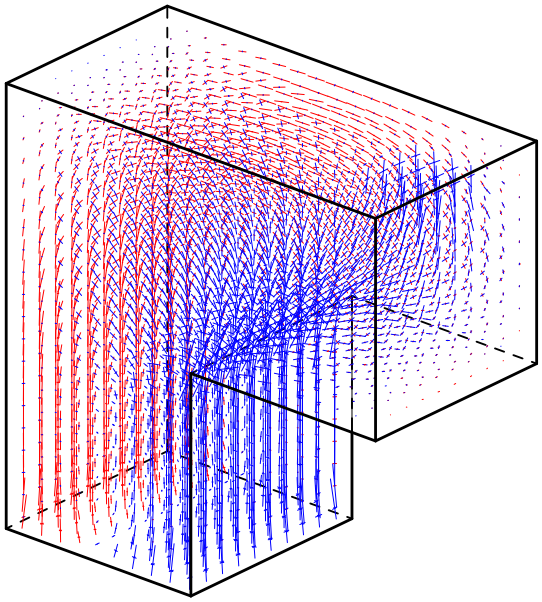
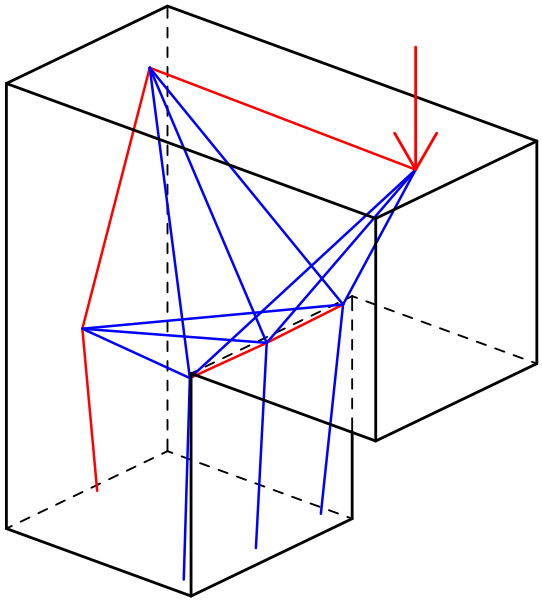
CHALMERS
UNIVERSITY OF TECHNOLOGY





3D Strut-and-Tie Modelling - Interactive Design Using Peridynamics

Development of a tool for generating strut-and-tie models using peridynamics combined with optimisation for an efficient reinforcement placement

Master's thesis in
Engineering Mathematics and Computational Science
Structural Engineering and Building Technology

KALLE THORSAGER
MATTIAS UDÉN

Department of Architecture and Civil Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Master's thesis ACEX30
Gothenburg, Sweden 2022

Master's thesis ACEX30

3D Strut-and-Tie Modelling - Interactive Design Using Peridynamics

Development of a tool for generating strut-and-tie models using peridynamics combined with optimisation for an efficient reinforcement placement

KALLE THORSAGER
MATTIAS UDÉN

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Architecture and Civil Engineering
Research Group for Architecture and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2022

3D Strut-and-Tie Modelling - Interactive Design Using Peridynamics
Development of a tool for generating strut-and-tie models using peridynamics combined with optimisation for an efficient reinforcement placement

KALLE THORSAGER
MATTIAS UDÉN

© KALLE THORSAGER, MATTIAS UDÉN, 2022.

Supervisor: MSc in Structural Engineering Johan Örnborg, VBK
Supervisor: Lic. Eng Jens Olsson, Department of Architecture and Civil Engineering, Chalmers University of Technology
Examiner: Senior Lecturer Mats Ander, Department of Industrial and Materials Science and Department of Architecture and Civil Engineering, Chalmers University of Technology

Master's Thesis 2022
Department of Architecture and Civil Engineering
Research group for Architecture and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Principal stress field and suggested strut-and-tied model for a corbel in 3D with a point load acting on it.

Typeset in LaTeX
Department of Architecture and Civil Engineering
Gothenburg, Sweden 2022

3D Strut-and-Tie Modelling - Interactive Design Using Peridynamics
Development of a tool for generating strut-and-tie models using peridynamics combined with optimisation for an efficient reinforcement placement
KALLE THORSAGER & MATTIAS UDÉN
Department of Architecture and Civil Engineering
Chalmers University of Technology

Abstract

The design of discontinuity regions in reinforced concrete structures is usually investigated in the ultimate limit state with a Strut-and-Tie Model (STM). By using STM the structure is designed by a lower bound approach according to the theory of plasticity. This makes the design conservative, assuming that the concrete structure can undergo sufficient plastic deformation. Therefore, it is good practice to construct the truss for the STM from the linear stress state in the structure.

A new efficient GPU accelerated method for the particle-based method peridynamics is proposed to produce the principal stress field in these regions interactively. Voxels are utilised to discretise the input volume into particles. The use of voxels in the peridynamics workflow has proved useful in terms that each particle's neighbour list is implied and data locality is assured. By implementing the peridynamics solver so that it utilises the GPU, the run-time of the simulation is drastically decreased on a personal computer. The boundary conditions are also treated with a new implicit method which reduces the influence of edge effects close to the boundary.

An initial truss topology is then produced algorithmically by tracing curves in the principal stress field. Nodes are created at local maximum von Mises points along these curves and new curves are traced from these nodes. Nodes are also inserted at relevant intersection points between the traced curves. Based on these principles an initial truss topology can be created for most structures.

A gradient-based optimisation is applied to the truss topology to get a truss with minimal reinforcement volume while still fulfilling the checks for the STM in a simplified manner. The gradient of displacement is used both with regards to the change of the initial node locations and element areas. The resulting truss and distribution of the reinforcement will be useful in an early design process but since the exact placement of the bars is not decided a more detailed analysis will be needed at a later stage.

The application is useful in terms of understanding the structural behaviour of 3D structures and estimating the reinforcement volume and placement.

Keywords: Strut-and-Tie, STM, Particle Methods, Peridynamics, Computational Design, Topology Optimisation, Gradient Based Optimisation.

Acknowledgements

We would like to direct our sincere gratitude to our supervisors, Johan Örnborg and Jens Olsson, as well as our examiner Mats Ander for sharing their knowledge and expertise. They have continuously supported us throughout this process and provided us with valuable feedback.

Special thanks to Filip Götborg for his critical review of our work.

Kalle Thorsager & Mattias Udén
Gothenburg, June 2022

Contents

List of Figures

List of Tables

1
Introduction

For most structures, there exist regions that do not adhere to ideal elements such as beams and plates. These parts of the structures are called discontinuity regions (D-regions) and are therefore not as straightforward to design. This makes it more challenging to determine the amount of reinforcement and its location for reinforced concrete structures. However, these parts are often crucial to the structure, and therefore it is important that they are designed correctly and that it can be verified that they will be able to sustain the loads acting on them. A common model to use for these parts is the Strut-and-Tie Model (STM) which will simplify the D-region into a truss with reinforcement ties and compressive struts in concrete. For complex 3D geometries, it is often hard to develop a suitable STM. In this project, it will be investigated if the particle-based method peridynamics can be utilised to come up with an appropriate STM which also will be optimised to reduce the reinforcement amount under certain constraints.

1.1 Background

The applicability of peridynamics in an interactive design tool was investigated by Edefors & Jonsson in [1]. They concluded that due to the mathematical formulation of peridynamics it is well suited for interactive design since alterations can be made during the run-time and also that based on the principal stress field an STM could be utilised to come up with the reinforcement placement. The following section contains a brief introduction to STM, peridynamics as well as how peridynamics can be used to develop the STM and optimisation algorithms.

1.1.1 Strut-and-Tie Modelling

The STM was first presented by Schlaich *et al.* in 1987 [2]. The method was developed to create better models for discontinuity regions in reinforced concrete structures. These regions are characterised by that the strain distribution is nonlinear over the cross-section which occurs in structures near concentrated loads, changes of the geometry, deep beams, etc. In these regions, common methods based on idealised elements cannot be utilised. Previously, before the development of STM, rules of thumb and engineering judgment were used to design these regions. However, these discontinuity regions are of great importance for the structures and therefore STM was developed to design them more coherently.

There are several different geometrical constraints that the STM must be able to fulfil. According to Engström (2015) [3], there are several limitations between both the relative angle of the applied forces (loads and reactions) and also the angles between the struts and ties. For some models, all of these criteria cannot be met. If that were to happen these limitations are most important for the struts and ties with the largest loads. It is also important that it is possible to place the needed reinforcement amount in the ties concerning the cover distance, the distance between the bars, and the anchorage of the reinforcement. The usual approach for the design of STMs is to sketch the load path of the specific problem, according to certain rules outlined by Schlaich and Schäfer in [4]. Figure 1.1 shows an example of the load paths and what a viable STM could look like. Instead of using the load path method, the linear stress field can be interpreted into an STM. The principal stresses for a deep beam can be seen in Figure 1.2, it can be seen that the strain distribution is non-linear over the cross-section and that the principal stresses in compression are somewhat similar to the load paths in Figure 1.1.

According to Eurocode 2 [5] the STM may be used for the design of parts of concrete structures where the stresses have a non-linear strain distribution over the cross-section. There are also requirements regarding the different nodal capacities in the STM and anchorage requirements.

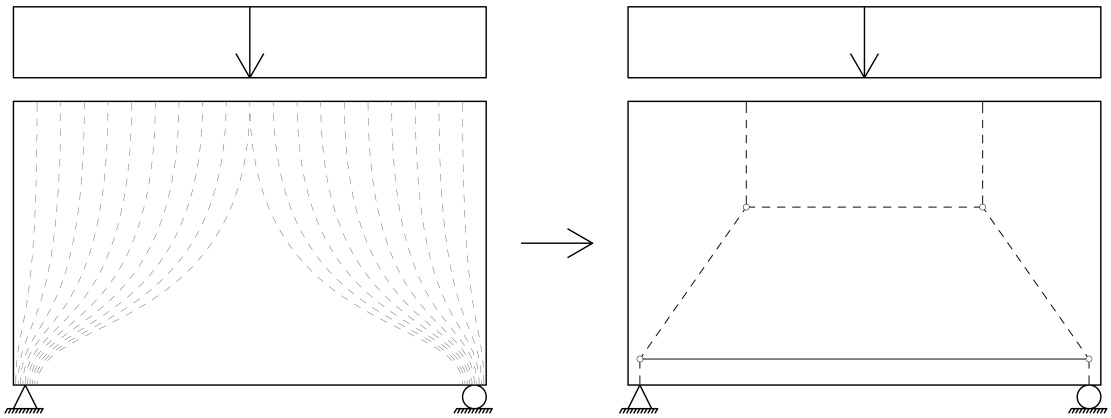


Figure 1.1: Load path and suggestion of what the corresponding STM could look like for a deep beam with an evenly distributed load.

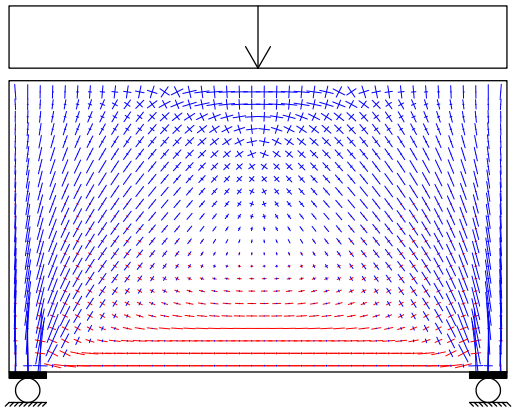


Figure 1.2: Principal stresses in a deep beam, blue indicate compressive stresses and red tensile.

1.1.2 Peridynamics

The theory of Peridynamics was first introduced by Silling in 2000 [6]. The driving idea was to have a theory which could handle the spontaneous formation of discontinuities such as cracks. Continuum mechanics can not handle cracks because it is based upon partial derivatives which are undefined at these discontinuities. Peridynamics is instead based upon integration, which is used to compute forces on a discrete set of material particles.

To model displacements using an equation without spatial derivative, Silling assumed forces could act over distances and that the interaction between two points in the continuum can be described by the displacement field. [6]
From this the peridynamic equation of motion is:

$$\rho \ddot{\boldsymbol{u}} = \int_H \boldsymbol{f}(\boldsymbol{u}' - \boldsymbol{u}, \boldsymbol{x}' - \boldsymbol{x}) dH + \boldsymbol{b}, \tag{1.1}$$

$(\cdot)'$ = value at centre of dH,
ρ = density [kg/m^3],
$\boldsymbol{x}$ = initial position [m],
$\boldsymbol{f}(\cdot,\cdot)$ = pairwise force density [N/m^6],
$\ddot{(\cdot)}$ = second derivative in time,
$\boldsymbol{u}$ = displacement [m],
$\boldsymbol{b}$ = bodyload [N/m^3],
H = Region of influence [m^3].

For simulations, a finite number of particles are assigned to represent the structure and Equation 1.1 can be simplified to a discrete sum. For an example of particles interacting with each other see Figure 1.3. These particles' motions are then tracked and continuously updated with time-stepping, where the steps must be small as the forces are dependent on all the other nearby particles' positions. In the case where a steady-state system is sought, dynamic relaxation techniques are used which involve damping and modifying the masses of the particles. [7]

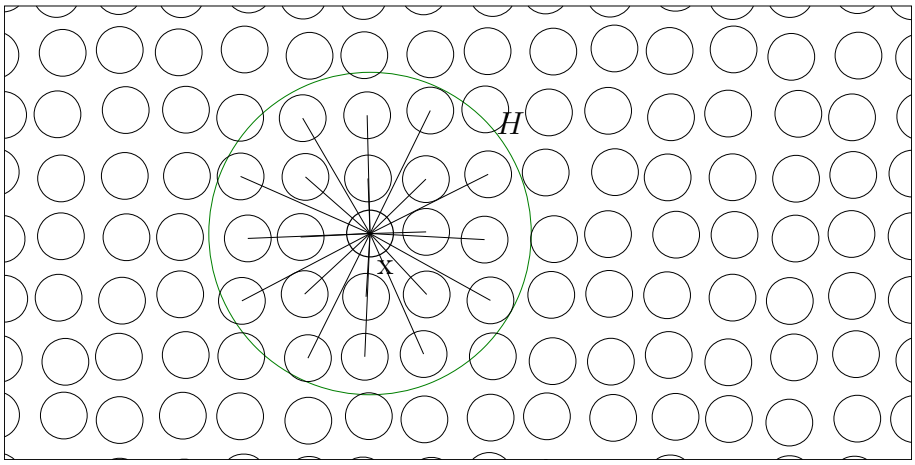


Figure 1.3: Illustration of how a particle at point **x** forms interactions with other particles within the region H.

Since peridynamics is an iterative approach, and the setup is mainly about finding the neighbours, it is suitable for interactive use. This is because the parameters of the model can be updated without disregarding the previous solution, essentially using the last time step as the initial state in the next simulation.

These properties allow one to go from some volume representation to a model, which can be updated by a user and changed by iterative optimisation, both for geometrical and material parameters, but also for incremental load increases for non-linear analysis, where the finished model can still be changed and optimised.

Time can be taken to note that there are other meshfree methods to which peridynamics bears a resemblance, such as Smooth Particle Hydrodynamics (SPH) as described by Gingold and Monaghan [8], and that since the introduction of peridynamics in 2000 further developments such as state-based with ordinary and non-ordinary peridynamics described Silling *et al.* [9] which allows for more accurate material models. Efforts to look at the subject from a statistical viewpoint have also led to the reproducing kernel peridynamics formulation as described by Hillman *et al.* [10] which uses the neighbouring particles as integration points to approximate the strain state of the particle. Furthermore using the similarities between SPH and peridynamics a force flux peridynamics has been conceptualised to deal with a variable horizon better as described by Olsson *et al.* [11].

1.1.3 Producing Truss Topology

To go from the peridynamic model to an STM, a truss topology needs to be developed. There exist several different methods that go from a continuous block of material into a truss model. A commonly used method is evolutionary structural optimisation which slowly removes the least stressed material as described by Xie *et al.* in [12]. This method only concentrates the material to where it is most needed and therefore does not generate a truss directly. This new geometry still needs to be translated into a truss model, which could be done by hand by an engineer or,

for example, with a neural network. Evolutionary structural optimisation is also a computationally heavy method that requires a lot of iterations before it reaches an appropriate result.

Another approach that can be used to generate the truss is instead based on the principal stress field in the material as described by Kwok *et al.* in [13]. This method is divided into two steps where the first one is to generate an initial truss that connects the load to the supports in the simplest way while still in general following the principal stresses in the material. The second step is introduced to expand the truss by inserting new nodes at appropriate positions and thus growing and optimising the truss topology. Whether a position is suitable or not for a new node is determined by the principal stress field. The goal is that the truss members will follow the principal stress field as closely as possible. [13]

In [13] comparisons are also made between this method based on principal stress field and evolutionary structural optimisation, see Figure 1.4. It can be seen that the structures generated based on the principal stresses required less computational time as well as structures much more closely related to the analytical Michell solution. The Michell truss is an optimal structure based on Maxwell's load-path theorem [14]. In this thesis, the truss topology will be generated by the principal stress field and not by evolutionary structural optimisation.

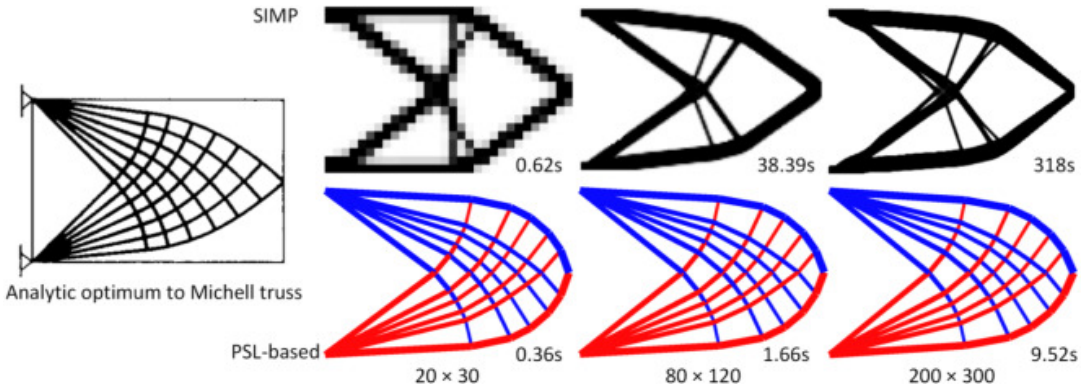


Figure 1.4: Comparison between the analytical Michell truss and a structure generated by both the evolutionary structural optimisation method Solid Isotropic Material with Penalisation (SIMP) and the principal stress line (PSL) method. The computational time to generate the different structures is also shown. [1]

[1]Reprinted from Computer-Aided Design, **80**, T. Kwok, Y. Li & Y. Chen, *A structural topology design method based on principal stress line*, 19-31, (2016), with permission from Elsevier.

1.1.4 Optimisation

The produced truss is based on the linear stress field but STM aims to model the lower bound plastic response. That is, a stress field is assumed which fulfils equilibrium in the structure. The actual capacity of the structure will be larger than the calculated capacity, under the condition that enough plastic redistribution can take place [3]. To limit the need for plastic redistribution, the linear stress field is utilised to construct the initial truss. However, STM is still based on plastic analysis and therefore optimisation will be used to coerce the solution to represent the plastic response.

The mathematical field of optimisation is in essence a subject about finding the minimum of a function $f(x)$ subject to some constraints on x,

$$\begin{aligned} \min_{x} \quad & f(x), \\ \text{subject to} \quad & x \in X. \end{aligned} \tag{1.2}$$

Since the evaluation of both the function $f(x)$ and the inclusion test of $x \in X$ may be expensive, quite a few methodologies have been developed to solve this.

We wish to minimise the amount of reinforcement used. It is however a coupled problem where the stress dictates where it is needed and its placement changes the stress. To deal with this problem the linear elastic stress field can be used to guide an initial truss placement, such that the topology of the truss is decided. Then as a secondary problem, the placement of the topology can be optimised, which will drastically reduce the number of variables, and allow the use of STM for constraints.

The optimisation is about displacing the nodes and element areas of the truss. The constraints should ensure that stress in the elements should not exceed a critical value, and a FEM solution for the truss can inexpensively be used and also have its derivative taken with regard to displacements. Further constraints are naturally that the minimum cover distance and all other physical constraints should hold during the deformation.

1.2 Aim

This thesis aims to develop an application that will go from a general 3D geometry with boundary conditions to an optimised reinforcement placement via a 3D STM based on principal stresses from peridynamics. The input should be a discontinuity region in a concrete structure with certain boundary conditions. The tool should translate the input geometry into a peridynamics model which calculates and display the principal stress field. The display of the principal stresses should be clear and it should be possible to interactively adapt the model based on this.

An STM should be created based on the stress field for the discontinuity region. The truss model should then be optimised, with the primary goal to reduce the needed

reinforcement amount while still fulfiling the necessary criteria. The user interface should be easy to understand and it should be possible to interact with the program to achieve a better result. The output from the program should be clear for the user and easily interpreted. The program should also run smoothly and efficiently solve the problem. The run-time should be within a reasonable time so that the tool will function with ease and still be user-friendly.

1.3 Outline of the Program

The main idea is to develop an application as outlined in Figure 1.5. The blue boxes represent input and outputs from the application and the orange boxes the different steps that need to be conducted. It should be noted that user input will be required in other parts of the process as well but to a smaller degree. The input mesh will be discretised into a voxel representation and used in the peridynamics solver with supplied boundary conditions. The peridynamics solver will calculate the principal stress field which will be utilised to construct an initial truss topology for the STM. The STM will be optimised with the primary goal to reduce the reinforcement amount and will use a signed distance field (created from the voxels) to make sure that the nodes stay within the domain.

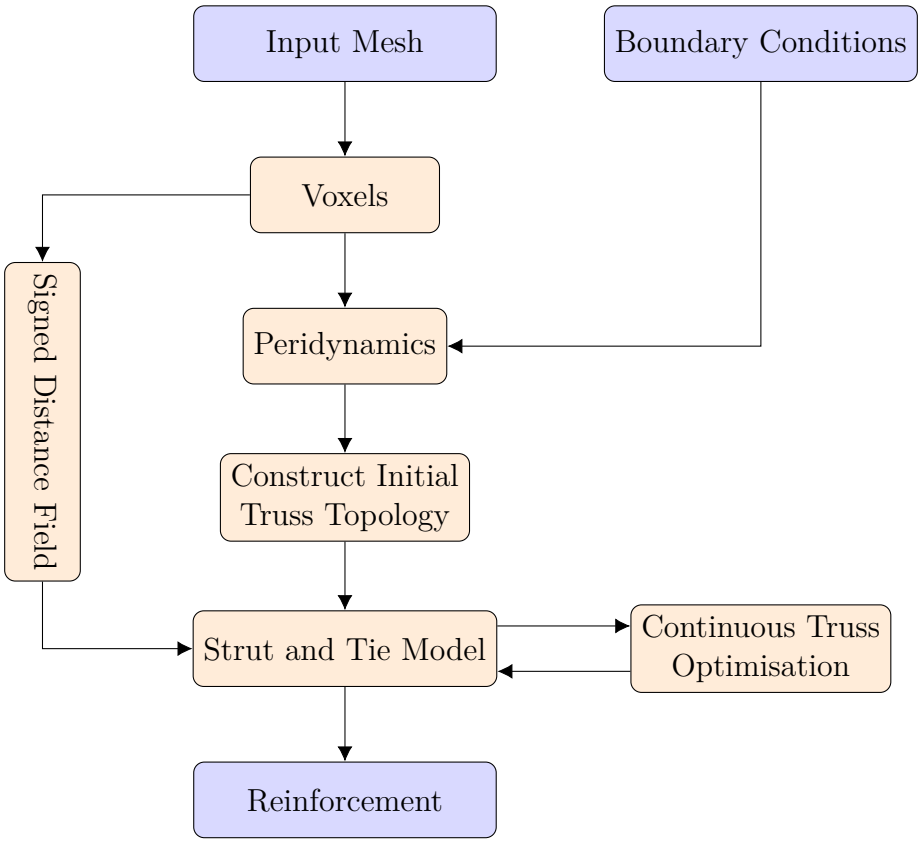


Figure 1.5: Flowchart of the full program plan.

The project will be implemented in the 3D computer graphics and computer-aided design program Rhino with the plug-in Grasshopper, which is a visual programming environment. Our application will be scripted in C# which is an object-oriented language suited for scripting components in Grasshopper and Rhino. Microsoft Visual Studio will be used as an integrated programming environment.

1.4 Research Questions

- Is the proposed program in Figure 1.5 a useful application, in terms of:
  - ease of use?
  - speed?
  - accuracy?
- How large impact do the changes of the peridynamics workflow have in terms of:
  - storing the points and neighbours implicitly by index as opposed to explicitly as points and lists?
  - how to handle the boundary conditions more coherently?
- Optimisation of the STM:
  - is it possible to generate a reasonable truss from the displacement field?
  - can we then apply optimisation to find a more optimal model and reduce the reinforcement amount?

1.5 Limitations

The application will only analyse the geometry in the Ultimate Limit State (ULS). For example, the cracking behaviour and deflections in the Serviceability Limit State (SLS) will not be analysed. STM is mainly used to analyse discontinuity regions at ULS and therefore this project is limited to this stage. If the STM topology is closely related to the linear elastic stress field only small plastic deformations will occur before equilibrium is reached in the geometry. With the implementation of plastic peridynamic analysis, it should not be too difficult to analyse the geometry in SLS as well. However, it would also require some extensive verification before the results could be trusted and used for design.

Most structures are subjected to different loads during their lifetime which will lead to different requirements. Therefore it is often needed to consider different load-cases when designing the structure. However, to simplify the optimisation procedure only one load case at a time will be considered.

The optimisation will aim to make use of as little reinforcement as possible. Hence no attempt will be made to optimise the volume in itself and the assigned ULS load will be used as an input. That is, it will find the best use of reinforcement for a problem, not the best way to use reinforcement.

The application will work as a guide to create an STM, which is an established method used a lot in practice today. Therefore the results will be verified according to the standard STM framework and Eurocode. The idea is not to create a tool that will validate the STM in itself.

The application will only verify the resulting STM against Eurocode, it will not verify against other design codes. This could probably be implemented with ease in a later stage if there is a need for it.

It would also be practical if in the future our application could be connected to a larger structural Finite Element (FE) model, which is a much more used method today. However, in the scope of this project, the focus will only be on our application and not on how to integrate it further with a FE model. If the tool were to be developed further this could be a natural step in making it more useful.

Part I

Theory

2

Strut-and-Tie Method

A detailed description of STM was presented in [4] by Schlaich *et al.* (1991). STM is in general utilised to design discontinuity regions in reinforced concrete structures, that is regions that do not adhere to ideal elements such as beams and plates. By employing STM, the structure is designed according to a lower bound approach based on the theory of plasticity. This enables a conservative design, as long as enough plastic deformations are possible within the structure. Therefore the chosen truss model should not deviate too much from the principal stress field in the linear state [3]. According to Schlaich *et al.* [4] the truss model can be optimised by minimising the inner strain energy

$$\sum_i F_i l_i \epsilon_{mi}, \tag{2.1}$$

F_i = force in strut or tie i [N], l_i = length of member i [m],
ϵ_{mi} = mean strain in member i [-].

It is in general enough to only consider the reinforcement ties in Equation 2.1 because compared to the concrete struts they can deform much more. Since this model deals with plasticity and large deformations in these regions engineering judgment is still needed to choose a truss model that will capture an appropriate simplification of the force flow.

2.1 Application Rules

According to Eurocode 2 [5] the STM may be used in parts of a structure with a non-linear strain distribution over the cross-section. There are then several different criteria that need to be fulfilled. The following factors and parameters are used in the following sections:

γ_c = partial safety factor for concrete,
f_{cd} = design compressive strength of concrete,
f_{ck} = characteristic compressive strength of concrete.

The recommended value of the effectiveness factor, ν, as defined in Eurocode 2 [5] is

$$\nu = 1 - \frac{f_{ck}[\text{MPa}]}{250}. \tag{2.2}$$

The effectiveness factor is used to calculate the equivalent plastic strength from the uniaxial strength. In general, it is not easy to define the effectiveness factor but in STM applications, Equation 2.2 can be used. [3]

2.1.1 Limitations of Angles

The following limitations of angles are given for the structure to have a reasonable behaviour in the service state and to not require extensive plastic deformation before reaching the designed equilibrium in the ultimate limit state. For some structures, it can be hard and even impossible to develop an STM that fulfils all of the limitations. In those cases the angles for more heavily loaded components should be prioritised, the entire model does not need to be discarded [3].

The following limitations between struts and ties in the model are recommended by Engström in [3].

- Deviation angle for concentrated loads entering the D-region: $\sim 30°$, but smaller than $45°$.
- Deviation angle for strut meeting ties in only one direction: $\sim 60°$, but larger than $45°$.
- Deviation angle for strut meeting ties in multiple directions: $\sim 45°$, but larger than $30°$.

2.1.2 Limitations of Nodes

The design capacity of a node depends on the stress state and the number of ties that pass through the node. If the node exists in a part of the structure with a distributed stress field it is called a distributed node and will not be critical in the design. For the opposite case when the node exists in a concentrated stress region it is called a concentrated node and its capacity needs to be verified. It is the compressive stresses in the nodes that need to be verified, the tensile stresses are considered when calculating the required steel area as described in Section 2.1.4. [3]

The following limitations of the node capacities are defined in Eurocode 2 with the recommended value for the factors k_{1-4} [5].
For compression nodes with no anchored reinforcement the capacity is determined as:

$$\sigma_{Rd,max} = k_1 \nu f_{cd} \quad \text{where} \quad k_1 = 1.0. \tag{2.3}$$

For compression - tension nodes with reinforcement in only one direction the capacity is determined as:

$$\sigma_{Rd,max} = k_2 \nu f_{cd} \quad \text{where} \quad k_2 = 0.85. \tag{2.4}$$

For compression - tension nodes with reinforcement in at least two directions the capacity is determined as:

$$\sigma_{Rd,max} = k_3 \nu f_{cd} \quad \text{where} \quad k_3 = 0.75. \tag{2.5}$$

The capacity of the compressive stress may be increased by 10% if at least one of the following conditions is fulfilled [5]:

- Triaxial compression is assured.
- All angles between struts and ties are larger than 55°.
- The stresses applied at supports or at point loads are uniform, and the node is confined by stirrups.
- The reinforcement is arranged in multiple layers.
- The node is reliably confined through bearing arrangement or friction.

For the special case with triaxially compressed nodes, the capacity may instead be determined as:

$$\sigma_{Rd,max} \leq k_4 \nu f_{cd} \quad \text{where} \quad k_4 = 3.0, \tag{2.6}$$

with a modified value of f_{cd} considering the triaxial stress state [5].

2.1.3 Limitations of Struts

The capacity of the concrete struts needs to be checked concerning unfavourable multiaxial stresses. For the case with transverse compression or no transverse stress, the design capacity of the strut can be determined according to

$$\sigma_{Rd,max} = f_{cd} \quad \text{where} \quad f_{cd} = \alpha_{cc} \frac{f_{ck}}{\gamma_c}, \tag{2.7}$$

and α_{cc} is a coefficient considering long term effects with recommended value 1.0 [3]. If the strut instead is subjected to transverse tension or is located in a cracked region the design capacity is determined as [3],

$$\sigma_{Rd,max} = 0.6 \nu f_{cd}. \tag{2.8}$$

According to Engström, it is in general not necessary to check the capacity of the struts if it is provided with transversal reinforcement and that the nodes have enough capacity. The minimum reinforcement can for some cases be assumed to fulfil this purpose or a more detailed STM can be used which includes the need for secondary reinforcement. [3]

2.1.4 Limitations of Ties

For the structure to have a ductile behaviour in the ultimate state, the stress in the reinforcement ties should reach the dimensioning yield stress of the steel, f_{yd}. The required steel area, A_s, can be determined by

$$A_s \geq \frac{T}{f_{yd}}, \tag{2.9}$$

where T is the tensile force in the tie [3]. The tensile capacity of the concrete is neglected.

