to introduce additional relations, that will form a closed system. In FSCI problems, we expect rather large deformations, therefore, it is beneficial to employ a geometrically nonlinear structural model. This approach com-

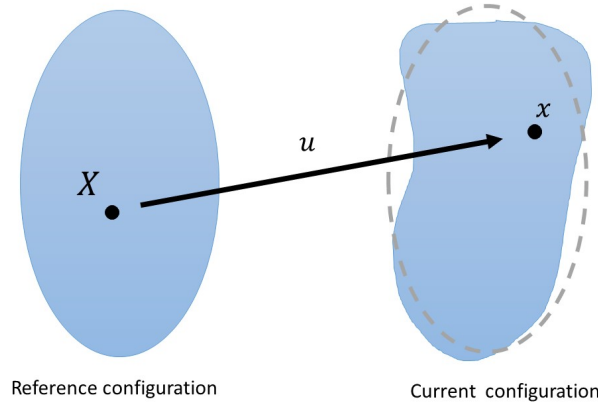


Figure 3: Deformed (current) and Undeformed (Previous) Configuration.

bined with a *total Lagrangian formulation*, yields the following strong form of our problem set:

$$\rho \mathbf{u}_{tt} = \text{Div}(\mathbf{F}\mathbf{S}) + \rho \mathbf{b}_0 \quad \text{in } \Omega_0^S \times (0, T), \quad (2.3a)$$

$$\mathbf{S} = \mathbf{C} : \mathbf{E} \quad \text{in } \Omega_0^S \times (0, T), \quad (2.3b)$$

$$\mathbf{E} = \frac{1}{2}(\mathbf{F}^T \mathbf{F} - \mathbf{I}) \quad \text{in } \Omega_0^S \times (0, T), \quad (2.3c)$$

$$\mathbf{u} = \tilde{\mathbf{u}} \quad \text{on } \Gamma_D^0 \times (0, T), \quad (2.3d)$$

$$\mathbf{F}\mathbf{S}\mathbf{N} = \bar{\mathbf{t}}_0 \quad \text{on } \Gamma_N^0 \times (0, T), \quad (2.3e)$$

$$\mathbf{u}(\mathbf{X}, t = 0) = \mathbf{u}_0(\mathbf{X}) \quad \text{in } \Omega_0^S. \quad (2.3f)$$

In Equation (2.3) we denote the second Piola stress tensor as \mathbf{S} , the material matrix as \mathbf{C} . \mathbf{E} stands for the Green-Langrange strain tensor and \mathbf{F} denotes the deformation gradient. As mention above, we are interested in nonlinear behaviour, which is introduced in Equation (2.3c). To have a full initial-boundary value problem, we introduce Dirichlet and Neumann boundary condition in Equation (2.3d) and (2.3e) along with our initial conditions in Equation (2.3f), which is usually set to zero.

To solve a structural problem, we first transform the strong form to a weak form through the *principle of virtual work*. The problem statement reads:

$$\begin{aligned} &\text{Find } \mathbf{u}(\mathbf{X}, t \in S^t = \{\mathbf{u} \in \mathbf{H}^1 \times (0, T) | \mathbf{u} = \tilde{\mathbf{u}}\} \quad \text{on } \Gamma_D^0 \forall t \quad \text{such that} \\ &\text{for } \forall \delta \mathbf{u} \in S_0^t \text{ holds} \\ &\partial \Pi(\mathbf{u}, \delta \mathbf{u}) = \int_{\Omega_0^{S_0}} \mathbf{S} : \delta \mathbf{E} dV + \int_{\Omega_0^{S_0}} \rho(\mathbf{u}_{tt} - \mathbf{b}_0) \cdot \delta \mathbf{u} dV - \int_{\Gamma_N^0} \bar{\mathbf{t}}_0 \delta \mathbf{u} dA = 0. \end{aligned} \quad (2.4)$$

We simply look for a combination of inner and outer stresses that would balance out for any virtual displacement $\delta \mathbf{u} \in S_0^t = \{\mathbf{H}^1 \times (0, T) | \delta \mathbf{u} = 0 \text{ on } \Gamma_D^0 \forall t\}$, where \mathbf{H}^1 denotes Sobolev's space $W^{1,2}$ [29], with a norm defined in Equation (2.5).

$$\|\mathbf{f}\|_{W^{1,2}} = \|\mathbf{f}\|_{H^1} = \left(\int_{\Omega} |\mathbf{f}|^2 + |\nabla \mathbf{f}|^2 \right)^{1/2} \quad (2.5)$$

The spatial discretization of the weak formulation in Equation (2.4) is done by *isogeometric analysis*. The basics of IGA are discussed in Section 2.3. This step yields a system of time-dependend nonlinear ordinary differential equations, that can be expressed in matrix form as

$$\mathbf{M}^{t+\Delta t} \ddot{\mathbf{U}}^{(i)} + {}^{t+\Delta t} \mathbf{K}(\mathbf{U}) {}^{t+\Delta t} \mathbf{U} = {}^{t+\Delta t} \mathbf{F}, \quad (2.6)$$

where we know the displacement field \mathbf{U} at time level t and we want to determine the field for $t + \Delta t$. Above, we denote the mass matrix as \mathbf{M} , the stiffness matrix as $\mathbf{K}(\mathbf{U})$ and the right-hand side vector is represented by \mathbf{F} . The upper left indices denote the associated time level.

The second term shows the non-linear behavior with respect to the displacement field \mathbf{U} . Therefore, we use an iterative procedure, in particular Newton - Raphson method, to linearize the matrix form (Equation (2.6)) [1]:

$$\mathbf{M}^{t+\Delta t(i)} + {}^{t+\Delta t} \mathbf{K}^{t+\Delta t} \mathbf{U} = {}^{t+\Delta t} \mathbf{F} - {}^{t+\Delta t} \mathbf{R}^{(i-1)}, \quad (2.7a)$$

$${}^{t+\Delta t} \mathbf{U}^{(i)} = {}^{t+\Delta t} \mathbf{U}^{(i-1)} + \Delta \mathbf{U}^{(i)}. \quad (2.7b)$$

The iteration levels are denoted as upper right indices. ${}^{t+\Delta t} \mathbf{R}^{(i-1)}$ represent the inner stresses determined in the previous iteration. The next time step is initialized by the converged solution from the preceding step by ${}^{t+\Delta t} \mathbf{U}^{(0)} = {}^t \mathbf{U}$, ${}^{t+\Delta t} \mathbf{K}^{(0)} = {}^t \mathbf{K}$ and ${}^{t+\Delta t} \mathbf{R}^{(0)} = {}^t \mathbf{R}$.

2.1.3 Coupling Conditions

In FSCI problems, the coupling conditions are used for the Fluid-Structure interaction. This set of equations create a link between the solutions in the structural and fluid domain through a specific set of boundary conditions at the FSI interface Γ_{FS} .

In general, we distinguish two types of coupling, that are used in different cases.

- *Volume coupled problems*: Problems, in which two or more problems share the same computational domain, or at least its part, e.g. combustion.
- *Surface coupled problems*: Coupling takes place on the interaction interface. We handle two, or more, spatial domains, e.g., FSCI.

FSCI problems belong to the latter one. The coupling equations in FSCI are sorted into two types [2]:

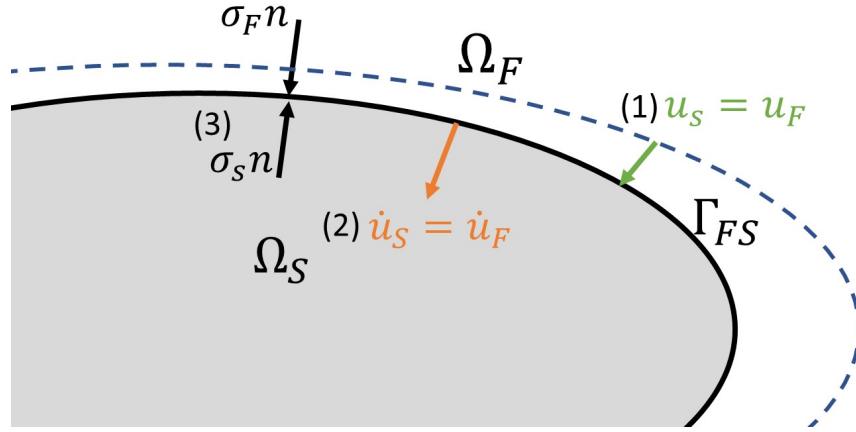


Figure 4: Coupling Conditions; (1) Displacement Coupling, (2) Velocity Coupling, (3) Stress Coupling.

Kinematic coupling conditions The kinematic coupling conditions imply, that at the interaction boundary, the displacement and the velocity of both interacting boundaries are equal. However, in general case Equation (2.8b) does not have to hold true[10, 19].

$$\mathbf{u}_F(\mathbf{x}, t) = \mathbf{u}_S(\mathbf{x}, t) \quad \forall t \geq 0, \mathbf{x} \in \Gamma_{FS} \quad (2.8a)$$

$$\dot{\mathbf{u}}_F(\mathbf{x}, t) = \dot{\mathbf{u}}_S(\mathbf{x}, t) \quad \forall t \geq 0, \mathbf{x} \in \Gamma_{FS} \quad (2.8b)$$

Dynamic coupling conditions The second type is the dynamic coupling condition, which couples stress over the shared interface.

$$\boldsymbol{\sigma}_S(\mathbf{x}, t)\mathbf{n}(\mathbf{x}, t) = -\boldsymbol{\sigma}_F(\mathbf{x}, t)\mathbf{n}(\mathbf{x}, t) \quad \forall t \geq 0, \mathbf{x} \in \Gamma_{FS} \quad (2.9)$$

Therein, $\boldsymbol{\sigma}_S$ and $\boldsymbol{\sigma}_F$ are stress tensors of the fluid and the structure respectively and \mathbf{n} is the normal vector of the coupled interface.

For FSI problems, the most beneficial coupling condition is through the conservation of mechanical power exchanged between the two domains involved. This approach diminishes the artificial creation or destruction of energy, which could lead to non-physical results. This approach is modified *Dirichlet-Neumann coupling*, which originally uses Equation (2.8a) as a boundary condition for the fluid domain and Equation (2.9)

as a boundary condition for the solid domain. This modified coupling condition is obtained by merging Equation (2.8b) and Equation (2.9) in one [3]. The final form, satisfying both conditions at once at all times t , reads

$$(\boldsymbol{\sigma}_F \mathbf{n}) \cdot \dot{\mathbf{u}}_F = -(\boldsymbol{\sigma}_S \mathbf{n}) \cdot \dot{\mathbf{u}}_S \quad \forall t \geq 0, \mathbf{x} \in \Gamma_{FS}. \quad (2.10)$$

2.2 Partitioned Algorithms

At the *Chair of Computational Analysis for Technical Systems (CATS)* at RWTH Aachen University, the ansatz of a **partitioned algorithm** is pursued in order to simulate problems of fluid-structure-contact interaction. This section will give a brief introduction to the underlying concept of partitioned algorithms along with its motivation and strengths as well as its downsides and shortcomings.

2.2.1 Solution Approaches for FSCI

Numerical simulation of coupled systems, i.e. FSCI, can be solved by employing either **monolithic** or **partitioned** solver [17].

Monolithic solvers first discretize all equations of FSCI problem along with its boundary conditions into one system, which is then solved by one solver at once. The biggest advantage of such a solver is that it does not introduce any instabilities due to splitting and partitioning steps, e.g., added-mass effect, and is thus stable and consistent. On the other hand, developing such a solver is an extremely tedious job. In fact, for most applications, one would need to create a brand new solver from a scratch for the particular problem in hand. Available solvers for FSCI are extremely rare compared to standalone solvers for CFD and CSD.

Having variety of different solvers for each part of the FSCI problem is the main motivation for partitioned solvers. The idea is to use specialized modules for each part of the partitioned problem and an adequate coupling algorithm, that transfers the solution data from one part into the right hand side of the other. In principle, we could use any combination of solvers based on the nature of the problem at hand. While having great success in terms of applicability in wide range of problems, partitioned algorithms are prone to instabilities, which stem mainly from the coupling procedure [7].

2.2.2 Transfer of Loads

The transfer of solution data from one domain to the other is done on two levels - spatial and temporal coupling.

Spatial coupling In FSI computations, we have two distinct meshes - one on the fluid side and a second on the structural side. Each of which needs to take care of different properties of the numerical problem. For example, in the fluid domain, we may want to capture boundary layers, while in the structural domain we may be interested only in the deformation. This may result in a significantly finer mesh on the fluid side. This leads to having two meshes with, in general, non-matching interfaces. This raises the question how to transfer the data from one domain to the other. In the special case of a *matching interface*, the process is somewhat easy. We exchange the data either via *simple integration*, which yields consistent nodal forces, or through *consistent boundary flux* [5]. Unfortunately, most of the cases have non-matching interfaces. Then, employing some sort of interpolation method is necessary.

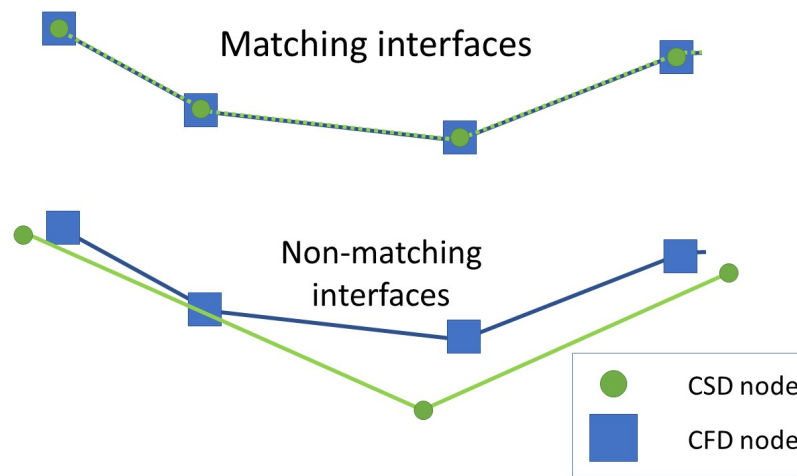


Figure 5: Example of Matching and Non-matching Interfaces.

One of the most naive approaches would be the *nearest neighbour interpolation*. The aim is to transfer load from position x_a on the CFD mesh to the CSD surface. The algorithm first searches the closest neighbour, in this case position x_b and assigns the full value of the applied force to the CSD node. Such an approach is easy to implement, however shows severe inconsistency.

The FSCI solver developed at CATS, RWTH Aachen, employs the *Finite Interpolation method*: Its first step is the orthogonal projection of a node onto the second surface [26].

The projection point needs to satisfy the following condition:

$$F(\xi) = (\mathbf{x}_F - \mathbf{x}_p) \cdot \mathbf{t} \stackrel{!}{=} 0, \quad (2.11)$$

where \mathbf{t} stands for the tangential vector and \mathbf{x}_F , \mathbf{x}_S are the coordinates of projecting and projected point. Once the projection point is found (i.e., Equation (2.11) is solved), the forces are transformed through weighting functions as depicted at Figure 8.

The very same steps are used in the other direction, projecting deformations of the solid body on to the fluid domain, however in the opposite direction.

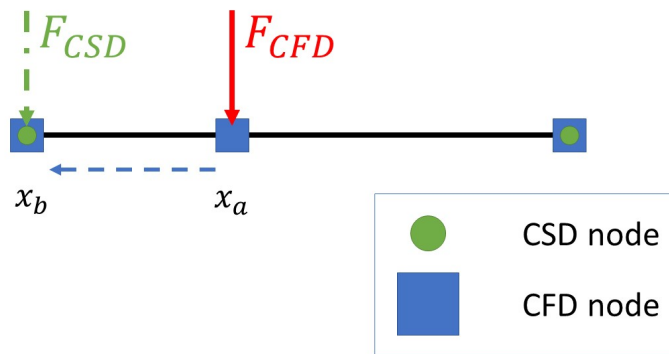


Figure 6: Non-matching Interfaces - Nearest Neighbour Method.

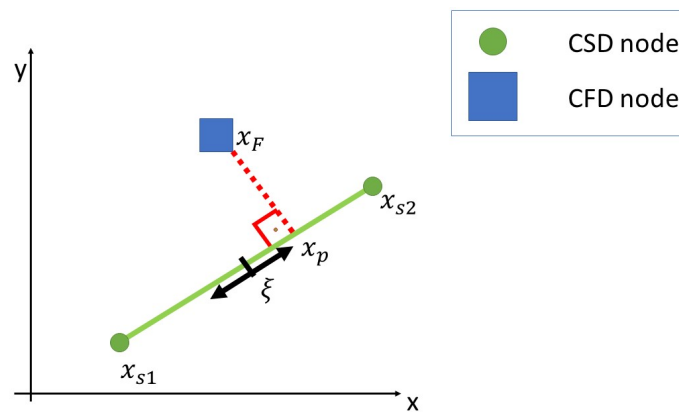


Figure 7: Point Projection.

Temporal coupling For time-dependent problems, such as temper rolling, we need to ensure the transfer of data between distinct time-levels. Therefore, we introduce a temporal discretization and a temporal coupling. The coupled nature of the problem requires us to perform the temporal coupling of the system only in distinct time levels, such that the coupled data come from the same physical time for both stand-alone solvers. Depending on the level of interaction between the fluid and the structure, we can either use **weak** or **strong** coupling. While weak coupling is mostly sufficient for slight interaction between both systems, e.g., aeroelastics, for other applications, such as FSCI problems, it introduces significant errors. The error stems from the underlying concept pictured at the following diagram. Excluding the 5th step, same diagram is relevant even for weak coupling.

- 1 Prescription of solid deformation in time $n+1$ such that $\hat{D}^{n+1} = D^n$
- 2 Computation of the mesh displacement, computation of the fluid field in the time level $n+1$ based on \hat{D}^{n+1}

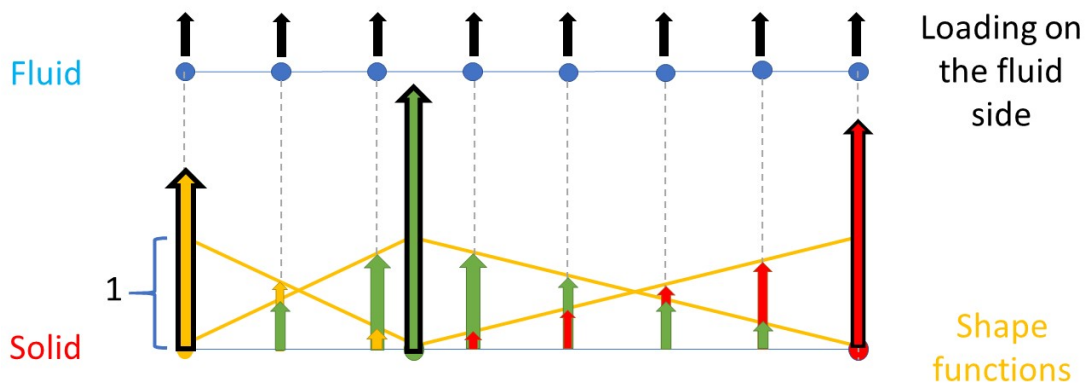


Figure 8: Finite Interpolation Method - Load Transfer.