2.2 Reinforcement

The placement of the required reinforcement needs to fulfil several different criteria as stated in Eurocode 2 [5]. In STM, it is assumed that the reinforcement bars can be fully anchored within the node region which needs to be verified. The required anchorage length, l_{bd}, can be calculated as

$$l_{bd} = \alpha_1\alpha_2\alpha_3\alpha_4\alpha_5 l_{b,rqd} \geq l_{b,min}. \tag{2.10}$$

Where α_{1-5} are coefficients defined in Eurocode 2, $l_{b,rqd}$ is the basic required anchorage length and $l_{b,min}$ is the minimum anchorage length. If there are not enough space to achieve the required anchorage length with straight bars, bent bars can be used or other methods such as welding the bars to a steel plate.

The minimum bar spacing in all directions also needs to be considered as well as the cover distance to ensure the durability of the structure. It might also occur that the ties in the strut-and-tie geometry are longer than the available reinforcements bars, or the length of the bar may be limited due to some other factor. These situations are usually solved by having overlapping bars, it is then important to make sure that these laps are done correctly so the forces can be transmitted between the bars. All of these requirements need to be verified in accordance with Eurocode 2 [5].

2.3 Enhanced Strut-and-Tie Method in 3D

To utilise STM in 3D some adaptations of the model need to be conducted. Chantelot & Mathern [15] have proposed how the nodal zones can consequently be characterised and the strut geometries defined for the 3D case. All of the stated verifications in Section 2.1 are valid in 3D as well, including the angle limitations and node capacities. The 3D nodal zones are considered to be cuboids and the struts then can be seen as projections of the cuboid along the system line [15], for example, see Figure 2.1.

As noted by Bohman & Chaudhry in [16], the system line of the strut, as defined by the STM, does not necessarily coincide with the centroid axis of the strut, see Figure 2.2. Only for the special case when both connecting cuboids are of the same size or the angle of the strut is 45 degrees in relation to the cuboids. The difference between the system line and centroid axis of the strut is in general relatively small, therefore this deviation is deemed acceptable in this thesis and not considered further.

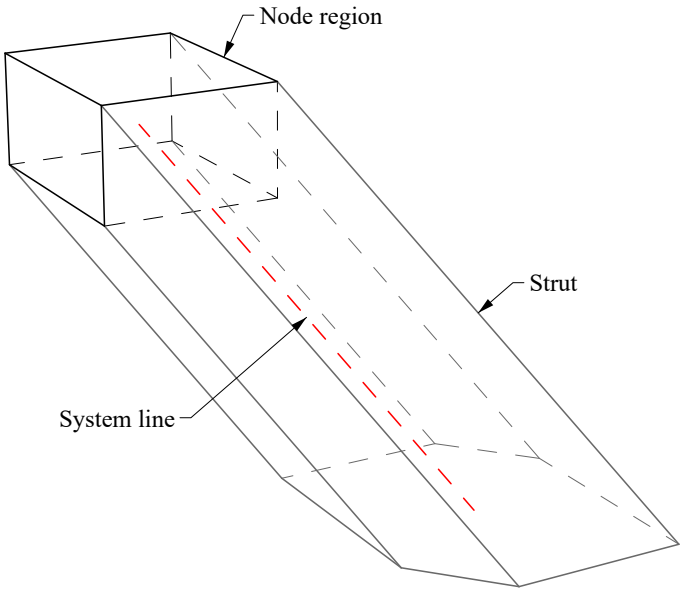


Figure 2.1: The considered 3D nodal zone as a cuboid and the strut as a projection of the cuboid along the system line.

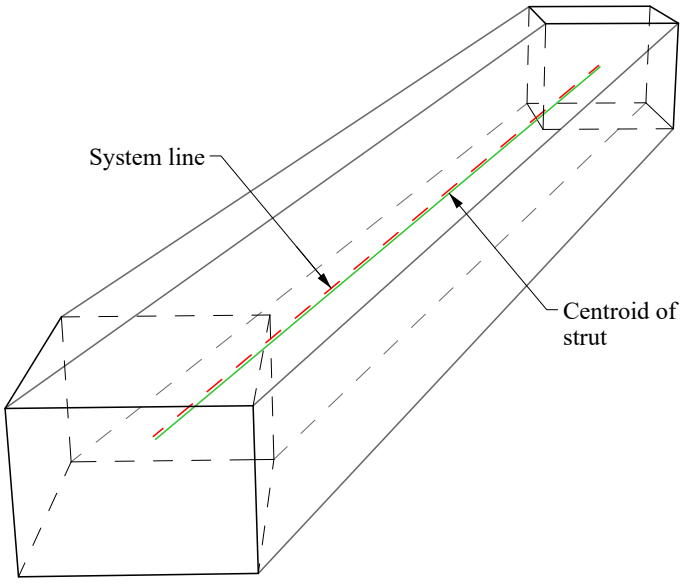


Figure 2.2: For a strut between node regions with different sizes and an arbitrary angle, the system line (dashed red) does not perfectly match the centroid line (solid green) of the strut.

3

Peridynamics

Peridynamics is a particle-based continuum theory first introduced by Silling in 2000 [6]. There are several different branches of peridynamics but the following sections will be focused on bond-based peridynamics.

3.1 Bond-based

The theory for bond-based peridynamics was first presented by Silling in [6]. He stated that the interaction between each pair of particles, within a body with region R, can be described by a vector-valued function $\boldsymbol{f}$, such that the force per unit reference volume due to interaction with other particles is a functional, $\boldsymbol{L}$, of the displacement field $\boldsymbol{u}$. For any point $\boldsymbol{x}$ in R and at any time t, the value of $\boldsymbol{L}$ is then given by:

$$\boldsymbol{L}_{\boldsymbol{u}}(\boldsymbol{x},t) = \int_R \boldsymbol{f}(\boldsymbol{u}(\boldsymbol{x}',t) - \boldsymbol{u}(\boldsymbol{x},t), \boldsymbol{x}' - \boldsymbol{x}) dV_{\boldsymbol{x}'}, \quad \forall \boldsymbol{x} \in R, \quad t \geq 0, \tag{3.1}$$

$\boldsymbol{x}$ = initial position [m], $\boldsymbol{u}$ = displacement [m],
$(\cdot)'$ = value at centre of $dV_{\boldsymbol{x}'}$.

This relation can be rewritten as:

$$\boldsymbol{L}_{\boldsymbol{u}}(\boldsymbol{x}) = \int_R \boldsymbol{f}(\boldsymbol{u}' - \boldsymbol{u}, \boldsymbol{x}' - \boldsymbol{x}) dV' \quad \text{on } R. \tag{3.2}$$

The peridynamic equation of motion is then given by dynamic equilibrium:

$$\rho \ddot{\boldsymbol{u}} = \boldsymbol{L}_{\boldsymbol{u}} + \boldsymbol{b} \quad \text{on} \quad R, \quad t \geq 0, \tag{3.3}$$

ρ = density [kg/m^3], $\ddot{(\cdot)}$ = second derivative in time,
$\boldsymbol{b}$ = bodyload [N/m^3].

The integral in Equation 3.2 can be expressed as integrating over the region of influence H for any point $\boldsymbol{x}$ in R. The region of influence, H, for each particle $\boldsymbol{x}$ can be defined by the horizon, δ. That is, particle $\boldsymbol{x}$ interacts with all other particles within the distance δ in R. This leads to the previously stated continuous peridynamics equation of motion:

$$\rho \ddot{\boldsymbol{u}} = \int_H \boldsymbol{f}(\boldsymbol{u}' - \boldsymbol{u}, \boldsymbol{x}' - \boldsymbol{x}) dH + \boldsymbol{b}. \tag{3.4}$$

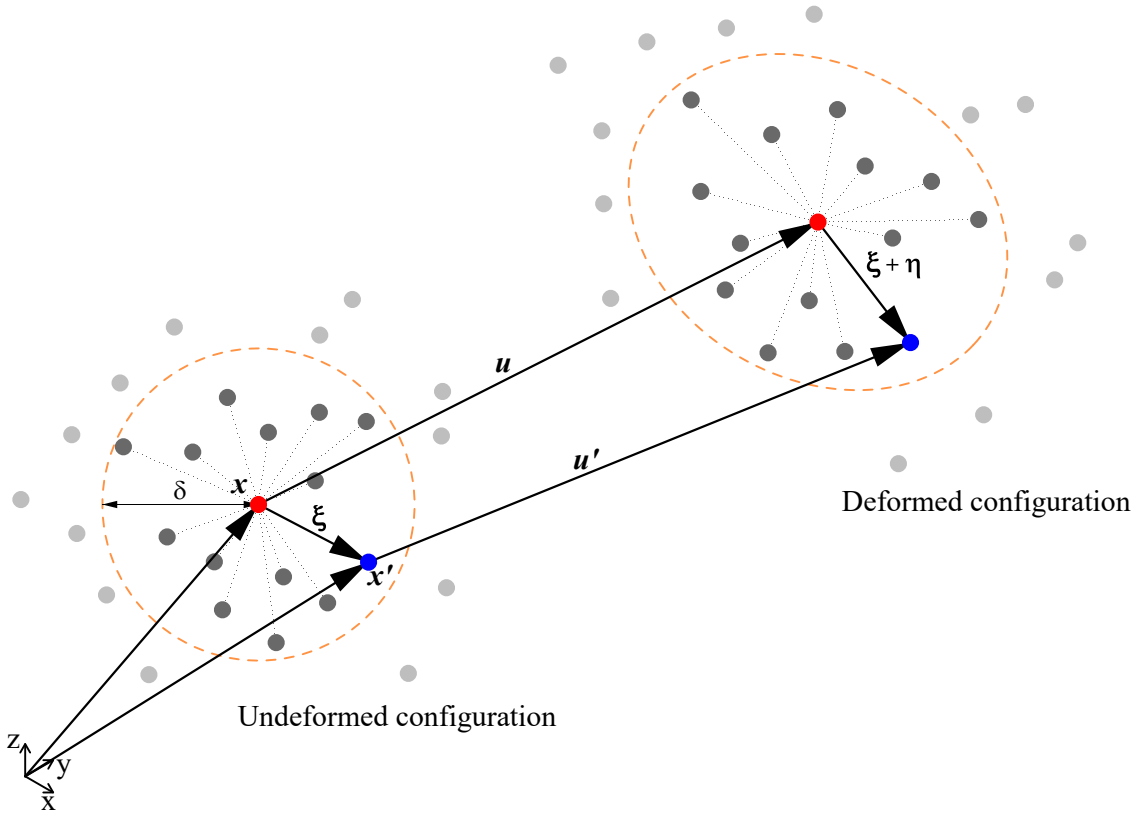


Figure 3.1: Definition of relevant vectors in the deformed and undeformed configuration.

The relative displacement vector and the relative position vector in the undeformed configuration can then be written as (see Figure 3.1):

$$\boldsymbol{\eta} = \boldsymbol{u}' - \boldsymbol{u}, \tag{3.5}$$

$$\boldsymbol{\xi} = \boldsymbol{x}' - \boldsymbol{x}, \tag{3.6}$$

$\boldsymbol{\eta}$ = relative displacement vector [m],
$\boldsymbol{\xi}$ = relative position vector in the undeformed configuration [m].

The relative position vector in the deformed configuration then becomes:

$$\boldsymbol{\xi} + \boldsymbol{\eta}. \tag{3.7}$$

The function $\boldsymbol{f}$ is called the pairwise force density function and it will contain all constitutive information needed for the model. It can be defined as the force per unit volume squared that the material point $\boldsymbol{x}'$ exerts on the material point $\boldsymbol{x}$. There are two important restrictions on $\boldsymbol{f}$, the first arises from Newton's Third Law:

$$\boldsymbol{f}(-\boldsymbol{\eta}, -\boldsymbol{\xi}) = -\boldsymbol{f}(\boldsymbol{\eta}, \boldsymbol{\xi}) \quad \forall \boldsymbol{\eta}, \boldsymbol{\xi}. \tag{3.8}$$

The second restriction comes from the conservation of angular momentum:

$$(\boldsymbol{\xi} + \boldsymbol{\eta}) \times \boldsymbol{f}(\boldsymbol{\eta}, \boldsymbol{\xi}) = 0 \quad \forall \boldsymbol{\eta}, \boldsymbol{\xi}. \tag{3.9}$$

Based on these two restrictions it can be concluded that the most general form of $\boldsymbol{f}$ is:

$$\boldsymbol{f}(\boldsymbol{\eta}, \boldsymbol{\xi}) = F(\boldsymbol{\eta}, \boldsymbol{\xi})(\boldsymbol{\xi} + \boldsymbol{\eta}), \quad \forall \boldsymbol{\eta}, \boldsymbol{\xi}, \tag{3.10}$$

where F is a scalar function that satisfies:

$$F(-\boldsymbol{\eta}, -\boldsymbol{\xi}) = F(\boldsymbol{\eta}, \boldsymbol{\xi}), \quad \forall \boldsymbol{\eta}, \boldsymbol{\xi}. \tag{3.11}$$

This is the basis for peridynamics as presented by Silling. [6]

For bond-based peridynamics the pairwise force density vector is assumed to be linearly dependent on the stretch between these material points and, if deformations caused by temperature changes are neglected, it can then be written on the form

$$\boldsymbol{f}(\boldsymbol{\eta}, \boldsymbol{\xi}) = cs(\boldsymbol{\eta}, \boldsymbol{\xi}) \frac{\boldsymbol{\xi} + \boldsymbol{\eta}}{|\boldsymbol{\xi} + \boldsymbol{\eta}|}, \tag{3.12}$$

$$c = \text{bond-constant } [\text{N/m}^6],$$

where $s(\boldsymbol{\eta}, \boldsymbol{\xi})$ is the stretch which can be interpreted as the engineering strain. It is then defined according to classical continuum theory as

$$s(\boldsymbol{\eta}, \boldsymbol{\xi}) = \frac{|\boldsymbol{\xi} + \boldsymbol{\eta}| - |\boldsymbol{\xi}|}{|\boldsymbol{\xi}|}. \tag{3.13}$$

This form of $\boldsymbol{f}$ fulfils the form presented in Equation 3.10 and 3.11. For bond-based peridynamics the Poisson's ratio is fixed to either 1/4 in 3D and in 2D under the assumption of plane strain, under the assumption of plane stress it instead becomes 1/3 in 2D. [7]

3.2 Discrete Formulation

To compute the forces numerically for a general system the region R is replaced with a finite number of particles encompassing the same region. Each particle tracks the properties of the system and together gives an approximation of the idealised system. For such a system the peridynamic equation of motion (Eq 3.4) can be rewritten as

$$\rho \ddot{\boldsymbol{u}}_i = \sum_{j \in H_i} \boldsymbol{f}(\boldsymbol{\eta}_{ij}, \boldsymbol{\xi}_{ij}) V_j + \boldsymbol{b}_i, \tag{3.14}$$

ρ = density [kg/m^3], $\quad \boldsymbol{u}_i$ = displacement of particle i [m],
V_j = volume of particle j [m^3], $\quad \boldsymbol{b}_i$ = bodyload acting on particle i [N/m^3].

$\boldsymbol{f}(\cdot,\cdot)$ = pairwise force density function, see Eq. 3.12 [N/m^6],
$\boldsymbol{\eta}_{ij} = \boldsymbol{\eta}$ (see Eq. 3.5) between particle i and j [m],
$\boldsymbol{\xi}_{ij} = \boldsymbol{\xi}$ (see Eq. 3.6) between particle i and j [m],
H_i = set of indices for all particles in the region H for particle i.

3.3 Surface Corrections

The stiffness of the bonds is calculated assuming full support from neighbouring particles, which is not the case near surfaces. One approach recommended by Le and Bobaru in [17] is the simple volume correction method, where bonds are strengthened if the particles do not have many neighbours. This is recommended not because it has the best results but because the improvements compared to the complexity of other solutions are marginal.

The bond stiffening factor proposed and outlined by Bobaru *et al.* in [18] is

$$\lambda = \frac{2V_0}{V(\boldsymbol{x}_i) + V(\boldsymbol{x}_j)}, \tag{3.15}$$

V_0 = volume of the full region of influence H [m^3],
$V(\boldsymbol{x}_i)$ = volume of material within the particle $\boldsymbol{x}_i$'s region of influence which is active [m^3].

This averages the lack of material around both particles which determines the correction factor. As the base volume, V_0, emulates a fully embedded particle we also know that this factor will always be greater than 1.

3.4 Quasi-Static Solution

The peridynamic equation of motion (Eq. 3.14) can be used to increment the system in time. Such a time integration is useful for things like impact problems, but our aim is a stress field for the system at rest. The system will be at rest when no particle experience acceleration, i.e. Equation 3.14 equals zero. Hence the whole system can be seen as a system of equations to solve

$$\sum_{j \in H_i} \boldsymbol{f}(\boldsymbol{\eta}_{ij}, \boldsymbol{\xi}_{ij}) V_j + \boldsymbol{b}_i = \boldsymbol{0} \quad \forall i. \tag{3.16}$$

This perspective means that a time step Δt is simply an integration step. Furthermore, as only Equation 3.16 needs to hold, values such as the density, ρ, will not affect the final solution. Concepts from the physical process can still be useful to solve this system. However, as a damped system is known to reach equilibrium, we know that values not contained in Equation 3.16 can be chosen for quicker convergence.

3.5 Density

In this stepping scheme, the concepts of time step and density both describe an integration step, as can be seen by considering the relation between impulse ΔtF and the difference in momentum, mv,

$$(mv_1 - mv_0) = \Delta tF, \tag{3.17}$$

$$v_1 - v_0 = \frac{\Delta t}{m}F, \tag{3.18}$$

$m =$ mass [kg], $\Delta t =$ duration [s],
$v =$ velocity [m/s], $F =$ force [N].

Hence the time step Δt will be set to 1 and a separate density will be set for each degree of freedom in the system according to the formulation described by Madenci *et al.* in [7] which approximates the largest safe step length as

$$\lambda_{ii} \geq \frac{1}{4}\sum_j |\mathbf{K}_{ij}|, \tag{3.19}$$

where $\mathbf{K}_{ij}$ is is the stiffness between particle i and j calculated according to Appendix A.1 to produce the direction dependent density λ_{ii}.

This density is the inverse of a step length and the manner it is connected to mass is through inertia and wholly disconnected from gravitational effects which would be applied through the bodyload $\boldsymbol{b}$.

3.6 Velocity Verlet

Velocity verlet is a commonly used method for the integration of trajectories of particles bound by Newton's equations of motion. As such the positions infer what the current acceleration is, which in turn updates the velocities and positions as described in [19].

The new displacement is determined through a second-order Taylor expansion centred at $t = 0$,

$$u^{n+1} = u^n + \dot{u}^n\Delta t + \frac{\ddot{u}^n\Delta t^2}{2}. \tag{3.20}$$

The new velocity is found with a Crank–Nicolson step (see Equation 8.1), which avoids the commonly implicit nature as the acceleration is not computed from the velocity,

$$\dot{u}^{n+1} = \dot{u}^n + \frac{\ddot{u}^n + \ddot{u}^{n+1}}{2}\Delta t. \tag{3.21}$$

3.7 Damping

Since the aim is to find static equilibrium, energy needs to be removed from the system. This will be done by damping and as the damping has no effect once equilibrium is achieved one can pick damping coefficients which facilitates a faster convergence. A method to do this is described by Madenci *et al.* in [7] based around the lowest frequency of the system, where the damping coefficient c is described as

$$c = 2\sqrt{\frac{\sum_i \boldsymbol{u}_i^2 K_{ii}}{\sum_i \boldsymbol{u}_i^2}}, \tag{3.22}$$

$$K_{ii} = -\frac{\boldsymbol{f}_i^n - \boldsymbol{f}_i^{n-1}}{\lambda_{ii}\dot{\boldsymbol{u}}_i^{n-1/2}}, \tag{3.23}$$

c = damping coefficient [-],
$\boldsymbol{u_i}$ = displacement for degree of freedom i [m],
$\boldsymbol{f}_i$ = resulting force density for degree of freedom i [N/m^3],
λ = directional density from Eq. 3.19 [kg/m^3],

where some small value is substituted for $\dot{u}_i^{n-1/2}$ in case of zero.

The local stiffness in Equation 3.23 is an explicit derivative, where the difference in force between steps is divided by the change in displacement during the step (where we remember that the time step is 1). λ_{ii} is also included for numerical stability reasons as explained in [7].

3.8 Stress Tensor

The stress over some surface is the total force passing through it over the area of the surface. Peridynamics is however non-local, and any surface has many bonds passing through it, which makes determining it non-trivial. The paper by Fallah *et al.* [20] shows that by assuming a smooth change of the displacement field the stress tensor can be computed directly from a dyadic product as

$$\boldsymbol{\sigma}_i = \frac{1}{2}\sum_j V_j \boldsymbol{f}(\boldsymbol{\eta}_j, \boldsymbol{\xi}_j) \otimes \boldsymbol{\xi}_j. \tag{3.24}$$

V_j = volume of particle j [m^3],
$\boldsymbol{f}(\cdot,\cdot)$ = pairwise force density vector (see Eq. 3.12) [N/m^6],
$\boldsymbol{\eta}_j$ = relative displacement vector (see Eq. 3.5) [m],
$\boldsymbol{\xi}_j$ = relative position vector (see Eq. 3.6) [m].

This expression derives the stress tensor from the peridynamic forces without the strain tensor and as such makes no assumptions on if the material is linearly isotropic. The strain tensor can still be computed as outlined in Appendix A.2, but this work has no specific use for that approach.

4

Principal Stress Based Design

As stated in [4] the strut-and-tie geometry should be derived from the principal stress field to reduce the need for plastic deformations before reaching the calculated yield point. Therefore, the method presented in [13] is studied further to be able to go from a principal stress field to a truss that is suitable for STM.

4.1 Principal Stress Field

The principal stresses are acquired for each point by solving the eigenvalue problem

$$\boldsymbol{\sigma}\hat{\boldsymbol{n}}_i = \lambda_i \hat{\boldsymbol{n}}_i. \tag{4.1}$$

Where $\boldsymbol{\sigma}$ is the Cauchy stress tensor and $\hat{\mathbf{n}}_i$ the principal direction for the principal stress λ_i. For each point three orthogonal principal directions will be acquired, a positive principal stress indicates tension and negative compression.

As described by Kwok *et al.* in [13] the principal stress field can be divided into five different ideal regions as seen in Figure 4.1. In R regions the material is in a state of uniaxial stress (either in compression or tension), therefore these regions are not suitable for nodes in the truss. Regions with pure compression or tension in all directions are denoted S regions. S regions are suitable for nodes since they have a multiaxial stress state, however, all of the connecting members should either be in compression or tension depending on the sign of the principal stress in the region. In T regions compression and tension, stresses meet, and as stated before they are always orthogonal to each other. It is these T regions that are suitable for further development of the truss and topology growth [13]. This classification of the principal stress field works in 2D as well as in 3D.

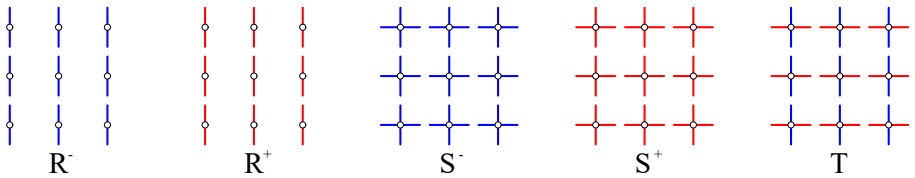


Figure 4.1: The different ideal regions in a principal stress field. Blue lines indicate compression and red tension. In R regions there are uniaxial stress, in S regions there are pure tension or compression in all directions and T regions there are both compression and tension.

4.2 Initial Truss Topology

The idea with the initial truss topology is to create a topology that connects the load to the support as directly as possible while still making sure that all members are within the design domain and somewhat follow the principal stress field. To do this, the principal stress field needs to be calculated for the domain as well as the effective von Mises stress. Local maximum points of the von Mises stress also need to be identified for further processing. The method to create the initial truss topology is divided into three different phases, the load phase, the local maximum point phase and the support phase. [13]

4.2.1 Load Phase (I)

In the load phase, principal stress curves are traced from the points of loading in all directions. If a curve connects to a support region that curve is kept and called a I curve. A line is drawn between its start point at the load and its endpoint at the support. If a curve instead reaches one of the local maximum points of the von Mises stress this point becomes a node in the truss. The curve is kept as a I curve and a line is drawn from its start point to the new node. All other curves traced from the points of loading are discarded. [13]

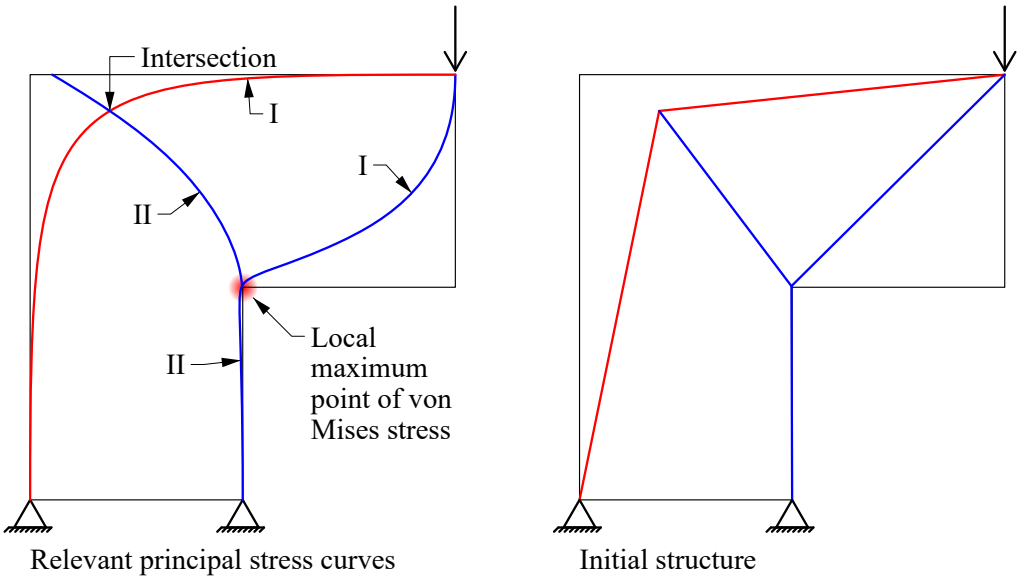


Figure 4.2: Generation of the initial structure for an L shaped cantilever with a point load. Curves indicated with I was generated in the first phase and the ones with II in the second.

4.2.2 Local Maximum Point Phase (II)

Phase II is only relevant if one of the I curves connects to a local maximum point, otherwise, it can be skipped. If a local maximum point is reached, new principal stress curves are traced from this node. The curves that reach a support region are

kept as II curves and lines are drawn from its endpoints. If instead a curve intersects with one of the previous I curves a node is created at this position. The line from the previous step is divided to include this node. The curve from the current step is kept as a II curve and a line is drawn from its endpoints. If any curve will create a duplicate line it is disregarded as well as other curves that do not fulfil any of the criteria above. In Figure 4.2 an example is shown for an L shaped cantilever with a point load. The curve in compression generated in phase I connects to the local maximum point and therefore phase II is activated. The new curves from the local maximum point connect both to the right support and the curve in tension from phase I, because of this that line is subdivided at the intersection. [13]

4.2.3 Support Phase (III)

The support phase is conducted when the loading and support positions are collinear with each other. Curves are traced from the part of the support regions that are under the highest stress in descending order. If the traced curve intersects with any I curve it is kept and a line is drawn from its endpoints. New lines are traced until all supports are connected to the structure unless an entire support region is unstressed. With the completion of this step, the initial truss structure is complete. In Figure 4.3 the method is applied on a structure with a point load. The curve generated in phase I does not connect to a local maximum point and therefore phase II is not conducted. The principal stress curve in compression is generated in phase III and intersects with the phase I curve in the middle of the structure. [13]

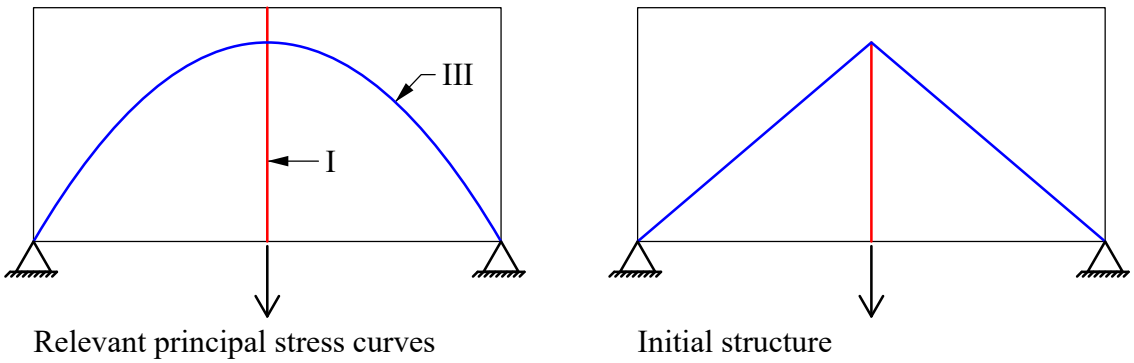


Figure 4.3: Generation of the initial truss for a structure with a point load. Curves indicated with I were generated in the first phase and the ones with III in the third.

4.3 Evolution of Topology

The initial truss generated according to the method mentioned above can be improved with the goal to reduce the strain energy of the system. This is done by minimising the deviation of the truss members from the principal stress curves from which it originates. This can be interpreted as minimising the area between the

truss members and the principal stress curve by creating new nodes, see Figure 4.4. At these nodes, new principal stress curves should be traced and new members created. The new nodes should only be created at T regions in the principal stress field, this is to reach equilibrium between the compressive and tensile forces. This is an iterative procedure and for each iteration, a new node is created at the position where the error between the truss member and the principal stress curve can be reduced the most. This procedure is repeated until the strain energy stabilises or until a certain number of subdivisions is done, specified by the user. [13]

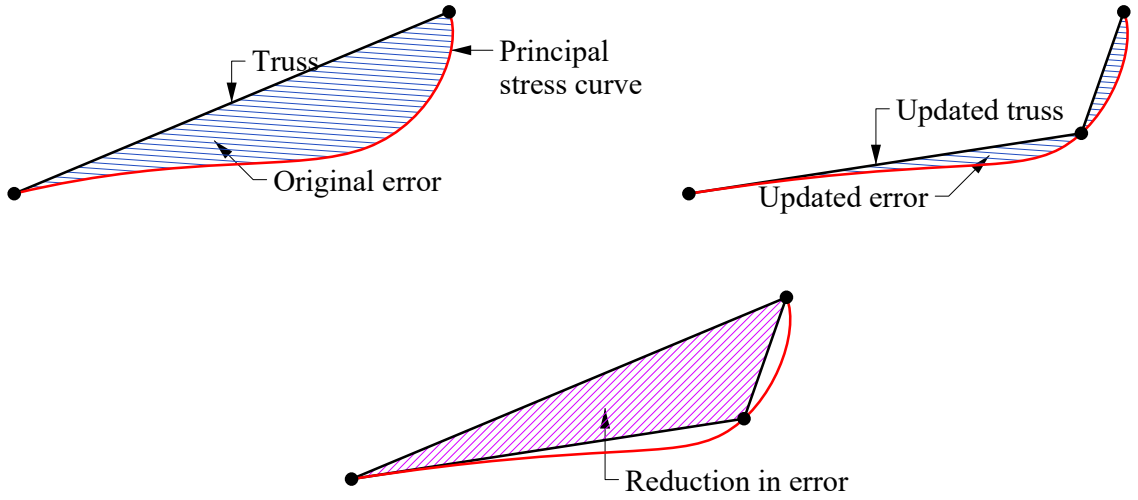


Figure 4.4: Reduction in the deviation between the truss member and principal stress curve by creating a new node.

4.4 Limitations

The described method developed by Kwok *et al.* in [13] have certain limitations that need to be addressed. For the generation of truss topologies only point loads are considered. Therefore the method needs to be developed to handle distributed loads. Only point supports are considered as well and more precisely only pinned supports are studied, roller supports are not investigated. The presented method is designed to work in 2D but with some adaptations, it could also be extended to 3D. In 3D the load paths and the traced principal stress curves become more complicated and there is also harder to distinguish intersections since curves may pass very close to each other but without actually intersecting each other. These issues need to be addressed for the method to function in 3D as well. [13]

5

Optimisation

Solving optimisation problems is sometimes possible analytically, but in general and in our case it must be solved by numerical methods. Those used in this thesis are described in this chapter.

5.1 Steepest Descent

The steepest descent method is an iterative algorithm for unconstrained optimisation problems and works by assuming that the function can be locally approximated by its gradient and as such needs a function with a defined gradient and more importantly one that is descriptive of the function. The gradient of a function describes the expected change of function value given a change of the corresponding variable. The negative gradient is the direction where this is minimised.

$$\arg\min_{\mathbf{p}} \mathbf{p}^T \nabla f(\mathbf{x}) = -\nabla f(\mathbf{x}) \tag{5.1}$$

if the length of $\mathbf{p}$ is constrained to the length of $\nabla f(\mathbf{x})$ [21].

Steepest descent works by picking an initial point $\mathbf{x}_0$ and then applying

$$\mathbf{x_{k+1}} = \mathbf{x}_k - \alpha \nabla f(\mathbf{x}_k), \tag{5.2}$$

until some end condition is met. The end condition can be by comparing the change of the point, function value or length of the gradient. The step length factor α can be chosen in some different manners such as fixed, line search or Armijo step rule.

5.2 Armijo Step Rule

The Armijo step rule is a method for picking the step length during the steepest descent method and is described by Andréasson *et al.* in [21]. The principle is based on that our knowledge of the system is its current value and gradient. With this, we can produce an expected value for any step length which we can compare to the actual value for that step length. This evaluation is done to determine if our step covered a part of the function which can be approximated linearly and avoids over-stepping into other local minimums. This also mostly maintains that there should be a connection between the gradient and the function value, which is useful for the steepest descent method which works on that assumption. If a step length α_k

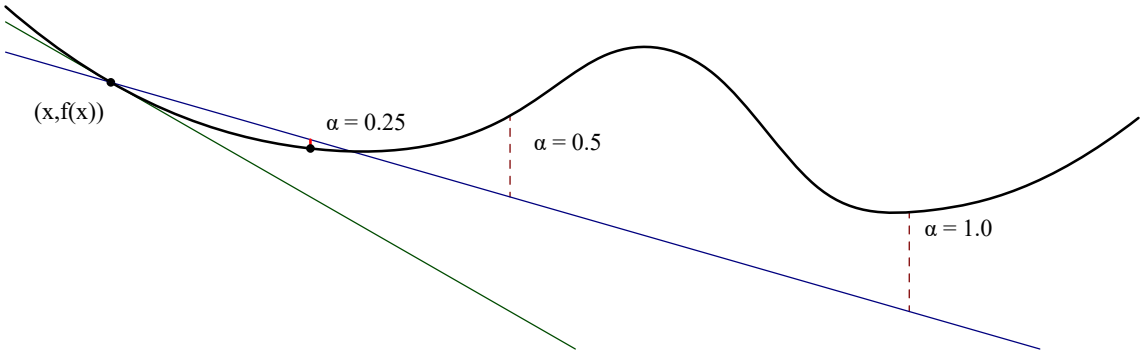


Figure 5.1: Armijo step rule for a 1-dimensional case where the green line is the gradient and the blue line is the reduced expectation. Trails are made for decreasing values of α where the function value is smaller than the expectation at 0.25.

does not satisfy the condition in Equation 5.3 it is halved until it does, and to allow it to find a suitable step length it can also increase by some factor between iterations.

Since the first-order approximation of a convex function is defined to be less or equal to the function, and any local minima for a smooth function is locally convex, the expected decrease in value is reduced by a factor μ, as illustrated in Figure 5.1.

$$\mathbf{f}(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \leq \mathbf{f}(\mathbf{x}_k) + \mu \alpha_k \mathbf{p}_k^T \nabla \mathbf{f}(\mathbf{x}_k), \tag{5.3}$$

where $\mathbf{p}_k$ is the step direction and α_k is the factor controlling the step length.

5.3 Gradient Projection

For constrained problems, one might exit the domain and one way to return to the feasible space is by gradient projection. As described in [21] a point is stationary if $\mathbf{y} = \mathbf{x}$ given Equation 5.4, where stationary has the same connotations as zero gradient in unconstrained optimisation.

$$\mathbf{y} = \text{Proj}_X[\mathbf{x} - \nabla f(\mathbf{x})], \tag{5.4}$$

where $\text{Proj}_X[\mathbf{w}]$ is the solution to the problem

$$\min_{\mathbf{x} \in X} ||\mathbf{x} - \mathbf{w}||. \tag{5.5}$$

Thus the base principle is that if the point exits the feasible space it is projected to the closest point inside the domain, see Figure 5.2. If the constraint is continuous and we are still sufficiently close, one can project along with the constraint's normal, where the normal of a constraint can be found by taking its gradient if it is specified as

$$g(\mathbf{x}) \leq 0, \tag{5.6a}$$

$$h(\mathbf{x}) = 0. \tag{5.6b}$$

The true normal of the constraint is the gradient of the projected point, but an approximation would be to use the gradient of the point where one projects from.

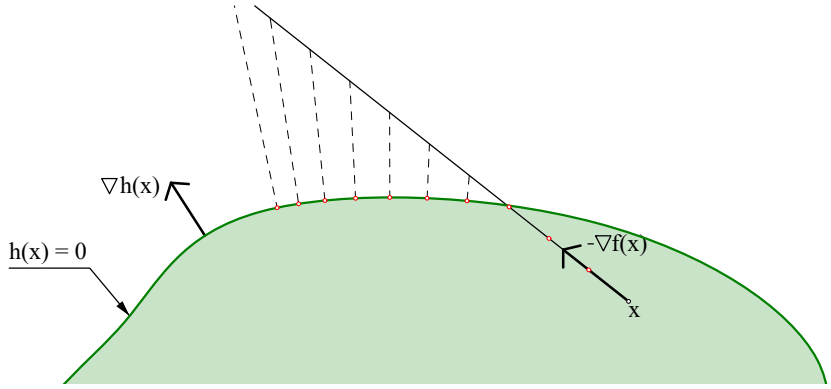


Figure 5.2: The potential next red step points in $\mathbb{R}^2$ from a point x along $-\nabla f(x)$ projected back to the feasible space along the constraint's gradient.

5.4 Exterior Penalty Method

As constrained optimisation has certain challenges, one can modify the problem statement into an unconstrained one with the exterior penalty method as described in [21]. As such, given a model on the form

$$\begin{aligned} \min_x \quad & f(x), \\ \text{s.t.} \quad & x \leq g_i(x) \quad \forall i, \end{aligned}$$

one can transform it to the equivalent unconstrained model where an indicator function has been added as a penalty to the objective function.

$$\min_x \quad f(x) + \sum_i \Phi_i(x),$$

$$\Phi_i(x) = \begin{cases} 0, & \text{if } x \leq g_i(x), \\ \infty, & \text{else.} \end{cases}$$

Optimising over discontinuous functions involving infinity is not much better but allows for the next step of this problem where we approach the indicator function $\Phi_i(x)$ smoothly using a limit,

$$\lim_{\nu \to \infty} \min_x \quad f(x) + \nu \sum_i \Phi_i',$$

$$\Phi_i'(x) = (\max\{0, g_i(x)\})^2.$$

The different stages are illustrated in Figure 5.3. The relaxed indicator function $\Phi_i'(x)$ is intentionally picked such that the value and derivative at zero are zero, which lets it connects to the rest of the function smoothly. Furthermore, it has a smoothly increasing derivative such that when added to some other function there will be a minimum with zero gradient.

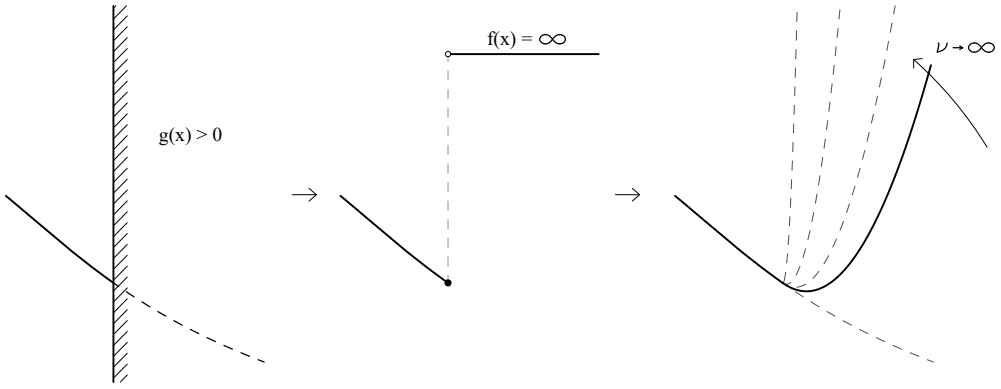


Figure 5.3: From the left: the constrained problem, the problem with an indicator function and the problem with a relaxed indicator function.

When attempting to approach the solution at the limit one begins with a small value for ν to aid in convergence as it keeps the function from changing the value too quickly. Once the solution is supported by the penalty function, the factor ν can be increased successively to approach the limit. This maintains numerical stability as the steps will be small when initialising it at the old solution which will be close to the new solution.

So in essence; find a smooth function which is negative in the feasible space and positive outside of it, then find the gradient of that function squared such that we can use it in the steepest descent method. Then solve the problem successively for increasing values of ν.

6
Parallelisation

The parallel nature of certain problems can be utilised by dividing the amount of work into different places and achieve higher effective execution speeds than what could be achieved otherwise. Peridynamics is a computational method which lends itself to parallelism since the force of each particle is solved individually. It is however not a trivially parallel problem as there is interaction between the iterations. This can be thought of in terms of expected performance gains in that the new execution speed will be the time to compute the problem divided by the number of parallel executions plus the synchronisation time. The synchronisation includes both the setup and collection of different threads and the communication between these threads as the executions run. [22]

This chapter will largely treat how massive parallelisation has been achieved on Graphical Processing Units (GPUs) and limitations on how it functions.

6.1 OpenCL

GPUs came about as many graphical processes are similar and highly parallel and were at first dedicated hardware. These did not have the possibility for general computation and those that wanted to use it at such had to reformulate the problem as a graphical one. Nvidia then launched CUDA and GPUs which could run it which was a general computation framework for GPUs which was vendor-specific. OpenCL was then born as an attempt to make parallel software development platform agnostic. [23]

OpenCL is a language standard for cross-platform general computation on accelerator devices such as GPUs. It allows for each device to compile the code on its own, which is important as GPUs are diverse, highly optimised and evolving. Furthermore, the architecture which allows it to receive data, distribute it to its cores, schedule it and return it will differ from device to device. OpenCL is also not restricted to computations for GPUs but also functions for CPUs and embedded devices.

The concept of general computation on a GPU is that the main program defines and launches the kernel, where the kernel is a function which runs on the device. The kernel will be run once for every work item assigned and can synchronise with other threads in the same workgroup. For example, matrix multiplication can be dealt with like this where each output component is assigned a work item. [24]

6.2 Memory Model

The GPU is commonly separated from the rest of the computer and has its own dedicated memory from which it has faster access. The host can communicate with that global memory but transfer speeds are comparatively low and are thus preferably only done in batches to assign input data and retrieve output data. As a result, some computations which would be quicker to do on the host should be done on the device to cut out transfer delays.

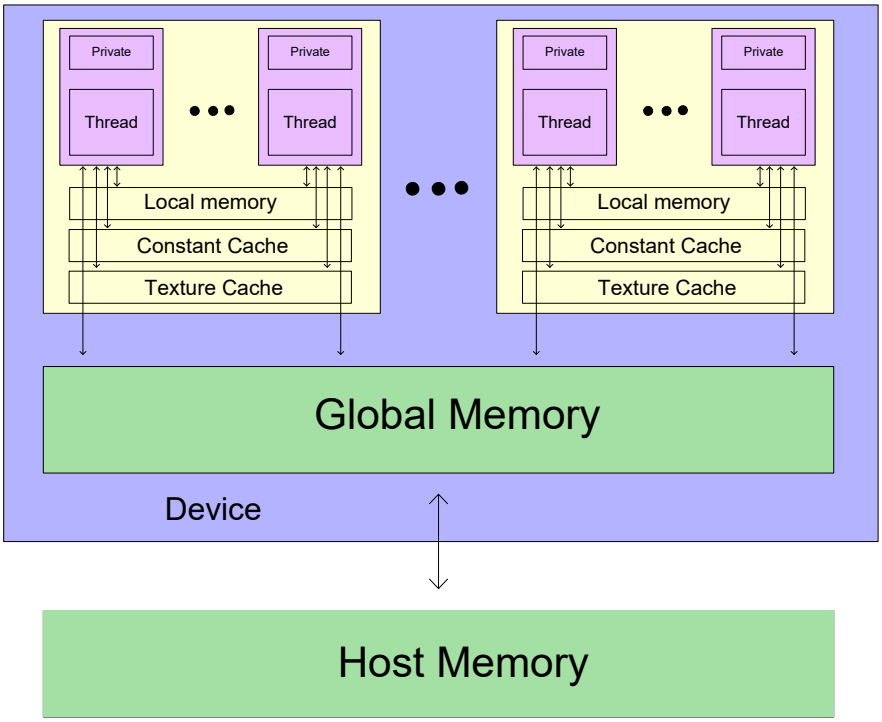


Figure 6.1: The memory model as described by OpenCL with some common additional abstracted hardware details. Caches are hardware specific but very common for GPUs.

Within the GPU there are different levels of memory which will have different access speeds, see Figure 6.1. The global memory is the largest and slowest memory which is accessible from all threads and the only memory which the host can communicate with. The local memory is a memory which is attached to each group of threads. These work groups contain the largest number of items which can synchronise during a kernel's execution and the local memory is a faster memory which is specific per work group. The final explicitly handled memory is the private memory which is specific per thread. Memory access for each step outwards from the thread is in general an order of magnitude slower which is why conscious memory management easily can become important. [24]

One important cache to talk about is the texture cache as it is used for image data and automatically manages data locality. That is during memory access for image structures the texture cache will cache both the fetched data and spatially adjacent values. This cached data will then be available for the current thread and adjacent threads where the threads can also be structured in a grid.

These memory access times are often obfuscated however due to a few different reasons. Threads do not run completely in parallel but are interlaced such that operations, like memory access, can be performed asynchronously before returning to the thread. Threads on the same core also share the same cache which is temporarily stored, fast access, global memory. This means that multiple memory calls might coalesce into one. Arithmetic operations may also run as the access is proceeding. [25]

6.3 Divergence

On a GPU the kernels are run in warps, small collections of threads, where each warp will execute the same instructions for all the threads in line with the Single Instruction Multiple Data (SIMD) framework. On a hardware level, this enables higher performance but there are some considerations for the software. Divergence is when different threads in a warp take different paths due to things like if- and else-statements. The GPU can solve this by running all the threads once for each path required. This means that such branches can be doubly bad for performance as they may both fail in branch prediction and need to run it multiple times. One can treat this with the concept of branch-less programming. This is a concept which is also relevant for CPU computations for other reasons and works by assuming that the cost of an if-statement might be higher than performing the computation and disregarding the result, which in some cases has the same result as not entering the if-statement.

This became relevant for peridynamics as each particle sums over its active neighbours, but not all neighbours will be active and neighbouring particles will have a different set of active neighbours, meaning that one might end up with as many divergent paths as there are threads in a warp (commonly 32). Instead, the force contribution of non-active particles is multiplied by zero to ensure that the result is the same.

If-statements should however not be disregarded for something one can expect that for most warps all threads will enter the same branch, meaning that those warps can exit sooner and free resources for other threads. [26]

Part II

Development and Implementation

7
Peridynamics

There exist several different kinds of peridynamics, but the kind implemented in this thesis is bond-based peridynamics, which was chosen due to its simplicity and speed. It has fewer possibilities for accurate modelling, but as the aim of the peridynamics model is to inform of a linear elastic stress state, it need only to be indicative, not exact. The following sections will detail how it has been implemented and concepts used to increase computational speed, accuracy at low resolution and other measures to evaluate material response.

The following external packages are used in the application:

- Grasshopper Async Component - Allows grasshopper components to compute in the background
- Silk.NET - C# wrapper for GPU computations in OpenCL

7.1 Voxels

Peridynamics simulations are relatively computationally expensive and to be able to compute them efficiently the data structure concept of voxels has been chosen. Voxels are pixels in 3D such that for some cuboid region every voxel is assigned values and can be accessed through an index.

The use of voxels affects things in some ways:

1. Since the voxels themselves already define a grid of particles, converting a mesh to a voxel representation is straightforward.
2. The voxels also implicitly define their position by an index, and as a result, can also define their neighbours as an offset to their index. This list of offsets is the same for all voxels.
3. Voxels need to define all values in some cuboid region, which can be a downside for slender or sparse structures, but the aim is to deal with 3D effects in material bulk where this inefficiency is negligible.
4. A uniform placement of particles means that the initial displacements $\boldsymbol{\xi}$ for all neighbourhoods are the same.
5. It allows the peridynamics simulation steps to more closely resemble image processing algorithms, and can thus leverage the considerable efforts placed into that field.

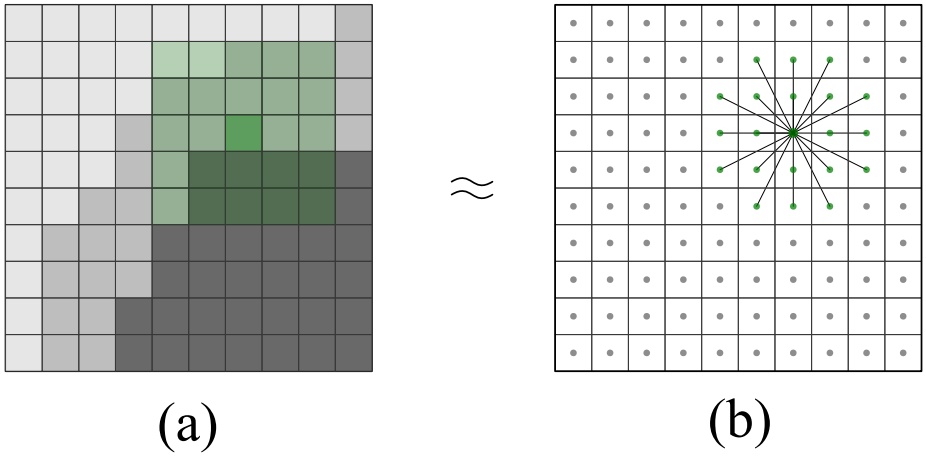


Figure 7.1: (a) Shows how a filter is applied on the pixel in green from its neighbours. (b) Shows how the force is computed for the particle in green from its neighbours.

Having identical initial displacements $\boldsymbol{\xi}$ means that these values can be pre-computed and since they are the same for each neighbourhood only a negligible amount of extra memory is needed. This is relevant as most of the computational effort is spent evaluating Equation 3.12 and 3.13, both heavy in use of $\boldsymbol{\xi}$ and the length of $\boldsymbol{\xi}$.

The peridynamic computation is similar to both applying filters to images and computing convolutions of images for neural networks, see Figure 7.1. It furthermore benefits from the spatial locality optimisations made on GPUs which have been made to enable image sub-sampling and distortion. All of these are operations where every data point will have the use of spatially adjacent data points. As pointed out in Section 6.2, managing the memory such that memory access is reduced can be dimensional and a voxel representation is an image structure which can leverage pre-existing solutions.

7.1.1 Mesh to Voxels

Defining voxels explicitly would be difficult, hence a conversion from a closed mesh representation to voxels is added as shown in Figure 7.2. Broadly this is done by marking all cells the mesh's faces intersect, this creates disjoint regions delimiting inside and outside the mesh which can be filled with the desired values.

The first step is to fill cells that the mesh intersects, which is done by spanning a grid of points over each mesh face. The grid has a spacing of $\Delta/\sqrt{2}$ to ensure that no cell is skipped over. The voxel structure implicitly defines the position of its values by indices, but this also means that one can do the reverse and acquire an index from a point. Using such a point-to-index conversion the correct voxel values are filled.

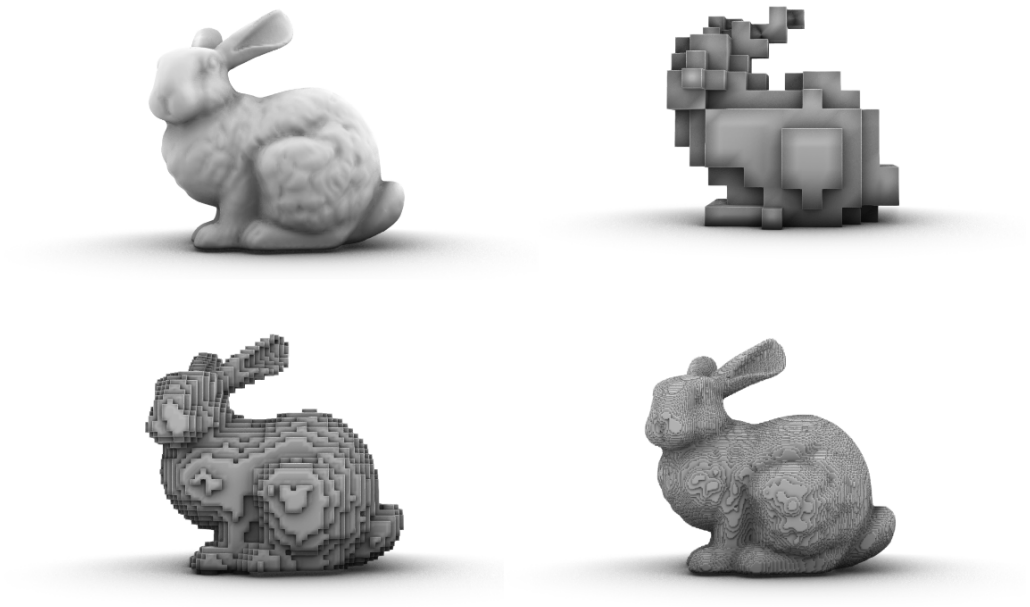


Figure 7.2: The Stanford bunny converted to a voxel representation at different resolutions with the original mesh in the upper left corner.

Once all the boundary cells are marked we know that there exist disjoint regions and each of these will be marked as inside or outside. To do so, find the first unmarked cell and check if its location is inside the mesh. Then a flood fill algorithm can be used to efficiently mark all values in that disjoint region to the same value. The version of flood fill used here is span fill as described in [27]. This is then repeated until there are no unmarked cells.

Finally to ensure a smoother voxel representation all boundary cell locations are checked for mesh inclusion.

These steps let one fill large numbers of voxel values with few mesh inclusion tests, which is good as that is a relatively complex and heavy operation. The other parts, spanning points on faces, point-to-index and flood fill, are on the other hand simple and quick. The bulk of the inclusion tests is done during the refinement step which is optional if one is willing to have a rougher model. Furthermore one could implement this method without inclusion tests if one was willing to assume that only regions connecting to the outer edge of the voxels are outside, i.e. no interior cavities.

7.2 Boundary Conditions for Peridynamics

As peridynamics is a model which relates particles to particles, applying boundary conditions based on surfaces is not trivial. One must also be aware of the non-local nature of peridynamics where the field should be smooth over multiple particles for valid results.

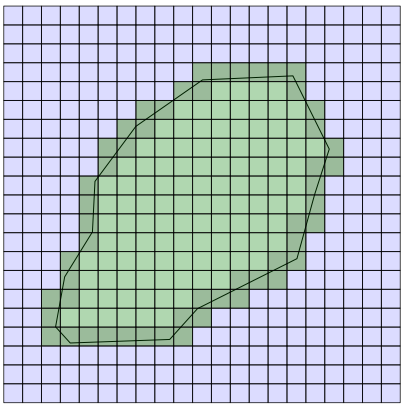


Figure 7.3: A 2D representation of how cells on the meshes border are marked to form disjoint regions which can be filled.

7.2.1 Implicit Dirichlet

The Dirichlet boundary condition prescribes the primary unknown field variable on the boundary, where the primary unknown, in this case, is the displacement

$$\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{x}) \quad \text{on } \partial\Omega. \tag{7.1}$$

Hence to facilitate a Dirichlet boundary condition the aim is to define an implicit surface where no displacements are made during the simulation. This is done by considering the bonds particles have with each other as springs. These springs have a stiffness but by setting any part of the spring past the surface to have infinite stiffness, they will implicitly attach to a stiff surface, see Figure 7.4.

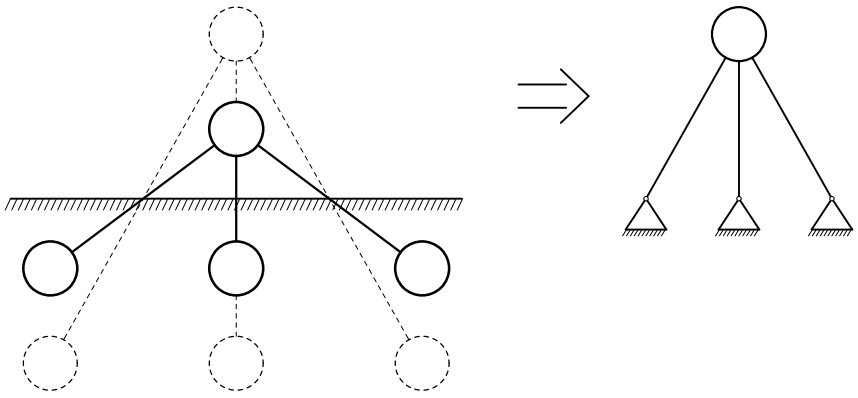


Figure 7.4: Dirichlet prescribes some displacement at a surface and this illustrates a transformation between a consistent model where each side of the boundary condition displaces to a model with only one side and shorter bonds.

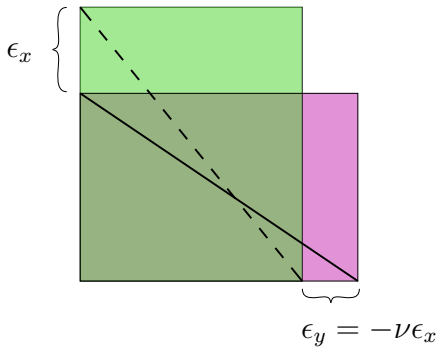


Figure 7.5: Poisson's ratio describes the relation between strains in different directions. For boundaries locked in different directions, this can be used to set the displacement at the border for free directions.

It is cumbersome to keep track of a large number of bonds with custom bond stiffness, therefore some simplifications can be made by using springs. Namely, each bond's first order stiffness approximation can be computed as in Appendix A.1 and as all springs act independently they can be considered parallel, meaning that their spring constants can be added into a single representative matrix. This matrix gives the first order stiffness approximation of the influence of all fictive particles beyond the surface. This can then be added to the peridynamics equation of motion (Eq. 3.14) to account for these fictive particles.

$$\rho\ddot{\boldsymbol{u}}_i = \sum_{j\in H_i} \boldsymbol{f}(\boldsymbol{\eta}_{ij}, \boldsymbol{\xi}_{ij})V_j + K_i\boldsymbol{u}_i + \boldsymbol{b}_i \tag{7.2}$$

ρ = density [kg/m^3], $\quad$ $\boldsymbol{u}_i$ = displacement of particle i [m],
V_j = volume of particle j [m^3], $\quad$ $\boldsymbol{b}_i$ = bodyload acting on particle i [N/m^3],

$\boldsymbol{f}(\cdot,\cdot)$ = pairwise force density function, see Eq. 3.12 [N/m^6],
$\boldsymbol{\eta}_{ij} = \boldsymbol{\eta}$ (see Eq. 3.5) between particle i and j [m],
$\boldsymbol{\xi}_{ij} = \boldsymbol{\xi}$ (see Eq. 3.6) between particle i and j [m],
H_i = set of indices for all particles in the region H for particle i,
K_i = local stiffness matrix representing the influence of the dirichlet boundary conditions [N/m^4].

Furthermore, it can be manipulated to only lock in certain directions, i.e. enabling rolling supports. To do so the transverse contraction needs to be accounted for with the Poisson's ratio, ν, see Figure 7.5. If it for example is locked in direction x and free to move in directions y and z the spring stiffness matrix K is

$$K = \begin{bmatrix} k_x & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad k_x = cv\frac{\boldsymbol{\xi}_x^2 - \nu\boldsymbol{\xi}_x\boldsymbol{\xi}_y - \nu\boldsymbol{\xi}_x\boldsymbol{\xi}_z}{|\boldsymbol{\xi}|^3}. \tag{7.3}$$

If the point instead is locked in directions x and y, and not in z the spring stiffness matrix K can be calculated as

$$K = \begin{bmatrix} k_x & k_{xy} & 0 \\ k_{xy} & k_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{7.4}$$

$$k_x = cv\frac{\boldsymbol{\xi}_x^2 - \nu\boldsymbol{\xi}_x\boldsymbol{\xi}_z}{|\boldsymbol{\xi}|^3}, \quad k_y = cv\frac{\boldsymbol{\xi}_y^2 - \nu\boldsymbol{\xi}_y\boldsymbol{\xi}_z}{|\boldsymbol{\xi}|^3}, \tag{7.5}$$

$$k_{xy} = cv\frac{\boldsymbol{\xi}_x\boldsymbol{\xi}_y - \nu\boldsymbol{\xi}_z^2}{|\boldsymbol{\xi}|^3}. \tag{7.6}$$

The matrix's zero elements are such that displacement in those directions does not induce force and that displacement does not induce force in those directions. i.e. in Equation 7.4 displacing in the z direction should not induce any force and it should never result in any force in the z direction since that is the free direction.

This matrix formulation and spring perspective also means that the stiffness can be modified to have spring supports (Robin condition) by using the formula for springs in a series,

$$k = \left(\frac{1}{k_1} + \frac{1}{k_2}\right)^{-1}. \tag{7.7}$$

Finally one can also apply a prescribed displacement other than zero by adding a body load $\boldsymbol{b}$ which cancels out the spring at the desired displacement $\boldsymbol{d}$ since the system is linear,

$$\boldsymbol{b} = -K\boldsymbol{d}. \tag{7.8}$$

7.2.2 Implicit Neumann

The Neumann boundary condition prescribes the spatial derivative along the normal for the primary field variable at a boundary

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} = \boldsymbol{f}(\boldsymbol{x}) \quad \text{on } \partial\Omega, \tag{7.9}$$

$\boldsymbol{u}$ = displacement vector [m], $\boldsymbol{f}(\cdot)$ = prescribed vector value at $\boldsymbol{x}$ [-],
$\boldsymbol{n}$ = normal vector of the boundary $\partial\Omega$ [-].

The prescribed value $\boldsymbol{f}(\cdot)$ is a strain, but as loads are more common to work with the input will be related to loads. Peridynamics does not have boundaries, instead, all bonds passing through the implicit surface will be assumed to be in a strain field aligned to the desired load. By then knowing the strain in those bonds, a bodyload can be computed and applied to the particles near the border directly by using the stiffness matrix in Appendix A.1.