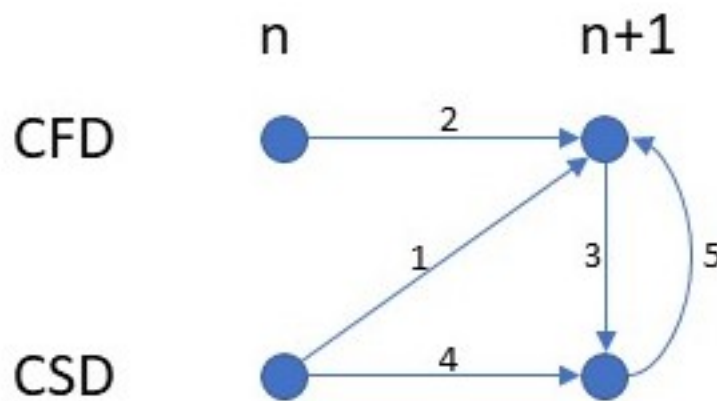


Figure 9: Weakly - Strongly Coupled Scheme.

- 3 Computation of loads acting on the structure
- 4 Adjustment of the solid deformation D^{n+1} .
- 5 If $|D^{n+1} - \hat{D}^{n+1}| \geq \epsilon$, we need to take another loop. We submit D^{n+1} as "new" \hat{D}^{n+1} and start the loop from point 2 until \hat{D}^{n+1} converges to D^{n+1}

In the first coupling iteration, the algorithm uses the solution from the previous time step as a initial guess for the next time step. For weakly coupled problems with strong interaction between the fields this solution may be far from converged solution and

thus introducing significant errors, which could eventually lead to a violation of coupling condition. Incorporation of the 5th step in the diagram above means a shift from weak to strong coupling scheme. In this step, we check the convergence of the newly obtained solution. In order to fully avoid the error from the coupling, one would need to theoretically take infinitely many iterations. This is, of course, computationally impossible, therefore we define a convergence criterion, with respect to which we minimize the error production in somewhat controlled manner. Such solution converges to the solution of the monolithic solver up to the predefined precision.

2.2.3 Dirichlet-Neumann Coupling Scheme

Contemporary standard among coupling approaches for FSCI problems is the **Dirichlet-Neumann** coupling scheme. It uses the kinematic coupling condition to transfer the data, in form of displacements, from the structural body to the fluid solver. For the inverse data transport we employ the dynamic coupling condition from the fluid part to the solid part. The transferred data enforces the load on the structural degrees of freedom as a Neumann boundary condition [25].

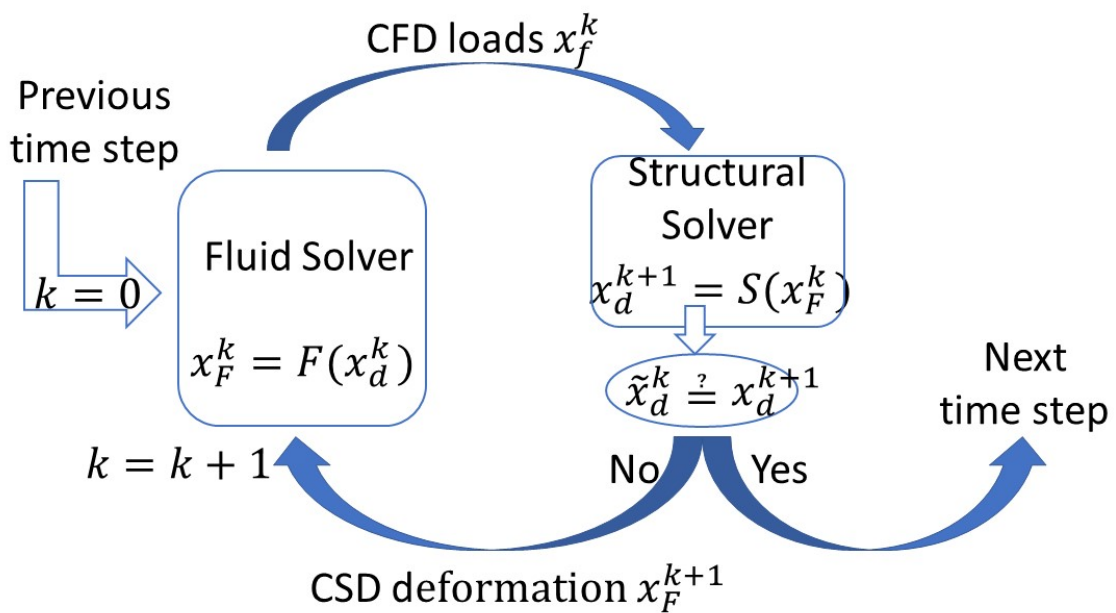


Figure 10: Scheme of the Dirichlet-Neumann Coupling.

2.2.4 Added-Mass Effect

As mentioned in Section 2.2.1, partitioned algorithms suffer from error production due to the splitting and partitioning steps, namely the **Added-Mass Effect** (AME). AME contributes through artificially increasing the inertia terms to the overall instability of the numerical system. This unfortunate phenomenon stems from the fact that fluid forces are computed on the basis of "guessed" structural interface displacement, which subsequently yields incorrect coupling forces. It has been proved in [9], that any sequentially staggered schemes for incompressible flows will get unstable provided the mass (density) ratio, of the fluid and the structure, is large enough. For weak coupling schemes, the coupling step is performed once per time step. Therefore the process is, in a sense, explicit and thus this error will be prominent despite the possibly implicit fluid or structural solvers.

A discrete mathematical quantifier of the AME for FSI problems has been derived in [9]. Simplifying assumptions, such as conforming discretization along the FS boundary, has been imposed (this is, however, not the case in this thesis). Nevertheless, it is still worth demonstrating the idea behind this phenomenon. The following *instability condition* for schemes with partial or full recursion¹ respectively has been derived [15].

$$\frac{m^F}{m^S} \max_i \mu_i > C \quad (2.12a)$$

$$\frac{m^F}{m^S} \max_i \mu_i > \frac{C}{n} \quad \forall n > 1 \quad (2.12b)$$

Therein, C is some constant dependent on the type of predictor employed for the computation, n stands for number of time steps and μ_i represents the eigenvalues of the system. From this condition and its derivation in [9] we may deduce the following:

- Mass ratio $\frac{m^F}{m^S}$, is the major factor to identify the stability of the computation. This leads to a conclusion that some combinations of material are numerically "forbidden".
- Counter-intuitively, the AME is more prominent as the time step Δt is decreased. This fact has a serious impact. It is, in fact, impossible to achieve a unconditional *stability* and an absolute *accuracy* at the same time, i.e., it is necessary to make a compromise between the accuracy and the stability.

The influence of the *Added Mass Effect* on the numerical solution is illustrated at Figure 11. As an example we used one of the common test cases in FSI problematic - flutter in the airflow. The dashed line represents the reference state at time level t^n . The dotted line

¹Schemes, that use the data from all previous time-steps in order to evaluate the new time-step.

is the unknown exact deformation due to the fluid-structure interaction. The full line, however, displays the obtained solution.

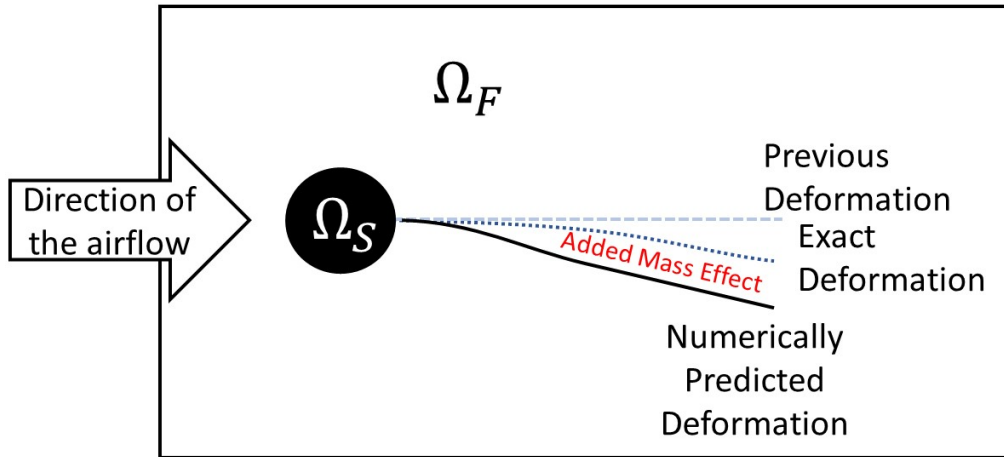


Figure 11: Simplified Illustration of the Principle of AME.

2.3 Isogeometrical Analysis

The spatial discretization of the structural problem described in Section 2.1.2 has been performed by means of *Isogeometrical Analysis* (IGA) introduced by Hughes et al. [4, 11]. This approach has been employed in order to provide exact a description of the discretized body and thus ensure higher precision of the whole computation. IGA, in fact, has become the golden standard for - *Computational Contact Mechanics*.

Most commercial and industrial finite-element codes discretize the problem with elements of the lowest degrees. The reason are the difficulties connected with higher order elements in terms of computational complexity (increasing number of nodes per element) or smoothness requirements on the element boundaries. These and many other problems may result in convergence problems. Another substantial problem of classical FE discretization is the FEA model creation itself. It turns out that adjusting the CAD model, such that it is suitable for FE discretization, and the subsequent FEA model creation takes up to 80% of the overall computational analysis of a problem [4]. Workflow of the typical FE simulation is depicted in Figure 12. However, the biggest shortcoming of FE discretization, is the approximate nature of the discretized interfaces. Partially due to the adjustments done to the CAD model as a pre-processing step before the mesh creation and partially due to the low order elements that are insufficient to model elaborate curvatures properly.

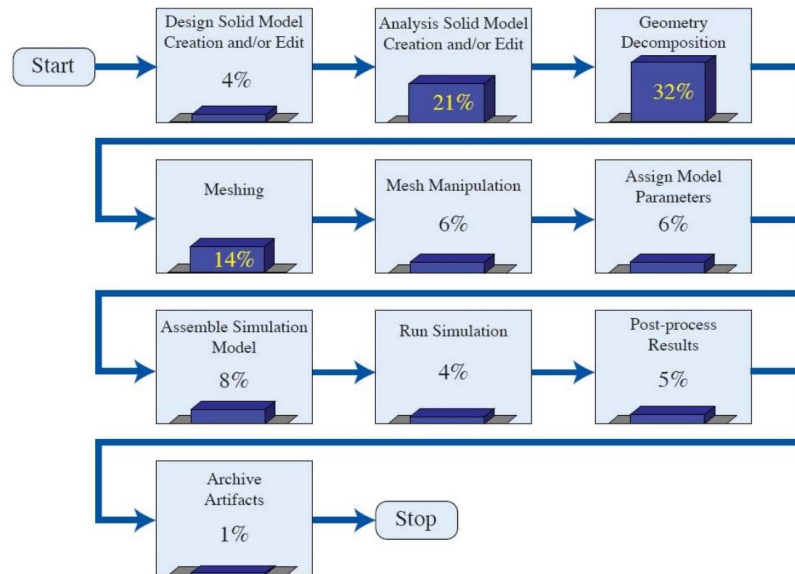
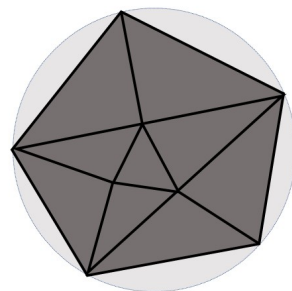
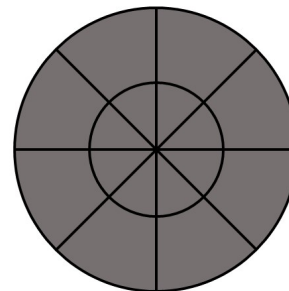


Figure 12: Anatomy of the General Engineering Analysis Process [4].

The idea of isogeometrical analysis is to substitute the classical polynomial basis functions. The motivation is to surpass the problematic part of standard FE approach, the FEA model creation, by reconstituting the FE analysis with in-CAD geometry. The intuitive and, to some extent, obvious choice for the basis functions is to adopt the very same basis as for the CAD model - *Non-Uniform Rational B-Splines (NURBS)*.



Finite Element Method



Isogeometric Analysis

Figure 13: Comparison of FE and IGA Spatial Discretization.

Non-Uniform Rational B-Splines Unlike classical FE basis functions, NURBS basis functions are usually not interpolatory. For body description, we define two distinct types of mesh - *control mesh* and *physical mesh*.

- The **control mesh**, defined by *control points*, generate a scaffold of the physical mesh. It resembles the standard FE discretization.

- The **physical mesh** decomposes the actual geometry through patches and knots, which are the equivalent to the FE elements. The basis functions are infinitely smooth in these elements (C^∞) and the over-the-boundary smoothness is defined as C^{p-m} , where p represents the polynomial degree and m stands for the multiplicity of the knot.

From mathematical point of view, NURBS are linear combination of n control points \mathbf{B}_I and with them associated basis functions $N_{I,p}(\xi)$, where p represents the degree of the NURBS and ξ stands for the coordinate in parametric space [4]. For higher dimensions, we create the basis function as simple product of its one-dimensional counterparts as seen in Equation (2.14b). The forms of 1D and 2D basis functions are shown in Equations (2.13) and (2.14a).

$$\mathbf{C}(\xi^1) = \sum_{I=1}^n N_I^p(\xi^1) \mathbf{B}_I \quad (2.13)$$

$$\mathbf{S}(\xi^1, \xi^2) = \sum_{I=1}^n N_I^{p,q}(\xi^1, \xi^2) \mathbf{B}_I \quad (2.14a)$$

$$N_I^{p,q}(\xi^1, \xi^2) = N_I^p(\xi^1) N_I^q(\xi^2) \quad (2.14b)$$

The NURBS definition of basis function is adopted from the definition of B-Splines. Therefore, we may generate basis function of any order (theoretically) through recursion.

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \quad (2.15a)$$

$$N_{i,0}(\xi) = \begin{cases} 1 & \xi_i \leq \xi \leq \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \quad (2.15b)$$

While Equation (2.15a) defines the recursive relation for the p^{th} order of the basis function, Equation (2.15b) is the starting piece-wise constant basis function for $p = 0$. ξ_i in the Equation (2.15a) stands for the parametric coordinate of i^{th} knot, that are either uniformly or non-uniformly distributed through the parametric space ($\xi_i \in [a, b]$). Knot coordinates are gathered in non-decreasing manner in 1D *knot vector* $\Theta \in \mathcal{R}^{n+p+1}$, where the superscript defines its length. This vector does define the characteristic behaviour of the resulting NURBS basis. The *knot vector* plays a significant role in terms of continuity of the basis functions at distinct knots.

For more in-depth information on IGA see [4].

Isogeometric Analysis	Finite Element Analysis
Exact geometry	Approximate geometry
Control points	Nodal points
Control variables	Nodal variables
Basis does not interpolate control points	Basis do interpolate the nodal points
NURBS basis	Polynomial basis
High, easy controlled continuity	C^0 -continuity, always fixed
hpk-refinement space	hp-refinement space
Point-wise positive basis	Basis not necessarily positive
Convex hull property	No convex hull property
Variation diminishing in the presence of discontinuous data	Oscillatory in the presence of discontinuous data

Table 1: Differences Between IG and FE Analysis [4].

Isogeometric analysis and Finite Element analysis
Isoparametric concept
Galerkin's method
Code architecture
Compact support
Bandwidth of matrices
Partition of unity
Affine co-variance
Patch test satisfied

Table 2: Common Features Shared by IG and FE Analysis [4].

2.4 Computational Contact Mechanics

Computational Contact Mechanics (CCM) deals with two or more solid bodies interacting with each other. In principle, it could be even only one body interacting with itself. Main challenges of CCM are:

- Detection of contact regions,
- Way of addressing numerical penetration of contact surfaces,
- Accounting for friction between surfaces.

CCM recognizes two types of bodies - **master** and **slave**. While slave bodies are deformable, the master bodies may be deemed as deformable or rigid. Master bodies are usually used to enforce a deformation to the slave body.

2.4.1 Contact Detection

A *Gauss Point-To-Segment* method has been used for the contact discretization. This discretization creates a link between a Gauss-point on the slave surface and a corresponding segment of the other surface, of the master surface, as shown at Figure 14. This method is valid for non-conforming meshes, problems with large deformations and large sliding. This method compared to similar methods, such as *Node-to-Surface* method, has the following advantages: Unlike in the case of *Node-to-Surface* method, which employs only the element nodes, the *Gauss-Point-to-Surface* approach scatters a number of Gauss-Points over the contact boundary. The advantage comes in a form of much more detailed description of the contact surface, which is especially advantageous for big surface deformations or curvatures. We further use the Gauss-points for the numerical integration. The downside of this approach comes at the computational price. Having more accurate description of the contact surface implicitly increases the computational costs.

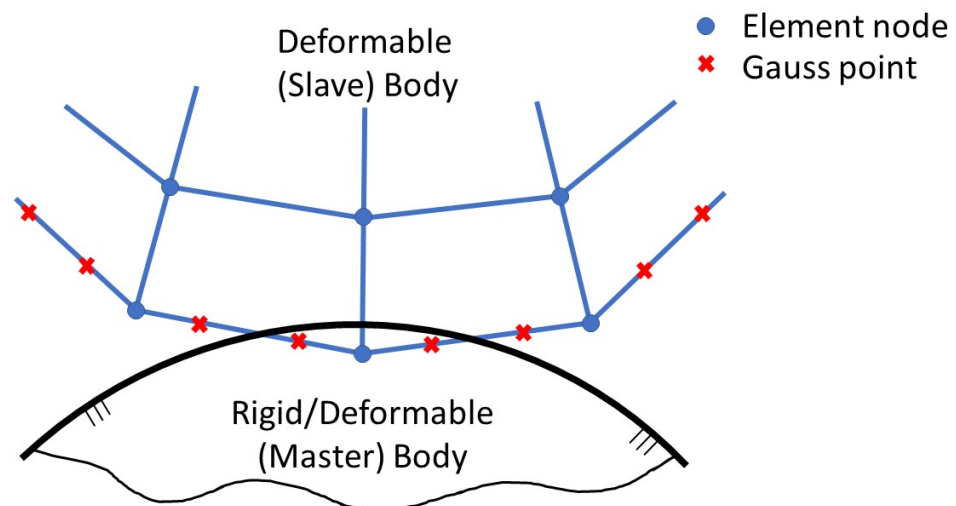


Figure 14: Illustration of Scattered Gauss Point over the Slave Contact Surface.

Finding contact regions effectively and with the least computational costs is of utmost importance. This process may be performed either at the beginning of each time step or at the beginning of each iteration (convergence step). The first approach, while being fast, with good convergence and stability, lacks accuracy due to the incremental displacements in convergence steps. The latter one provides better accuracy in exchange for low performance in terms of speed, convergence and stability. In this thesis, the contact is checked at the beginning of every Newton iteration.