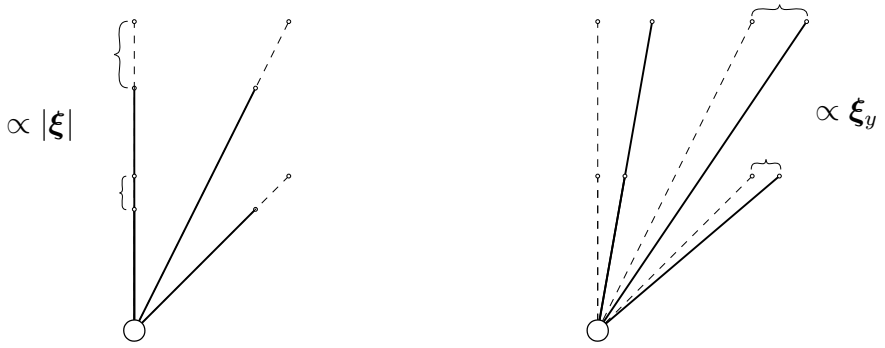


Figure 7.6: Relative displacement for uniform strain. Isotropic expansion is directly proportional to the initial distance, and simple shear is proportional to the projected distance.

The applied strain field is set to have its magnitude increase along the normal of the surface and the strain vectors in the field aligned to the applied load. Using this strain field and local stiffness matrix produces a relative load

$$\frac{\boldsymbol{\xi}_i \boldsymbol{\xi}_j}{|\boldsymbol{\xi}|^3}(\boldsymbol{\xi} \cdot \boldsymbol{n})\boldsymbol{l}, \tag{7.10}$$

$\boldsymbol{n}$ = normal vector of surface [-], $\boldsymbol{l}$ = load vector direction [-].

And once this relative load has been computed for all particles they can be scaled such that the total load matches whatever was prescribed.

This method is made such that the bounds of the system need not be modified and such that there exists a smooth transition of load into the system. The smooth transition of the load is important as peridynamics is non-local and assumes that the displacement field within each region of influence is roughly linear or representative of the system on a micro-scale.

7.3 End Condition

The simulation aims to solve the system of equations in Eq. 3.16. This is done in an iterative manner where each step should move the solution closer to equilibrium. Hence even if this was not done numerically a true solution would not be reached. Therefore an end condition should be set such that the simulation terminates when the solution is close enough. This is done by considering the residual of the system of equations (Eq. 3.16), where the residual is the absolute average of all equations.

Physically this residual will be the average resulting forces' magnitude still in the model, but the magnitude of this number will depend heavily on the model which is dealt with. To normalise the residual it is divided by the total amount of force applied to the system

$$F = \sum_i \mathbf{b}_i \tag{7.11}$$

$\mathbf{b}_i =$ bodyload applied to particle i [N].

7.4 Discrete Bond Constant

The peridynamic bond constant, c, from Equation 3.12 sets the strength of the individual bonds and should be adjusted such that it resembles a material with some Young's Modulus E. This is done by comparing the strain energy density W_{el} under isotropic expansion for both peridynamics and continuum mechanics. The strain energy density for peridynamics is normally computed assuming that the region of influence is a sphere, but the discrete system this will be used in has its region of influence determined by particles. Ganzenmüller *et al.* in [28] shows that computing the bond constant from the discrete system provides a better match as variations to δ will only change the bond constant if the size of the discrete region changes as illustrated in Figure 7.7.

The strain energy density in continuum mechanics for a material under isotropic expansion as shown in [7] is

$$W_{el} = \frac{9\kappa s^2}{2} \quad (3D), \qquad W_{el} = 2\kappa s^2 \quad (2D). \tag{7.12}$$

$\kappa =$ Bulk modulus [Pa], $s =$ strain [-].

The equivalent measure in peridynamics for both an idealised and discrete system is

$$W_{el} = \frac{1}{2}\int_H w(s)dH \approx \frac{1}{2}\sum_{j\in H_i} w(s)V_j \tag{7.13}$$

$H =$ region of influence [m^3], $V_j =$ volume of particle j [m^3],

where the micropotential is

$$w(s) = -\int f(s(\boldsymbol{\eta}), \xi)d\boldsymbol{\eta} = \frac{1}{2}cs^2\xi, \tag{7.14}$$

$f(\cdot,\cdot) =$ microforce, scalar version of Eq. 3.12 [N/m^6],

$\boldsymbol{\eta} =$ relative displacement vector see Eq. 3.5 [m], $\xi =$ length of $\boldsymbol{\xi}$ see Eq. 3.6 [m].

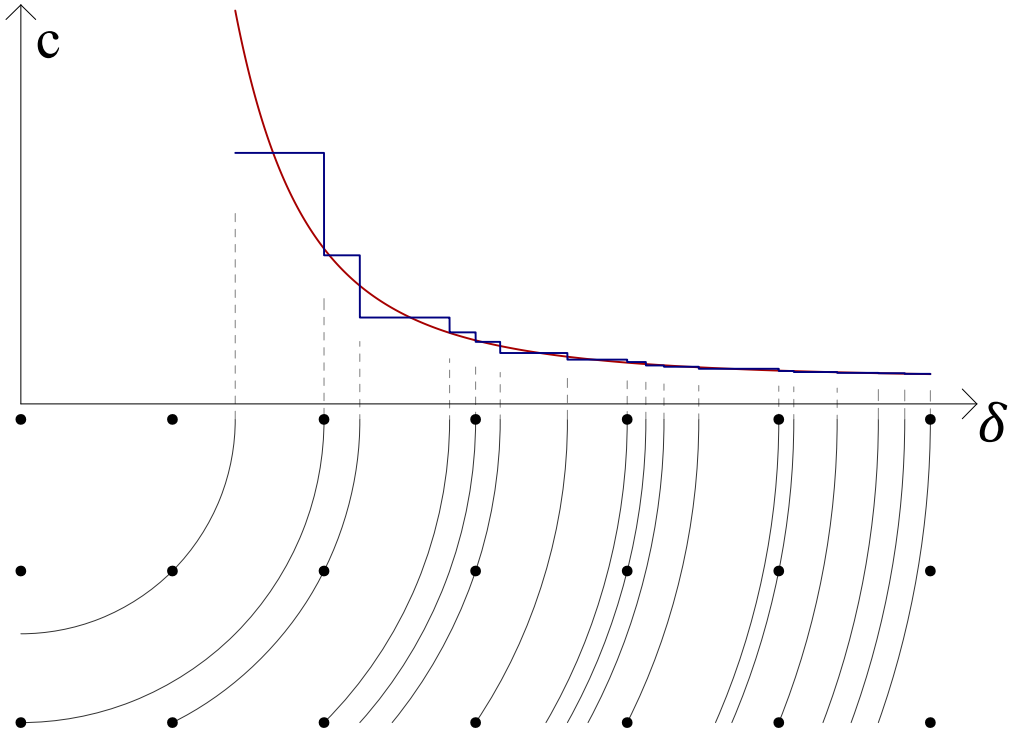


Figure 7.7: Graphing the bond constant c in 2D against δ and illustrating that while the idealised solution in red changes values continuously, no new bonds are formed per particle, resulting in different effective stiffnesses based on δ.

By then comparing the peridynamics and continuum mechanics solution we can solve for c using the discrete system

$$c = \frac{18\kappa}{\sum_j \boldsymbol{\xi}_j V_j} \quad (3D), \qquad c = \frac{8\kappa}{\sum_j \boldsymbol{\xi}_j V_j} \quad (2D). \tag{7.15}$$

The idealised values for reference are

$$c = \frac{18\kappa}{\pi\delta^4} \quad (3D), \qquad c = \frac{12\kappa}{\pi\delta^3} \quad (2D). \tag{7.16}$$

7.5 Interactivity

The interactive possibilities of peridynamics are compelling and to facilitate that efforts have been made to allow for some changes to the model as it is running.

The current progress of the solution is wholly contained in all the positions and velocities of the particles, with the caveat that to calculate the damping as we do one also needs the previous time step's forces. To let the simulation continue from a previous time step we hence need to map these values from the old model to the new one. Therefore these values are considered as data points in a vector field which can be used for interpolation. Here we used linear interpolation as that seemed sufficient.

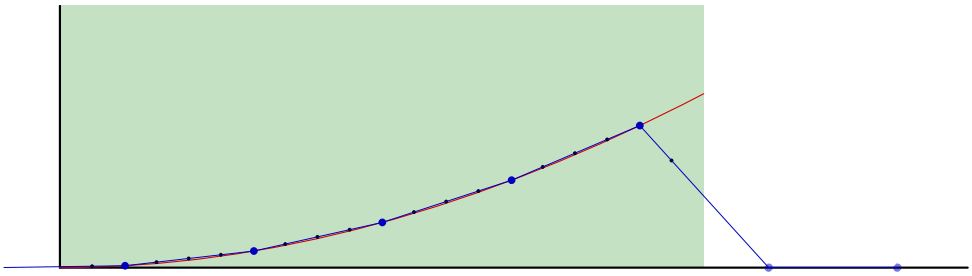


Figure 7.8: A true displacement field in red is approximated by linear interpolation from some data points in blue. The points in black showcase that changing the point resolution gives problems at borders with high displacement as it interpolates using undefined values.

The remaining problem is that while the field has defined values inside the old solid, no values exist outside of it. This creates a problem as peridynamics does not produce surface values, as such, increasing the fidelity means interpolating between values from inside the domain and undefined ones, see Figure 7.8.

To solve this a finishing step was added when the old simulation completes, approximating the values outside of the domain. The approximation is done by taking the average of nearby defined values. This keeps the edge values similar to their true values and provides a smooth interface as one extends the domain, which is sufficient for keeping the simulation stable. An example showing the smoothed interface when extending a domain and increasing the resolution can be seen in Figure 7.9.

Together with the simplicity of storing and reading from fields and the ease with which a new model is set up (see Section 7.1.1) the model is in essence interactive.

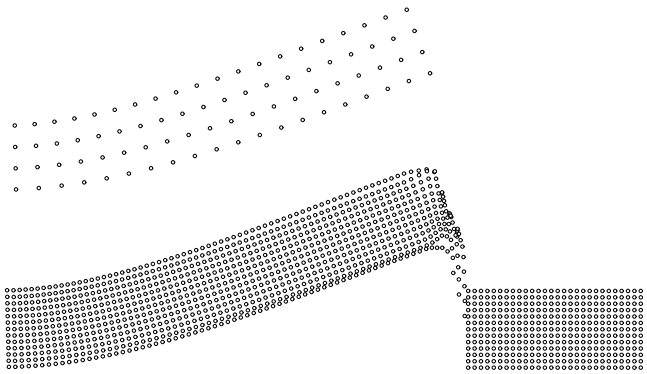


Figure 7.9: The top shows a converged solution for a cantilever and below is an interpolation of the displacement field both within the original shape and in an extension of the field to the right.

7.6 Data Footprint

Keeping the amount of data stored and manipulated during the simulation down is of interest as it both lets one run larger simulations and makes the data which is to be manipulated during the simulation more local and hence faster to access. As this implementation uses voxels, every cell will have memory allocated even if that data is not manipulated during the simulation.

Spatial data is stored in image objects which can store 1, 2 or 4 primitive values per voxel. Therefore the volume factor is stored with the displacement vector as it is a scalar value and can be used to indicate if the voxel is active. Furthermore, the data which is to be modified is allocated twice so that one image can be used to read and the other to write.

Table 7.1: Memory use of the simulation on a GPU.

Image Data		
Use	**#**	**bytes / voxel**
Displacement & Volume factor	2	4·sizeof(float)
Velocity	2	4·sizeof(float)
Force	2	4·sizeof(float)
Density	1	4·sizeof(float)
Bodyload	1	4·sizeof(float)
Springs	1	4·sizeof(float)

Linear Array Data	
Use	**bytes / voxel**
Damping Buffer 1	2·sizeof(float)
Residual Buffer 1	sizeof(float)
Damping Buffer 2	$\lceil 2/(256 \cdot 512) \cdot$ sizeof(float) $\rceil$
Residual Buffer 2	$\lceil 1/(256 \cdot 512) \cdot$ sizeof(float) $\rceil$

Constant Data	
Use	**bytes**
$\boldsymbol{\xi}$ & $\|\|\boldsymbol{\xi}\|\|$	$(\lfloor\delta\rfloor \cdot 2 + 1)^3$·sizeof(float)

The pre-computed $\boldsymbol{\xi}$ is stored with both the vector itself and its length as both are used, but this only depends on δ and is relativity small.

The result is slightly more than 156 bytes per voxel as a float is 4 bytes, which for reference fits about 6.4 million voxels in 1 GB of memory.

7.7 Particle Importance

It can often be useful to have a measure of how relevant any part of a structure is for the total capacity, where a common use case is Evolutionary Structural Optimisation (ESO) and the measure is von Mises stresses. For peridynamics however, von Mises stresses are computed non-locally through the stress tensor (see Section 3.8) and can be perturbed easily when removing material nearby (which is what ESO does). Hence an alternate measure is proposed.

The idea is to measure how much the solution would change if the particle was removed, but a true evaluation of that for each particle would be unwieldy. Instead, the measure is in how much the residual would change (see Section 7.3).

Removing a particle would remove its corresponding equation in the system of Equation 3.16 but it would also remove that particle's contributions in its neighbours' equations. Assuming that the system is at equilibrium, then removing the equation will have no impact, but that particle being removed impacts the neighbouring equations through the forces in the bonds. The force in a bond is equal and opposite for the other particle (see Equation 3.8) and since we are in equilibrium a change in any direction is detrimental. Hence the measure proposed is

$$\sum_j |f_{ij}| + |\boldsymbol{b}_i|. \tag{7.17}$$

f_{ij} = Force applied by the bond between i and j [N], $\boldsymbol{b}$ = bodyload [N].

This is useful as it is trivial to compute as all values are computed during a step regardless and can be used to determine what particles are important. For the full workflow proposed in this thesis, however, this is not used but showed promise during development for other tasks.

7.8 Non-Linear Material Behaviour

Linear elastic solids often carry their loads in a distributed manner where large swats of material might be engaged marginally. We are however interested in how to apply reinforcement efficiently to it where one might assume that the concrete has cracked.

One option to make the solid act more like that is by changing the material to more closely resemble that of concrete and reinforcement. To do this we employ an impossible material where the force in a bond is set to zero if the tension is in a medium amount of strain. This model how the concrete cracks, and losses carrying capacity, while emulating that areas with large tensile strains would receive reinforcement by a designer. It should be stressed that the curve in Figure 7.10 is not a stress-strain curve for a material, but a force-strain curve per bond.

To alleviate the amount of input needed to be provided by a user, and ensure that the model does not fall apart the strains needed are picked automatically by assuming that the model is in the ultimate limit state. As such the largest strain should correspond to the largest strain concrete/steel can bear.

Using this material can be illuminating in how a solid can redistribute loads as it cracks but is not directly used in the workflow proposed by this thesis.

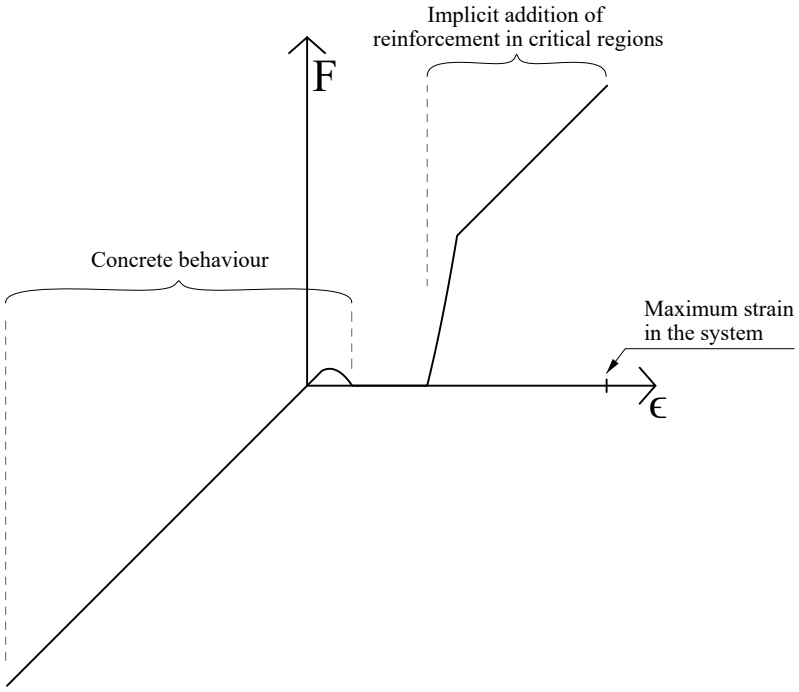


Figure 7.10: The force per bond is plotted against the strain, where it has been modified to emulate a designed concrete structure in the ultimate limit state.

8

Principal Stress Based Truss Generation

In this section, the method of how to generate the initial truss topology is developed from the principle described in Section 4.2.

8.1 Tracing Principal Stress Curves

To trace curves based on the principal stresses the principal stress field in the domain needs to be known as well as the start position and a start vector. It is necessary to input a start vector to determine in which direction the principal stress should be traced. Each voxel will have three different directions and the curve needs to follow the correct one. Since the current position to be evaluated will not perfectly coincide with a voxel a trilinear interpolation based on the eight closest voxels in 3D and four in 2D according to the method described in [29] will be utilised. This method works by systematically doing linear interpolation in each global coordinate direction, see Figure 8.1 for an example in 3D. First a linear interpolation is conducted along the x axis generating the values c_{00}, c_{01}, c_{10} and c_{11}. These values are in turn interpolated along the y axis, generating the values c_0 and c_1. Lastly, these two values are interpolated along the z axis which gives the sought value c at the relevant position.

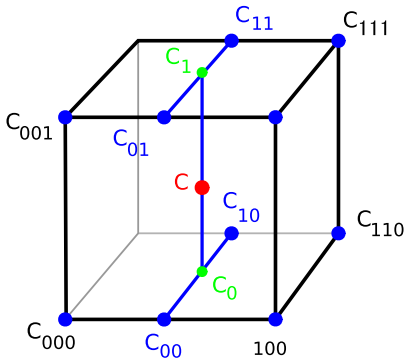


Figure 8.1: Trilinear interpolation for the value c at the position of the red dot. [1]

[1] Created by Marmelad, licensed under CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4355112

To evaluate the correct principal direction for each point, the scalar product between each principal direction and the vector used for the previous step is computed. The direction with the largest scalar product should be used for the linear interpolation. For unitised vectors, the scalar product is large if the direction of the vectors is similar.

To calculate the next vector for the tracing of the curve the Crank-Nicolson method is used as described in Equation 8.1. Where $\boldsymbol{x}_0$ is the current position, $\boldsymbol{x}_1$ is the next position that should be traced to and $\mathbf{F}(\boldsymbol{x})$ is the interpolated unit vector of the principal stress direction at the position $\boldsymbol{x}$. The value of α_l is a factor deciding the distance between the point $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$. This is an iterative method since the next step is decided both from the value of the field at the start and end position. In general, very few iterations are needed to find the position of the next point $\boldsymbol{x}_1$. The curve is automatically stopped if it exists through the domain, loops around to its beginning or if the value of the interpolated stress at the point $\boldsymbol{x}_1$ is considered to be too low. The cutoff limit is decided as a factor of the maximum principal stress in the domain.

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \alpha_l \frac{\mathbf{F}(\boldsymbol{x}_1) + \mathbf{F}(\boldsymbol{x}_0)}{2}, \tag{8.1}$$

$\boldsymbol{x}$ = location, α_l = step length,
$\mathbf{F}(\cdot): \quad \mathbb{R}^3 \mapsto \mathbb{R}^3 \quad =$ vector valued function.

8.2 Principal Stress Curves Along the Boundary

If one or more of the closest voxels (but not all) are outside of the domain the interpolation needs to be handled differently for the curve to either asymptotically stay inside of the domain or exit perpendicular to the edge. This is because there must exist equilibrium along an unloaded boundary, see Figure 8.2. Therefore the direction of the principal stress with a non-neglectable magnitude needs to be parallel to the edge. For the voxels that are outside of the domain but still next to the current position, the normal vector for the boundary at that voxel is calculated. This normal vector is calculated by adding up vectors from the current voxel to the voxels inside of the domain that is within the horizon, see Figure 8.3. The average principal direction of the closest voxels to the position is mirrored with this normal vector and this value is assigned to the voxel that is outside of the domain. For the case with a loaded boundary, the principal stress will not be parallel to the edge but its direction will be affected by the load. However, the same procedure is used for both a loaded and unloaded boundary. This will introduce a small error on the traced curve but since it will still exit the boundary at approximately the same point it is deemed acceptable and will not affect the generation of the initial truss topology to a large degree.

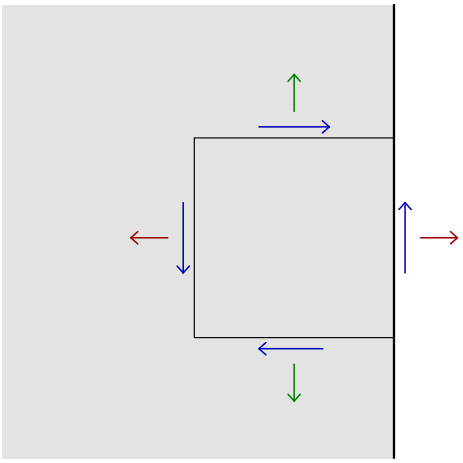


Figure 8.2: An infinitesimal piece of material on a free boundary. The free surface has no forces applied to it, implying that the blue and red forces are zero through global equilibrium.

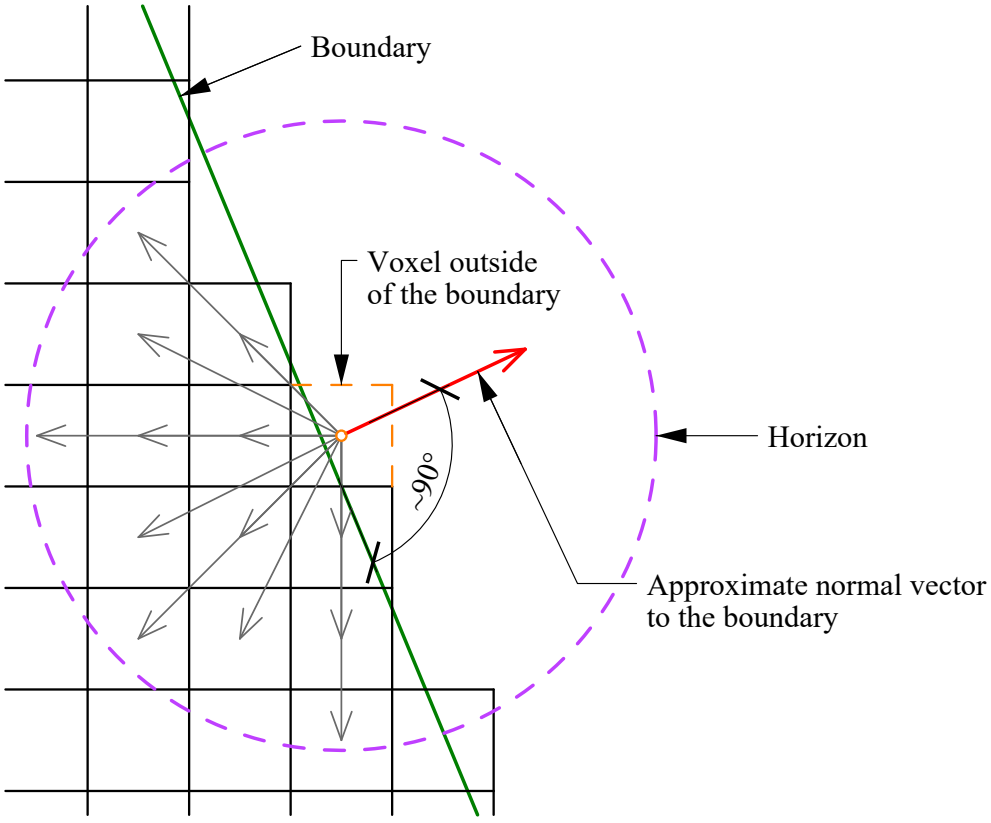


Figure 8.3: Figure showing how the normal vector to the boundary is calculated based on the position of the voxels. Even though the image is showing a 2D case the same principle is applicable in 3D as well.

8.3 Generation of the Initial Truss Topology

The method to generate the initial truss topology is based on the method presented in Section 4.2 developed in [13]. Changes are made to the method to better match the aim of this thesis and successfully generate typologies for more cases and overcome some of the stated limitations and difficulties presented in Section 4.4.

Phase I

1. Tracing principal stress curves from the loading position/s in all directions. For distributed loads, the user manually decides how to discretise the load.
2. If these curves connect to a new boundary edge/surface or the beginning of another one of these curves, it is kept and called a phase I curve.
3. If the curve did not fulfil any criteria in the previous step, a check is conducted to determine if it intersects one of the previously defined phase I curves, if so this curve is kept and also added to the other phase I curves.
4. Local maximum points of von Mises stress along with these phase I curves are identified. These local maximum points are kept if they are not located close to a boundary edge/surface and if the stress state at that point is multi-axial.

Phase II

1. From all of the kept local maximum curves new principal stress curves are traced. These curves do not begin precisely at the local maximum point but are offset a certain distance (specified by the user) in the direction of the tangent to the phase I curve it originates from and also in the main directions of the normal plane to the same curve.
2. If the newly traced curve intersects a phase I curve it is kept and called a phase II curve. A line is drawn from the local maximum to the intersection point. The line based on that phase I curve should now also not connect directly from its start to an endpoint but via this intermediate point.
3. If one of the traced curves from the local maximum point connects to a boundary in a new way the curve is kept as a phase II curve and a new line is drawn.
4. For the rest of the curves that do not fulfil any of the previous criteria it is checked if they intersect with an already defined phase II curve. If so, a line is not drawn directly to the intersection point but instead to the closest endpoint of the phase II curve to create as few nodes as necessary.

Phase III

1. New curves are traced from the support boundaries and if they connect any endpoint of a previously defined curve a new line is drawn.
2. If the loading and support conditions are collinear, curves are traced from the supports regions in descending order based on their von Mises stress.
3. If one of the traced curves intersects with any curve traced in phase I that curve is kept and a new line is drawn. This process is repeated until all stressed supports are connected to the truss topology.

For all of the steps mentioned above, it is checked so that different nodes are not created too close to each other (by a tolerance supplied by the user) but instead merged into one node to simplify the truss topology. It is not in this stage prob-

lematic if the algorithm should generate duplicate lines since these will easily be removed at a later stage. Therefore such a check is not included in the algorithm. However, it is still important to not create nodes that are too close to each other and therefore such a check is conducted.

In Phase I step 3 is new and that is because in some cases relevant curves traced from the loading positions are discarded without it. The first step in Phase II is modified by beginning the tracing of new curves at an offset distance from the local maximum point. This is because the principal stress field will not change direction exactly at that point but over a region instead, see Figure 8.4. Step 4 in Phase II is also a new edition added to identify other crucial members in the truss topology. New nodes are not inserted for the topology to not become overly complex. In Phase III the first step is added to be able to handle structures with roller supports.

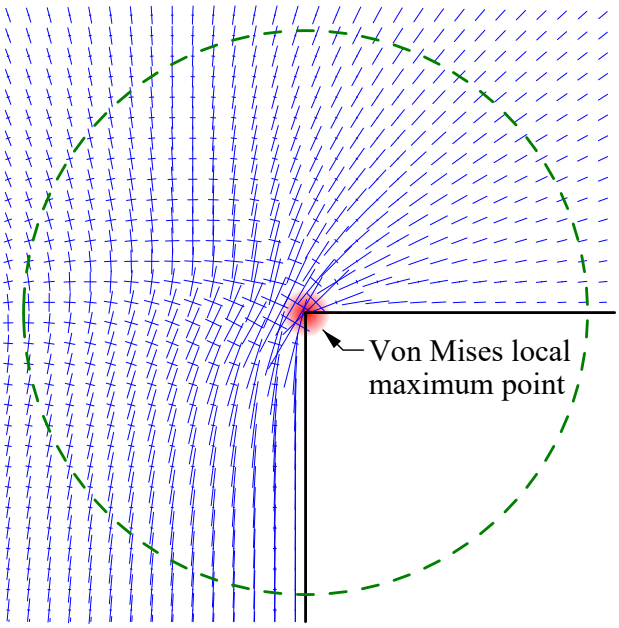


Figure 8.4: The principal stress field is visualised with the blue lines (compression) at discrete points. The local maximum point of the von Mises stress is depicted in red and the region in which the principal stress field changes direction is within the green dashed circle.

For the 3D case, it is also important to note that principal stress curves that ought to intersect each other and construct a node there in fact very seldom will intersect each other. Therefore an extra tolerance is required to find these curves that almost form an intersection with each other. For the two dimensional cases, this is not a problem since all curves are located in the same plane.

In Figure 8.5 it can be seen how the described method works for a deep beam subjected to a distributed load with a hole. The von Mises stress distribution and the principal stress field were calculated from our peridynamics solution and visualised per voxel. The principal stress curves are traced in all of the phases described above.

For this case, it has been chosen by the user to divide the distributed load into two equal parts and to start the generation from the middle of these parts. In phase I, the load points connect to the support points but also finds the connection between the load points. The local maximum points are identified on these curves and new curves are traced, the start position of these curves is offset a bit to find the tension line because of the distribution of the node region described earlier. In phase III the support points are connected to each other. The tension tie is located a little too far down and currently is not contained within the domain, however, since this is only the initial topology that is not regarded as an issue. Similarly, the bottom tension tie is also located too far down and the top compression strut too high.

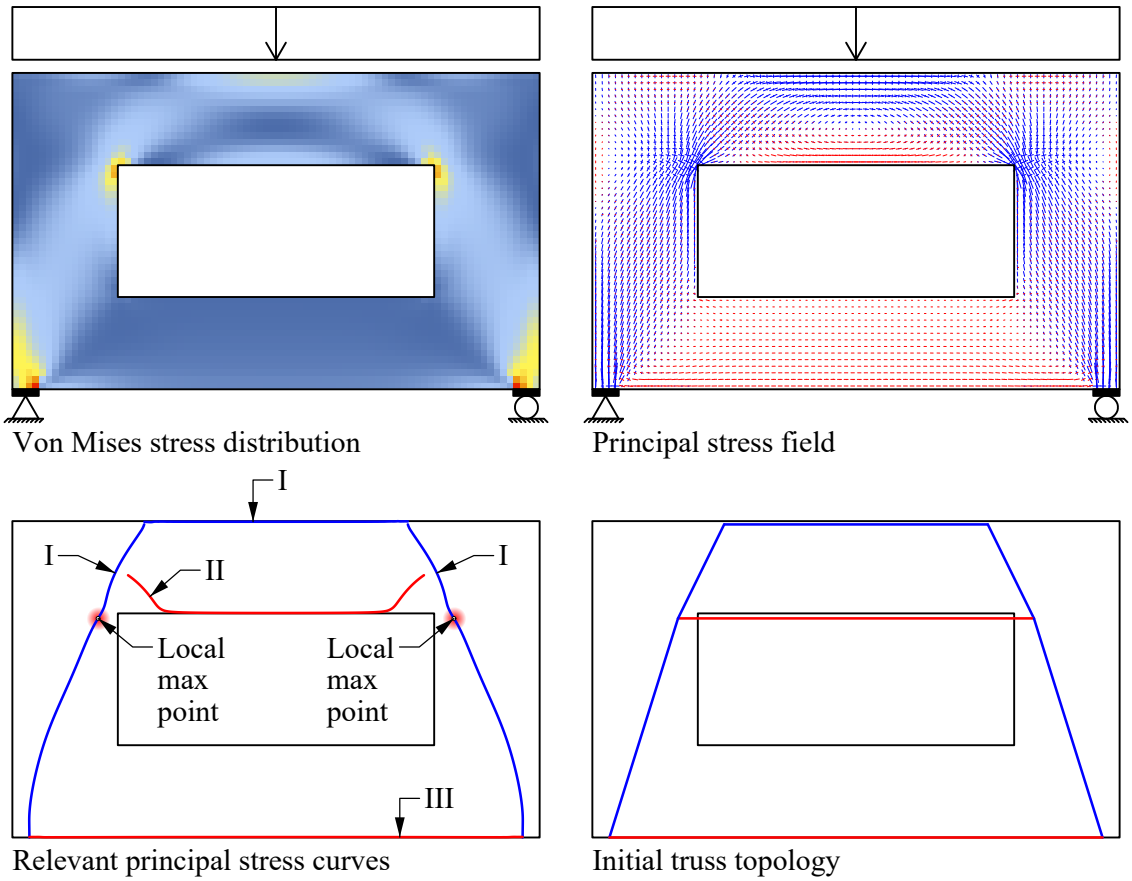


Figure 8.5: Generation of the initial truss topology for a deep beam with a hole.