The mathematical formulation of the contact detection is described by the following equation.

$$\rho^* \in \Gamma_m : \forall \rho \in \Gamma_m; |\mathbf{r}_s - \rho^*| \leq |\mathbf{r}_s - \rho| \quad (2.16)$$

Therein, ρ^* represents the closest point on the master surface, while \mathbf{r}_s represents the given slave node. In general case, we obtain non-linear minimization problem stated in Equation (2.17).

$$\min_{\xi \in [0,1]} F(\mathbf{r}_s, \xi) \longrightarrow \xi^* : \forall \xi \in [0, 1], |\mathbf{r}_s - \rho(\xi^*)| \leq |\mathbf{r}(\xi) - \rho|. \quad (2.17)$$

Equation (2.17) is equivalent to

$$(\mathbf{r}_s - \rho(\xi^*)) \cdot \left. \frac{\partial \rho}{\partial \xi} \right|_{\xi^*} = 0 \quad (2.18)$$

Equation (2.18) states that for the projection point \mathbf{r}_s the projection vector $\mathbf{r}_s - \rho(\xi^*)$ needs to be orthogonal to the tangential vector $\frac{\partial \rho}{\partial \xi}$ or at least very close to being orthogonal, i.e., usually projection tolerance angle is defined - $\epsilon \ll 1$. At Figure 15 we can see the final state of projection point search. It is important to note, that projection does not always have to exist or does not necessarily have to be unique. This is the case for surfaces with kinks or convex sections as illustrated at Figure 16.

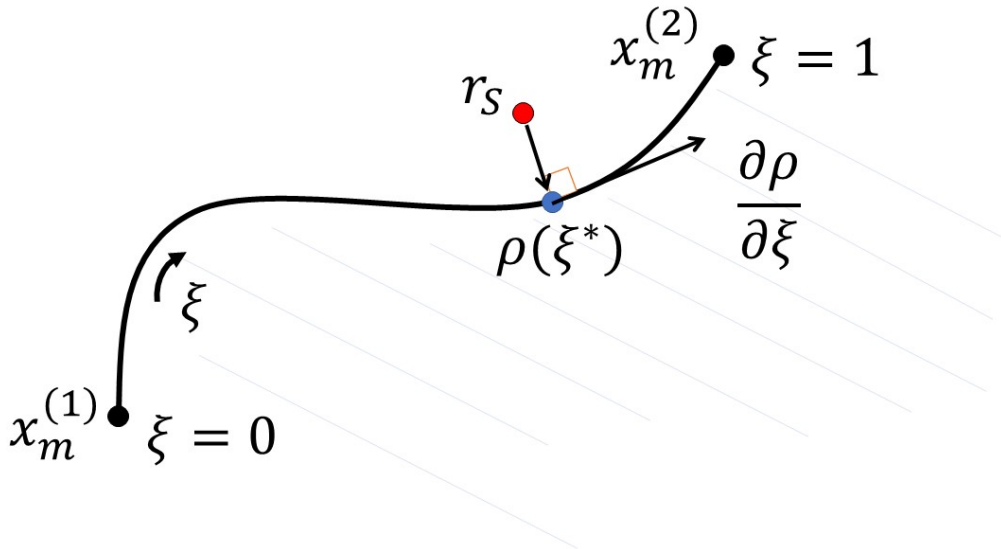


Figure 15: Closest Point Detection.

Brute force methods The most naive method are based on brute force, such as *All-to-all detection - Maximal Detection Distance Concept*. While being simple to implement, this

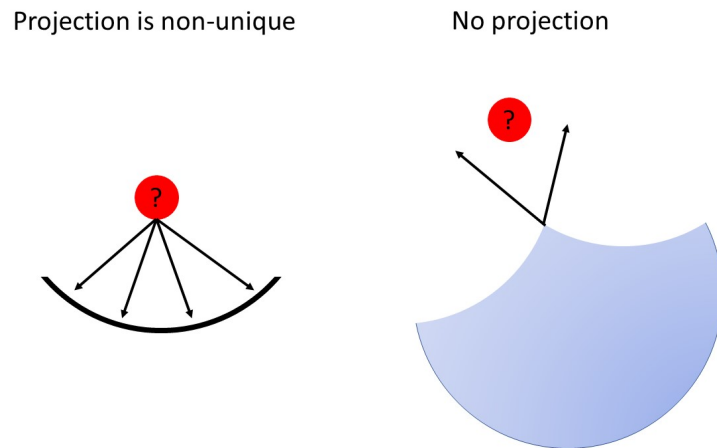


Figure 16: Projection Point - Errors.

simplification comes at high computational price. Checking the contact criterion (distance) between all computational nodes on the surface of one solid with respect to all surface nodes of the second solid is rather inconvenient. This step requires $O(nm)$ operations, where n and m is the number of the contact nodes of two distinct surfaces. This procedure can easily become the most expensive step in the whole simulation(!). Second downside of this approach is the fact that this kind of detection techniques do have **blind spots**, i.e., it is possible to miss a node especially by surfaces with high curvature[28]. These shortcomings has been a great motivation for faster, more efficient algorithms, ideally without blind spots.

Bounding Box method One of the more elaborate search strategies is the *Bounding Box method*. This approach has two phases. The first phase is a *coarse (global) search*, where we identify possible contact regions, i.e., we pinpoint contact nodes, where interaction is expected, with the help of *bounding box algorithm*. In this software framework, *axis-aligned bounding box* method has been implemented. The algorithm looks for the top-, right-, left- and bottom-most points of the whole domain, connects them and creates a rectangular bounding box around one body.

This approach is not the most efficient. Mainly, because it triggers the second phase of contact detection in many more cases than actually necessary. As depicted at Figure 18, two bounding boxes overlap, but in fact no collision will occur, since both systems are still far apart. Nevertheless, the local search has been triggered anyway.

If the global search detects the overlap of two (or more) bounding boxes, the second phase, *local search - Gauss-point-level search*, of the contact detection is triggered. Since we have already narrowed the search down to a fraction of the computational nodes it is plausible to use brute force search. In fact, that is usually the way. The goal of the local search is to find distance between the projection point and the projected point. Then, depending on the positive or negative value of the normal gap shown

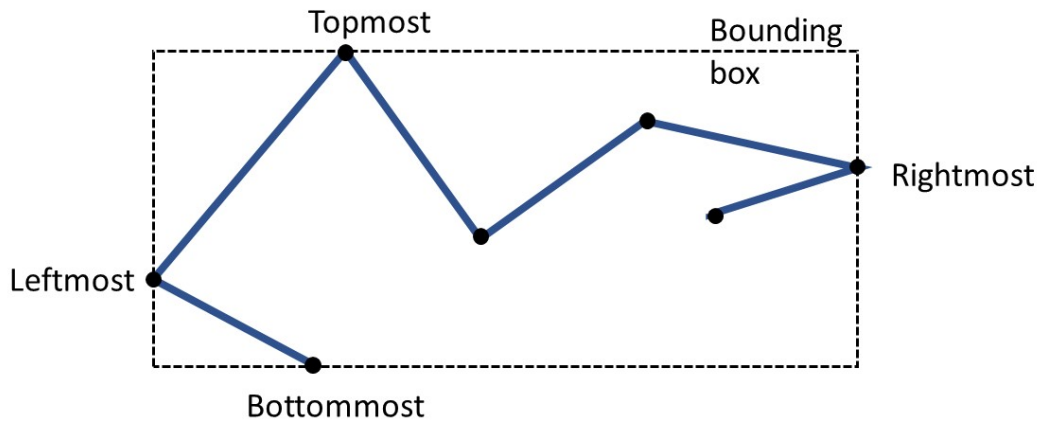


Figure 17: Axis-aligned Bounding Box Method.

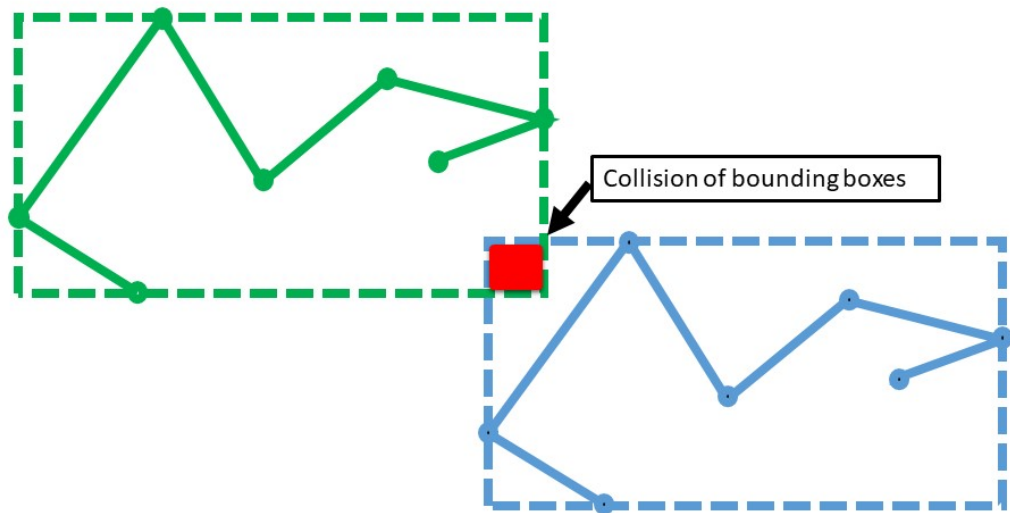


Figure 18: Contact Detection - Drawbacks.

at Equation (2.19), the algorithm decides whether or not will the node in question be involved in the contact computation.

$$g_N \geq 0 \quad (2.19)$$

Therein, g_N represents the normal gap between surfaces. Equation (2.19) is a part of a more general condition, so called *Hertz-Signorini-Moreau condition* depicted in Equation (2.20).

$$g_N \geq 0, \quad t_N \leq 0, \quad t_N g_N = 0 \quad (2.20)$$

Where t_N stands for the magnitude of normal contact traction[12]. Equations in Equation (2.20) are usually referred to as *contact*, *compression* and *complementary condition*, respectively. The following steps of the computation vary, from mathematical point of view, with respect to the problem setup. The implications of frictionless or frictional case will be discussed in the following chapter.

3 Solution of the Contact Problem

The goal of this thesis is to implement a frictional contact solver.

In *Computational Contact Mechanics* (CCM), it is distinguished between two types of contact - *frictionless* and *frictional*. In both cases, the problem breaks down to invoking the principle of stationary potential energy for quasi-static problems. The potential energy for contact problems can be expressed in following manner[22].

$$\Pi = \Pi_{int} + \Pi_C - \Pi_{ext} \quad (3.1)$$

Therein, lower indices distinguish between *internal* potential energy Π_{int} , *contact* potential energy Π_C and *external* potential energy Π_{ext} . Weak form of this problem corresponds with the variation of Equation (3.1) [22],

$$\delta\Pi = \delta\Pi_{int} + \delta\Pi_C - \delta\Pi_{ext} = 0 \quad \forall \varphi \in V, \quad (3.2)$$

where V represents a suitable function space for $\delta\varphi$ - the admissible mechanical variational displacement. As mentioned before, there are 2 types of contact problems. We need to adjust the definition of contact potential energy respectively.

- **Frictionless contact** - $\Pi_C = \Pi_N$
- **Frictional contact** - $\Pi_C = \Pi_N - \Pi_T$

In other words, the virtual work of the contact forces is made up from Π_N - virtual normal energy, and Π_T - virtual work of tangential forces.

Frictionless contact Frictionless contact is a simplification of frictional contact. While it may seem to be rather small adjustment to the final equation, from a mathematical point of view this distinction has significant implications. If we consider the boundary value problem from Section 2.1.2, ignoring the friction conditions, the principle of virtual work for the elastostatic, unilateral contact problem is given in the following equation [18].

$$\int_{\Omega} \sigma_{ij} \delta \epsilon_{ij} d\Omega = \int_{\Omega} b_i \delta u_i d\Omega + \int_{\Gamma_s} T_i \delta u_i dS + \int_{\Gamma_c^2} t_i^n \delta u_i dS + \sum_n F_i^k \delta u_i^k \quad (3.3)$$

Therein, $\delta u_i = v_i - u_i$ is an admissible virtual displacement, $\delta \epsilon_{ij}$ stands for the virtual strain corresponding to δu_i and t_i^n is the normal contact traction. We further denote the concentrated forces as F_i^k , body forces per unit volume as b_i , the stress tensor as σ_{ij} and stress vector as $T_i = \sigma_{ij} N_j$, where N_j is a unit vector. This equation is an extended form of Equation (3.2). Last integral in Equation (3.3) is called *virtual contact work* and it has been proved, that this term is always equal or greater than 0 [12]. Therefore, by dropping out this last term, we obtain the variational inequality, which characterizes the solution on a unilateral, frictionless contact problem. The new variational inequality formulation reads [18]:

$$\text{Find } \mathbf{u} \in \mathbf{K} : \int_{\Omega} \sigma_{ij} \delta \epsilon_{ij} d\Omega \geq \int_{\Omega} b_i \delta u_i d\Omega + \int_{\Gamma_s} T_i \delta u_i dS + \sum_n F_i^k \delta u_i^k, \quad \forall \mathbf{v} \in \mathbf{K}. \quad (3.4)$$

Therein, $\mathbf{K} = \{\mathbf{v} \in \mathbf{V} | \mathbf{v} \cdot \mathbf{n} - d_0 \leq 0\}$, where \mathbf{V} is a set of admissible displacements. Considering the definition of total potential energy in Equation (2.4), then from the optimization theory, we know, that there exists a unique displacement field $\mathbf{u} \in \mathbf{K}$, which minimizes the total potential energy Π on \mathbf{K} .

$$\Pi(\mathbf{u}) \leq \Pi(\mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{K} \quad (3.5)$$

Equation (3.5) can be reformulated in a form of a standard optimization problem with total potential energy Π as an objective function in the following manner:

$$\min_u \Pi(u) \quad \text{subjected to} \quad \mathbf{u} \cdot \mathbf{n} - d_0 \leq 0; \quad \forall a^2 \in \Gamma_c^2. \quad (3.6)$$

It has been shown in [12], that minimizer \mathbf{u} of the optimization problem stated in Equation (3.6) is a solution of the variational inequality in Equation (3.4) [20, 21].

3.1 Variational Formulation for the Frictional Contact

Opposite to frictionless contact problems, frictional contact problems do not have an equivalent optimization problem, i.e., such problem can not be formulated in this manner. We rewrite and keep the virtual work equation as equality given by [18]

$$\int_{\Omega} \sigma_{ij} \delta \epsilon_{ij} d\Omega = \int_{\Omega} b_i \delta u_i d\Omega + \int_{\Gamma_s} T_i \delta u_i dS + \int_{\Gamma_{\bar{C}}} t_i^n \delta u_i dS + \int_{\Gamma_{\bar{C}}} t_i^f \delta u_i dS + \sum_f F_i^k \delta u_i^k. \quad (3.7)$$

Therein, t_i^n and t_i^f stand for normal and tangential traction, which corresponds to friction on the contact surface $\Gamma_{\bar{C}}$, which is a subspace of Γ_C^2 - candidate contact surface. Now, the frictional contact problem reads [12]:

Find displacement field u_i , normal contact traction t_i^n and tangential contact traction t_i^f such that all of the boundary conditions are satisfied.

Since contact surface, normal and tangential tractions are all unknown, the employment of incremental solver, in our case non-linear Newton-Raphson method, is necessary. The fulfilment of the contact conditions is generally enforced by one of the following methods.

Penalty regularization The idea behind penalty regularization is to add a fictional force generated by a numerical spring. This takes care of the appropriate deformation of a slave body according to the master body shape as well as for the sticking or sliding state in frictional contact. Such approach has rather simple implementation (compared to other possibilities). It has intuitive physical interpretation (see Figure 19) and does not introduce any other degrees of freedom to the system. The main drawback is, that the provided solution is only approximate and is its dependent on the chosen penalty parameters.

- ϵ_N - normal penalty parameter
- ϵ_T - tangential penalty parameter

Small values of penalty factor lead to big, unphysical penetration, while huge values contribute to ill-conditioning of the global matrix. In theory, the penalty method could yield exact results, however, the penalty factors would need to be infinity. Even more distressful is the fact that the proper choice of penalty factors is case sensitive and is dependent on many variables, such as time step, rate of prescribed motion, etc.

Lagrange Multipliers In this approach, the contact tractions are treated as additional unknowns to the problem at hand. Alongside the additional degrees of freedom, which

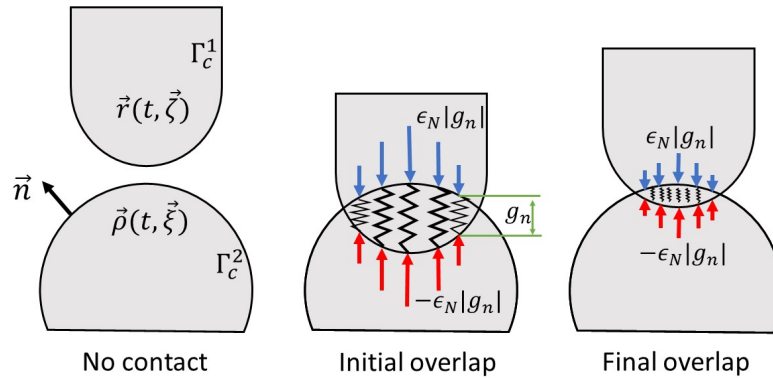


Figure 19: Physical Interpretation of Penalty Method; **Left** - reference state, **Middle** - initial penetration introduces a force of the numerical spring into the system, forcing the penetration to decrease in order to reach the converged state, **Right** - converged state with some residual penetration.

may significantly influence the problems computational cost, this method suffers from a non-smooth Lagrangian [28]. On the other hand, it yields a exact solution of the contact conditions.

The drawbacks of the Lagrange Multipliers method can be circumvented by combining both of the earlier mentioned approaches, i.e., employing the *Augmented Lagrangian Method*. The only disadvantage that still prevails are the additional degrees of freedom introduced to the system [28].

3.2 Implementation

The overall contact algorithm implementation may be divided into two subsequent parts - normal and tangential contact. This order needs to be preserved and originates from the *sliding criterion*, which will be discussed in the following sub-chapters.

Once the detection algorithm spots a possible contact, procedures described at Section 2.4.1 follow. The algorithm detects the projection vector, projection point, evaluates the surface normal and checks whether or not the Gauss point in question takes part in the contact computation in the current iteration.