8.4 Local Maximum Points

The method to find the local maximum points on curves is also developed to generate more stable results. In [13] local maximum points are searched for over the entire structure and then activating the points if a traced curve is passing close to one of them. This method might generate a lot of uninteresting points and also no check is directly done if the point is subjected to a multiaxial stress state.

In the improved method, the von Mises stress is interpolated along the traced curve and local maximum points along the curve are instead located. To determine if it is a suitable node location with a multiaxial stress state the principal stress field is studied. If the principal stress in one direction is much larger than the other two the position has uniaxial stress and is not suitable for a node since the stresses in general only flow in one direction.

8.5 Stability of Truss Topology

The method described will generate a truss topology that for the input boundary conditions will carry the loads efficiently to the supports. However, there is no guarantee that the truss will be stable and it might be a mechanism, in the sense of FE analysis, the determinant of the row reduced stiffness matrix may be zero. To remedy this extra diagonal truss members need to be created, however, the forces in the original truss members ought to be much larger than in the new members since they were generated by the principal stresses.

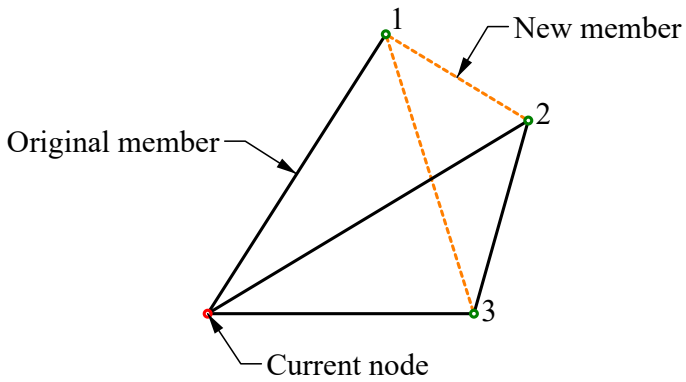


Figure 8.6: Triangulating the truss by checking if the nodes that the current node is connected to are in turn connected to each other.

The extra diagonals required could be generated directly by the user but for larger trusses, this might be cumbersome and therefore a method to do so based on the original topology is created. For each node, it is checked if the nodes that it is connected to are connected to each other, see Figure 8.6. If they are not connected a new member is constructed between them. For the example shown the current node is connected to nodes 1-3, since nodes 1 and 2 are not originally connected a new member is created between them. If the new truss member should pass outside of the domain, duplicate an existing member or pass through a node it is discarded. This is a simple method to triangulate the truss and thus make sure that it is stable. This method will in some cases generate more new members than required to ensure a stable structure but since these members are only added for stability reasons the

forces in them should ideally be close to zero. The user can of course also remove undesired members manually. In Figure 8.7 the 3D topology in the upper parts is braced using the described method resulting in the topology in the lower figure with the new lines in orange. The original structure is arbitrarily drawn and does not originate from a principal stress field.

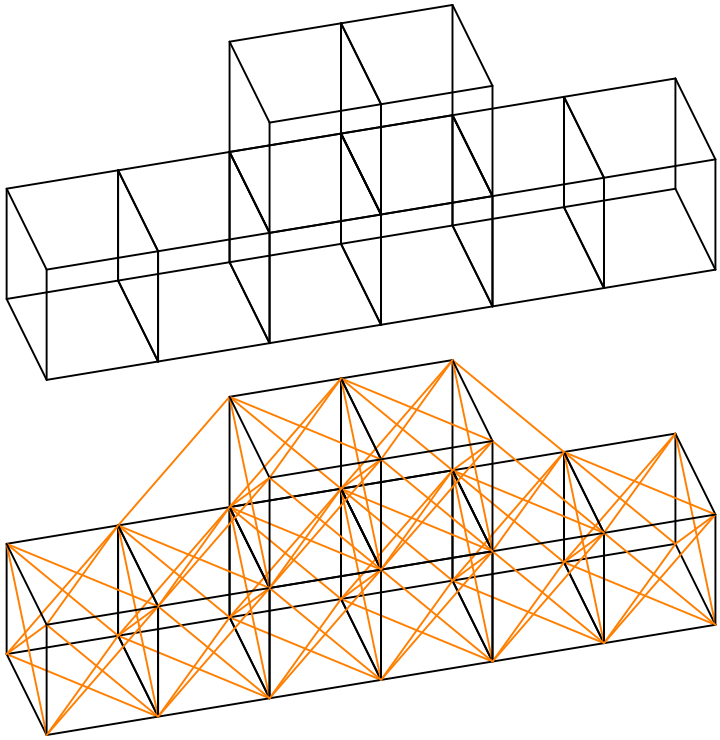


Figure 8.7: The original topology shown in the upper part is braced with the orange lines in the lower part.

9
Truss Optimisation

The preceding truss topology was based on a linear stress field and is made to produce an efficient truss, yet for STM, what is approximated is the plastic response and since it is encased in concrete regardless, the importance of struts is less than that of ties. The aim here is to take the existing topology and change it such that the required reinforcement is minimised. This change must still enforce that all elements hold and are inside the solid. The model is then solved with the steepest descent method (Section 5.1) using gradient projection (Section 5.3), exterior penalty method (Section 5.4) and Armijo's step rule (Section 5.2).

Finding the optimal truss position is a non-linear problem which often has many local optimums which can make it difficult to find the global optimum. It should be noted that it is not the global optimum that is sought for since the aim is to find the efficient plastic load paths for which there exists a continuous change between the linear state and the plastic one. Hence given that the initial truss models the linear state, a continuous gradient-based approach which finds a local minimum resembles the physical process. While this does not find the global optimum of the objective function this should find the truss describing the system.

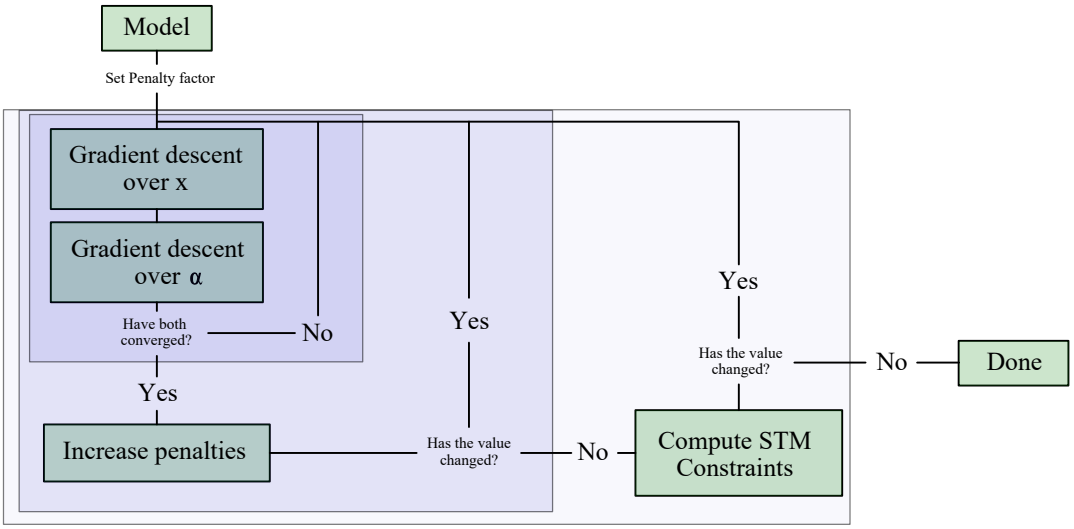


Figure 9.1: Flowchart of the various steps worked through as the optimisation solver converges.

Figure 9.1 gives a high-level outline of how the optimisation process will operate, where the penalties refer to the exterior penalty method (see Section 5.4) and the computation of STM constraints are detailed in Section 9.5.

The following package will also be used in the application:

- CSparse - Sparse matrix solver package for C#

9.1 Variables and Sets

The variables for the problem are the position of the nodes and the area of the elements, at each end. But the available areas are highly dependent on the position of the nodes, and to avoid having free-floating variables loosely connected by some constraint, the variables will instead be the positions of the nodes and reduction factors for the areas, where a maximum area is set by a function dependent on the nodal positions.

$\boldsymbol{x}$ = Initial coordinate for each degree of freedom. [m]

$\boldsymbol{\alpha}$ = Reduction factor of the maximum possible area of an element. [-]

The area function for an element's end is the area of the maximal cuboid which fits in the domain projected along with the normal vector of the element, see Section 2.3,

$$A(\mathbf{x}) = \text{Proj}_A\{\max\{\text{Cube}(\mathbf{x})\}, \mathbf{n}\}. \tag{9.1}$$

Details on how this is done and done efficiently can be seen in Appendix A.4 and A.5.

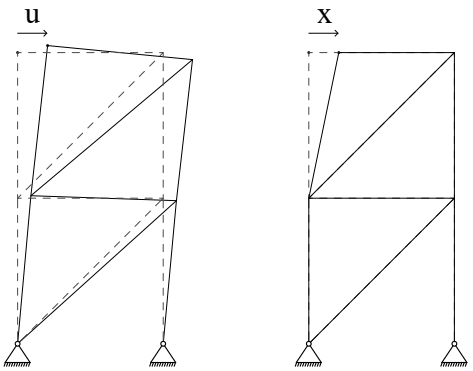


Figure 9.2: $\boldsymbol{u}$ denotes displacement of some structure due to boundary conditions. $\boldsymbol{x}$ denotes a change in the initial placement of the structure.

To aid in notation we will also define the sets:

$$\mathbf{N} = \text{indices of all nodes in the model,} \tag{9.2}$$

$$\mathbf{E} = \text{indices of all elements in the model,} \tag{9.3}$$

$$\mathbf{I} = \{i \in \mathbf{E} \quad | \quad \epsilon_i > 0\}, \tag{9.4}$$

$$\mathbf{J} = \{j \in \mathbf{E} \quad | \quad \epsilon_j \leq 0\}. \tag{9.5}$$

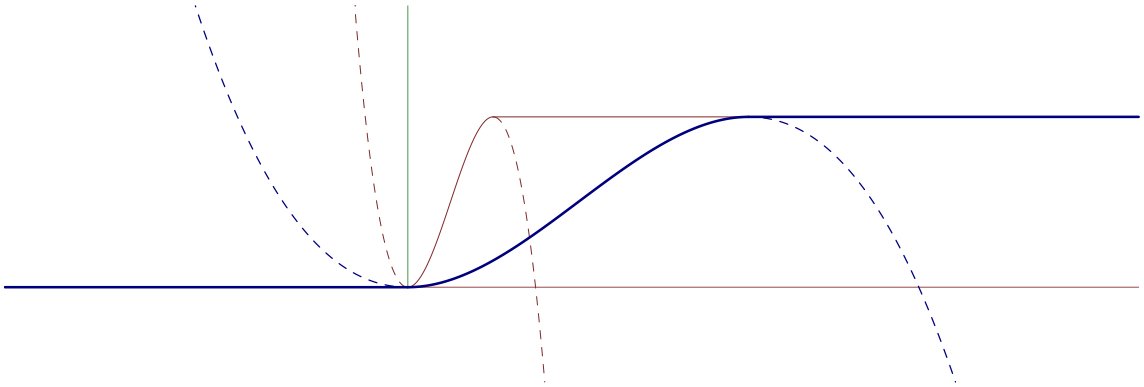


Figure 9.3: An indicator function $I(x)$ with a smooth connection illustrating that the transition region can have its extent changed to alleviate convergence. The different colours indicate different region sizes and the dashed lines indicate unused parts of the polynomial.

9.2 Functions

9.2.1 Objective Function

As noted in [2] minimising over the strain energy (Equation 2.1) is normally said to produce an optimal model, but a truss with minimal strain energy will increase its element areas to reduce the strain. Hence another function is used.

The goal is a model with a low amount of reinforcement, a natural option would be to minimise over the volume. This would however create an issue as it creates a discontinuity when the strain of an element passes zero. This discontinuity means that the objective function value is not well approximated by its derivative. Hence this discontinuity is smoothed by fitting a polynomial such that the values and derivatives are continuous, see Figure 9.3, and in a similar manner as with the exterior point method, we let this transition region tend towards the initial function. This will let the method "know" that it can also remove volume by turning it from a tie to a strut.

Hence the objective function is

$$f(\boldsymbol{x}, \boldsymbol{\alpha}) = \sum_i A_i l_i I(\epsilon_i). \tag{9.6}$$

A_i = Mean area of element i [m^2], l_i = length of element i [m],
$I(\cdot)$ = smoothed indicator function [-], ϵ_i = mean strain in element i [-].

9.2.2 Constraints

A functional STM is the end goal for this optimisation and constraints must be added to ensure that its specifications are maintained.

9.2.2.1 Utilisation

The struts utilisation is calculated such that for each node, the reduction factors are calculated based on the number of ties that connect to it as described in Section 2.1.2. The stress in both ends of the element is then compared against the maximal allowed stress. We will also note that we are using the simplification that tapered struts stiffness can be approximated by the average area of the ends and that this will make the element slightly stiffer, see Appendix A.3.

$$\frac{EA_m\epsilon}{A_e} \leq \sigma_d\beta, \tag{9.7}$$

is rewritten to be zero at equality and normalised such that the penalties will be unit-less

$$g_j^2(\boldsymbol{x}, \boldsymbol{\alpha}) = \frac{EA_m\epsilon}{A_e\sigma_d\beta} - 1. \tag{9.8}$$

E = Young's modulus for the concrete [Pa], ϵ = mean strain of the element [-],
A_m = mean area of the element [m^2], A_e = area of the smallest end [m^2],
σ_d = design stress for the concrete [Pa], β = reduction factor [-].

The ties have a similar constraint where their utilisation should not be overused. The ties however should also be designed such that they fail first as such an equality constraint is implied, but since the objective function enforces as little reinforcement as possible we relax that constraint to only consider the utilisation and trust that it will converge towards equality regardless.

$$E_{steel}\epsilon \leq f_{yd}, \tag{9.9}$$

$$g_i^1(\boldsymbol{x}, \boldsymbol{\alpha}) = \frac{E_{steel}\epsilon}{f_{yd}} - 1, \tag{9.10}$$

E_{steel} = Young's modulus for the steel [Pa], ϵ = strain in the element [-],
f_{yd} = design yield strength for the steel [Pa].

9.2.2.2 Geometric Constraints

The truss will only be valid for the STM if the material used comes from within the solid, hence the nodes should be constrained such that the whole truss is within the valid domain. This is done implicitly by connecting the area of the elements to the available space in the model where if an element approaches an edge, the area and hence stiffness will tend towards zero. If the element is needed, the reduction in stiffness will keep it from exiting the domain, and if it is not needed it does not matter.

To control how the solution develops, some nodes are locked onto lines. This is achieved with gradient projection, i.e. the steps will be constrained to the direction of the lines,

$$h(x) = \text{Dist}\{x, line\} = 0. \tag{9.11}$$

9.2.2.3 Orthogonality

To facilitate the constructability of the produced trusses, the reinforcements are preferably placed in an axis oriented manner. This can be encouraged with a penalty function targeting non-orthogonal elements in tension,

$$\text{Orth}(x) = \min\left\{f, \frac{||\mathbf{n} - \mathbf{n}_{max}||^2}{||\mathbf{n}||^2}\right\}. \tag{9.12}$$

Where $\mathbf{n}_{max}$ is a vector containing only the maximum coordinate of $\mathbf{n}$. The function is limited by a maximum value f as very non-orthogonal elements may have cause to be so, and more importantly, as a slightly off-axis element is similarly bothersome as a very off-axis one. One can also note that for numerical reasons the switch between the two functions is smoothed out.

9.3 Model

The model is thus defined as:

$$\begin{aligned}
\text{minimise} \quad & f(\boldsymbol{x}, \boldsymbol{\alpha}) + \sum_{i \in \mathbf{I}} \text{Orth}(\boldsymbol{x}, i) && && \text{(9.13a)}\\
\text{subject to} \quad & g_i^1(\boldsymbol{x}, \boldsymbol{\alpha}) \leq 0 && \forall i \in \mathbf{I}, && \text{(9.13b)}\\
& g_j^2(\boldsymbol{x}, \boldsymbol{\alpha}) \leq 0 && \forall j \in \mathbf{J}, && \text{(9.13c)}\\
& 0 \leq \boldsymbol{\alpha}_k \leq 1 && \forall k \in \mathbf{E}, && \text{(9.13d)}\\
& h(\boldsymbol{x}_i) = 0 && \forall i \in \mathbf{N}. && \text{(9.13e)}
\end{aligned}$$

The model can then be rewritten according to the exterior penalty method (Section 5.4) for the complicating constraints.

$$\begin{aligned}
\text{minimise} \quad & f(\boldsymbol{x}, \boldsymbol{\alpha}) + \sum_{i \in \mathbf{I}} \text{Orth}(\boldsymbol{x}, i) \quad + && && \text{(9.14a)}\\
& \nu \sum_{i \in \mathbf{I}} \max\{0, g_i^1(\boldsymbol{x}, \boldsymbol{\alpha})\}^2 \quad + && && \text{(9.14b)}\\
& \nu \sum_{j \in \mathbf{J}} \max\{0, g_j^2(\boldsymbol{x}, \boldsymbol{\alpha})\}^2 && && \text{(9.14c)}\\
\text{subject to} \quad & 0 \leq \boldsymbol{\alpha}_k \leq 1 && \forall k \in \mathbf{E}, && \text{(9.14d)}\\
& h(\boldsymbol{x}_i) = 0 && \forall i \in \mathbf{N}. && \text{(9.14e)}
\end{aligned}$$

9.4 Gradients

For the steepest descent the gradient of the objective function and penalty functions are needed, and as such we will begin by expanding a generic derivative $(\cdot)'$ for each relevant function.

$$f'(\boldsymbol{x}, \boldsymbol{\alpha}) = \sum_i A_i' l_i I(\epsilon_i) + A_i l_i' I(\epsilon_i) + A_i l_i I'(\epsilon_i)\epsilon_i', \tag{9.15}$$

$$\left(g_j^2(\boldsymbol{x},\boldsymbol{\alpha})^2\right)' = 2\left(\frac{EA_m\epsilon}{A_e\sigma_d\beta} - 1\right)\frac{EA'_m\epsilon A_e + EA'_m\epsilon' A_e - EA_m\epsilon A'_e}{A_e^2\sigma_d\beta}, \tag{9.16}$$

$$\left(g_i^1(\boldsymbol{x},\boldsymbol{\alpha})^2\right)' = 2\left(\frac{E_{steel}\epsilon}{f_{yd}} - 1\right)\frac{E_{steel}\epsilon'}{f_{yd}}. \tag{9.17}$$

Where one can identify three main derivatives to be determined: ϵ', l', A'.

9.4.1 Length

The length of an element will depend on the location of its endpoints, $\boldsymbol{e}(\boldsymbol{x})$ and $\boldsymbol{s}(\boldsymbol{x})$

$$l = \sqrt{(\boldsymbol{e}-\boldsymbol{s})^2}, \tag{9.18}$$

$$\frac{\partial l}{\partial \boldsymbol{s}_i} = -\frac{\boldsymbol{e}_i - \boldsymbol{s}_i}{l}, \qquad \frac{\partial l}{\partial \boldsymbol{e}_i} = \frac{\boldsymbol{e}_i - \boldsymbol{s}_i}{l}. \tag{9.19}$$

9.4.2 Maximum Area

The maximum area an element can have is dependent on $\boldsymbol{x}$ and computed as the average area of the endpoint areas (Appendix A.3)

$$A = \frac{A_s + A_e}{2}, \tag{9.20}$$

$$\frac{\partial A}{\partial x} = \frac{A'_s + A'_e}{2}, \tag{9.21}$$

$$\frac{\partial A_s}{\partial x} = \mathbf{n}' \cdot \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} + \mathbf{n} \cdot \begin{bmatrix} A'_x \\ A'_y \\ A'_z \end{bmatrix}, \tag{9.22}$$

where

$$A'_x = \text{width} \cdot \text{length}' + \text{width}' \cdot \text{length}, \quad \text{etc.} \tag{9.23}$$

The maximal cuboid dimensions are evaluated through a linear interpolation of a slightly treated field as specified in Appendix A.4. As such the function is the linear interpolation which is trivial to take the derivative of.

$\mathbf{n}$ denotes the positive normal which derivatives are

$$\frac{\partial \mathbf{n}}{\partial x} = \frac{\partial}{\partial x}\left(\begin{bmatrix} |x|/l \\ |y|/l \\ |z|/l \end{bmatrix}\right), \tag{9.24}$$

$$\frac{\partial \mathbf{n}_x}{\partial x} = \frac{(x^2+y^2+z^2)\frac{\partial |x|}{\partial x} - x|x|}{(x^2+y^2+z^2)^{3/2}}, \tag{9.25}$$

$$\frac{\partial \mathbf{n}_y}{\partial x} = -\frac{x|y|}{(x^2+y^2+z^2)^{3/2}}, \tag{9.26}$$

where further cases follow from symmetry.
One can also note that due to the absolute value the derivative is discontinuous at zero.

9.4.3 Strain

Small axial strains for a FEM model are determined by the end nodes displacement $\mathbf{u}_s$ and $\mathbf{u}_e$

$$\epsilon = \frac{\mathbf{n}(\mathbf{u}_e - \mathbf{u}_s)}{l}, \tag{9.27}$$

i.e. the projection of the displacement onto the original element divided by the length.

$$\epsilon' = \frac{\mathbf{n}'(\mathbf{u}_e - \mathbf{u}_s)l + \mathbf{n}(\mathbf{u}'_e - \mathbf{u}'_s)l - \mathbf{n}(\mathbf{u}_e - \mathbf{u}_s)l'}{l^2} \tag{9.28}$$

The derivative of the normal is much like that of the positive normal:

$$\frac{\partial \mathbf{n}_x}{\partial x} = \frac{y^2 + z^2}{(x^2 + y^2 + z^2)^{3/2}}, \tag{9.29}$$

$$\frac{\partial \mathbf{n}_y}{\partial x} = -\frac{xy}{(x^2 + y^2 + z^2)^{3/2}}, \tag{9.30}$$

where further cases follow from symmetry.

9.4.4 Displacement

Then we have the displacement vector $\mathbf{u}$ which is determined through FEM, see [30], as

$$\mathbf{K}\mathbf{u} = \mathbf{f}. \tag{9.31}$$

Which through implicit differentiation becomes

$$\mathbf{K}'\mathbf{u} + \mathbf{K}\mathbf{u}' = \mathbf{0}, \tag{9.32}$$

$$\mathbf{K}\mathbf{u}' = -\mathbf{K}'\mathbf{u}. \tag{9.33}$$

This is a linear system of equations where $\mathbf{K}$ and $\mathbf{u}$ are constant for all derivative directions. By using sparse methods as such as LDL decomposition [31] as opposed to the explicit inverse one can solve the system while keeping it sparse. This coupled with that derivative of the stiffness matrix will be zero for all elements not connected to the variable which is modified. Hence the method is that the elements of the derivative of the stiffness matrix are both assembled and multiplied with $\mathbf{u}$ to form a new vector which with the existing LDL decomposition can be used to solve for $\mathbf{u}'$ which is the expected change in displacements given a change to the initial node positions $\boldsymbol{x}$.

The stiffness matrix derivative $\mathbf{K}'$ can be assembled from its local parts much like the stiffness matrix can. Where the different derivatives are specified as:

$$\mathbf{K}^{e\prime} = \mathbf{T}^{T\prime}\bar{\mathbf{K}}^e\mathbf{T} + \mathbf{T}^T\bar{\mathbf{K}}^{e\prime}\mathbf{T} + \mathbf{T}^T\bar{\mathbf{K}}^e\mathbf{T}', \tag{9.34}$$

$$\frac{\partial \bar{\mathbf{K}}^e}{\partial x} = E\alpha \frac{lA' - l'A}{l^2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \tag{9.35}$$

$$\frac{\partial \bar{\mathbf{K}}^e}{\partial \alpha} = \frac{EA}{l} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \tag{9.36}$$

$$\frac{\partial \mathbf{T}}{\partial x} = \begin{bmatrix} \mathbf{n}' & \mathbf{0} \\ \mathbf{0} & \mathbf{n}' \end{bmatrix}^T, \tag{9.37}$$

$$\frac{\partial \mathbf{T}}{\partial \alpha} = \mathbf{0}. \tag{9.38}$$

9.4.5 Orthogonality

The chosen function in Equation 9.12 is segmented into some different parts, but symmetry means that they will be very similar and we will only need to examine one of them. So assuming that the value is less than f and that x is the maximum coordinate we have:

$$g = \frac{y^2 + z^2}{x^2 + y^2 + z^2}, \tag{9.39}$$

$$\frac{\partial g}{\partial x} = -\frac{2x(y^2 + z^2)}{(x^2 + y^2 + z^2)^2}, \tag{9.40}$$

$$\frac{\partial g}{\partial y} = \frac{2x^2 y}{(x^2 + y^2 + z^2)^2}. \tag{9.41}$$

We can see from Equation 9.40 that the gradient will smooth out at the minimum and for the cut off to the plateau, we make use of interpolation down to a close to zero value. Some value is left such that the system knows how to change given that nothing is opposing it.

9.5 Evaluation of Strut-and-Tie Model

The truss optimisation uses some relaxed constraints to simplify evaluation and make it more feasible to take derivatives, as such the next step is to evaluate the model more fully and feed that back to the model as fixed constraints. Fixed constraints are constraints which should change due to the variables, but are fixed to the value evaluated at this point. A FEM solution will be used to solve for the forces in the truss and the evaluation will be based on the most critical nodes and the ties according to Eurocode [5], where full element containment and areas are computed.

The restriction which was relaxed in the continuous optimisation was that if multiple elements connect to a node, then all of their areas must be the projections from the same cuboid along each element's normals (Section 2.3). Since each element is connected to two nodes and each node updates based on the areas of the element, a new bounding box and neighbouring areas must be calculated repeatedly until

the areas stop decreasing. This iterative nature makes it both slow to evaluate and difficult to take the derivative of, which is why it has been placed as a different kind of constraint.

From the converged relaxed solution we know the current forces in the elements and can determine the minimum area necessary $A_{minimum}$ assuming that a change in the area does not affect the forces. This is not true, since the system is almost always statically indeterminate, but it is an approximation. From the current areas $A_{current}$, we can also find the true areas A_{real} which fulfil the restricted condition given that non of the current areas can increase. The difference between these two vectors, A_{real} and $A_{minimum}$, shows which elements have too little area, and with the assumption that the relative area between two elements connecting to the same node will be similar, we can estimate what areas all elements need to have at a minimum. This assumption about connections means that we can apply the penalty function directly to the smaller elements which induce the reduction after the restriction is done here. That is important as the elements which would break think they have sufficient area in the relaxed model.

The final step in Equation 9.44 is to apply the iterative area check on the increased set of areas, as non-critical elements might have received impossibly high areas as a result of the linear scaling of all areas. Applying the same method again means that the solution which we enforce the continuous optimisation to converge against is similar to what will be produced here.

$$A_{real} = \text{REDUCE}(A_{current}), \tag{9.42}$$

$$A_{minimum} \leq \lambda A_{real}, \tag{9.43}$$

$$A_{constraints} = \text{REDUCE}(\lambda A_{real}), \tag{9.44}$$

where REDUCE$(\cdot)$ is to apply the iterative procedure to find the largest consistent areas without increasing any areas and λ is the smallest number greater than 1 such that Equation 9.43 is fulfilled.

The resulting constraints $A_{constraints}$ are applied as minimum element areas to fulfil in the relaxed model using the exterior penalty method. Then the model is allowed to converge again following the schematics in Figure 9.1. The following subsections detail how the REDUCE$(\cdot)$ step finds the true areas.

9.5.1 Optimising Cuboid Size

As such, the desire for each node is to have the largest axis oriented cuboid which fits inside the domain where non of the projected areas are larger than some initial set areas. The areas are the current areas of all connecting elements and the produced box will inform them of their new areas.

This is a problem which will be solved as an optimisation problem by maximising the largeness of the cuboid such that the projected area in the element directions does not exceed some values and such that the cuboid remains in the domain.

The objective function measuring the size of the cuboid, and thus determining what being largest means, is chosen to be the ratio between the volume to the surface area. This is because the volume grows quicker than surface area making it prefer large cuboids and the ratio makes it tend towards a cube-like shape.

The projected area of a cuboid is described in Appendix A.5. The optimisation problem is as such:

$$\begin{aligned} \max_{x,y,z} \quad & \frac{xyz}{xy+xz+yz} \\ \text{subject to} \quad & n_x^i yz + n_y^i xz + n_z^i xy \leq A_i \quad \forall i \in \mathbf{I}, \\ & x \leq X, \quad y \leq Y, \quad z \leq Z, \end{aligned}$$

where X, Y, Z are the maximum width, depth and height allowable by the domain and $\mathbf{I}$ is the set of indices for constraints containing the positive projection vector and the maximum area of the element.

It is solved with the steepest descent (Section 5.1) and gradient projection (Section 5.3), where the gradient of the objective function f and constraint g is

$$\nabla f = \begin{bmatrix} \frac{y^2z^2}{(yz+x(y+z))^2} \\ \frac{x^2z^2}{(xz+y(x+z))^2} \\ \frac{x^2y^2}{(xy+z(x+y))^2} \end{bmatrix}, \quad \nabla g = \begin{bmatrix} \boldsymbol{n}_y z + \boldsymbol{n}_z y \\ \boldsymbol{n}_x z + \boldsymbol{n}_z x \\ \boldsymbol{n}_x y + \boldsymbol{n}_y x \end{bmatrix}. \tag{9.45}$$

This gradient can be used to form a ray which can return us to the feasible space. The ray $\boldsymbol{r}(t)$ is formed from some stepping point $\boldsymbol{p}$ along the closest violated constraint's gradient ∇g

$$\boldsymbol{r}(t) = \boldsymbol{p} + \nabla g \cdot t, \tag{9.46}$$

which can be used to solve an constraint to equality

$$g(\boldsymbol{r}(t)) = A, \tag{9.47}$$

resulting in

$$at^2 + bt^2 + c = A, \tag{9.48}$$

$$a = \boldsymbol{n}^T \begin{bmatrix} \nabla g_y \nabla g_z \\ \nabla g_x \nabla g_z \\ \nabla g_x \nabla g_y \end{bmatrix}, \quad b = \boldsymbol{n}^T \begin{bmatrix} \nabla g_y \boldsymbol{p}_z + \nabla g_z \boldsymbol{p}_y \\ \nabla g_x \boldsymbol{p}_z + \nabla g_z \boldsymbol{p}_x \\ \nabla g_x \boldsymbol{p}_y + \nabla g_y \boldsymbol{p}_x \end{bmatrix}, \quad c = \boldsymbol{n}^T \begin{bmatrix} \boldsymbol{p}_y \boldsymbol{p}_z \\ \boldsymbol{p}_x \boldsymbol{p}_z \\ \boldsymbol{p}_x \boldsymbol{p}_y \end{bmatrix}, \tag{9.49}$$

which provides us with the new projected point $\boldsymbol{r}(t)$ which will fulfil this constraint. The other constraints projections are trivial.