Equation (3.1) can be rewritten to Equation (3.9) through the following relation:

$$\Pi = - \int_{\partial\beta_k} \beta_k^s \mathbf{f} da_k, \quad (3.8)$$

where \mathbf{f} stand for the general force that acts on the body. In other words, in order to solve a contact problem we need to solve non-linear equation

$$\mathbf{f}(\mathbf{u}) = \mathbf{f}_{int} + \mathbf{f}_C + \mathbf{f}_{ext}, \quad (3.9)$$

where right hand side terms represent, analogously to the potential energy, internal, contact and external forces. Overall force contributions are assembled in the conventional FE manner. While we know internal and external forces from inner space or outer surface, we need to take special care for the contact forces. This will be discussed in two parts - normal contact and tangential contact.

$$\text{Element contact force } \mathbf{f}_C^e = - \int_{\Gamma_0^e} \mathbf{N}^T \mathbf{t}_C \quad (3.10a)$$

$$\mathbf{t}_C = \mathbf{t}_N - \mathbf{t}_T \quad (3.10b)$$

The element contact force is dependent not only on the nodal displacement of the element, but at the same time on the contact gap between master and slave surfaces - mathematically stated as $\mathbf{f}_s^e = f(\mathbf{u}_s^e \text{ on } \Gamma_s^e; \mathbf{u}_m^e \text{ on } \Gamma_m^e)$. Linearization of this relation with respect to \mathbf{u}_s^e and \mathbf{u}_m^e , displacements of slave and master body, yields

$$\mathbf{f}_{C_s}^e(\mathbf{u}_s^e - \Delta \mathbf{u}_s^e, \mathbf{u}_m^e + \Delta \mathbf{u}_m^e) \approx \mathbf{f}_{C_s}^e(\mathbf{u}_s^e, \mathbf{u}_m^e) + \Delta \mathbf{f}_{C_s}^e(\mathbf{u}_s^e, \mathbf{u}_m^e), \quad (3.11)$$

where

$$\Delta \mathbf{f}_{C_s}^e(\mathbf{u}_s^e, \mathbf{u}_m^e) = \frac{\partial \mathbf{f}_{C_s}^e}{\partial \mathbf{u}_s^e} \Delta \mathbf{u}_s^e + \frac{\partial \mathbf{f}_{C_s}^e}{\partial \mathbf{u}_m^e} \Delta \mathbf{u}_m^e \xrightarrow{\text{for rigid master body}} \frac{\partial \mathbf{f}_{C_s}^e}{\partial \mathbf{u}_s^e} \Delta \mathbf{u}_s^e, \quad (3.12a)$$

$$\underline{K}_{C_{ss}}^e = \frac{\partial \mathbf{f}_{C_s}^e}{\partial \mathbf{u}_s^e} = - \int_{\Gamma_s^e} \mathbf{N}_s^T \frac{\partial \mathbf{t}_C}{\partial \mathbf{u}_s^e} dA_s - \int_{\Gamma_s^c} \mathbf{N}_s^T \mathbf{t}_C \otimes \mathbf{a}_s^\alpha \mathbf{N}_{s,\alpha} dA_s. \quad (3.12b)$$

The term $\mathbf{N}_{s,\alpha}$ in Equations (3.12b), stands for the shape functions on the slave element Γ_s^e . It is important to remark, that in Equation (3.12b) \mathbf{a}_s^α stands for the **contra-variant tangent**. In Equation (3.12a), the rigid master body assumption simplifies the matter significantly [22]. As seen from the Equation (3.12b), the only unknown term is the partial derivative of contact traction.

$$\frac{\partial \mathbf{t}_C}{\partial \mathbf{u}_s^e} \xrightarrow{\text{from Equation (3.10b)}} \frac{\partial \mathbf{t}_C}{\partial \mathbf{u}_s^e} = \frac{\partial \mathbf{t}_N}{\partial \mathbf{u}_s^e} - \frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e} \quad (3.13)$$

The evaluation of each of the right hand side terms will be discussed in Section 3.2.1 and Section 3.2.2.

3.2.1 Normal Contact

The first goal of the implementation section of this thesis was to extend normal contact code in *contactModul.F90*, internal CATS contact module, into third dimension. It meant deriving and discretizing a formula for traction force derivative in 2D. Contact takes place on the surfaces of the contact bodies - 3D bodies have 2D contact surfaces. In this regard, we adopted the derivation from [22].

$$\text{Normal gap } g_N = (\mathbf{x}_s - \mathbf{x}_p) \cdot \mathbf{n}_p \quad (3.14a)$$

$$\text{Normal traction } \mathbf{t}_N := \epsilon_N d_N \mathbf{n}_p \quad ^2 \quad (3.14b)$$

As defined in [22], this falls in to the point interaction class, i.e., p-class of contact interaction. This publication also provides detailed derivation of the normal traction derivative. Note, that for our purposes - rigid master, we will list only the relevant terms. From the definition of t_N , we get

$$\frac{\partial \mathbf{t}_N}{\partial \mathbf{u}_s^e} = -\epsilon_N \mathbf{n}_p \frac{\partial g_N}{\partial \mathbf{u}_s^e} - \epsilon_N g_N \frac{\partial \mathbf{n}_p}{\partial \mathbf{u}_s^e}. \quad (3.15)$$

It can be shown that [22]

$$\frac{\partial g_N}{\partial \mathbf{u}_s^e} = \mathbf{n}_p^T \mathbf{N}_s, \quad (3.16a)$$

$$\frac{\partial \mathbf{n}_p}{\partial \mathbf{u}_s^e} = \frac{1}{g_N} [\mathbf{I} - \mathbf{n}_p \otimes \mathbf{n}_p - c_p^{\alpha\beta} \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p] \mathbf{N}_s. \quad (3.16b)$$

Combining all above mentioned equation, we obtain

$$\frac{\partial \mathbf{t}_N}{\partial \mathbf{u}_s^e} = \frac{\partial \mathbf{t}_N}{\partial \mathbf{x}_s} \mathbf{N}_s \quad (3.17a)$$

$$\frac{\partial \mathbf{t}_N}{\partial \mathbf{x}_s} = -\epsilon_N [\mathbf{I} - c_p^{\alpha\beta} \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p] \quad (3.17b)$$

² $d_N = -g_N$

Therein, \mathbf{a}_α^p and \mathbf{a}_β^p stand for the co-variant tangents defined in Equation (3.19), \mathbf{I} represents the identity matrix, ϵ_N stands for the normal penalty parameter and $c_p^{\alpha\beta}$ is matrix defined as

$$[c_p^{\alpha\beta}] = [a_{\alpha\beta}^p - g_N b_{\alpha\beta}^p]^{-1}, \quad (3.18a)$$

$$\text{with } a_{\alpha\beta}^p := \mathbf{a}_\alpha^p \cdot \mathbf{a}_\beta^p, \quad (3.18b)$$

$$b_{\alpha\beta}^p := \mathbf{n}_p \cdot \mathbf{a}_{\alpha,\beta}^p. \quad (3.18c)$$

Here, \mathbf{a}_α^p and \mathbf{a}_β^p represent the **co-variant metric tensor**, while $\mathbf{a}_{\alpha,\beta}^p$ denotes the **curvature tensor**.

$$\begin{aligned} \mathbf{a}_p^\alpha &= a_p^{\alpha\beta} \mathbf{a}_\alpha^p, & [a_p^{\alpha\beta}] &= [a_{\alpha\beta}^p]^{-1}, & \mathbf{a}_{\alpha\beta}^p &= \mathbf{a}_\alpha^p \cdot \mathbf{a}_\beta^p \\ \mathbf{a}_p^\beta &= a_p^{\alpha\beta} \mathbf{a}_\beta^p, \end{aligned} \quad (3.19)$$

3.2.2 Tangential Contact

Once the normal traction is evaluated, we proceed to the tangential contact, i.e., **friction**. As apparent from Equation (3.12b), the aim is to evaluate the tangential traction as well as its derivative similarly as in case of the normal contact. In general, we distinguish two different frictional behaviours: Either *sticking* or *sliding*. To decide which one of these two possibilities is triggered, we use the *slip criterion*, which has the following form.

$$f_{slip} = \|\mathbf{t}_T\| - \mu p = \begin{cases} < 0 & \textit{sticking} \\ = 0 & \textit{sliding} \end{cases} \quad (3.20)$$

Therein, $p := \|\mathbf{t}_N\|$, where the normal traction is defined in Section 3.2.1, and μ denotes the friction coefficient. The slip criterion function used in this thesis can be visualized as a cone in $\{p, \mathbf{t}_T\}$ -space, pictured at Figure 20.

Depending on the slip criterion value, we either constrain the traction \mathbf{t}_T , such that no relative motion takes place - sticking; or in case of sliding we define the traction through *Coulomb's law*

$$\mathbf{t}_T = -\mu \|\mathbf{t}_N\| \frac{\dot{\mathbf{g}}_T}{\|\dot{\mathbf{g}}_T\|}. \quad (3.21)$$

Therein, $\dot{\mathbf{g}}_T$ represent the sliding velocity between contact surfaces. However, for the implementation purposes, or rather for implementation checks, it is beneficial to realize, that according to the derivation in [23] (Chapter 2.6), the last fraction in Equation (3.21) turns out to be

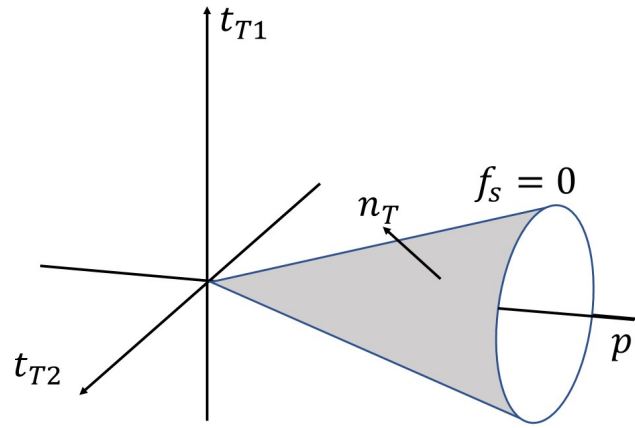


Figure 20: Interpretation of Slip Criterion Function.

$$\mathbf{t}_T = -\mu \|\mathbf{t}_N\| \mathbf{n}_t, \quad (3.22)$$

where $\mathbf{n}_t = \frac{\mathbf{t}_T}{\|\mathbf{t}_T\|}$ stands for unit tangential vector. Equation (3.22) holds true if and only if $\epsilon_T = \epsilon_N$. The relation for $\epsilon_T \neq \epsilon_N$ will be introduced later.

For the friction implementation, we adopted the so-called *predictor-corrector* approach.

1. Elastic Predictor Step In this step we define the predictor in a form of *trial tangential traction* \mathbf{t}_T^{trial} . The proposed term in [23] is described in Equation (3.23).

$$\mathbf{t}_{T,n+1}^{trial} = \epsilon_T (\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_p^{n+1}) - \mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^n)) \quad (3.23)$$

Therein, $\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_p^{n+1})$ is the physical position at time-level t^{n+1} evaluated at the current projection parametric coordinate $\boldsymbol{\xi}_p^{n+1}$, while $\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^n)$ is also a physical position at time-level t^{n+1} , however, evaluated at the previous parametric sliding coordinate.

For computations with larger time steps Equation (3.23) becomes inaccurate. This drawback is further discussed in [6]. Sauer and De Lorenzis, 2014, further propose an alternative formulation, which is based on the idea of projecting the $\mathbf{t}_{T,n+1}^{trial}$ into the tangential plane $\partial\beta_m^{n+1}$. The proposed formulation reads

$$\mathbf{t}_{T,n+1}^{trial} = \epsilon_T (\mathbf{I} - \mathbf{n}_p^{n+1} \otimes \mathbf{n}_p^{n+1}) (\mathbf{x}_s^{n+1} - \mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^n)). \quad (3.24)$$

However, it turns out, that for small time steps, as considered in this thesis, this equation is equivalent to Equation (3.23) up to marginal differences.

2. Slip criterion Once the trial tangential traction is evaluated, we check the slip criterion introduced in Equation (3.20). As indicated in this equation, two outcomes are

possible. This law can be obtained by observing the dissipation of energy throughout the friction. The dissipation is defined as

$$\mathcal{D} = \mathbf{t}_T \cdot \mathcal{L}\mathbf{g}_{sl}, \quad (3.25)$$

where

$$\mathcal{L}\mathbf{g}_{sl} = \dot{\xi}_{sl}^\alpha \mathbf{a}_\alpha^{sl}, \quad \text{with} \quad \mathbf{a}_\alpha^{sl} = \left. \frac{\partial \mathbf{x}_m}{\partial \xi^\alpha} \right|_{\xi_{sl}} \quad (3.26)$$

is the *Lie derivative* applied at the inelastic slip vector \mathbf{g}_{sl} . \mathbf{a}_α^{sl} represents the co-variant tangent vector at the parametric coordinate ξ_{sl} . Using the maximum dissipation principle, it is possible to prove that the Lie derivative of the inelastic slip vector is parallel to partial derivative of the slip criterion with respect to the tangential traction.

$$\mathcal{L}\mathbf{g}_{sl} \parallel \frac{\partial f_{sl}}{\partial \mathbf{t}_T} \quad (3.27)$$

In other words, we find the subsequent evolution law for inelastic slip

$$\mathcal{L}\mathbf{g}_{sl} = \gamma \mathbf{n}_t, \quad \text{where} \quad \mathbf{n}_t : \frac{\partial f_{sl}}{\partial \mathbf{t}_T} = \frac{\mathbf{t}_T}{\|\mathbf{t}_T\|}. \quad (3.28)$$

By combining Equation (3.26) and Equation (3.28) we obtain

$$\dot{\xi}_{sl}^\alpha = \gamma \mathbf{n}_t \cdot \mathbf{a}_{sl}^\alpha. \quad (3.29)$$

3. Inelastic Corrector Step The corrector step, from its definition, adjusts the elastic predictor step in order to make it more accurate. The desired update rules will be stated.

Discretization of Equation (3.29) by the means of implicit Euler yields

$$\xi_{sl,n+1}^\alpha = \xi_{sl,n}^\alpha + \Delta \xi_{sl,n+1}^\alpha, \quad (3.30)$$

where

$$\Delta \xi_{sl,n+1}^\alpha \approx \Delta \gamma_{n+1} \mathbf{n}_t^{n+1} \cdot \mathbf{a}_{s,n+1}^\alpha. \quad (3.31)$$

Therein, $\mathbf{a}_{sl,n+1}^\alpha$ is the contra-variant tangent vector at position ξ_{sl} . $\Delta \gamma_{n+1}$ is a constant, that numerically takes care of the distinction between the sticking and the sliding case by adjusting the parametric position as well as the resulting trial traction force as shown in Equation (3.33). Equation (3.30) can be interpreted as an update to the position, where we "attach" the numerical spring, which is the physical interpretation of

the penalty method. Mapping $\boldsymbol{\xi}_{sl}$ to the surface $\partial\beta_m$ at $n + 1$ time level, we obtain the following update prescription

$$\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^{n+1}) = \mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^n) + \Delta\mathbf{x}_m^{n+1}, \quad \text{with} \quad \Delta\mathbf{x}_m^{n+1} \approx \Delta\gamma_{n+1}\mathbf{n}_t^{n+1}. \quad (3.32)$$

The tangential traction at time level $n+1$ is given as

$$\mathbf{t}_T^{n+1} = \epsilon_T(\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_p^{n+1}) - \mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^{n+1})) = \mathbf{t}_{T,n+1}^{trial} - \epsilon_T\Delta\gamma_{n+1}\mathbf{n}^{n+1}. \quad (3.33)$$

Solving the final form of Equation (3.33) for $\Delta\gamma_{n+1}$, we obtain its update rule for sliding:

$$\Delta\gamma_{n+1} = \frac{\|\mathbf{t}_{T,n+1}^{trial}\| - \mu p_{n+1}}{\epsilon_T}. \quad (3.34)$$

Plugging Equation (3.34) into Equation (3.33), and for $\epsilon_N = \epsilon_T$, we formally derive Equation (3.22). In general, ϵ_T and ϵ_N do not have to be the same.

In case of sticking condition, $\Delta\gamma_{n+1} = 0$, i.e., the update of the parametric coordinate stated in Equation (3.30) only overwrites the new parametric coordinate with the old one. Plugging this into Equation (3.30) yields no change in the parametric coordinate.

Traction Derivative Evaluation Same as in Section 3.2.1, it is important to derive the update rules for tangential traction \mathbf{t}_T , which has been shown above as well as for the derivative $\frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e}$. As before, even this step will be split in to two section - sticking and sliding case.

For the **sticking case** the tangential traction is defined as [23]

$$\mathbf{t}_T = \epsilon_T(\mathbf{x}_m(\boldsymbol{\xi}_p) - \mathbf{x}_m(\boldsymbol{\xi}_{sl})) = \epsilon_T(\mathbf{x}_p - \mathbf{x}_m(\boldsymbol{\xi}_{sl})). \quad (3.35)$$

For simplicity and better readability of this derivation, the superscript $n + 1$ will be omitted and it will be inherently assumed that every term is in this time level, unless stated differently. The derivative of Equation (3.35) with respect to \mathbf{u}_s^e reads

$$\frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e} = \epsilon_T \frac{\partial \mathbf{x}_p}{\partial \mathbf{u}_s^e}. \quad (3.36)$$

With

$$\frac{\partial \mathbf{x}_p}{\partial \mathbf{u}_s^e} = c_p^{\alpha\beta} \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p \mathbf{N}_s \quad (3.37)$$

we find the derivative to be

$$\frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e} = \epsilon_T c_p^{\alpha\beta} \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p \mathbf{N}_s. \quad (3.38)$$

Therein, \mathbf{a}_α^p and \mathbf{a}_β^p are the co-variant tangents, \mathbf{N}_s is a shape function evaluated at ξ_p and $c_p^{\alpha\beta}$ are the components of a matrix defined in Equation (3.18) [23].

Alternatively, for the **sliding case** we define the traction through Equation (3.22). As in the previous case, superscripts $n + 1$ will be omitted. If we take the derivative of Equation (3.22), we find

$$\frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e} = -\mu \epsilon_N \mathbf{n}_T \otimes \mathbf{n}_T \mathbf{N}_s + \mu p \frac{\partial \mathbf{n}_T}{\partial \mathbf{u}_s^e}. \quad (3.39)$$

The derivative of the last term can be expressed as

$$\frac{\partial \mathbf{n}_T}{\partial \mathbf{u}_s^e} = \frac{1}{\|\mathbf{t}_T^{trial}\|} [\mathbf{I} - \mathbf{n}_T \otimes \mathbf{n}_T] \frac{\partial \mathbf{t}_T^{trial}}{\partial \mathbf{u}_s^e}, \quad (3.40)$$

where the last fraction is given in Section 3.2.1. Merging everything yields

$$\frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e} = \frac{\partial \mathbf{t}_T}{\partial \mathbf{x}_s^e} \mathbf{N}_s, \quad (3.41a)$$

$$\frac{\partial \mathbf{t}_T}{\partial \mathbf{x}_s^e} = -\mu \epsilon_N \mathbf{n}_T \otimes \mathbf{n}_T + \epsilon_T c_p^{\alpha\beta} \mathbf{P} \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p, \quad (3.41b)$$

$$\text{with } \mathbf{P} := \frac{\mu p}{\|\mathbf{t}_T^{trial}\|} [\mathbf{I} - \mathbf{n}_T \otimes \mathbf{n}_T]. \quad (3.41c)$$

3.2.3 Algorithm for the Frictional Contact

Algorithm 1: Frictional Contact Workflow - Commented

Result: Assembled stiffness matrix and right hand side vector

- Contact Detection - Section 2.4.1

- Initialize all variables

for all Gauss points on slave surface **do**

- Find projection point - $F(\xi) = (\mathbf{x}_F - \mathbf{x}_p) \cdot \mathbf{t} \stackrel{!}{=} 0$

- Compute projection vector \mathbf{v}_{proj} and outer normal \mathbf{n} of the master surface

- Compute the normal gap $g_N = (\mathbf{x}_s - \mathbf{x}_p) \cdot \mathbf{n}_p$ and the angle between \mathbf{v}_{proj} and \mathbf{n}

if $g_N > 0$ **or** $\mathbf{v}_{proj} > tolAngle$ **then**

 - Current Gauss point **is not** active in the current time step

 - Proceed to the next Gauss point

else

 - Current Gauss point **is** active in the current time step

end

NORMAL TRACTION EVALUATION - Section 3.2.1

• Compute normal traction - $\mathbf{t}_N := \epsilon_N d_N \mathbf{n}_p$

• Compute the transformation matrix - $[c_p^{\alpha\beta}] = [\mathbf{a}_\alpha^p \cdot \mathbf{a}_\beta^p - g_N \mathbf{n}_p \cdot \mathbf{a}_{\alpha,\beta}^p]^{-1}$

• Evaluate the traction derivative - $\frac{\partial \mathbf{t}_N}{\partial \mathbf{u}_s^e} = -\epsilon_N [\mathbf{I} - c_p^{\alpha\beta} \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p] \mathbf{N}_s$

TANGENTIAL TRACTION EVALUATION - Section 3.2.2

• Elastic predictor step - Evaluate $\mathbf{t}_T^{trial} = \epsilon_T (\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_p^{n+1}) - \mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^n))$

• Evaluate the slip criterion $f_{slip} = \|\mathbf{t}_T\| - \mu p$

 - $f_{slip} \leq 0$: sticking state - $\Delta\gamma_{n+1} = 0$

 - $f_{slip} > 0$: sliding state - $\Delta\gamma_{n+1} = \frac{\|\mathbf{t}_{T,n+1}^{trial}\| - \mu p_{n+1}}{\epsilon_T}$

• Update parametric coordinates $\xi_{sl,n+1}^\alpha = \xi_{sl,n}^\alpha + \Delta\xi_{sl,n+1}^\alpha$ and tangential traction $\mathbf{t}_T^{n+1} = \epsilon_T (\mathbf{x}_m^{n+1}(\boldsymbol{\xi}_p^{n+1}) - \mathbf{x}_m^{n+1}(\boldsymbol{\xi}_{sl}^{n+1})) = \mathbf{t}_{T,n+1}^{trial} - \epsilon_T \Delta\gamma_{n+1} \mathbf{n}^{n+1}$

• Evaluate the tangential traction derivative -

$$\frac{\partial \mathbf{t}_T}{\partial \mathbf{u}_s^e} = -\mu \epsilon_N \mathbf{n}_T \otimes \mathbf{n}_T + \epsilon_T c_p^{\alpha\beta} \frac{\mu p}{\|\mathbf{t}_T^{trial}\|} [\mathbf{I} - \mathbf{n}_T \otimes \mathbf{n}_T] \mathbf{a}_\alpha^p \otimes \mathbf{a}_\beta^p \mathbf{N}_s$$

- Assemble stiffness matrix $\underline{K}_{C_{ss}}^e = - \int_{\Gamma_s^e} \mathbf{N}_s^T \frac{\partial \mathbf{t}_C}{\partial \mathbf{u}_s^e} dA_s - \int_{\Gamma_C^e} \mathbf{N}_s^T \mathbf{t}_C \otimes \mathbf{a}_s^\alpha \mathbf{N}_{s,\alpha} dA_s$

- Assemble right hand side vector through $\mathbf{f}_{cs}^e = - \int_{\Gamma_s^e} \mathbf{N}_s^T \mathbf{t}_s da_s$

end

4 Software Environment

The software framework for simulation of FSCI problems follows the concepts of partitioned algorithms. For fluid-structure-contact interaction in particular, we employ the flow solver XNS, the structural solver FEAFa and the coupling module ACM. In fact, the enhancement of FEAFa module has been the main objective of this thesis.

4.1 Flow Solver XNS

XNS is an in-house finite-element flow solver developed at CATS institute, RWTH Aachen. Besides employing this solver for FSCI problems, it plays a major role in many other research fields such as free-surface flow (e.g. [13]), hemolysis, plastic extrusion and turbulent flows. Apart from compressible and incompressible Navier-Stokes equations, XNS features simplified models such as Stokes equations as well as shallow water equations or advection-diffusion problems.

4.2 Structural Solver FEAFa

FEAFa ("**Finite Element Analysis For Aeroelasticity**") is another in-house finite-element solver of CATS institute. While originally designed as a classical finite-element solver, it has undergone considerable extension works in terms of isogeometric analysis (as discussed in Section 2.3), which is slowly becoming a global mainstream in terms of computational solid mechanics as well as in computational contact mechanics.

Time discretization is done by a generalized α -scheme (discussed in Section 2.1.2). It further features the possibility of steady or transient computations. FEAFa can be used as a stand-alone solver as well as a library that can be included by other program packages, e.g., in simulations of FSCI problems [16].

4.3 Coupling Module ACM

Another software developed at CATS institute, RWTH Aachen is the **Aeroelastic Coupling Module** (ACM), which serves as an interface between fluid and structural solvers and allows the exchange of coupling-relevant data. While being broadly used for coupling between XNS and FEAFa solver, ACM may also be used as a coupling module for other solvers. In this regard, its modular implementation is extremely beneficial, especially in terms of implementation effort.

ACM features both weak and strong coupling schemes (as discussed in Section 2.2), various projection methods capable of handling conforming as well as non-conforming discretizations of the FSCI interface. In terms of structural solution post-processing

user can choose between constant or Aitken's dynamic relaxation or interface quasi-Newton method (more on this in [24]). Similarly to FEAFa, ACM can be also used in stand-alone or library mode [15].

4.4 Splinelib Library

All above mentioned softwares do make use of splines at some point of the numerical simulation. The functionality as well as wide range of spline types is provided by Fortran Splinelib library developed at CATS institute, RWTH Aachen. This library is applicable in wide range of tasks starting from spline evaluation or setter or getter functions of various kinds, computation of normal vectors all the way to closest-point projection.

5 Numerical Results

This section is designated for the numerical application of the implemented code. We will first demonstrate the performance of our implementation of the proposed frictional model - in Section 5.1. Further, our implementation will be compared to a benchmark computation from [14], Chapter 6.1.4 - in Section 5.2. In the following Section 5.3, we will discuss the effect of chosen parameters on the computation. Last part will be appointed to the 3D applicability of this implementation - in Section 5.4.

5.1 Sliding Rectangular

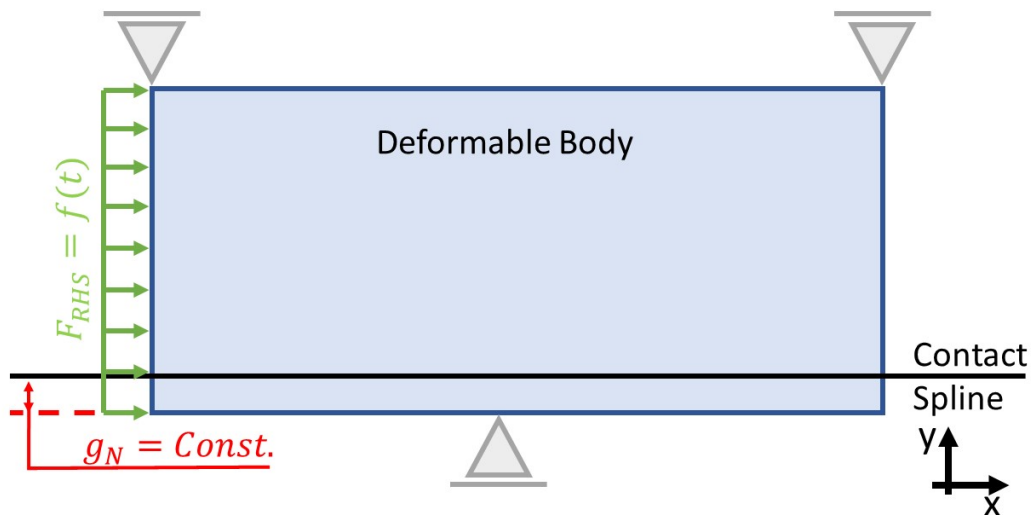


Figure 21: Sliding Rectangular - Problem Setup.

5.1.1 Problem Description

In this computation we will move a block, with predefined dimensions and physical parameters, from one point to the other, while accounting for the friction between the contact spline and the deformable body. The values of the normal and tangential traction are directly influenced by the magnitude of normal gap g_N and are thus not constant over time. In this case, however, we set all degrees of freedom in y -direction to zero, i.e., the block can only move horizontally and the normal gap g_N is constant over the whole computation. The penalty parameters ϵ_N and ϵ_T have been set to the same value through *autoAdaptPenaltyFactor* routine, that sets the values according to the following relation

$$\epsilon_N = \epsilon_T = \alpha \cdot \max(\tilde{\rho} \cdot \tilde{a}, \tilde{E}/L0), \quad (5.1)$$

where α is a user-defined parameter which has been set to 10 for this computation. Further, $\tilde{\rho}$ stands for average density, \tilde{a} represent maximal acceleration, \tilde{E} is the average Young's modulus and $L0$ some characteristic length of the problem.

Parameters	Rectangular Body - 2D
Dimensions [m]	0.005x0.015
Number of elements	50x8
Spline Degree	2
Young's Modulus [Pa]	$75 \cdot 10^8$
Poisson number [-]	0.3
Density [$\frac{kg}{m^3}$]	2710
Friction Coefficient [-]	0.3

Table 3: Sliding Rectangular - Problem Setup.

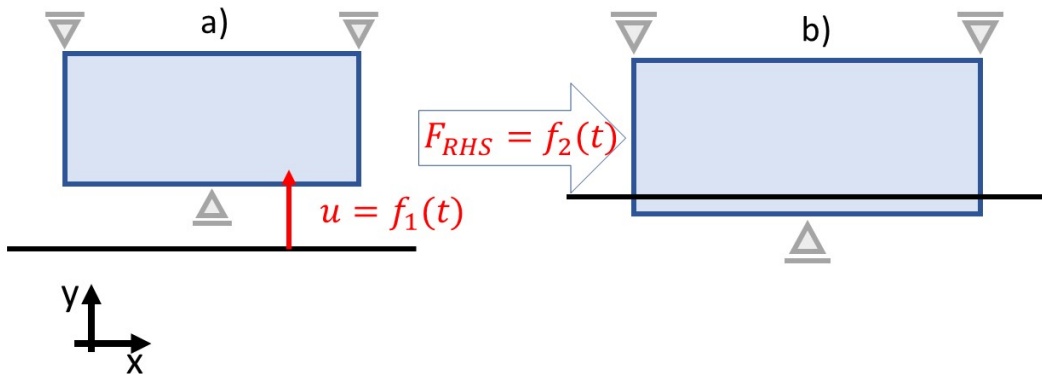


Figure 22: Sliding Rectangular - Phases.

The simulation is divided into 2 parts. The computation starts as shown in Figure 22-a), i.e., no contact. In the first phase, the contact spline moves with prescribed displacement into the contact with the deformable body. In the second part b), the time-dependent external force is applied.

$$u = f_1(t) = \begin{cases} f_{1x}(t) := 0 \\ f_{1y}(t) := \begin{cases} 0.012 \cdot t & t \leq 1 \cdot dt \\ 0.0012 & t > 2 \cdot dt \end{cases} \end{cases} \quad (5.2)$$

$$F_{RHS} = f_2(t) = \begin{cases} f_{2x}(t) := 1000 \cdot \sin(10\pi t); & t > 6 \cdot dt \\ f_{2y}(t) := 0 & \end{cases} \quad (5.3)$$

Therein, t represents the current physical time and $dt = 0.001s$ stands for the time-step size.

5.1.2 Results

NOTE - This numerical example is only for demonstration purposes. No physical results are pursued. The aim of this computation is to show the mechanism of *slip criterion* (Equation (3.20)). For simplicity, we introduced some assumptions, that make the working of the implemented code easier to understand.

Horizontal and Vertical Forces A direct consequence of the constant normal gap is the constant vertical force. From the definition of the *Inelastic corrector step* in Equation (3.33) and the *trial tangential traction* in Equation (3.24), it is evident that tangential traction depends on the normal traction. As shown in Figure 23, the constant normal gap influences also the horizontal force. If we compare the course of this force and the time-dependent right hand side force from Figure 24, we may easily see the correlation with the exception of the sliding state interval ($t \in (0.036, 0.07)s$), which stems from the definition of slip criterion.

Number of Newton-Raphson Iterations The overall dependence of the number of Newton-Raphson iterations on the external forces and the overall motion state is evident from Figure 24. We can easily distinguish between the sliding state and the sticking state. It can be further concluded that the sliding state is, in general, computationally more expensive. This is caused by the change in parametric coordinate in every Newton-Raphson iteration taken.

Conclusion While this simulation did not yield any practical results, the demonstration of functioning friction algorithm has been successful.

5.2 2D Shallow Ironing Problem with Friction - Benchmark Computation

5.2.1 Problem Description

The benchmark computation setup from [14] is shown at Figure 27. We have adopted the spatial problem setup, i.e., dimensions and discretization. The boundary condi-

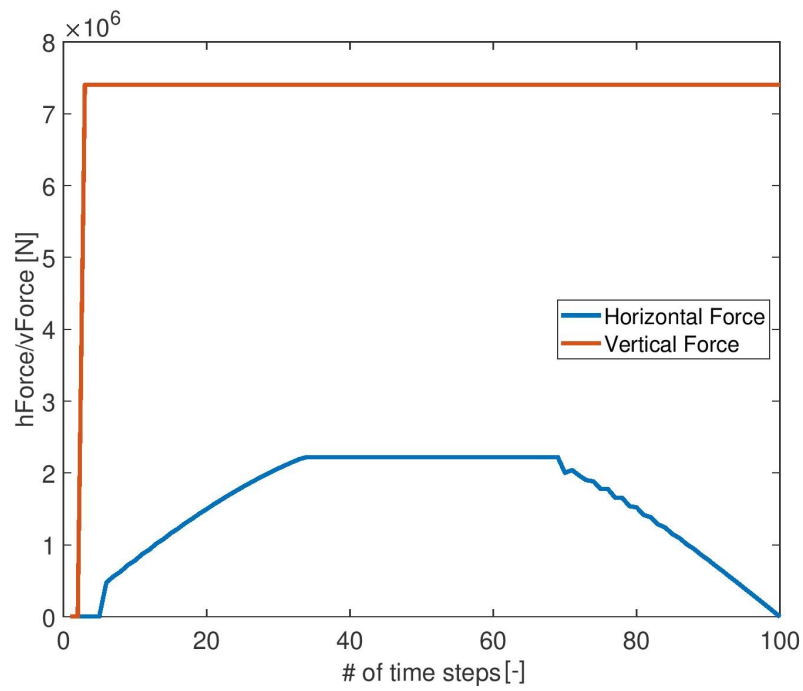


Figure 23: Sliding Rectangular - Horizontal and Vertical Contact Forces.

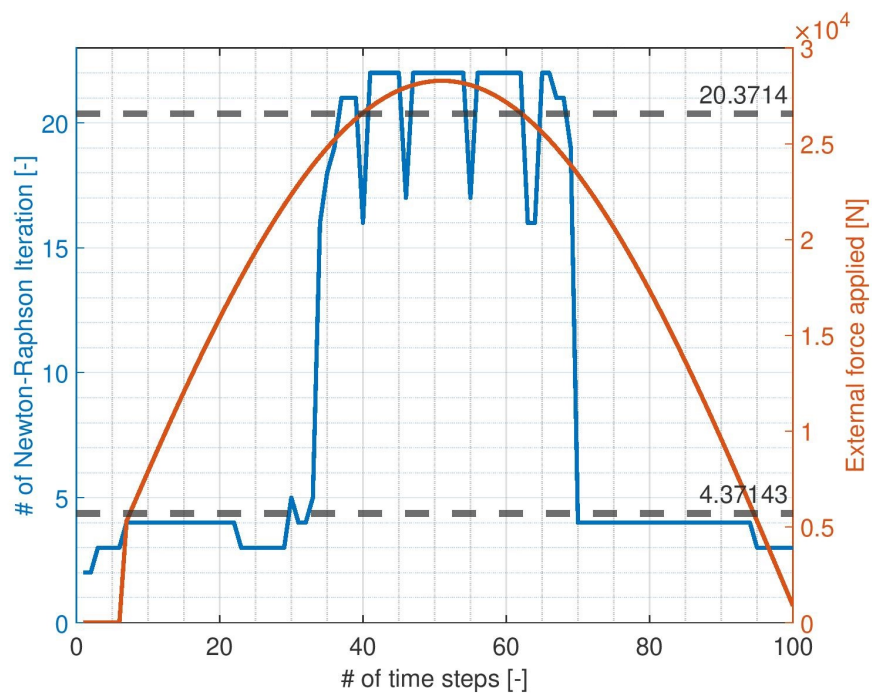


Figure 24: Sliding Rectangular - Influence of the External Load on the Number of Newton-Raphson Iterations.

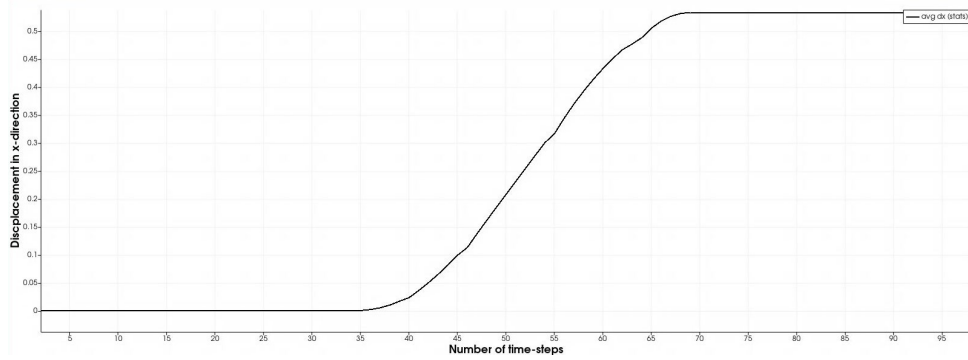


Figure 25: Sliding Rectangular - Displacement Over Time-Steps.