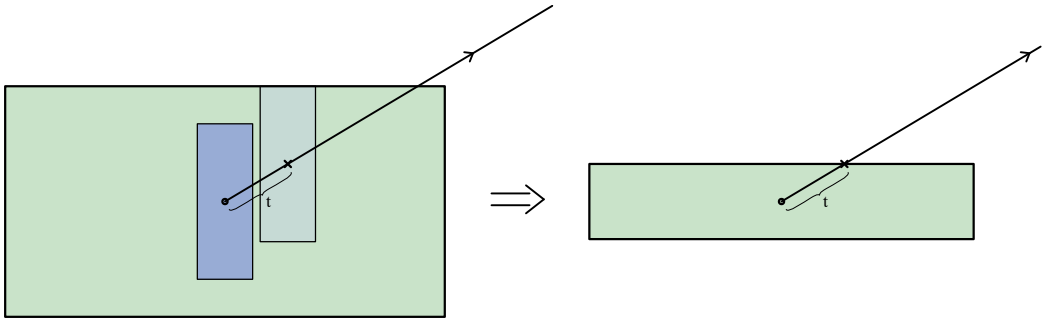


Figure 9.4: Furthest step length such that an enclosed box remains enclosed can be represented as a ray intersection of a single box.

9.5.2 Evaluate Element Containment

Once the node sizes are determined they implicitly set the shape of the elements, and those need to be checked for containment. For this, the signed distance field from Appendix A.4 will be used.

The size of the nodes is known and the shape at any point in between them can be interpolated from them. Through the SDF we can also evaluate the largest box centred at any point, thus beginning at one node one evaluates how far along with the element one can move the box such that the large box domain is not exceeded. This will inform us of a new stepping point where a new largest and interpolated box can be evaluated. As such one can step along the element and evaluate if it exceeds the size of the domain. The step size can however become 0, either because an intersection is found or that the field is growing in that direction and we evaluate it at a point. This is worked around by taking a trial step forward to see if it still fits.

The step size is determined by reducing the size of the enclosing box by the size of the inner box and performing a ray intersection with all the positive face planes and finally returning the intersection with the shortest step length, see Figure 9.4,

$$t = \min_i \frac{B_i - b_i}{2n_i}. \tag{9.50}$$

$B =$ Vector with the enclosing box's dimensions.
$b =$ Vector with inner box's dimensions.
$n =$ Positive step direction as a unit vector.

We also define $\frac{1}{0} = \infty$ as it would involve moving parallel to the plane with which to intersect.

It tentatively attempts to remedy elements which are not contained by projecting a cuboid which does fit where the intersection is. There are however limits to how much information can be backpropagated as the area is a scalar and the containment

might be very directionally dependent. The second alternative would be to alter the sizes of the cuboids connecting the element, but the ability to make a good decision on both which of them and how they should change such that the containment is valid and that it does not disturb connecting elements in the wrong way, is limited.

Part III
Findings

10

Peridynamics

Results evaluating the general implementation, augmentations and measures described in Chapter 7.

10.1 Reference Solution

Madenci *et al.* [7] specifies some suitable reference solutions for 3D peridynamics solvers which have idealised analytical solutions.

Both examples use a block of material fully locked at one end with a Young's Modulus E of 200 GPa. The horizon parameter δ used is 3.015Δ and the particles are placed in a 100x10x10 grid.

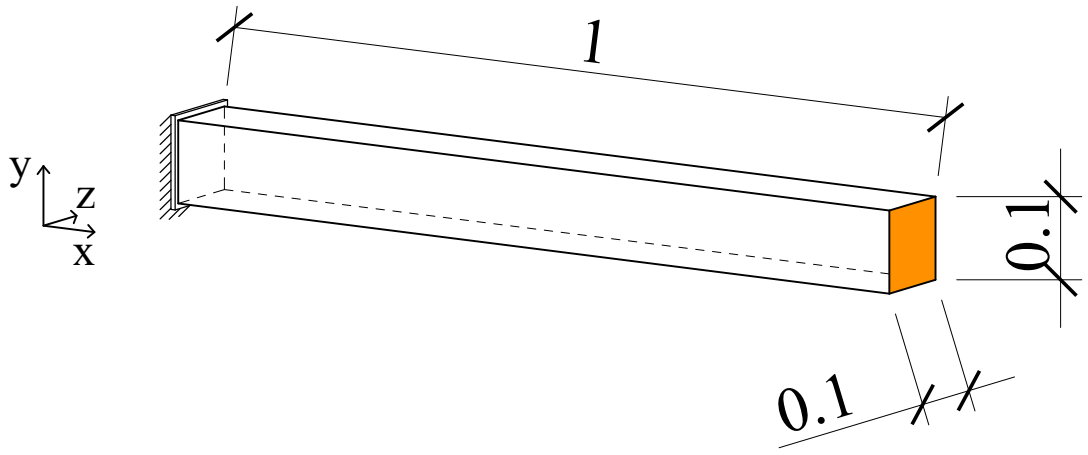


Figure 10.1: Test setup for the subsequent reference tests.

10.1.1 Block of Material Under Tension

This test applies an tensile pressure of 200 MPa along the x-axis to the orange area in Figure 10.1. The analytical solution for displacements at the center line of the block in the different axis is

$$u_x(x, 0, 0) = \frac{P}{E}x, \tag{10.1a}$$

$$u_y(0, y, 0) = -\nu\frac{P}{E}y. \tag{10.1b}$$

Graphs comparing the implemented peridynamics solution against the analytical ones is displayed in Figure 10.2 and 10.3. Furthermore, the von Mises stresses and the principal stress field as calculated by the implementation are displayed in Figure 10.4.

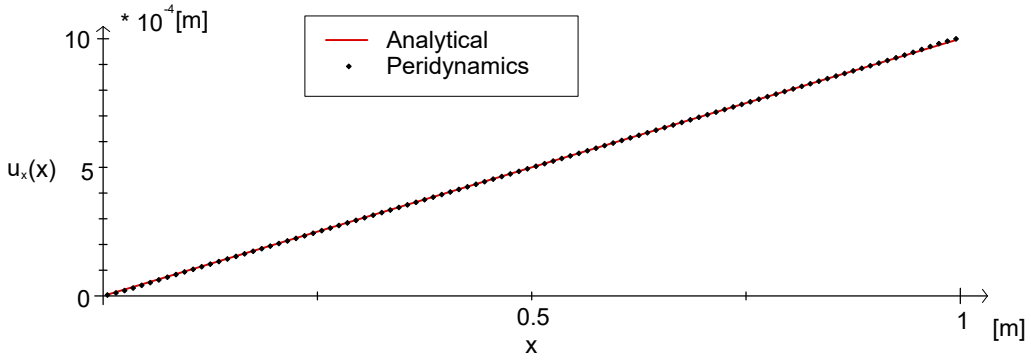


Figure 10.2: Axial displacement along a centre line in the bar under axial loading both the analytical and peridynamic solution.

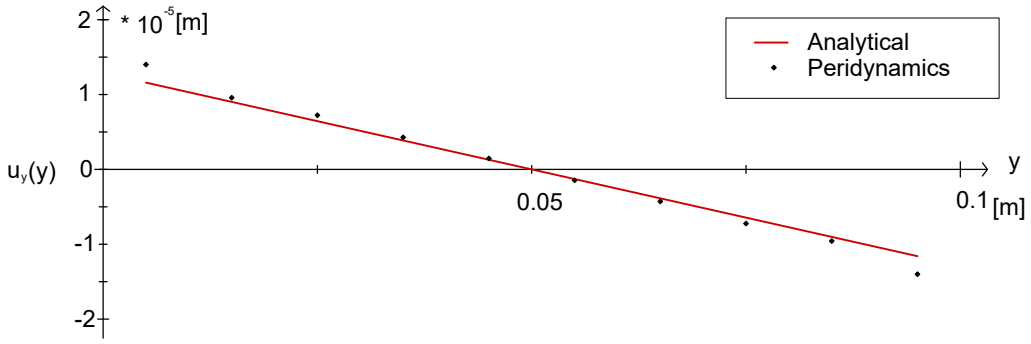


Figure 10.3: Displacement along the y-axis in a centre line of the bar under axial loading, both the analytical and peridynamic solution.

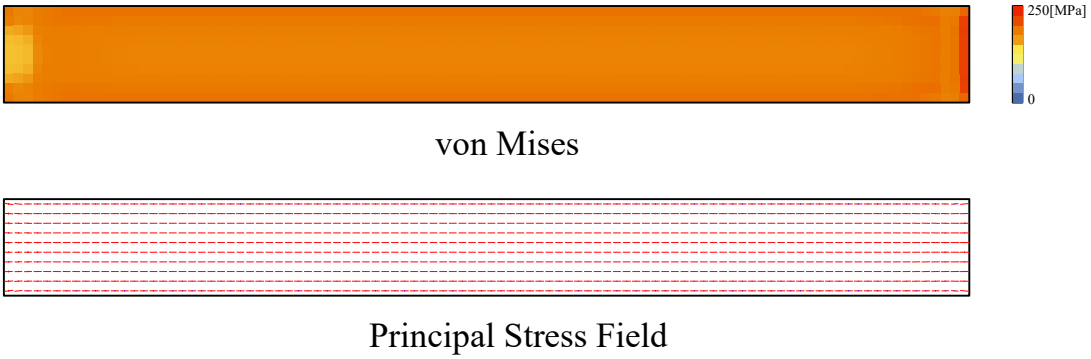


Figure 10.4: The von Mises and principal stress through a section of the bar under axial loading.

10.1.2 Block of Material Under Transverse Loading

This test applies a force of 5 kN along the y-axis to the orange area in Figure 10.1.

The Euler-Bernoulli analytical solution for displacement in the y-direction along the beams centre line is

$$u_y(x, 0, 0) = \frac{P}{6EI}(3L - x)x^2. \tag{10.2}$$

The Euler-Bernoulli solution is a good approximation of the actual solution to the 3D continuous problem for the dimensions of the tested cantilever.

The graph comparing the results is in Figure 10.5 and the implementations computed von Mises stresses and principal stresses are in Figure 10.6.

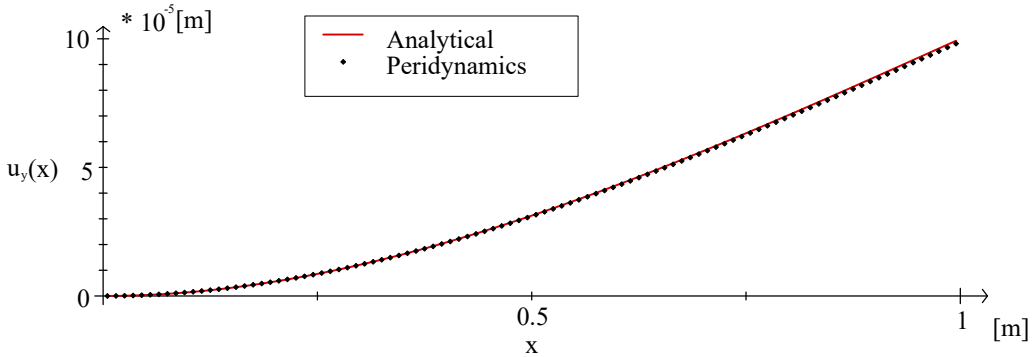


Figure 10.5: Transverse displacement along a centre line of a cantilever under transverse loading for both the analytical and peridynamics solution.

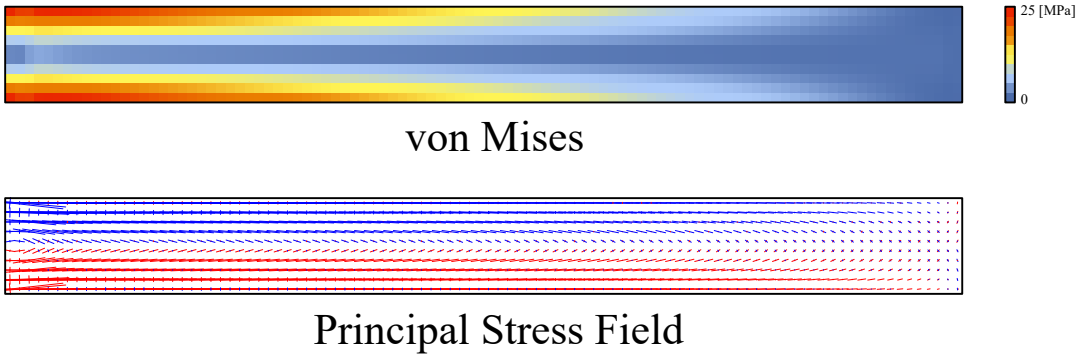


Figure 10.6: Von Mises and principal stress through a section of the cantilever.

10.2 Performance

As peridynamics is a computationally intensive method and the intent is for interactivity, the performance is of importance and compared in the following subsections.

The Computational Processing Unit (CPU) used for testing was:
Intel(R) Core(TM) i7-7820HQ CPU 2.90GHz
The Grapical Proccessing Unit (GPU) used for testing was:
NVIDIA Quadro M2200

10.2.1 CPU vs GPU

The computational time for an iteration has been measured for both the CPU and GPU with the given implementation, where the CPU is single-threaded. The test used $\delta = 2.5$ and such that most voxels were active particles. The test shows that the GPU is around 200 - 300 times faster than the CPU and that the CPU has a relative slowdown as the number of particles increases and the GPU the reverse.

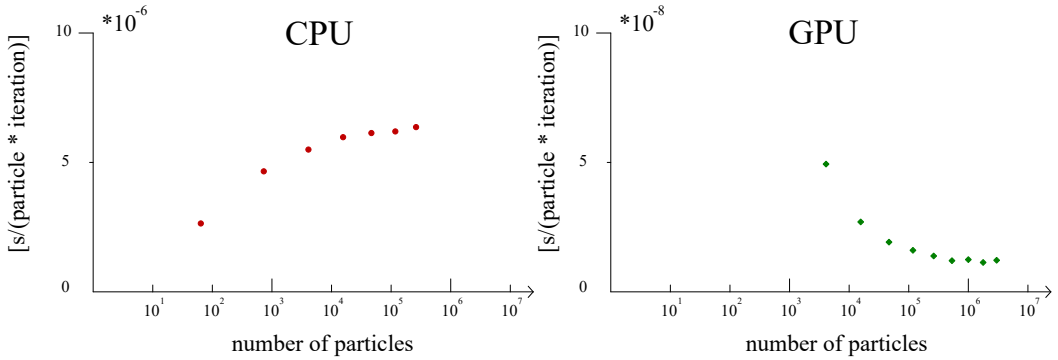


Figure 10.7: Seconds per iteration and number of particles in the system plotted against the size of the simulation for the CPU and GPU implementation. Note that the scales differ by a factor of 100.

10.2.2 PeriPy Comparison

PeriPy is a Python implementation of bond-based peridynamics which uses an extension to call OpenCL commands through C++ which is described in the paper by Boys *et al.* [32]. The data is structured such that each particle's position and neighbour list are stored and that each bond is set as a work item. That is, each particle has a work group to compute its force. PeriPy uses a constant damping factor, which reduces the amount of global synchronisation. PeriPy produces its particles by constructing a 3D mesh which it then fills and then computes all the neighbours. One should note that the setup time is non-negligible but not included in the test results as it measures how quickly it can iterate.

The test was run by using example two in PeriPy's example projects, containing 13539 particles with a $\delta = \pi$ which ran for 5000 iterations. The time from the implementation in this thesis is made using the same $\delta = \pi$ and is modified to use a constant damping factor and forgo residual checks to emulate similar behaviour as the simulation in PeriPy. See Table 10.1 for the results.

Table 10.1: Speed comparison against PeriPy for seconds per iteration and particle.

PeriPy		2.76e-7
Presented GPU		2.53e-8
Speed-up		≈10x

10.2.3 Pre-computed ξ

The execution speed of the force calculation has been compared with and without a pre-computation of $\boldsymbol{\xi}$ as described in Section 7.1 and displayed in Table 10.2.

Table 10.2: Improvements due to a pre-computation of $\boldsymbol{\xi}$ for both the CPU and GPU.

	Pre-computed $\boldsymbol{\xi}$	variable $\boldsymbol{\xi}$	Factor
CPU	4.03e-6	7.83e-6	195%
GPU	1.67e-8	2.72e-8	163%

10.3 Boundary Conditions

The classic peridynamic boundary conditions have been revised by this thesis and the following subsections compare these new implicit ones against the classical and what an analytical solution produces.

10.3.1 Dirichlet

The classic displacement peridynamic boundary conditions as described by Silling in [6] specify a region of material outside the domain which has the displacement

prescribed. This introduces softening within the particle influence region of boundary surfaces. This is compared against the implicit solution as presented in this thesis in Section 7.2.1 and against an analytical solution. To illustrate these effects a short bar with few particles is subjected to a prescribed tensile displacement along the x-axis of 0.05 meters at each end, see Figure 10.8. The green areas in the figure are the particles subjected to the displacement along the x-axis, it should be noted that they are not restrained in any other direction.

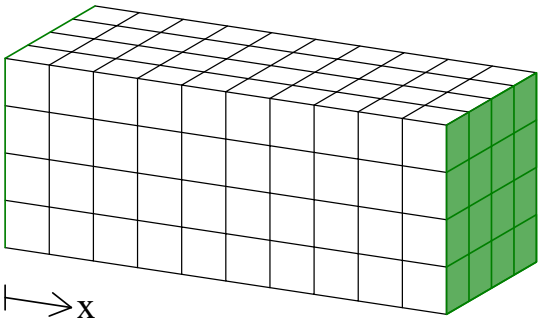


Figure 10.8: Applied boundary conditions to the short bar, the green areas indicate that it is subjected to Dirichlet boundary conditions.

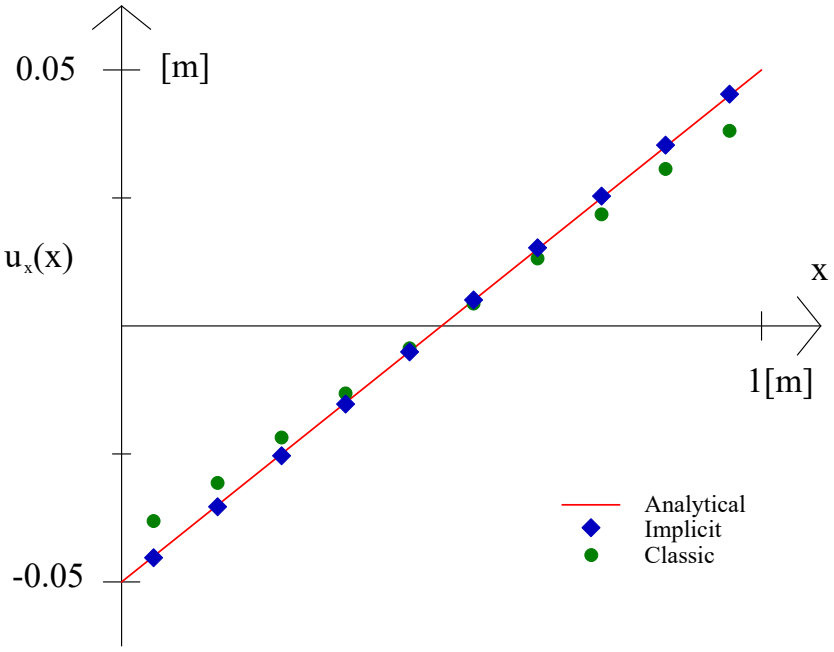


Figure 10.9: Axial displacement for a bar fixed at both ends comparing classic peridynamic boundary conditions against implicit ones.

10.3.2 Neumann

The classic peridynamics traction boundary condition as described by Kilic in [33] increases the amount of material which can skew results for coarse discretisations. To illustrate this a cantilever with a uniformly distributed load is compared against the analytical solution (Equation 10.3) and the boundary condition presented in Section 7.2.2, see Figure 10.10 for the test setup.

A choice was made that the added material does not receive any volume stiffness due to being on an edge. It was chosen as it produced the most accurate results for the classical solution.

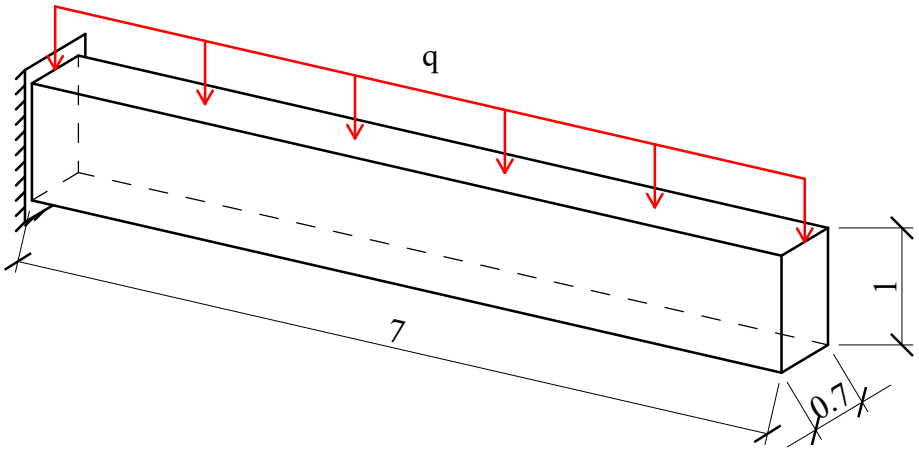


Figure 10.10: Boundary conditions and dimensions for the cantilever.

E =1000 Pa, L =7 m, b =0.7 m,
h =1 m, $q = \frac{1}{70}$ N/m in x-direction,
Δ =0.1, δ =3.015,
number of particles =4900.

The analytical solution for an Euler-Bernoulli beam is

$$u_y(x) = \frac{qx^2L^2}{24EI}\left(6 - \frac{4x}{L} + \frac{x^2}{L^2}\right). \tag{10.3}$$

Based on the measurements of the tested cantilever, the Euler-Bernoulli solution is a good approximation of the actual solution of the 3D continuous problem.

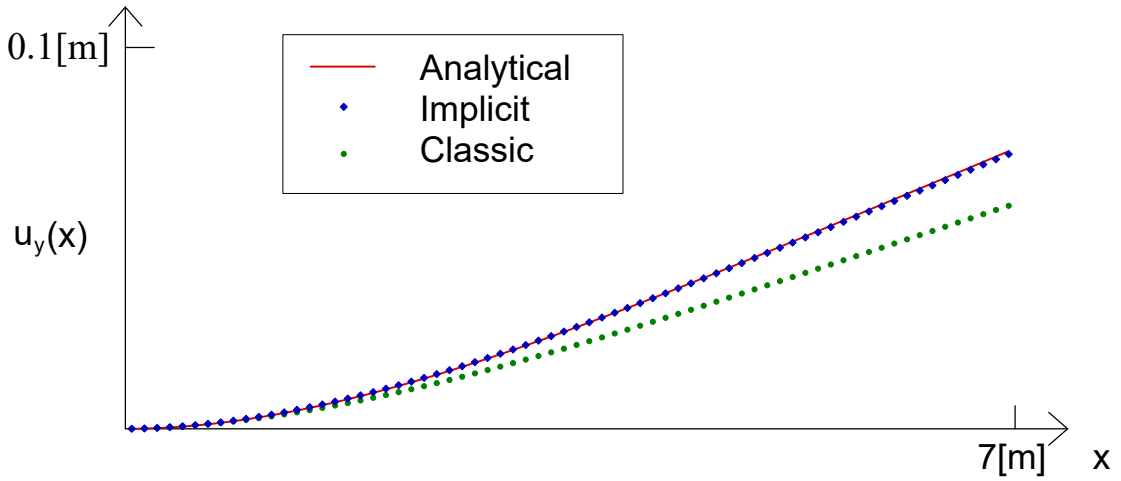


Figure 10.11: Vertical displacement for a cantilever comparing classic peridynamic boundary conditions against implicit ones.

10.4 Particle Importance

To showcase the concept of particle importance (as described in Section 7.7) an evolutionary structural optimisation procedure is conducted on a cantilever with dimensions and boundary conditions as presented in Figure 10.12. This is done two times, the first when particles are removed based on each particle's von Mises stress and in the second material is removed based on the importance of each particle. In both cases, this is an iterative procedure and first, all particles less than 1% of the current measurement are removed. For each iteration, this percentage is increased until it has reached an efficient solution. For each step, it is made sure that a converging solution is found. In Figure 10.13 the particles are removed based on their von Mises stress and in Figure 10.14 the particles are instead removed based on their importance. It should be noted that the figures are very closely related to each other and indicate a similar behaviour. The optimisation based on von Mises stress contains a little bit more noise and is also less symmetrical. Even if the structure in Figure 10.13 looks like it already has collapsed it should be noted that the horizon for each particle is set to 2.5 and therefore all particles are in some way still connected to the rest.

Table 10.3: Time per iteration and particle while using particle importance and von Mises.

	Standard	Particle importance	von Mises
time	5.69e-6	5.77e-6	6.88e-6
relative	100%	101.4%	120.9%

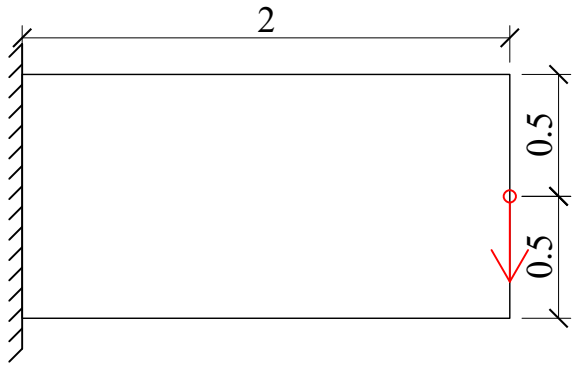


Figure 10.12: Dimensions and boundary conditions for the cantilever, subjected to evolutionary structural optimisation.

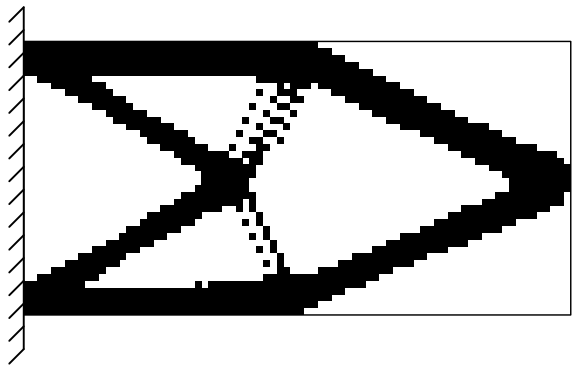


Figure 10.13: Evolutionary structural optimisation based on von Mises stress.

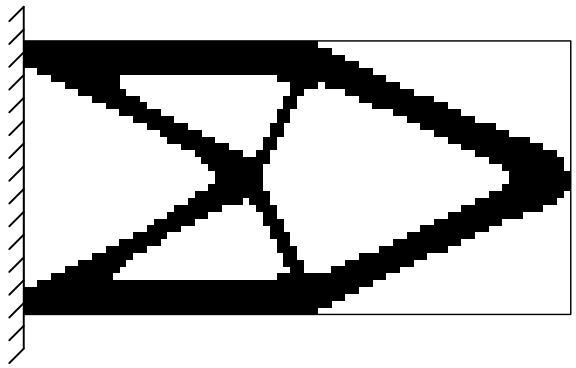


Figure 10.14: Evolutionary structural optimisation based on particle importance.

Particle importance is also a more efficient value to compute than von Mises as can be seen in Table 10.3, which compares computing a single iteration without computing any extra values, against also computing von Mises or Particle importance during the step. It is done during the step as the intermediate values when computing the force on the particle is also used for these measures.

10.5 Non-Linear Material

The material described in Section 7.8 can show how a plastic redistribution may happen within a structure. To illustrate this a T-section with an asymmetric load is analysed, both with a linear isotropic material and without. Using this non-linear material is quite detrimental on the convergence as the material definition is continuously updating.

The test was performed on a T-section with a Young's Modulus of 1000 Pa as seen in Figure 10.15.

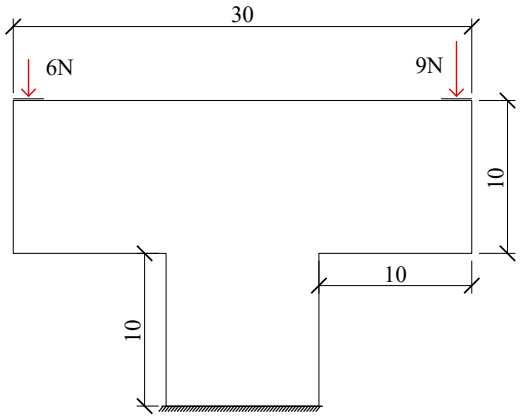


Figure 10.15: Dimensions and load applications for the T-section.

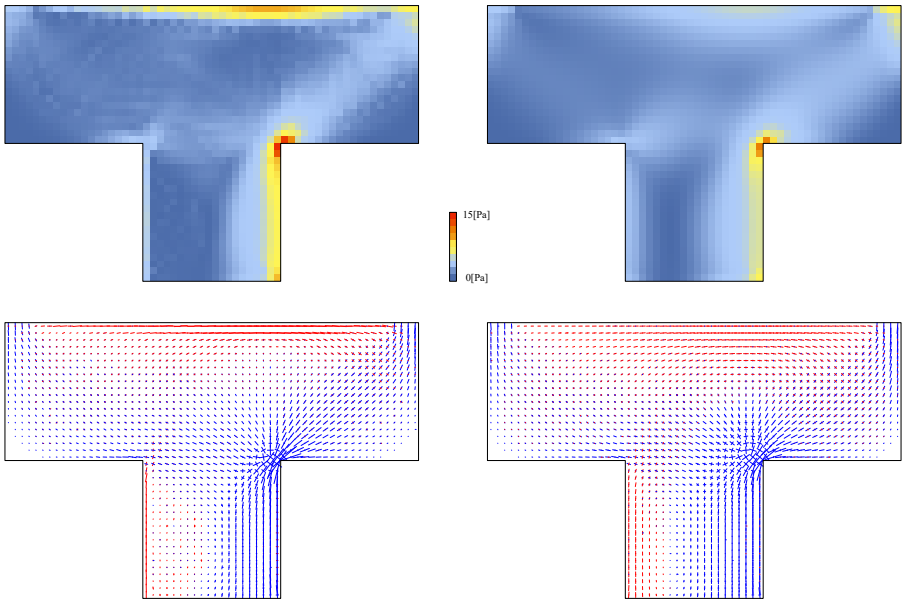


Figure 10.16: The von Mises stresses and the principal stress field for an asymmetrically loaded T-section with the linear elastic solution to the right and the modified material solution to the left.

11

Initial Truss Topology

The method described in Chapter 8 to find an initial truss topology is used for some different cases described below.

11.1 Corbel

Boundary conditions and dimensions of the tested corbel can be seen in Figure 11.1. The specific magnitude of the load and the elastic modulus of the material is not of interest since an elastic analysis is conducted and only the principal stress field is of interest. The effective von Mises stresses on the surface of the corbel can be seen in Figure 11.2 and the principal stress field in Figure 11.3. For von Mises stresses and principal stresses in different sections through the body, see Appendix B.1.