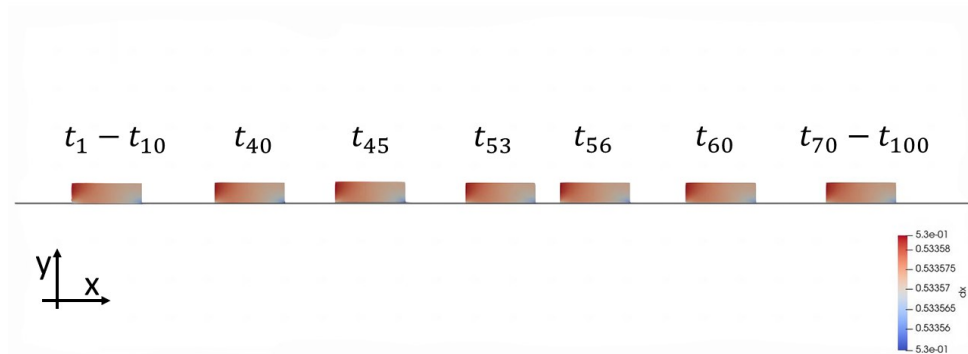


Figure 26: Sliding Rectangular - Visualization of the Displacement.

Parameters	Rectangular Body - 2D
$L_1[m]$	12
$L_2[m]$	4
$L_3[m]$	1.2
$L_4[m]$	0.9
$L_5[m]$	0.3
Young's Modulus E [Pa]	$68.69 \cdot 10^7$
Poisson's ratio ν [-]	0.32
Number of elements	50×8
Friction coefficient μ [-]	0.3
Penalty factors $\epsilon_N = \epsilon_T$ [-]	$1 \cdot 10^{10}$

Table 4: Benchmark 2D Shallow Ironing Problem with Friction - Setup Parameters.

tions for the slave body define zero degrees of freedom for its bottom boundary and

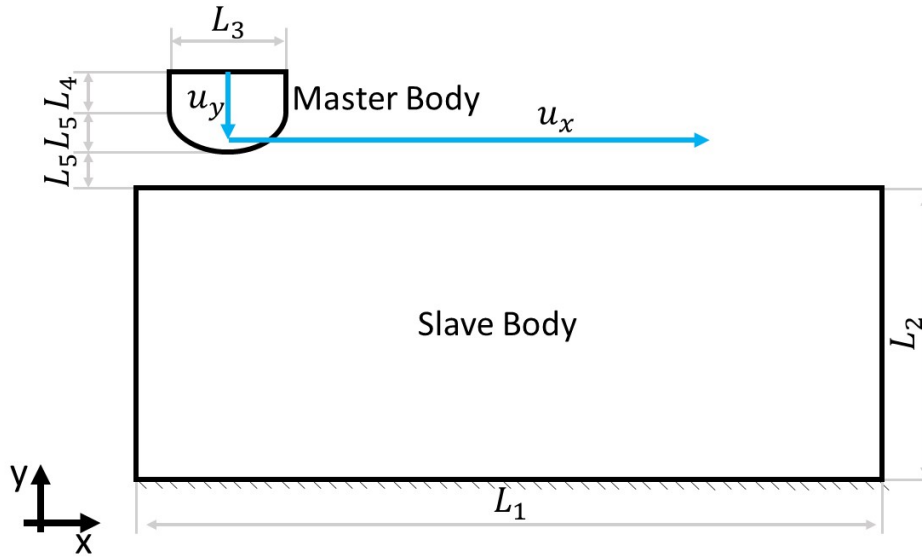


Figure 27: 2D Shallow Ironing Problem - Setup.

allow only y-deformation for the vertical boundaries, i.e., degrees of freedom in x-direction are set to zero.

Prescribed displacement of the master body for this simulation is specified in the following equation. We prescribed total of 2 000 time steps.

$$u(t) = \begin{cases} u_x(t) := 10 \cdot t & t > 1000 \cdot dt \\ u_y(t) := \begin{cases} -21 \cdot t & t \leq 1000 \cdot dt \\ -0.21 & t > 1000 \cdot dt \end{cases} \end{cases}, \text{ for } dt = 0.001s \quad (5.4)$$

5.2.2 Results

While comparing the obtained results with the benchmark, we need to keep in mind the following differences between the cases.

- The master body, the indenter, is **NOT** deformable as in the benchmark computation. This adjustment has been proposed due to the fact that the current implementation of our frictional module is not yet adjusted for the interaction of two deformable bodies.
- The rigid body (indenter) is in our case represented with only a half circle. This is, however, only visual difference. Since we do not have the ambitions to compute the deformations or stress in the indenter itself, it suffices to mimic only the part of the indenter which actually comes into the contact with the deformable body.

Due to these differences, we will use the results from [14] only as a guideline. Therefore, we will present our obtained results and then we will discuss the differences and shared features.

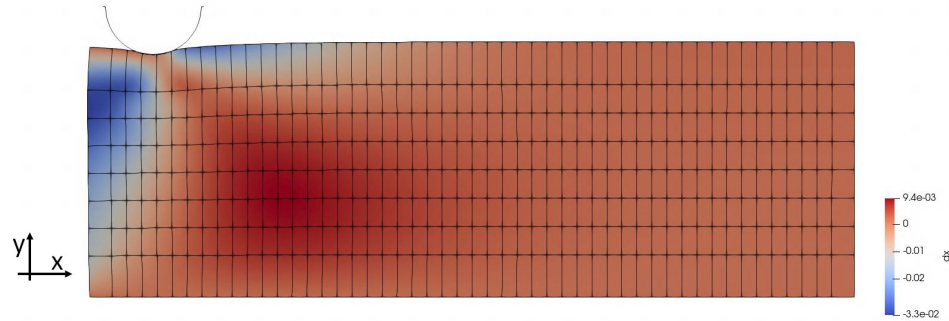


Figure 28: 2D Shallow Ironing Problem - $t = 1$ s

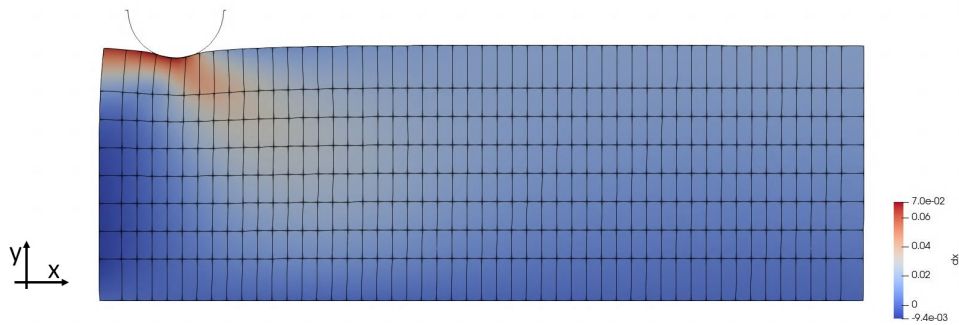


Figure 29: 2D Shallow Ironing Problem - $t = 1.2$ s

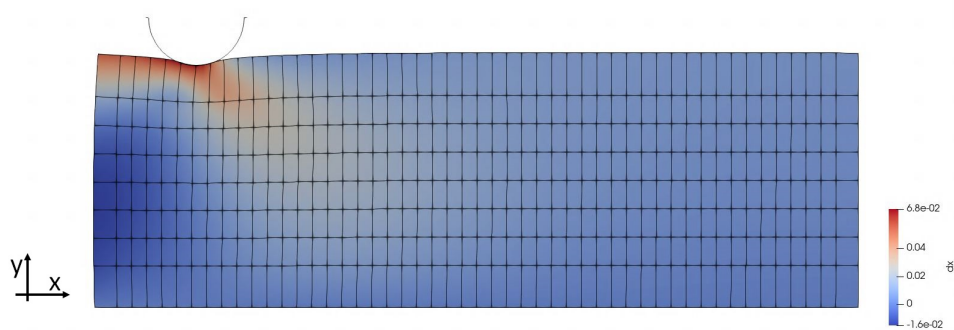


Figure 30: 2D Shallow Ironing Problem - $t = 1.6$ s

Figures 28 to 31 show the computation in distinct times. All pictures show the displacement field in x direction. If we compare the evolution of this field with the evolution of the stress field [14], the resemblance is obvious. For instance, the area of orange-red displacement to the southeast from the indenter at Figure 31 represents a compressed part of the slave body due to the displacement gradient in the southeast direction. Thus, the direct implication is the existence of higher stress region, which is at Figure 32 also

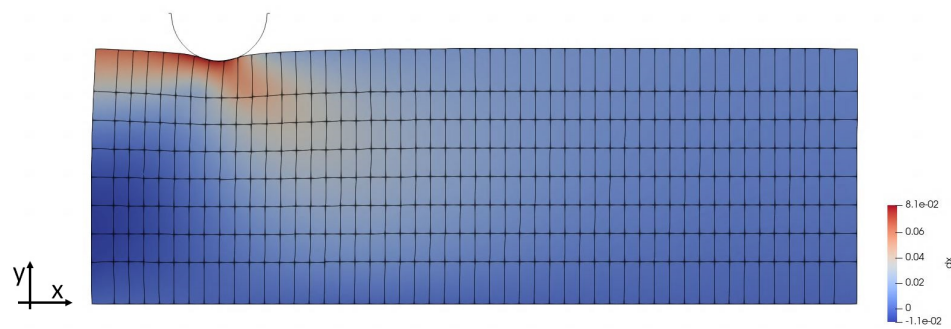
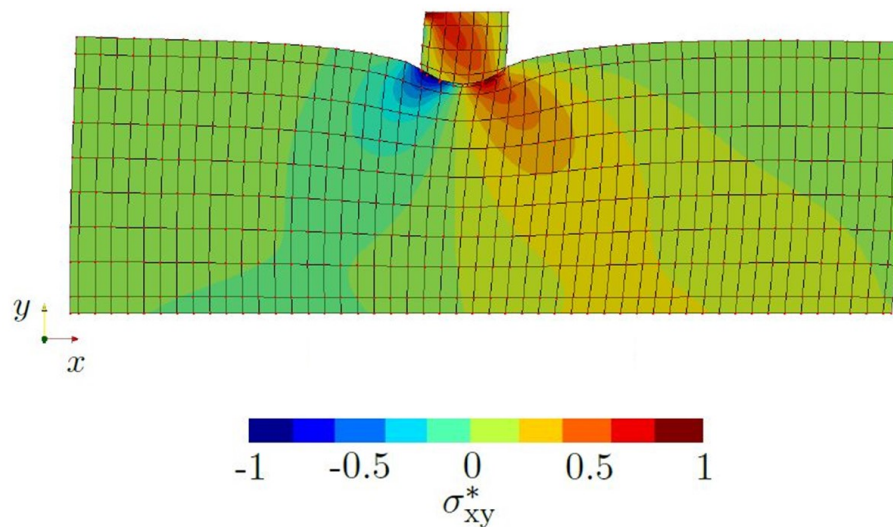
Figure 31: 2D Shallow Ironing Problem - $t = 2$ s

Figure 32: 2D Shallow Ironing Problem - Stress Field from [14].

depicted by orange-red area. The similar logic could be applied to the region left of the indenter. While the displacement field also indicates deformation in the positive direction of x axis, the inverse nature of the deformation gradient, with respect to the previous region, indicates the stretch of this area. This also corresponds to the result at Figure 32, where the stretch is represented by the shades of blue.

It is evident that both computations yield the same shape of the horizontal and vertical forces. The contrast comes in a form of force magnitude. While in our computations both components of the contact force are of an order 10^6 , components from the benchmark computation are two orders higher. This distinction can be explained by the difference in handling the master body. While our computation regards the master body as rigid, the benchmark computation deems it deformable. The deformation of the master body itself is believed to account for the order difference in the contact force. While considering the elastic model for both bodies, the deformation is accompanied by additional elastic forces acting on the slave body.

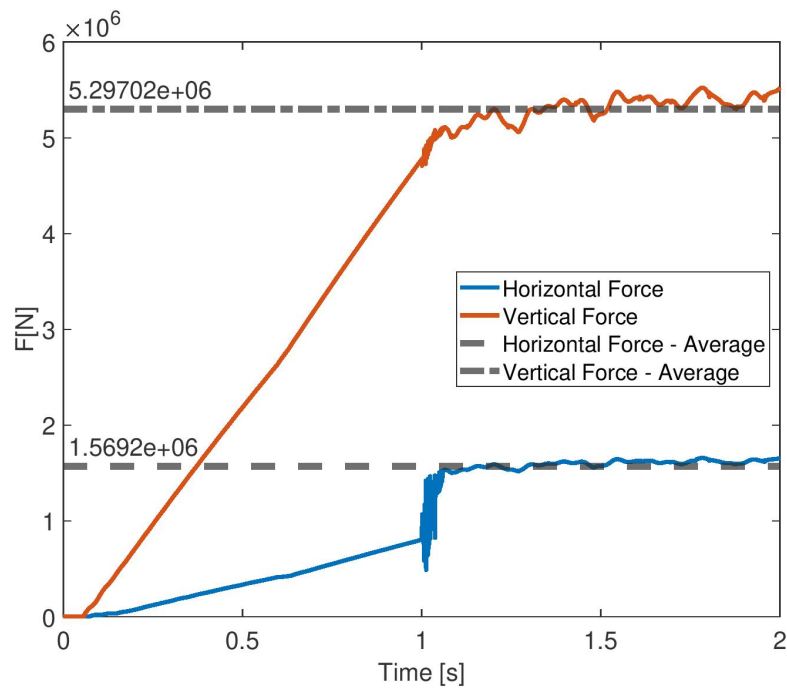


Figure 33: 2D Shallow Ironing Problem - Horizontal and Vertical Contact Forces.

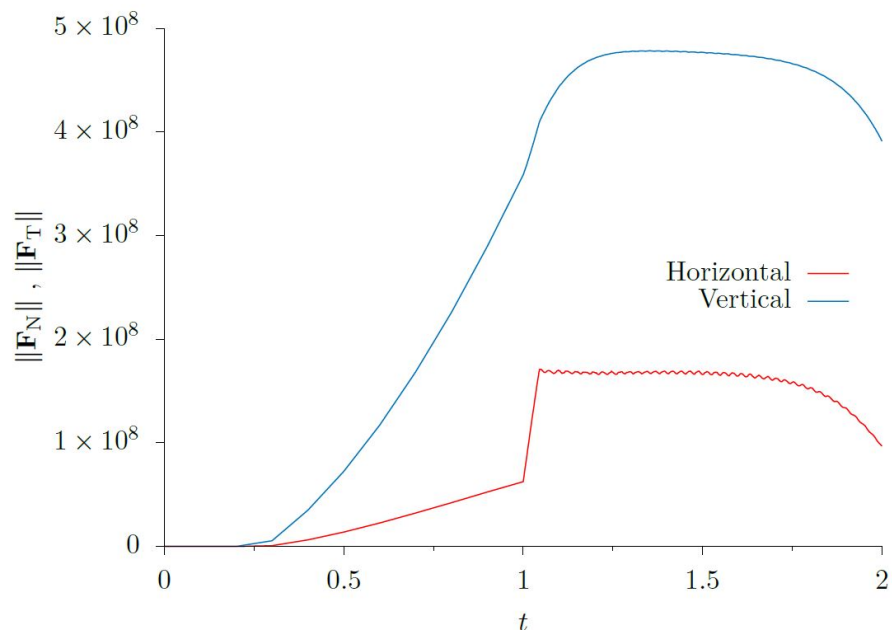


Figure 34: 2D Shallow Ironing Problem - Horizontal and Vertical Contact Forces from the Benchmark Computation [14].

Besides the qualitative and quantitative resemblance of our results, we are also interested in the overall performance of our implementation. One way to assess this feature is the convergence rate, i.e., the number of iterations needed for the time-step to reach

the converged state. Figure 35 depicts the number of taken iterations throughout the computation. It is interesting to note that the average number of iteration somewhat coincide with the average computed in the Moving Rectangular problem in the sliding phase - Section 5.1.

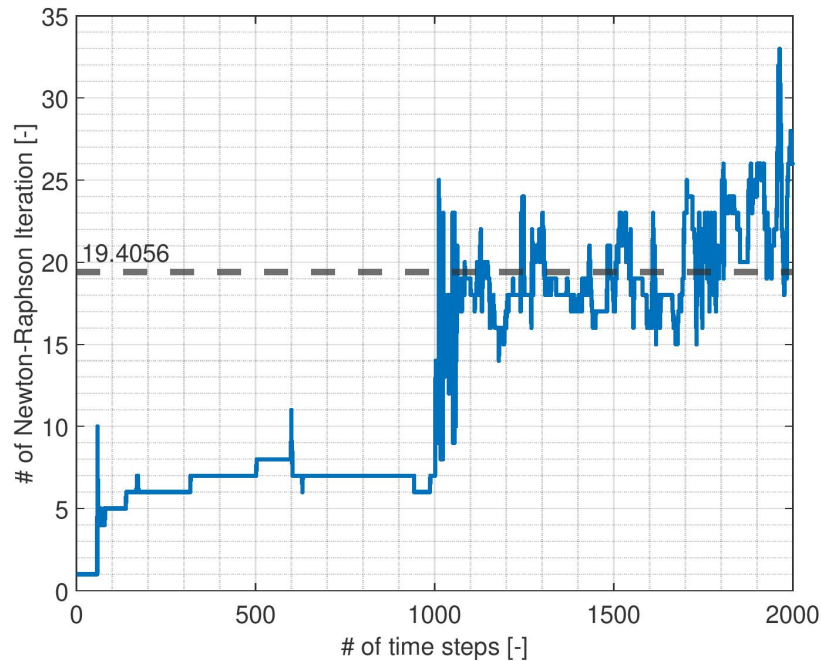


Figure 35: 2D Shallow Ironing Problem - Number of Newton-Raphson Iterations throughout the Computation.

Conclusion Despite the fact that the benchmark computation differs in some aspects, the goal of this comparison was to assess the resemblance in the computational behaviour, which has been shown and proven.

5.3 Sensitivity Analysis

This section will present results from six computations, that will describe different behaviour of the computation with respect to changes in the setup parameters, such as the magnitude of the prescribed displacement, the penalty parameters, as well as the magnitude of deformation of the slave body or the time-step size.

The general setup of the problem is the same as in the previous section, described in Figure 27 and Table 4, unless stated differently.

5.3.1 Influence of the Penalty Parameters

As discussed in Section 3.1, penalty parameters are case dependent and the result is directly influenced by their choice. For that reason we conducted two computations, in which the penalty parameters $\epsilon_N = \epsilon_T$ were chosen one order higher and one order lower than the value in Section 5.2.

Increase of the Penalty Parameters - Results

The increase of the penalty parameters did yield the expected results - the computation did fail already at 2nd iteration.

The question remains whether the error is due to the change of the penalty parameters or some other, so far unresolved problem. The first claim could be backed by the fact that higher values of penalty parameters contribute to the singularity of the final equation system. Since this error occurred already in the second time-step, the effect of the value increase would yield a problematic input in the next time-step, which would then lead to the fatal error in the projection.

Decrease of the Penalty Parameters - Results

The direct consequence of smaller penalty parameters is, in general, better convergence and bigger overlap of the slave and master body. This statement has been proved in our computations.