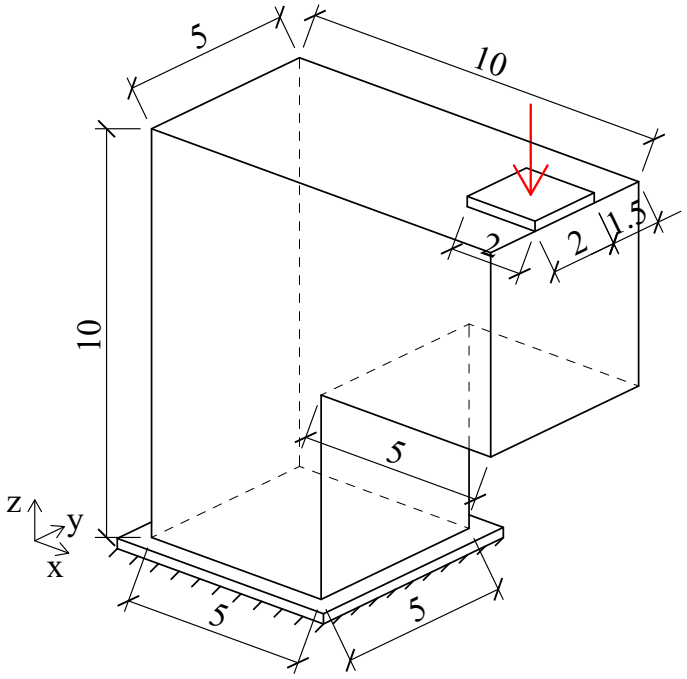


Figure 11.1: Dimensions for the corbel. A point load is acting on the corbel and it is fixed at the bottom.

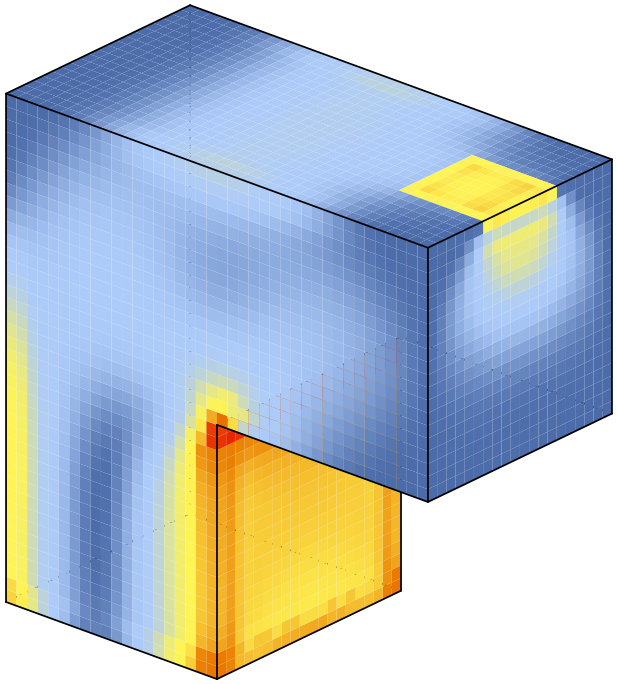


Figure 11.2: Distribution of the von Mises stresses on the surface of the corbel.

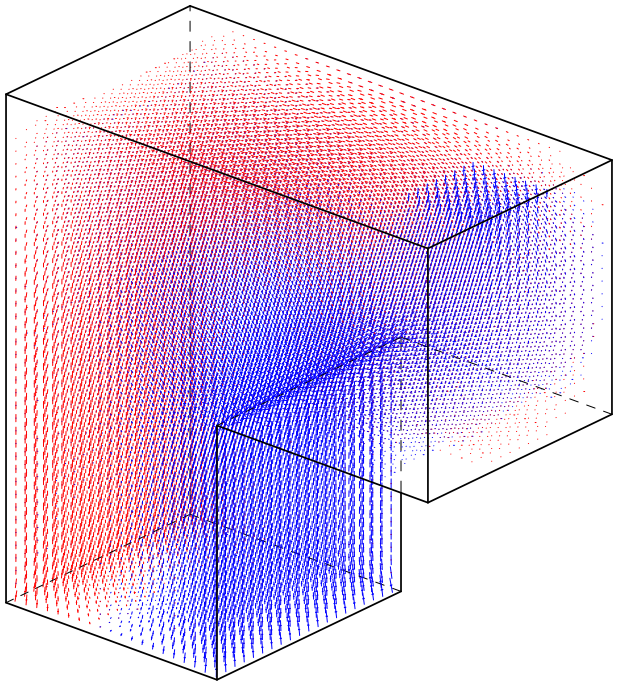


Figure 11.3: Visualisation of the principal stress field in the corbel.

It is chosen to start the tracing of the principal stress curves in phase I from the middle of the load plate. The traced curves from phase I are visualised in Figure 11.4. The local maximum points of von Mises stress are identified on the three curves following the principal stresses in compression. New curves are traced based on these points and with their intersections, the initial truss topology becomes as in

Figure 11.5. The different tolerances used for creating the initial truss are presented in Table 11.1. The node tolerance indicates how close different nodes are allowed to be, the offset tolerance indicates how large offsets are used from the local maximum points of von Mises stress before tracing the Phase II curves and the intersection tolerance indicates how close two curves need to be before it is considered as an intersection.

Table 11.1: The different tolerances used for the corbel.

Tolerances	
Node	3.0
Offset	0.6
Intersection	3.5

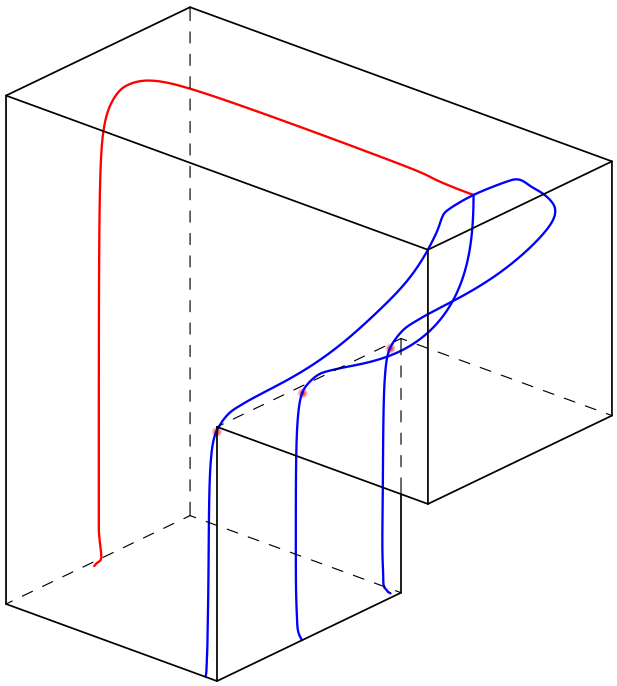


Figure 11.4: Relevant principal stress curves generated in phase I. The local maximum points of von Mises stress along the curves are marked with red.

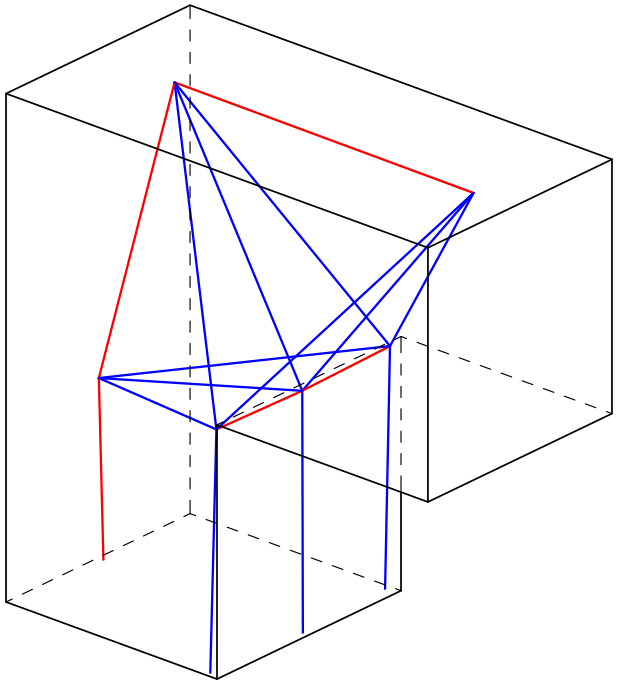


Figure 11.5: Initial truss topology for the corbel.

11.2 Symmetric Pile Cap

The dimensions of the symmetric pile cap and applied boundary conditions are visualised in Figure 11.6. A point load is distributed over a plate on the top part of the pile cap, it is simply supported in the four bottom corners of the cuboid. The effective von Mises stresses on the outside surface can be seen in Figure 11.7 and the principal stress field in Figure 11.8. For von Mises stresses and principal stresses in different sections through the body, see Appendix B.2.

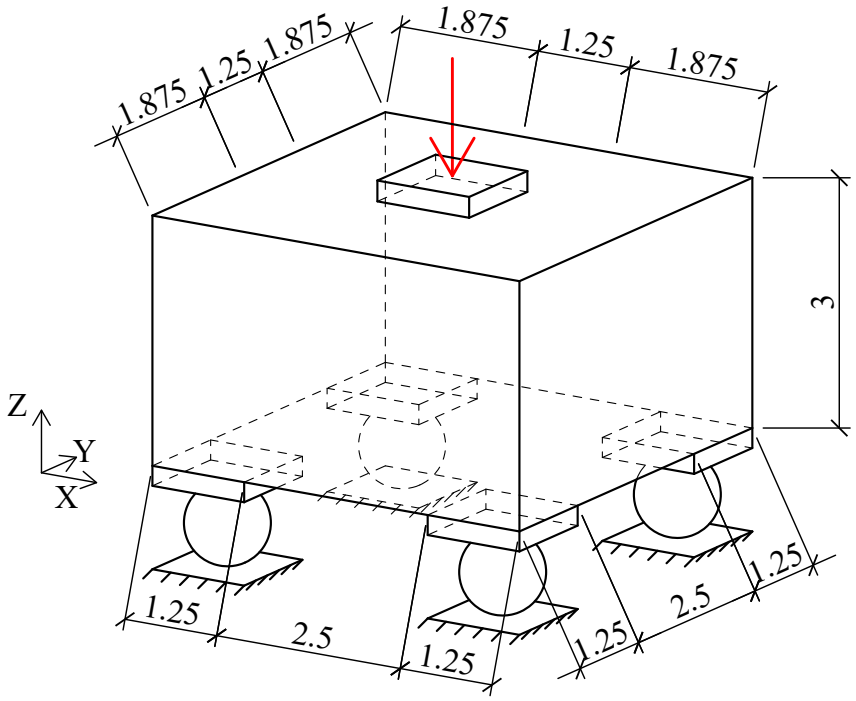


Figure 11.6: Dimensions of the pile cap and boundary conditions. The pile cap is simply supported on four supports with a point load acting on a plate.

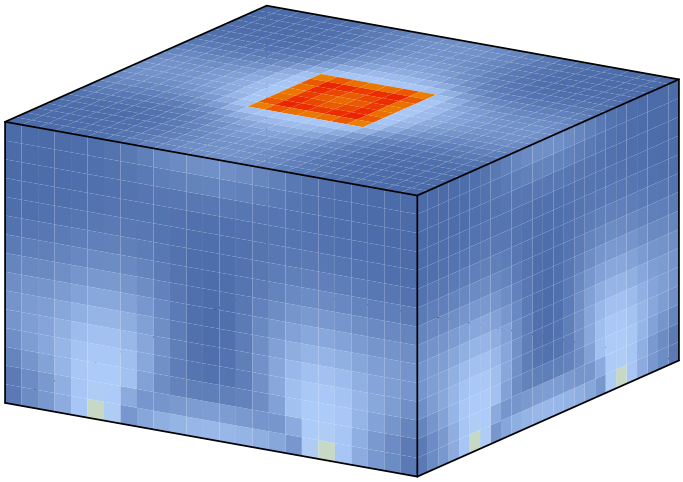


Figure 11.7: Distribution of von Mises stresses on the surface of the pile cap.

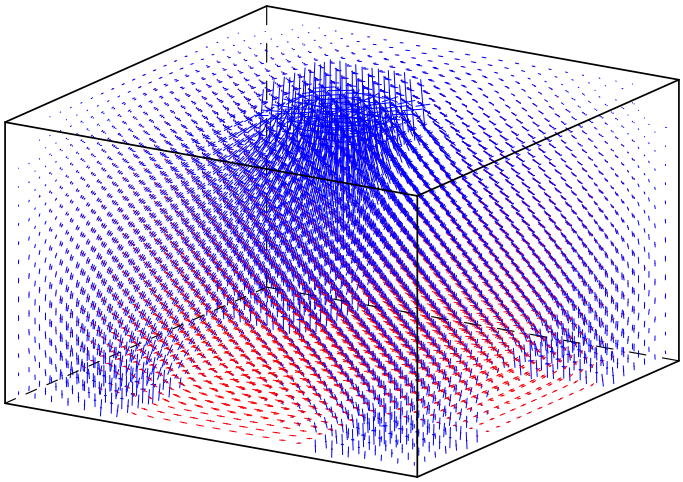


Figure 11.8: Visualisation of the principal stress field for the pile cap.

The load plate is divided into four equal parts and the principal stress curves are traced from the middle point of these parts. The traced curves can be seen in Figure 11.9. The blue curves are traced in phase I and are following the compressive stresses. There are no local maximum points of von Mises stress and therefore phase II is not conducted. In phase III the principal stress curves following the tensile stresses are traced. The resulting initial truss topology can be seen in Figure 11.10. The different tolerances used for tracing the principal stress curves and constructing the initial truss are presented in Table 11.2.

Table 11.2: The different tolerances used for the symmetric pile cap.

Tolerances	
Node	1.0
Offset	0.3
Intersection	1.0

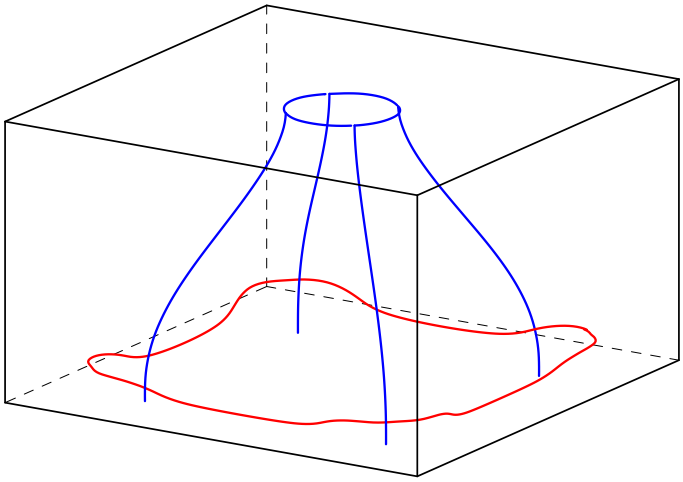


Figure 11.9: Relevant principal stress curves for the symmetric pile cap.

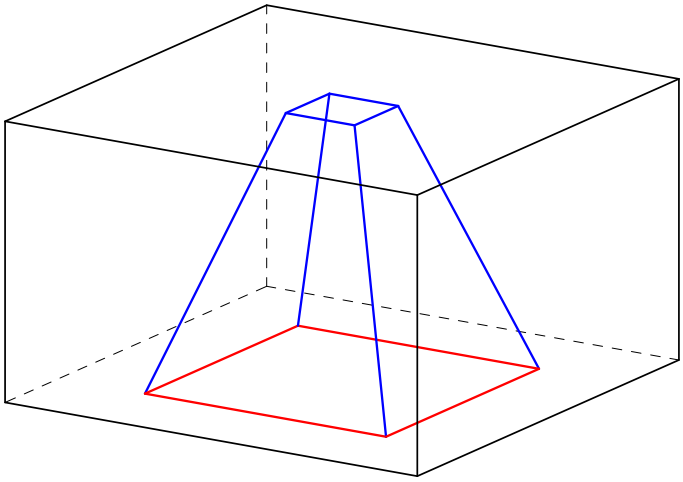


Figure 11.10: Initial truss topology for the symmetric pile cap.

11.3 Cantilever

The dimensions of the cantilever and applied boundary conditions are visualised in Figure 11.11. A point load is distributed over a plate on the top part right part of the cantilever and the surface to the left is completely fixed. The effective von Mises stresses on the outside surface can be seen in Figure 11.12 and the principal stress field in Figure 11.13. For von Mises stresses and principal stresses in different sections through the body, see Appendix B.3.

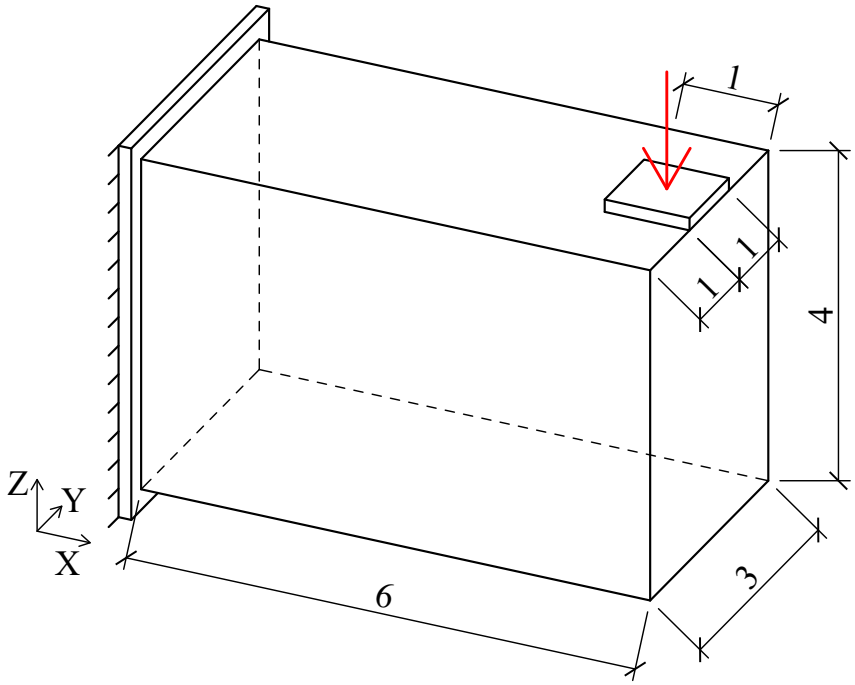


Figure 11.11: Dimensions of the cantilever and boundary conditions. The cantilever is fixed on one side and with a point load acting on a plate on the other.

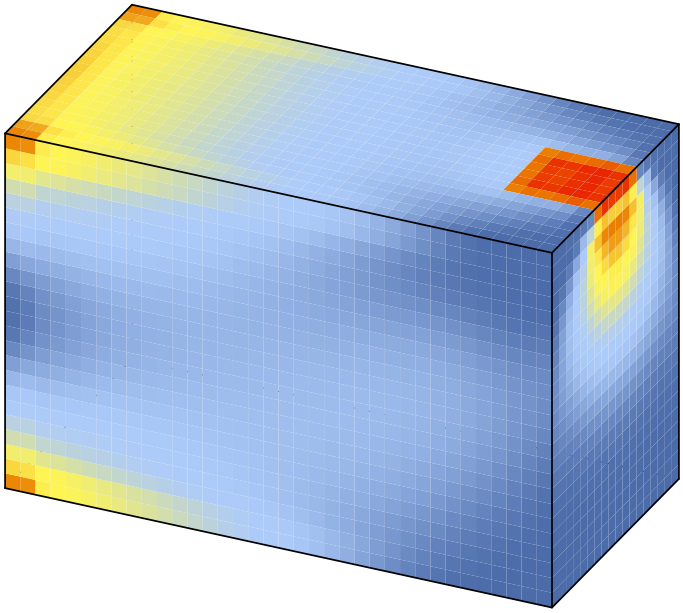


Figure 11.12: Distribution of von Mises stresses on the surface of the cantilever.

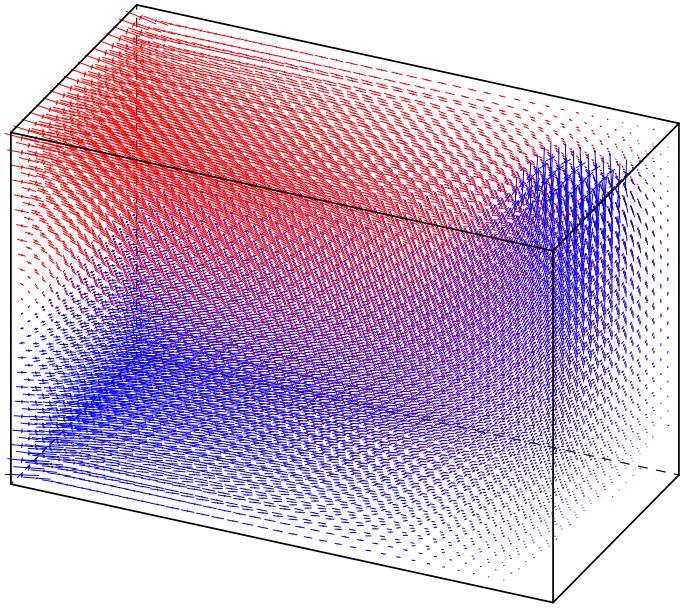


Figure 11.13: Visualisation of the principal stress field for the cantilever.

The principal stress curves are traced from the middle of the load plate and can be seen in Figure 11.14. All of the curves are traced in phase I, there are no local maximum points of von Mises stress and only one fixed support boundary, therefore both phase II and III can be skipped. The resulting initial truss topology can be seen in Figure 11.15. Since the truss generating procedure never enters phase II or III the initial truss topology shown will not change significantly if the tolerances in Table 11.3 would be altered. The used tolerances are nonetheless presented to give the complete picture.

Table 11.3: The different tolerances used for the cantilever.

Tolerances	
Node	0.5
Offset	0.5
Intersection	0.5

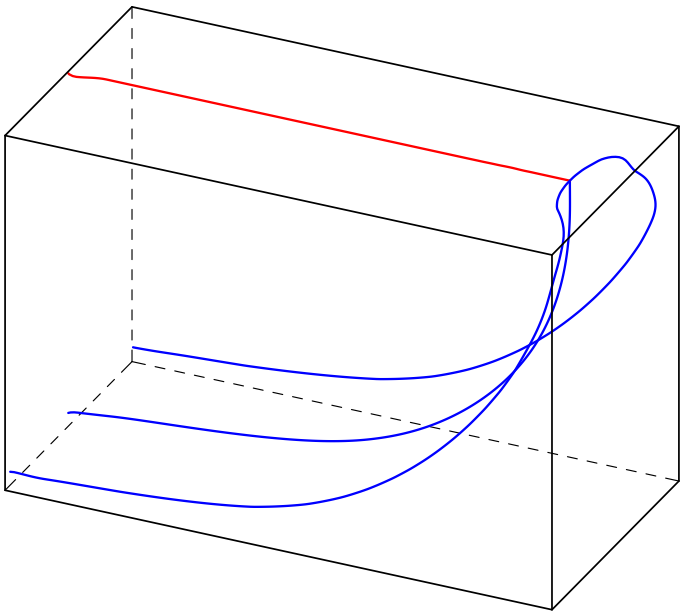


Figure 11.14: Relevant principal stress curves for the cantilever.

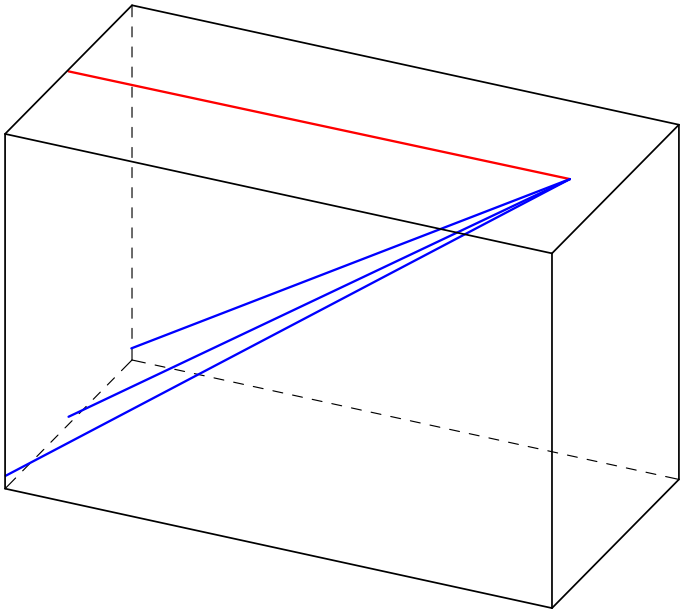


Figure 11.15: Initial truss topology for the cantilever.

11.4 Structure with Cavity

The structure depicted in Figure 11.16 can be described as a cuboid with another, smaller, cuboid subtracted from it. It is supported on three roller supports and with a load acting on the top of a plate. The principal stress field within the structure can be seen in Figure 11.17. For von Mises stresses and principal stress in different sections, see Appendix B.4.

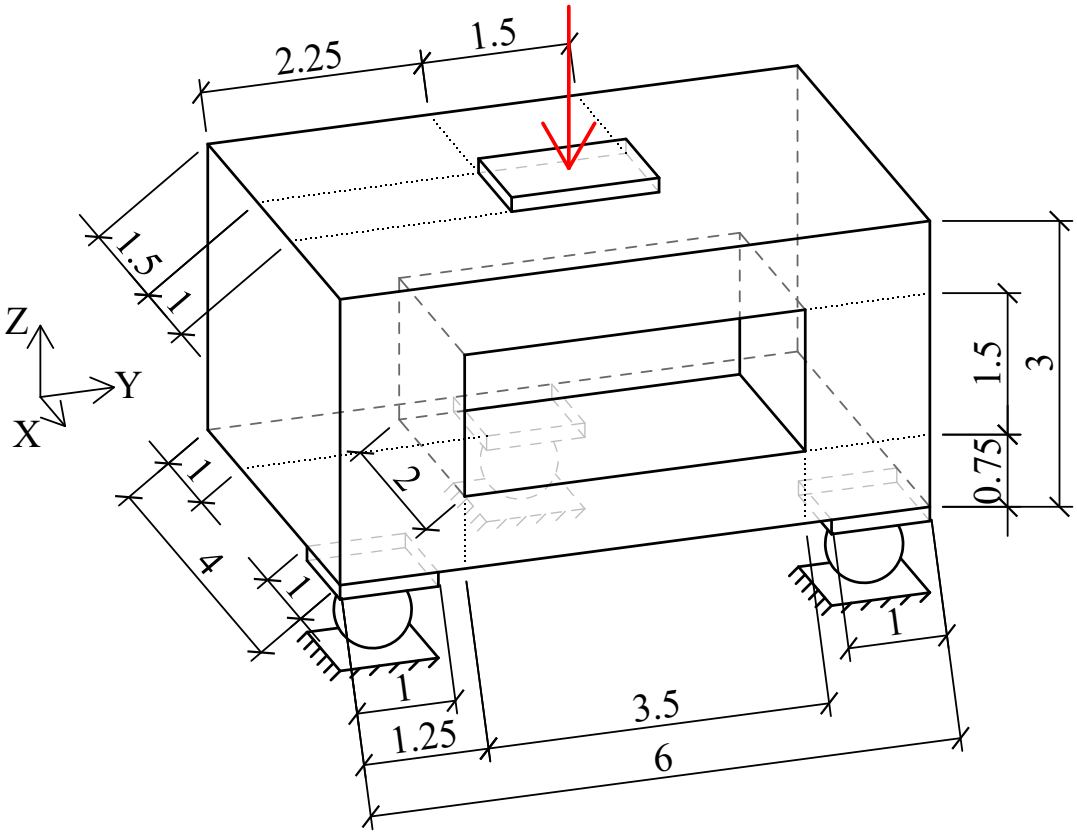


Figure 11.16: Dimensions of the domain and boundary conditions. The structure is simply supported on three supports with a point load acting on a plate.

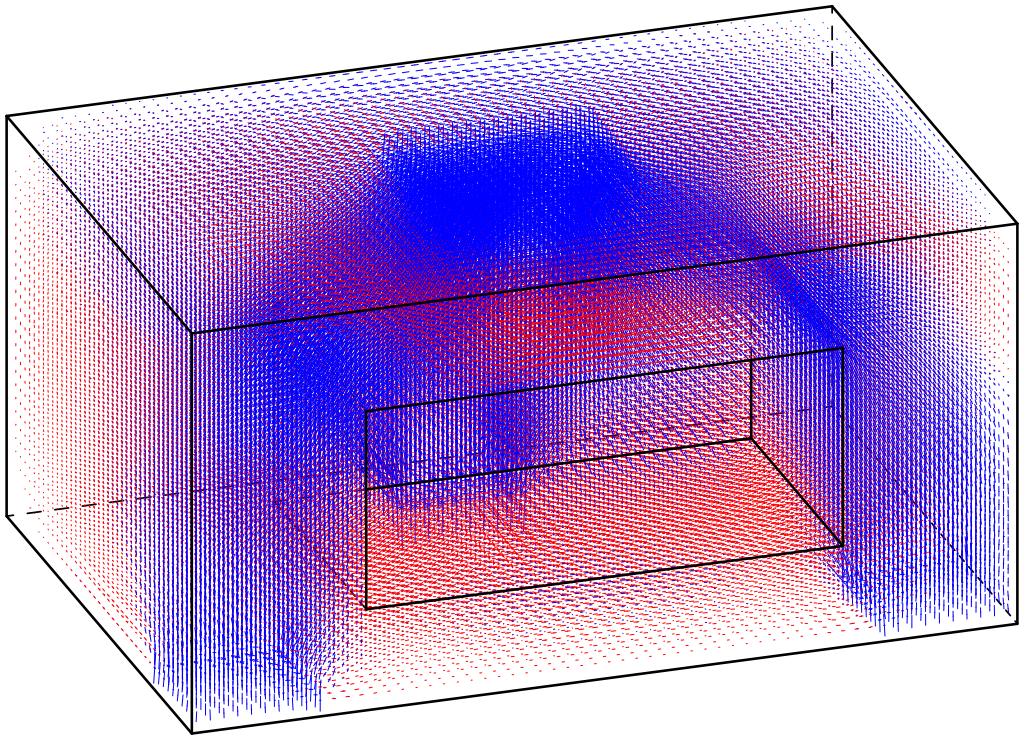


Figure 11.17: Visualisation of the principal stress field within the structure.

The loaded plate is divided into four equal parts where principal stress curves are traced from. Curves are traced in all three phases and the relevant curves can be seen in Figure 11.18. Based on these curves the initial truss topology can be seen in Figure 11.19 and 11.20. In Figure 11.20 the initial truss topology is also stabilised. The tolerances used are presented in Table 11.4.

Table 11.4: The different tolerances used for the structure.