The course of the horizontal and vertical component of the contact force is pictured at Figure 36. We may observe a qualitative resemblance of the forces compared to the results in previous section. Comparing Figures 33 and 36 in terms of force magnitude, we find that the decreased penalty parameter decreases the overall horizontal ...

Mention the overlap!

Another implication of the decreased penalty parameters may be observed in terms of Newton-Raphson iterations in distinct time-steps. In the comparison to the benchmark computation, the average number has decreased significantly.

5.3.2 The Influence of the Ironing Depth

In this part of the sensitivity analysis, we will study the influence of the ironing depth on the computational performance and on the components of the contact force. Therefore, we have changed the motion prescription in the following manner³. Due to the

³The changes with respect to the Equation (5.4) are marked as bold.

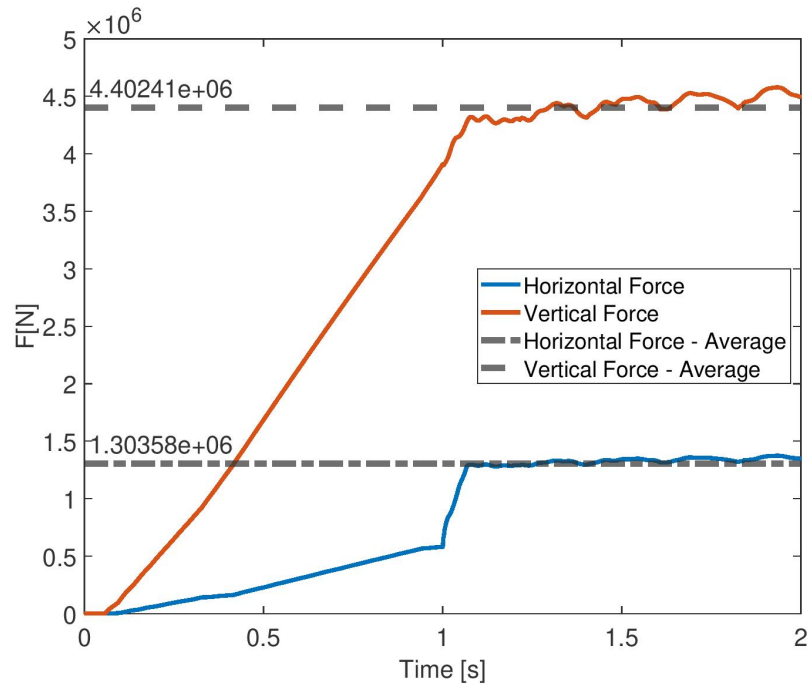


Figure 36: Sensitivity Analysis - Influence of Lower Penalty Parameters - Horizontal and Vertical Forces.

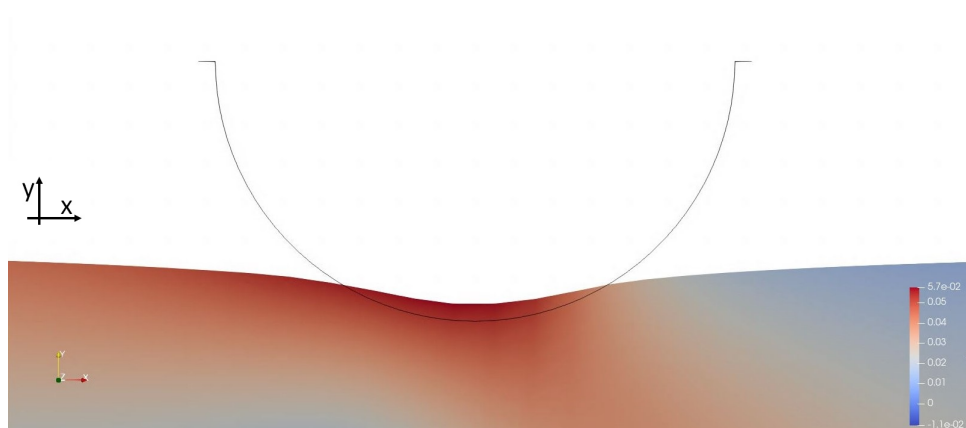


Figure 37: Sensitivity Analysis - Influence of the Lower Penalty Parameters $t = 2$ s.

convergence problems, we had to change the displacement prescription for the master body.

$$u(t) = \begin{cases} u_x(t) := & \mathbf{1} \cdot t & t > 1000 \cdot dt \\ u_y(t) := & \begin{cases} -\mathbf{31} \cdot t & t \leq 1000 \cdot dt \\ -\mathbf{0.31} & t > 1000 \cdot dt \end{cases} \end{cases}, \text{ for } dt = 0.001s \quad (5.5)$$

This sensitivity test has been conducted in order to gain some more insight in the effect of the ironing depth on the stability and speed of the computation. From the presented

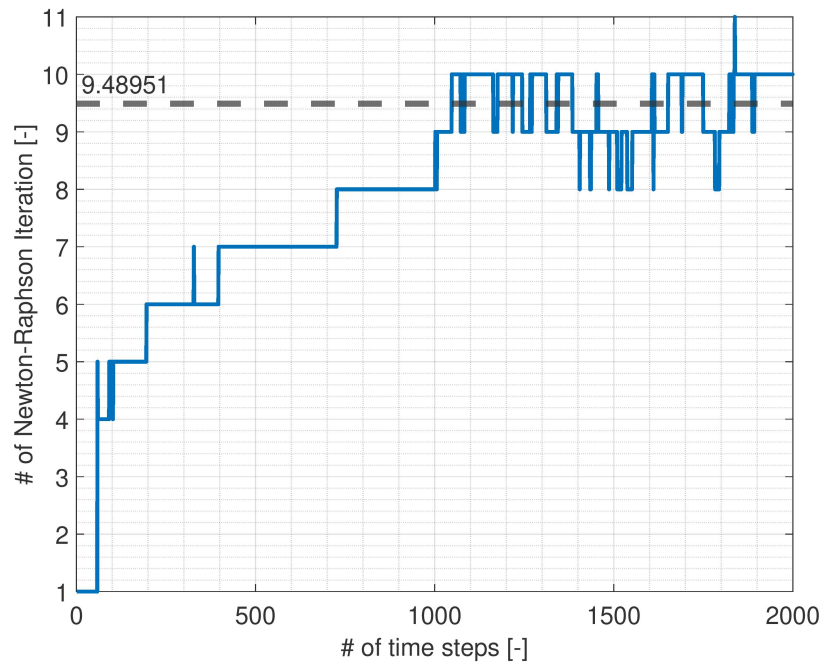


Figure 38: Sensitivity Analysis - Influence of Lower Penalty Parameters on Number of Newton-Raphson Iterations.

results we may conclude, that the ironing depth increases the contact force and thus also the range of vertical force oscillation. This is not a unexpected result. Having employed the elastic model for the slave body, the stresses alongside with the forces are linearly dependent on the strain.

The decrease in the average number of Newton-Raphson iterations is not very significant.

The overall influence of the ironing depth lies in terms of increased contact force. Other effects has not been observed.

5.3.3 Influence of the Time-Step

For any numerical simulation the time-step size is of utmost importance. Its value determines how much physical time the computation actually describes. In general, the smaller the time-step is the more precise the results get. As usually, exceptions exist as already mentioned in Section 2.2.4.

On the other hand, numerical computations, such as FSCI problems, are computationally very expensive problems especially for small time-step sizes. Therefore, the possibility of increasing the time-step size, can make the use of numerical simulations much

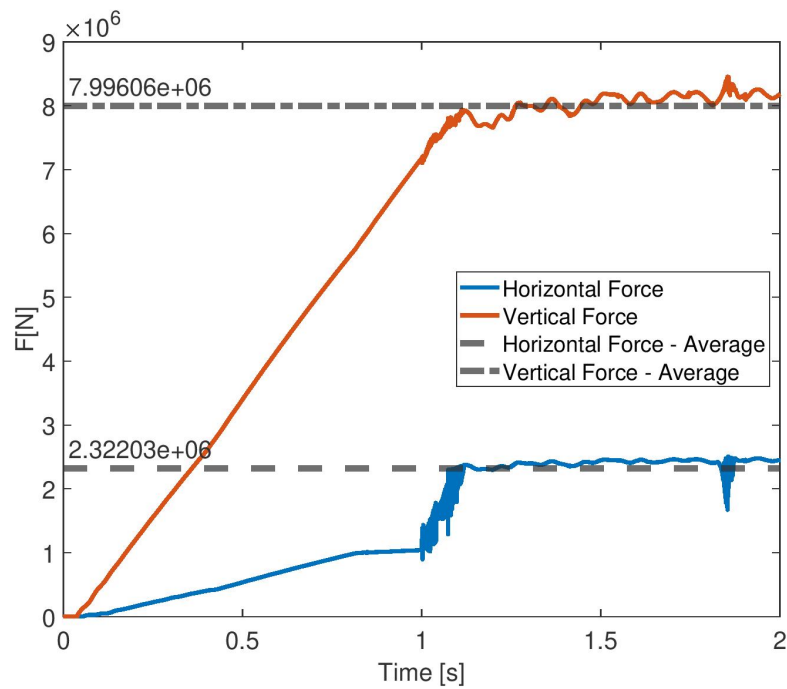


Figure 39: Influence of the Ironing Depth - Components of the Contact Force.

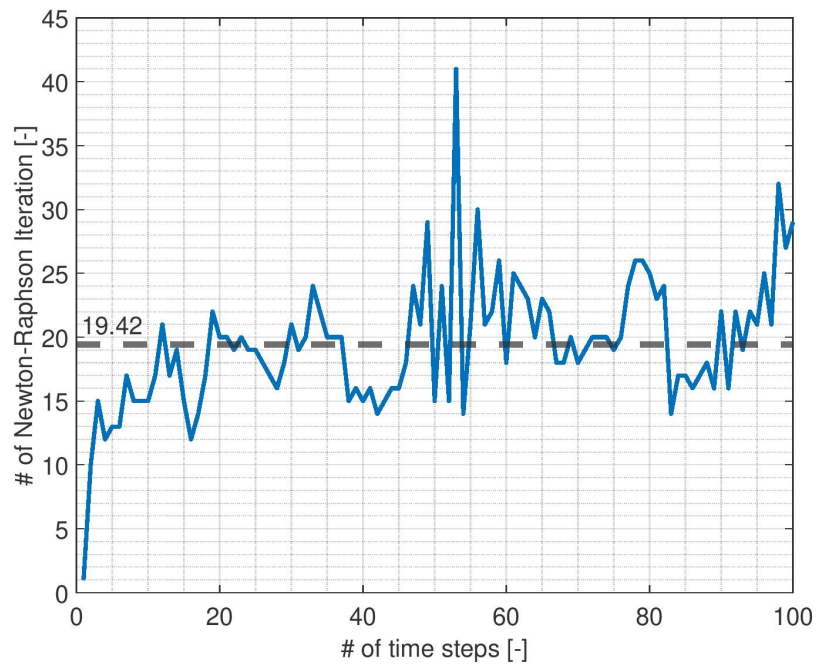


Figure 40: Influence of the Ironing Depth - Newton-Raphson Iterations.

more applicable. This facts has been the motivation to conduct a sensitivity analysis in terms of time-step value.

Prescribed displacement for the master body is described by the following equation. The prescribed number of time-steps was set to 50.

$$u(t) = \begin{cases} u_x(t) := \mathcal{H}(0.01 \cdot dt - t) \cdot (-20 * t) - 0.2 \cdot \mathcal{H}(t - 1 \cdot dt) \\ u_y(t) := 30 \cdot (t - 13 \cdot dt) \cdot \mathcal{H}(t - 13 \cdot dt) \end{cases} ; dt = 0.001s \quad (5.6)$$

Results The result is summarized in Table 5. It shows that for this setup the computation is extremely sensitive in terms of time-step value. The direct implication of the greater time-step is a greater displacement between two distinct time levels and thus creates a false input for the projection routine, that subsequently outputs the error.

dt	Number of completed iterations
0.01 s	1
0.005 s	1
0.001 s	50

Table 5: Influence of the Time-Step Value

5.4 3D simulation of Shallow Ironing

This section is designated for testing of the frictional contact implementation for 3D cases. All parts of the code, as well as the workflow described in Algorithm 1, are relevant for 3D problems as well.

5.4.1 Problem Setup

The setup of this computation has been inspired by the overall goal of this team work - temper rolling. Therefore, we will simulate a contact interaction between a deformable block of a material and a rigid master body in the form of a cylinder, which is supposed to represent the roller.

	Slave body
Number of elements	$[x, y, z] = [4, 2, 4]$
Degree of the elements	$[2, 2, 2]$
Young's Modulus [Pa]	$25 \cdot 10^6$
Poisson ration [-]	0.4
Density $[\frac{kg}{m^3}]$	2730

Table 6: 3D computation - Setup Parameters.

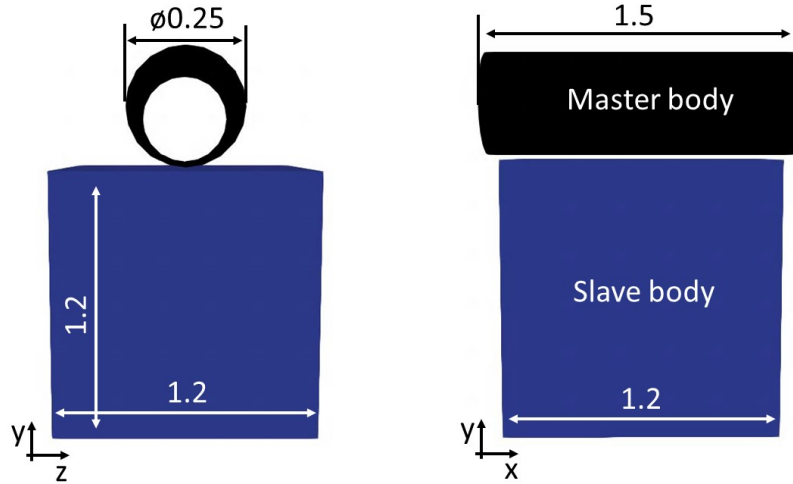


Figure 41: 3D Computation - Setup

Prescribed displacement for the master body is described by the following equation. The prescribed number of time-steps was set to 1350.

$$u(t) = \begin{cases} u_x(t) := & 0 \\ u_y(t) := & \begin{cases} -0.22 \cdot t & t \leq 1000 \cdot dt \\ -0.22 & t > 1000 \cdot dt \end{cases} \\ u_z(t) := & \begin{cases} 10 \cdot t & t > 1000 \cdot dt \end{cases} \end{cases}, \text{ for } dt = 0.001s \quad (5.7)$$

For the following computation, the values of both ϵ_N and ϵ_T were set automatically as described in Equation (5.1), with $\alpha = 10$.

5.4.2 Results

Figure 42 shows that the contact force components behave in the same manner as in previous 2D computations. The fact, that the extension of the problem to the third dimension shows same behaviour, gives us the confidence to claim that the implementation done in this thesis is indeed relevant even for 3D cases. As in the 2D cases, we may observe a difference in the forces magnitude. However, as in the previous cases, this can be accounted to the difference in the value of Young's modulus.

The average number of Newton-Raphson iterations is somewhat higher than in the 2D cases. This could be either caused by the extension to the third dimension itself or by some implementation imperfections. It is important to remark, that we decreased the convergence criterion to 10^{-9} . This criterion value is still reasonable and at the same time boosts the computation in terms of computational speed.

Figures 44 to 52 show the contact force component in z-direction in distinct time-steps.

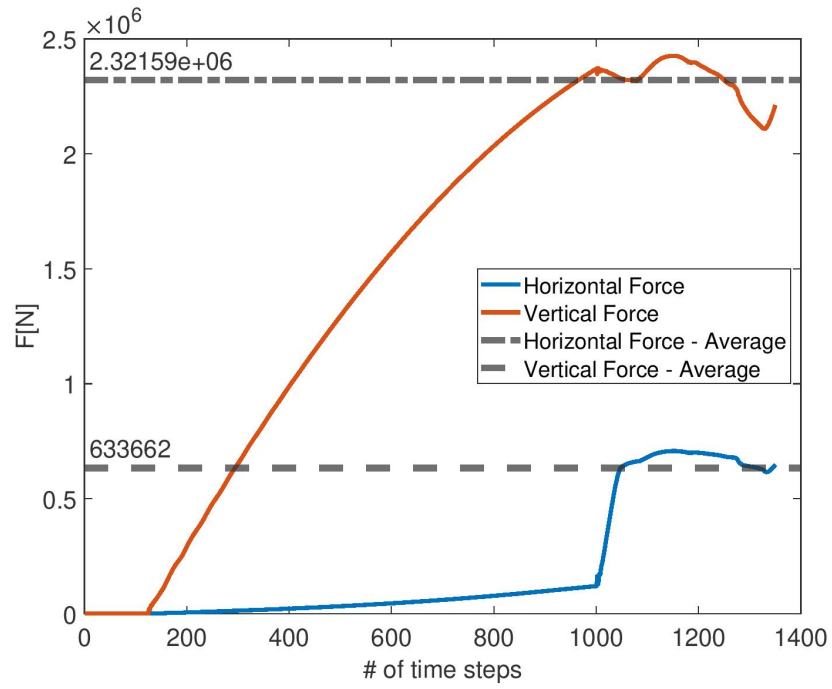


Figure 42: 3D Ironing - Contact Force.

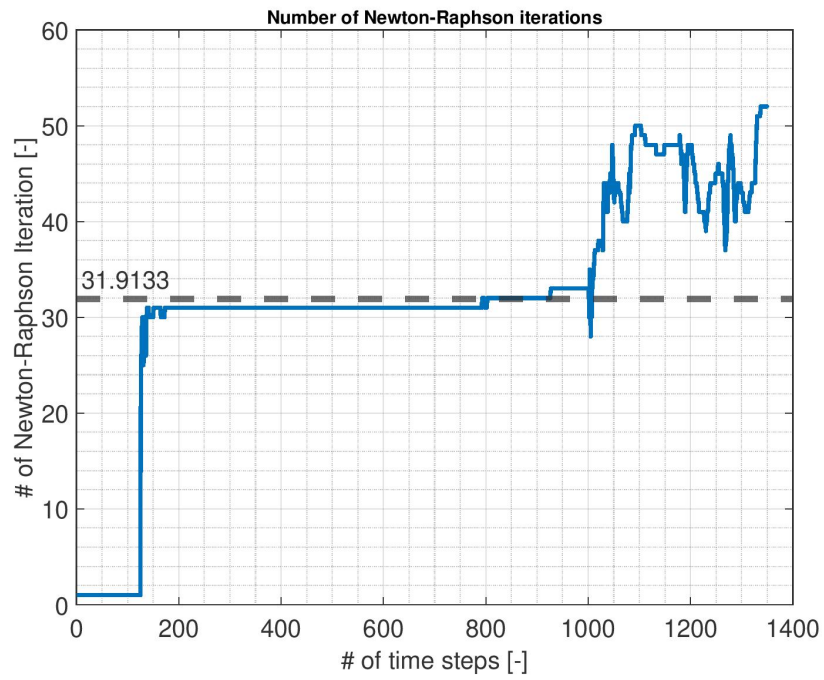


Figure 43: 3D ironing - Newton-Raphson Iterations.

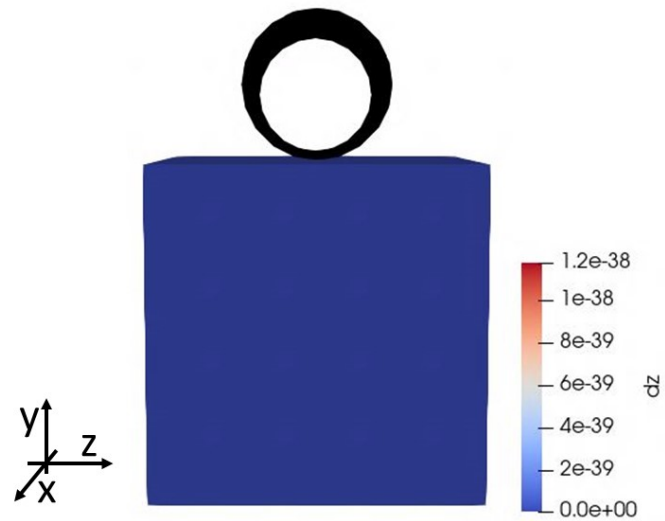


Figure 44: 3D Ironing - Force in z-direction $t = 0$ s.

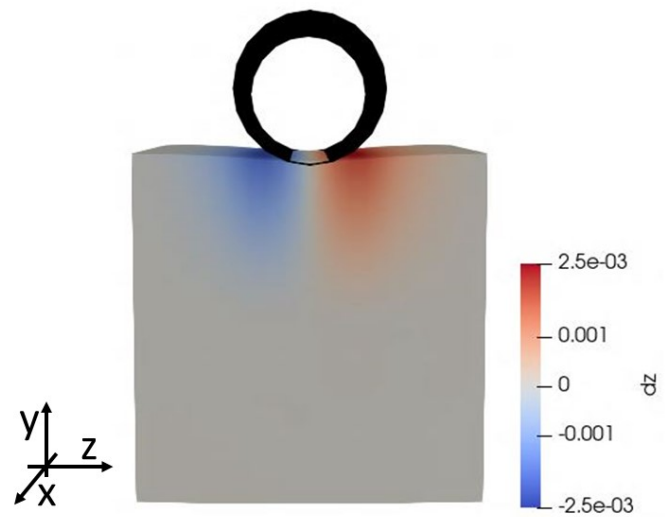


Figure 45: 3D Ironing - Force in z-direction $t = 0.25$ s.

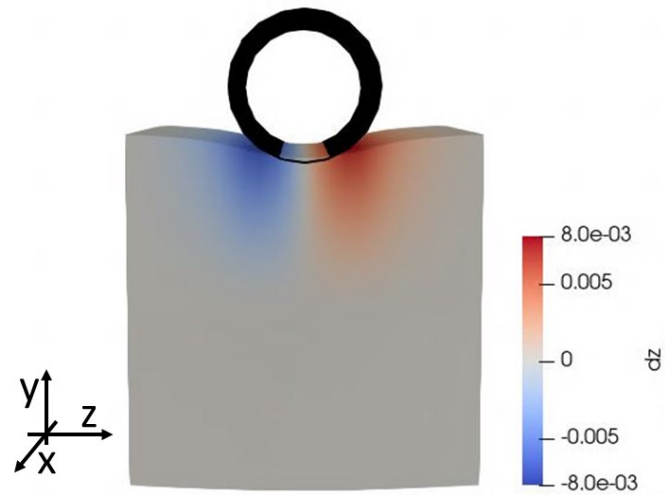


Figure 46: 3D Ironing - Force in z-direction $t = 0.5$ s.

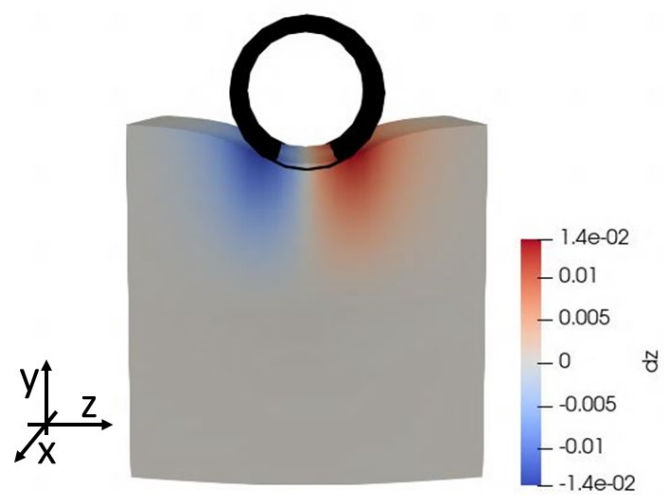


Figure 47: 3D Ironing - Force in z-direction $t = 0.75$ s.

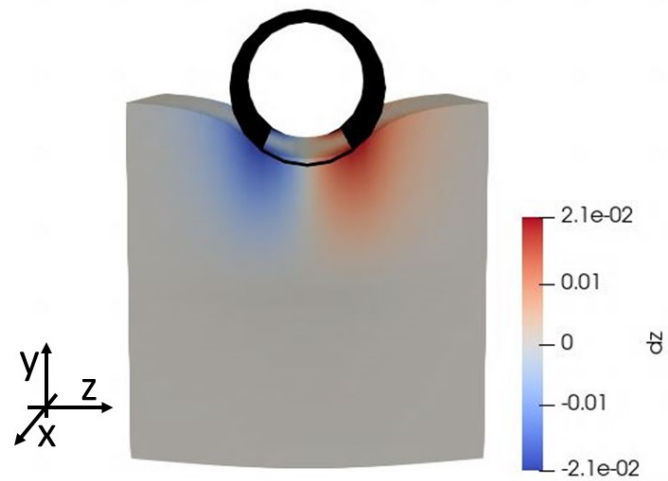


Figure 48: 3D Ironing - Force in z-direction $t = 1$ s.

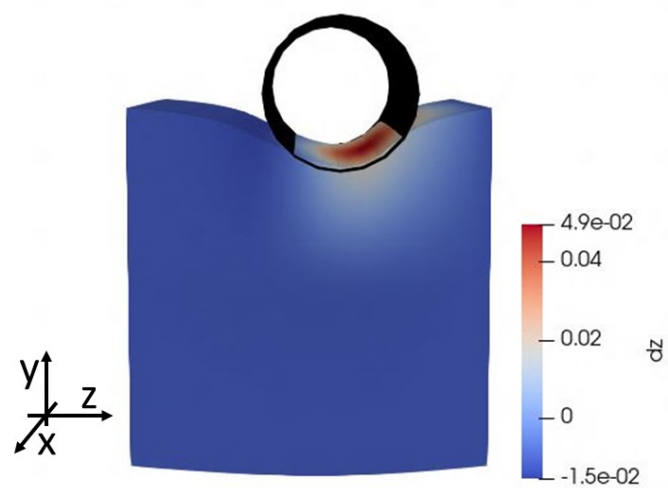


Figure 49: 3D Ironing - Force in z-direction $t = 1.1$ s.

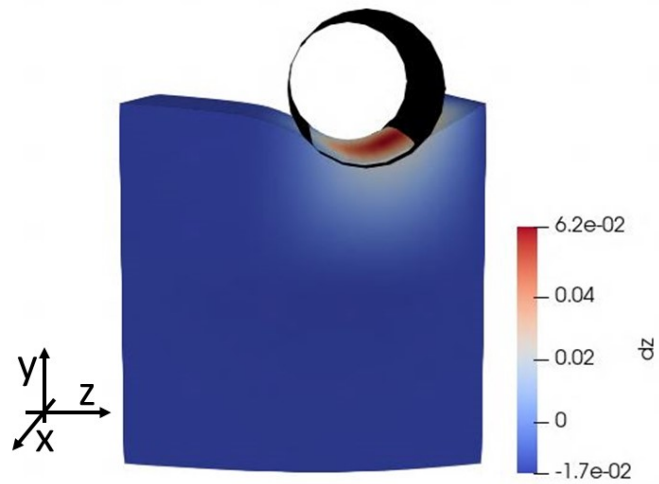


Figure 50: 3D Ironing - Force in z-direction $t = 1$ s.

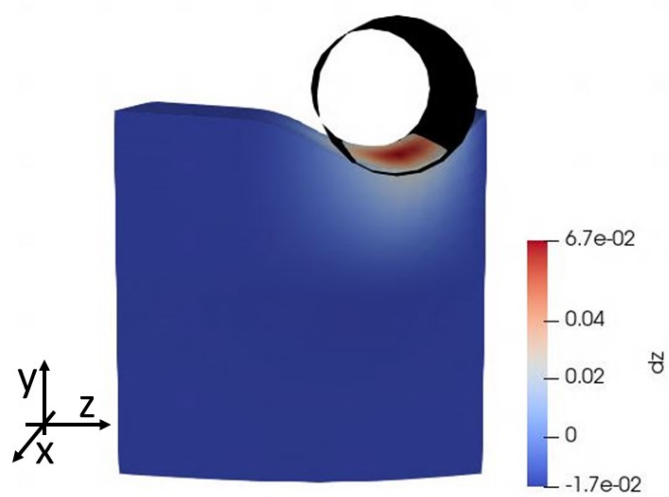


Figure 51: 3D Ironing - Force in z-direction $t = 1.3$ s.

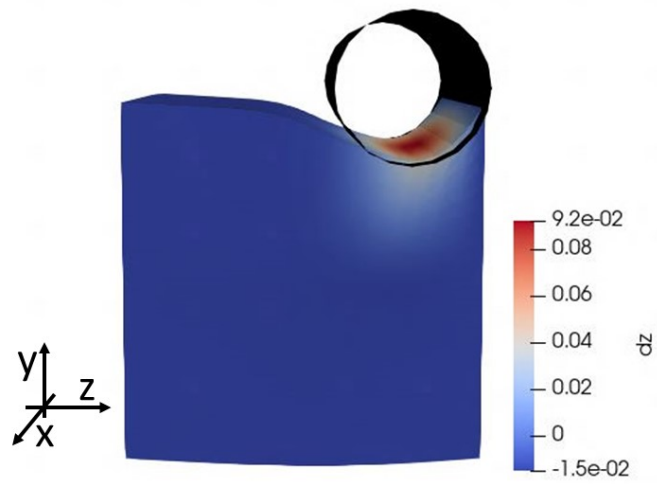


Figure 52: 3D Ironing - Force in z-direction $t = 1.35$ s.

6 Conclusion

This thesis has been designated to the implementation of frictional module to the already existing software framework of CATS institute at RWTH. The underlying theory of Fluid-Structure-Contact interaction has been introduced and it has been further elaborated on the theory of contact mechanics, namely friction, and its computational aspects, implementation and its advantages and shortcomings in terms of application in the area of temper rolling. To test our implementation and to find the weak spots of our implementation, we have conducted and described number of computations described in detail in Section 5.

The description of friction has been done by the Coulomb's law. This is generally accepted as a good approximation, however it does not account for the dynamic change of friction parameter. It would be interesting to determine whether different friction description would be beneficial in terms of FSCI computations.

From the performed computations in Section 5 we may conclude the following statements about the average number of Newton-Raphson iterations and the components of the contact force.

The average number of Newton-Raphson iterations depends mainly on the sliding status and the values or rather the proportion between the Young's modulus and the penalty parameters. Regardless of the prescribed displacement of the master body, ironing depth or the number of Gauss points involved, the average number of iterations in the performed 2D computation was 20.1858. The increase/decrease of the penalty parameters results in significant increase/decrease in the average number of iterations. The extension to the third dimension further increases the average number of Newton-Raphson iterations. It is, however, not yet clear, whether this jump is due to the additional problem dimension itself or due to some implementation imperfection.

All performed computations feature a similar evolution of the contact force over the computational time. The magnitude is influenced by the computation setup, namely the penalty parameters and Young's modulus of the slave body. The resemblance of the components of the contact force among the benchmark results and the results obtained through the performed 2D and 3D computations could be interpreted as a successful implementation of frictional extension into the existing framework.

While the desired implementation seems to output reasonable results, we must not forget about the shortcoming of the current state. As discussed in Section 2.4.1, the current method for determining the possible contact is the *Axis-aligned bounding box*. By employing more elaborate type of the bounding box method, e.g. *oriented bounding box* or *k faced discrete orientation polytope* [27], could speed up the computation considerably.

Acknowledgments

I would like to address my thanks to the head of the Chair of Computational Analysis of Technical Systems (CATS) at RWTH Aachen University, Prof. Marek Behr, Ph.D., and Priv.-Doz. Dr.-Ing. Stefanie Elgeti, for giving me the opportunity of participating in a field of research as challenging and fascinating as computational fluid-structure-contact interaction. Further, I would like to thank Dr.-Ing. Norber Hosters, whose lectures on Fluid-Structure Interaction ignited my interest and, through that, encouraged me to take the decisive step - applying for a master thesis at CATS institute.

I would like to express special thanks to my mentor Thomas Spenke, M. Sc.. Despite rather difficult times, he was always available and excited to answer any questions regarding the matter, even the silly ones. However, I do have one regret. Due to the current situation, I was not able to shake his hand nor could I express my gratitude in person for all his effort he invested in me. Hopefully some other time. Thank you.

In the end, I would like to thank the Department of Technical Mathematics at Faculty of Mechanical Engineering at Czech Technical University in Prague, namely to doc. Ing. Jan Halama, Ph.D., who gave me the opportunity to write my master thesis while being an Erasmus student at RWTH Aachen.

References

- [1] Klaus-Jurgen Bathe. *Finite element procedures* prentice-hall. *New Jersey*, 1037, 1996.
- [2] Carsten Braun. *Ein modulares Verfahren für die numerische aeroelastische Analyse von Luftfahrzeugen*. PhD thesis, Doctoral thesis, RWTH Aachen, Germany, 2007.
- [3] Carsten Braun. *Ein modulares Verfahren für die numerische aeroelastische Analyse von Luftfahrzeugen*. PhD thesis, Doctoral thesis, RWTH Aachen, Germany, 2007.
- [4] J Austin Cottrell, Thomas JR Hughes, and Yuri Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.
- [5] Aukje de Boer, Alexander H van Zuijlen, and Hester Bijl. Comparison of conservative and consistent approaches for the coupling of non-matching meshes. *Computer Methods in Applied Mechanics and Engineering*, 197(49-50):4284–4297, 2008.
- [6] C Agelet de Saracibar. A new frictional time integration algorithm for large slip multi-body frictional contact problems. *Computer Methods in Applied Mechanics and Engineering*, 142(3-4):303–334, 1997.
- [7] Joris Degroote, Robby Haelterman, Sebastiaan Annerel, Peter Bruggeman, and Jan Vierendeels. Performance of partitioned procedures in fluid–structure interaction. *Computers & structures*, 88(7-8):446–457, 2010.
- [8] Jean Donea and Antonio Huerta. *Finite element methods for flow problems*. John Wiley & Sons, 2003.
- [9] Christiane Förster, Wolfgang A Wall, and Ekkehard Ramm. Artificial added mass instabilities in sequential staggered coupling of nonlinear structures and incompressible viscous flows. *Computer methods in applied mechanics and engineering*, 196(7):1278–1293, 2007.
- [10] Bernhard Gatzhammer. *Efficient and Flexible Partitioned Simulation of Fluid-Structure Interactions*. Dissertation, Technische Universität München, München, 2014.
- [11] Thomas JR Hughes, John A Cottrell, and Yuri Bazilevs. Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer methods in applied mechanics and engineering*, 194(39-41):4135–4195, 2005.
- [12] Noboru Kikuchi and John Tinsley Oden. *Contact problems in elasticity: a study of variational inequalities and finite element methods*. SIAM, 1988.
- [13] Philipp Knechtges. *Simulation of viscoelastic free-surface flows*. Verlag Dr. Hut, 2018.

-
- [14] Ján Kopačka. Efficient and robust numerical solution of contact problems by the finite element method. 2017.
- [15] C. Braun L. Reimer, G. Wellmer and N. Hosters. *Dokumentation des Aeroelastischen Kopplungsmoduls (ACM)*. Chair for Computational Analysis of Technical Systems (CATS), RWTH Aachen University, 2016.
- [16] C. Braun N. Hosters M. BrÄEderlin E. Ongut L. Reimer, G. Wellmer and M. Make. *Documentation of the Structural Preprocessor Finite Element Analysis for Aeroelasticity (FEAFA)*. Chair for Computational Analysis of Technical Systems (CATS), RWTH Aachen University, 2016.
- [17] Hermann G Matthies and Jan Steindorf. Partitioned strong coupling algorithms for fluid–structure interaction. *Computers & structures*, 81(8-11):805–812, 2003.
- [18] Anand R Mijar and Jasbir S Arora. Study of variational inequality and equality formulations for elastostatic frictional contact problems. *Archives of Computational Methods in Engineering*, 7(4):387–449, 2000.
- [19] Boris Muha and Sunčica Čanić. Existence of a weak solution to a fluid–elastic structure interaction problem with the navier slip boundary condition. *Journal of Differential Equations*, 260(12):8550–8589, 2016.
- [20] M Aslam Noor. Some algorithms for general monotone mixed variational inequalities. *Mathematical and Computer Modelling*, 29(7):1–9, 1999.
- [21] Muhammad Aslam Noor. General variational inequalities. *Applied mathematics letters*, 1(2):119–122, 1988.
- [22] Roger A Sauer and Laura De Lorenzis. A computational contact formulation based on surface potentials. *Computer Methods in Applied Mechanics and Engineering*, 253:369–395, 2013.
- [23] Roger A Sauer and Laura De Lorenzis. An unbiased computational contact formulation for 3d friction. *International Journal for Numerical Methods in Engineering*, 101(4):251–280, 2015.
- [24] Thomas Spenke, Norbert Hosters, and Marek Behr. A multi-vector interface quasi-newton method with linear complexity for partitioned fluid–structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 361:112810, 2020.
- [25] Benjamin Walter Uekermann. *Partitioned fluid-structure interaction on massively parallel systems*. PhD thesis, Technische Universität München, 2016.
- [26] Georg Wellmer, Lars Reimer, Horst Flister, Marek Behr, and Josef Ballmann. A comparison of fluid/structure coupling methods for reduced structural models.

In *Management and Minimisation of Uncertainties and Errors in Numerical Aerodynamics*, pages 181–218. Springer, 2013.

- [27] Bin Yang and Tod A Laursen. A contact searching algorithm including bounding volume trees applied to finite sliding mortar formulations. *Computational Mechanics*, 41(2):189–205, 2008.
- [28] Vladislav Yastrebov. *Computational contact mechanics: geometry, detection and numerical techniques*. PhD thesis, École Nationale Supérieure des Mines de Paris, 2011.
- [29] David Young. The numerical solution of elliptic and parabolic partial differential equations. In *Modern Mathematics for the Engineer*, pages 373–419. McGraw-Hill Book Company, Inc., 1961.