Tolerances	
Node	1.3
Offset	0.2
Intersection	1.3

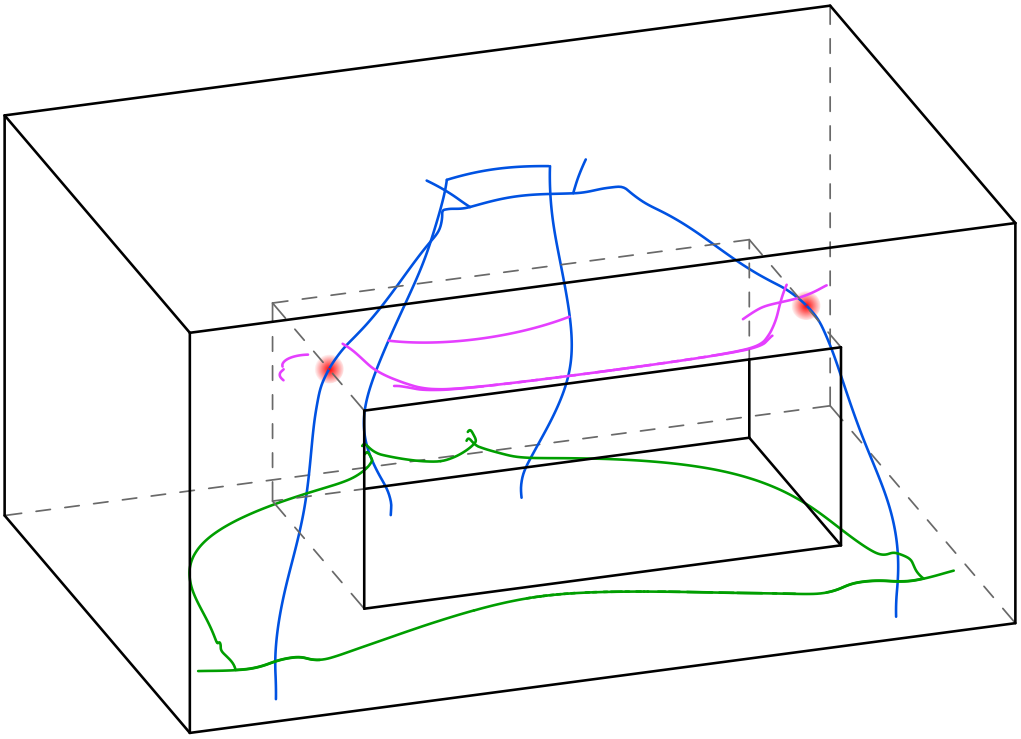


Figure 11.18: Relevant traced principal stress curves for the structure. The blue curves are traced in phase I and their local maximum points of von Mises stress are marked with red dots. The purple curves are traced in phase II and the green in phase III.

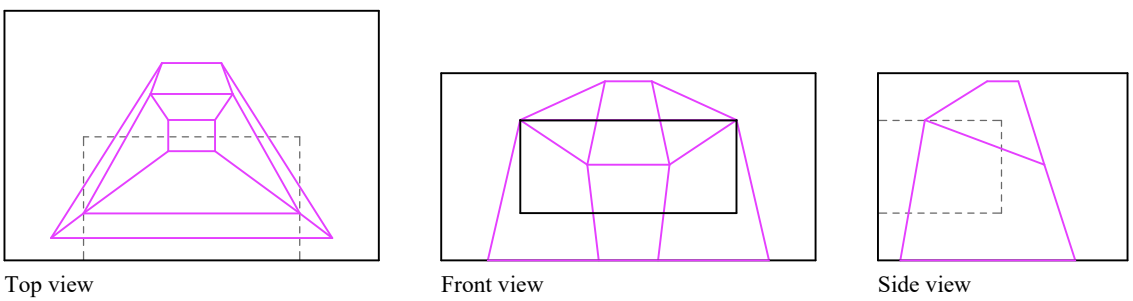


Figure 11.19: Different views over the initial truss topology.

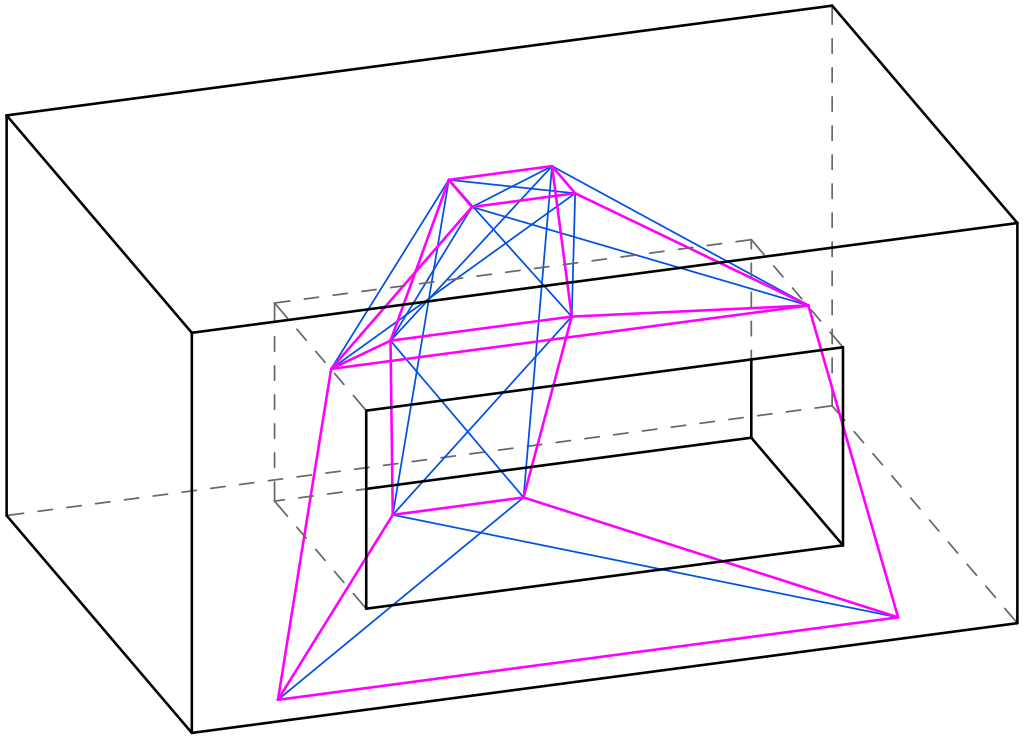


Figure 11.20: The initial truss topology in purple and stabilising members in blue.

12

Truss Optimisation

The truss optimisation procedure from Chapter 9 is applied to some of the presented initial truss topologies with the results of that presented below.

12.1 Corbel

The corbel described in Section 11.1 and the initial truss topology generated is used as a basis for truss optimisation with STM constraints. The dimensions of the corbel can be seen in Figure 11.1 and additional input parameters are presented in Table 12.1, ν is the effectiveness factor as previously defined in 2.2. The goal with the optimisation of the corbel is to minimise the total volume required for the ties.

Table 12.1: Input parameters for the corbel.

Load	5 MN
Concrete	
E_c	35 GPa
f_{ck}	40 MPa
f_{cd}	26.67 MPa
ν	0.84
Reinforcement	
E_s	200 GPa
f_{yd}	435 MPa

Our application requires 1267 iterations before it has reached a converged result. In Figure 12.1 the objective value and penalty during the iterations can be seen. The background of the graph shows if it is conducting a location- or area iteration. The iterations where it modifies the penalties are also shown and it can be seen that it conducts a final STM modification at the end which the solution passes. The objective value is the total reinforcement area as an equivalent concrete material in the entire system. The penalty is then added if the ties and/or struts at the nodes have exceeded their capacity.

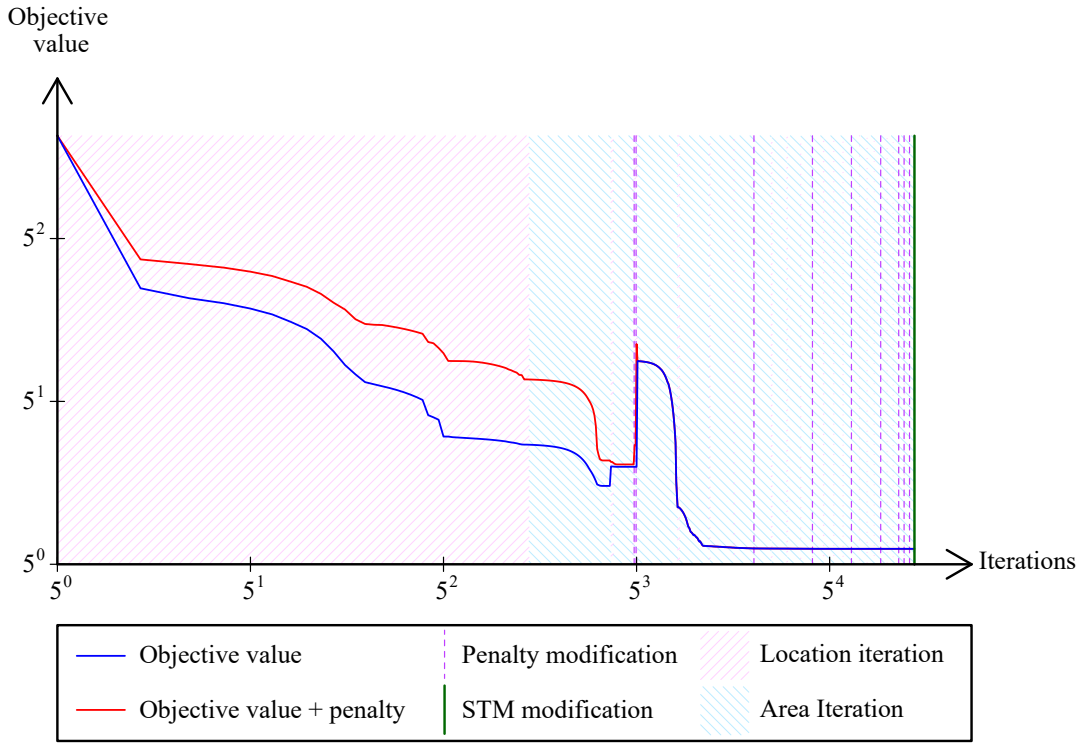


Figure 12.1: Graph over the objective value and penalty during the iterations for the corbel.

The final position of the nodes and the connected members are visualised in Figure 12.2, and the struts, ties and nodes are also annotated. For this structure, the tie and node capacities and their corresponding stress are presented in Table 12.2. Ties T2 and T3 have utilisation ratios very close to their yield strength and none of the struts at the nodes have a utilisation ratio over 90%.

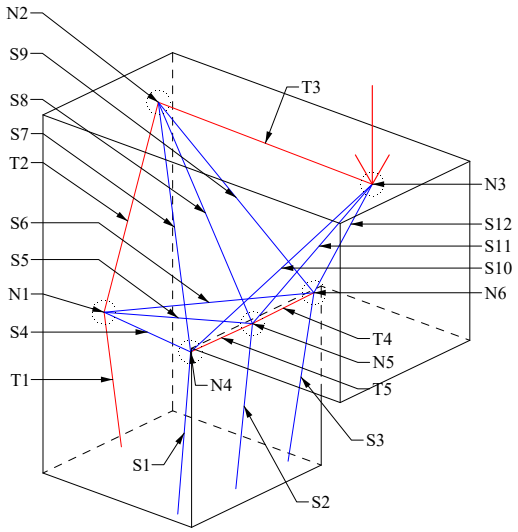


Figure 12.2: Annotation of struts, ties and nodes for the corbel.

The factors k as defined in section 2.1.2 are

$$k_1 = 1.0, \quad k_2 = 0.85, \quad k_3 = 0.75.$$

Table 12.2: Check of ties and nodes for the optimised corbel.

	Force [MN]	**Area [m^2]**	**Stress [MPa]**	**Utilisation**
Ties			$f_{yd} = 435$	
T1	3.8	0.0095	402.2	92%
T2	4.4	0.010	433.1	100%
T3	3.9	0.0091	435.0	100%
T4	0.54	0.011	46.2	11%
T5	0.52	0.0031	173.5	40%
Nodes			$\sigma_{Rd} = \nu f_{cd} = 22.4$	
N1			$\sigma_{Rd} k_3 = 16.8$	
S4	0.46	0.095	4.9	29%
S5	1.1	0.076	15.1	90%
S6	0.45	0.095	4.7	28%
N2			$\sigma_{Rd} k_3 = 16.8$	
S7	0.0	0.0	0.0	0%
S8	5.1	2.3	2.1	12%
S9	0.041	2.0	0.020	0%
N3			$\sigma_{Rd} k_3 = 16.8$	
S10	0.82	9.6	0.085	1%
S11	4.7	0.43	11.0	65%
S12	0.83	9.6	0.087	1%
N4			$\sigma_{Rd} k_2 = 19.0$	
S1	0.50	0.095	5.1	27%
S4	0.46	0.59	0.77	4%
S7	0.0	0.0	0.0	0%
S10	0.82	0.43	1.85	10%
N5			$\sigma_{Rd} k_3 = 16.8$	
S2	7.8	5.1	1.5	9%
S5	1.1	34.3	0.033	0%
S8	5.1	19.1	0.27	2%
S11	4.7	0.99	4.7	28%
N6			$\sigma_{Rd} k_2 = 19.0$	
S3	0.54	0.085	6.4	34%
S6	0.45	0.53	0.84	4%
S9	0.041	0.33	0.12	1%
S12	0.83	0.39	2.1	11%

12.2 Pile Cap

The pile cap first presented in Section 11.2 and the initial truss topology generated there are optimised with STM constraints and with the goal to minimise the total required reinforcement volume (the dimensions of the pile cap were presented in Figure 11.6). The input parameters set for the optimisation are presented in Table 12.3. The total load is divided into four equal loads and placed in the middle of the loaded plate, in the same manner as when tracing the initial truss topology.

Table 12.3: Input parameters for the pile cap.

Total load	20 MN
Concrete	
E_c	35 GPa
f_{ck}	40 MPa
f_{cd}	26.67 MPa
ν	0.84
Reinforcement	
E_s	200 GPa
f_{yd}	435 MPa

The changes of the objective value and penalties during the iterations can be seen in Figure 12.3. The objective value can be interpreted as the total tie volume as an equivalent concrete material. The background of the diagram shows whether a location- or area iteration is done, the locations when the penalty is modified are also marked. The pile cap requires 2849 iterations before it has reached a sufficiently converged solution.

The final structure can be seen in Figure 12.4 with struts, ties and nodes annotated. The truss displayed is, in fact, a mechanism and therefore FE analysis cannot be conducted directly, to overcome this the bottom and top rectangles are prevented from twisting. This is only to have a truss that can be analysed with FE analysis, the extra applied boundary conditions do not generate any forces and therefore do not affect the results. The result of the final tie and node check is presented in Table 12.4. It should also be noted that struts S13 and S14 cross each other without a node being indicated there. That is because the forces and stresses in the struts remain constant over the intersection and only compression forces are present, therefore the capacity at the nodal zone does not need to be reduced. The ties T5 and T6 also cross each other in a similar way that will not reduce the capacity of the structure. However, it needs to be considered when designing the exact reinforcement placement.

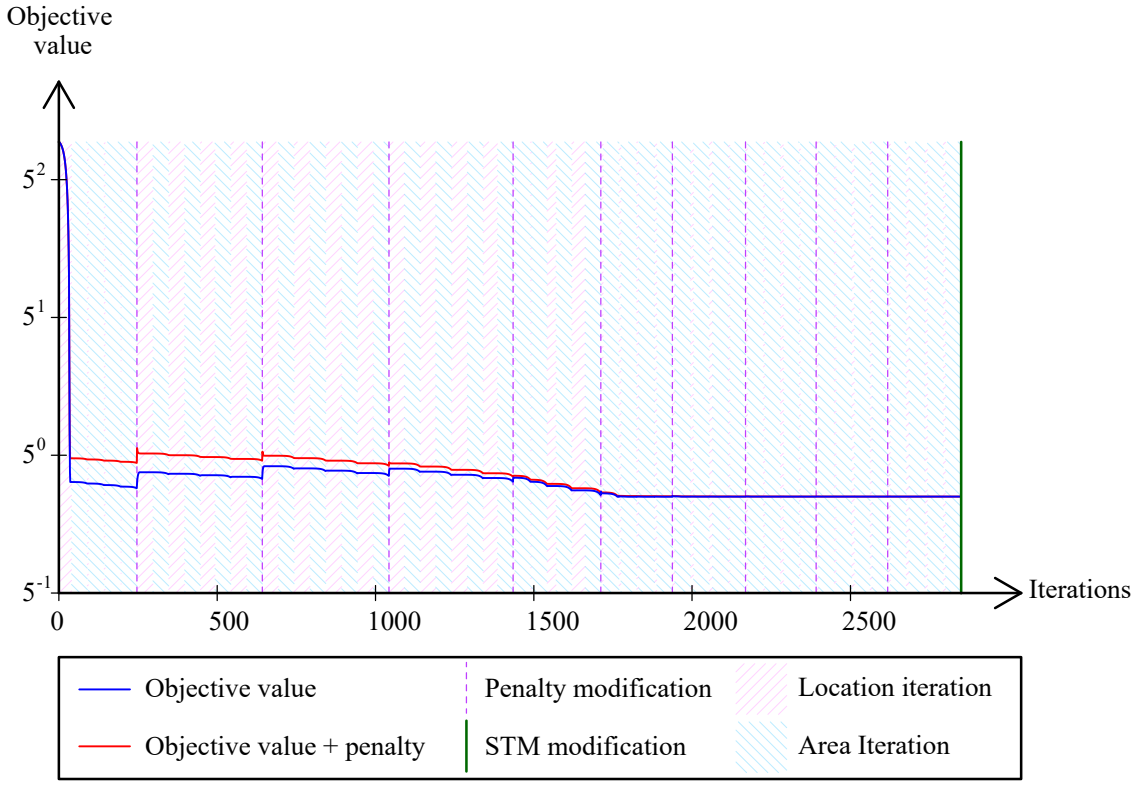


Figure 12.3: Graph over the objective value and penalty during the iterations for the pile cap.

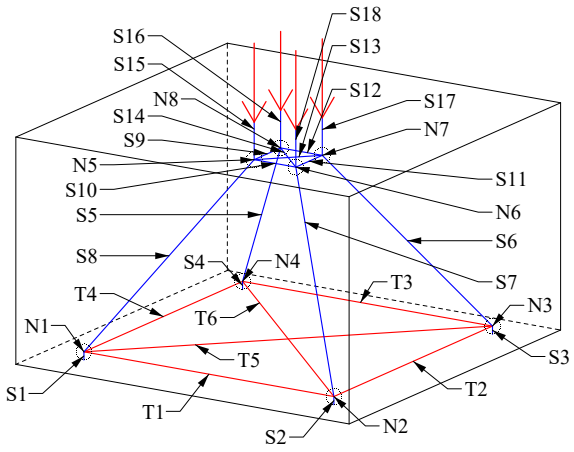


Figure 12.4: Annotation of struts, ties and nodes for the pile cap.

The factors k as defined in section 2.1.2 are

$$k_1 = 1.0, \quad k_2 = 0.85, \quad k_3 = 0.75.$$

Table 12.4: Check of ties and nodes for the optimised pile cap.

	Force [MN]	**Area [m^2]**	**Stress [MPa]**	**Utilisation**
Ties			$f_{yd} = 435$	
T1	2.1	0.0049	435.0	100%
T2	2.1	0.0049	435.0	100%
T3	2.1	0.0049	435.0	100%
T4	2.1	0.0049	435.0	100%
T5	1.4	0.0033	435.0	100%
T6	1.4	0.0033	435.0	100%
Nodes			$\sigma_{Rd} = \nu f_{cd} = 22.4$	
N1			$\sigma_{Rd} k_3 = 16.8$	
S1	5.0	0.54	9.3	55%
S8	6.7	0.44	15.1	90%
N2			$\sigma_{Rd} k_3 = 16.8$	
S2	5.0	0.54	9.3	55%
S7	6.7	0.44	15.1	90%
N3			$\sigma_{Rd} k_3 = 16.8$	
S3	5.0	0.54	9.3	55%
S6	6.7	0.44	15.1	90%
N4			$\sigma_{Rd} k_3 = 16.8$	
S4	5.0	0.54	9.3	55%
S5	6.7	0.44	15.1	90%
N5			$\sigma_{Rd} k_1 = 22.4$	
S8	6.7	18.2	0.37	2%
S9	1.6	4.2	0.38	2%
S10	1.6	4.2	0.38	2%
S13	2.2	5.9	0.38	2%
S15	5.0	0.39	12.8	57%
N6			$\sigma_{Rd} k_1 = 22.4$	
S7	6.7	18.2	0.37	2%
S10	1.6	4.2	0.38	2%
S11	1.6	4.2	0.37	2%
S14	2.2	5.9	0.38	2%
S18	5.0	0.39	12.8	57%
N7			$\sigma_{Rd} k_1 = 22.4$	
S6	3.7	18.2	0.37	2%
S11	1.6	4.2	0.37	2%
S12	1.6	4.2	0.38	2%
S13	2.2	5.9	0.38	2%
S17	5.0	0.39	12.8	57%
N8			$\sigma_{Rd} k_1 = 22.4$	
S5	6.7	18.2	0.37	2%
S9	1.6	4.2	0.38	2%
S12	1.6	4.2	0.38	2%
S14	2.2	5.9	0.38	2%
S16	5.0	0.39	12.8	57%

13
Discussion

The discussion will focus on the results and the research questions presented in Section 1.4 which were:

- Is the proposed program in Figure 1.5 a useful application, in terms of:
  - ease of use?
  - speed?
  - accuracy?
- How large impact do the changes of the peridynamics workflow have in terms of:
  - storing the points and neighbours implicitly by index as opposed to explicitly as points and lists?
  - how to handle the boundary conditions more coherently?
- Optimisation of the STM:
  - is it possible to generate a reasonable truss from the displacement field?
  - can we then apply optimisation to find a more optimal model and reduce the reinforcement amount?

13.1 Changes to the Peridynamics Workflow

In the following sections the different changes implemented to the peridynamics workflow described in Chapter 7 will be discussed.

13.1.1 Voxels

The speed gains over PeriPy presented in Section 10.2.2 should be mainly attributed to the use of image structures with voxels as the pre-computation of $\boldsymbol{\xi}$ only gave a two-fold increase and the only other notable difference is that they parallelise over bonds instead of particles, which they claimed improved their performance. Hence one can say that using voxels gives advantages but it also places limitations since it is a more rigid data structure.

For voxels the memory must be allocated for some cuboid like shape which encases the model, this can lead to a lot of waste in slender structures. This is not seen as a problem as the 3D effects are usually in bulky parts and slender structures are well approximated by other theories. It could however be worked around by splitting it into many simulations all working on some separate cuboid shape, which would

then communicate their results in-between iterations. Such a solution would involve some overhead and code complexity but would also add more flexibility.

The distance between particles must not be as uniform as done in this thesis, but the use of voxels does restrain how variable the distribution of particles can be as the neighbour list is implied. That is, one could add a vector for each particle which offsets its initial position, but it would have the same neighbours and needs to have a position which is consistent with that.

Attempting to expand the peridynamics simulation to include collisions could also be less straightforward as it would include collision detection between different voxel systems which must then be merged. A process which would be made extra difficult if the systems are not aligned.

13.1.2 Speed Comparisons and Limitations

The results in Section 10.2.1 show that the GPU is well suited to handle reasonably sized peridynamics simulations implemented with the voxel concept on a personal computer. The values presented between the CPU and the GPU should be taken in relation to each other however as this works performance improvements have been limited to the concepts outlined in this thesis and is otherwise implemented in a rudimentary manner. The implementations are also implemented in different languages, OpenCL and C# where the biggest difference, apart from CPU vs GPU, is that C# is Just In Time (JIT) compiled through an intermediate language and is a memory managed language. Both of which in general means slower execution speed, but since the JIT compilation is fairly optimized towards simple arithmetic operations within loops and that the memory is pre-allocated such that the garbage collector is not stressed. This comparison is also between performance on a CPU vs a GPU, and while these are on the same system they are different pieces of hardware and there is no reason to believe that some different system should have the same relative performance from their GPU and CPU. All that said it means that the values are highly indicative but subject to a substantial amount of change if it were to be re-implemented in a different language and ran on different hardware.

This work has largely focused on peridynamics for interactive work performed on a personal computer, but for bigger simulations different hardware might be an option. For such simulations parallelism may be better served by CPU or GPU clusters, where the differences between the two alternatives may differ from what was presented here as the circumstances are substantially different.

Section 10.2.1 shows that the CPU has a relative slowdown as the number of particles increases, this is due to that the size of the CPUs cache is limited and has different levels in regards to how quickly the CPU can fetch data from it. This means that as the size of the simulation increases, there is more congestion and cache misses as the data can not remain available. This effect eventually flattens out as the cache is fully utilised and further data is simply filling out the slow memory on the heap.

The GPU however has the inverse effect where increasing the number of particles decreases the relative time consumption. This is because while similar data locality problems plague the GPU as the CPU, the image structure has it more prepared for our usage and not as affected. However, the reason it decreases is that not all possible threads were used for a lower number of particles, and the time for one thread to complete one work item is fairly constant. Increasing the number of particles, therefore, produces better throughput up until the maximum number of threads are in use.

13.1.3 Boundary Conditions

The implicit boundary conditions show promise as they eliminate the peridynamic edge effects otherwise seen. This seems most promising when used with varying particle densities as the boundary regions are not required to have as much resolution.

The edge effect for peridynamic boundary conditions is not usually seen as a large issue since it largely deals with natural crack simulations which both happen away from these boundary conditions and already requires a large number of particles.

The remaining difficulty for the method presented is that when coupled with the volume correction surface correction method, one needs to know the geometry beyond the boundary condition to set those particles' stiffness to then produce the implicit stiffness values for the model. This can naturally be done by either modelling the outside or assuming its shape but would be preferable if the method was wholly contained to the surface definitions.

This thesis does also only deal with implementing these implicit constraints for bond-based peridynamics and an extension for state-based peridynamics would be interesting.

13.1.4 Interactivity

The tool is interactive in the sense that it:

1. Continuously previews the deformations during the simulation.
2. Allows changes to all parameters at run time, including δ and Δ.
3. Has a quick setup time due to the simplicity of the voxels when coming from input meshes and no need to build neighbour lists.

This interactivity is dependent on the number of particles and does not get particularly bogged down until around 250 000 particles. At that point, there is a mix of performance issues concerning Rhino's viewport, the application of boundary conditions and to transfer the data to and from the GPU. These issues have not been given any special consideration as the focus has been the run-time and model build time. Improvements in those areas are still a possibility.

The issues surrounding approximating the fields at the edge of the domain mentioned in Section 7.5 are treated in this thesis such that the solution remains stable. This is a minimum requirement for it to be interactive but there are more possibilities. One could imagine that if the approximation was more exact one could use solutions at low resolution to prime those at higher resolutions. That is, solve a very coarse model, step up the fidelity and repeat. This would allow the large scale deformations to propagate through the model quickly with few particles and that the iterations at higher resolution mostly resolve details in the fields. If coupled with some conditions for particle density convergence one could imagine a system for adaptive resolution.

A promising method for approximating continuations of fields in 3D seems to be the reproducing kernel method. Furthermore, a thought for particle convergence is comparing the divergence of the stress tensor to the applied load, as a stress field which is consistent with the model is usually sought.

13.1.5 Particle Importance

Particle importance allowed for faster convergence and more even results which makes it a promising measure for evolutionary structural optimisation when using peridynamics.

This can be compared against the von Mises stress which gives a much spottier result. Since stress is a less straightforward measure in peridynamics there are different ways it can be calculated and there may be other ways which would provide consistency, but given particle importance's simplicity, it is hard to imagine such a method to have comparative performance. Currently, the method to calculate the stress described in Section 3.8 seems to suffer somewhat from that the changes to the system disturb the linearity assumptions and perhaps also that the particles' surface correction is not updated due to how the material is removed. The particle importance on the other hand provides a more direct connection between what removing a particle may do for the system without involving physical assumptions.

13.1.6 Non-Linear Material Behaviour

Setting the material as described in Section 7.8 does provide insight into how a material might react during something akin to plastic redistribution, even if the tensile reinforcement here is applied by the material itself. This knowledge is illuminating but the final workflow presented has no direct use for it and a deeper investigation has not been conducted.

In regards to connecting this theoretical material to the actual material at hand one would need to do further investigations.

1. Could the final converged solution be representing a reinforced concrete structure?
2. Deeper check on the 3D behaviour of such a material.

3. Attempt to more closely resemble concrete and steel by using curves with correct stiffness for the two materials in each part.
4. Include that both steel and concrete are non-linear near yielding.

Furthermore, the convergence rate of this kind of simulation should be investigated as the step sizes (Section 3.5) and damping coefficients (Section 3.7) used here are determined assuming linear elasticity.

13.2 Initial Truss Topology

The idea to construct a truss topology based on the principal stress field and the method outlined in Chapter 8 seems to be working well and can generate a topology suitable for STM. Compared to evolutionary structural optimisation the principal stress-based approach directly generates a truss topology with nodes and members in between which is an advantage since it can then be directly connected to an STM optimisation. However, it can still be useful to run a structural optimisation since it can be very interesting to compare those results with the initial truss topology generated by principal stresses. It should also be noted that evolutionary structural optimisation is an iterative method and it can require quite a lot of iterations before it has reached a solution, especially in 3D.

In STM the truss used to model the behaviour of the structure should also be closely related to the linear stress state since that will decrease the need for plastic redistribution. By using a topology based on the principal stress field the truss will, at least to some degree, resemble the linear state. However, there is no guarantee that the traced curves will always follow the highest concentrations of principal stresses since that depends on the chosen start point and direction. To some degree, plastic redistribution will always occur before the stress state reaches that studied in the STM. The generated topologies in Chapter 11 can all be seen as approximations of the principal stress field in the linear state and therefore also in varying degree be suitable as an initial topology for STM. It should also be noted that it is difficult to quantify how much a discretised truss deviates from the linear stress state, in 3D it becomes even harder.

The initial truss topology generated for the cantilever in Section 11.3 is an example where quite a lot of plastic redistribution needs to take place before the stress state approximated by the truss will be reached. A better approximation would be to introduce a kink in the compression struts which would also lead to the need for shear reinforcement will be included in the truss model. The generated truss topology is too simple and more nodes and members would need to be introduced for it to be an appropriate approximation of the linear stress state and a good STM. This could be done according to the principles outlined in Section 4.3 which introduces a systematic approach on how to expand the topology.

Regions with large shear will be seen in the principal stress field where the tensile and compressive stresses are approximate of the same magnitude, for example, see

Figure B.20 in Appendix B.3. When curves are traced in these regions they will either follow the compressive or tensile stress direction, the traced curves will never directly flow in the direction of the shear stresses. This is quite obvious since the curves are traced in the principal stress field but still means that it will never find a member in the direction of the shear stress. In STM applications it is quite common with ties in that direction that represent the shear reinforcement. However, this does not necessarily cause any problems since the direction of these ties can be changed at a later stage in the optimisation procedure.

The results presented in Chapter 11 also depend on the chosen tolerances used for the different cases. For example, how close two curves should be in 3D for it to be considered as an intersection varies for the different cases and is specific for each case, it would be difficult for the application to choose this value by itself. How large each node is considered to be is also highly case-specific and needs to be chosen by the user by studying the principal stress field. If these parameters are not chosen in a good manner the initial topology might become very chaotic with a lot of unnecessary nodes and members, or it might completely overlook crucial members needed in the truss.

The user also needs to decide on how the load should be divided. For loads acting as point loads or on very small areas, there might be no need for a subdivision and the principal stress curves could be traced from only one point in the centre of the loaded area. However, if the load is acting over a larger area a subdivision is necessary to properly capture the behaviour of the structure, as required for STM. The decided number of divisions also depends on the support conditions since they have a very large impact on the flow of forces. To help the user decide it could be useful to study the principal stress field and simply trace some curves in it and test different positions, without doing any additional phases to construct the initial topology. That is a great way of getting a better understanding of the studied structure.

The modifications presented in Section 8.3 to the method presented in [13] for creating an initial truss topology are working well and making the method more general. It is now well suited for handling roller supports which are quite common in structural applications. Searching for maximum von Mises stress points along the traced curves instead of the entire regions also makes the results more stable and the locations of the inserted nodes more appropriate. These nodes due to high concentrations of von Mises stress must be found since otherwise the generated truss topologies might be far too simplified.

13.3 Strut-and-Tie Model Assumptions and Limitations

The ties in the STM are considered in a simplified way where the sizes of the tie are not directly related to a specific reinforcement distribution but to the volume needed if the ties are considered with an equivalent concrete stiffness with the α

factor, $\alpha = E_s/E_c$. When checking the capacities of the ties the actual steel stresses are considered. Therefore the application does not consider the exact reinforcement distribution and is only suitable in preliminary design situations. The reinforcement does affect the nodal zones that it crosses and this will have an effect on the final design. However, the application does still provide a sketch of where the reinforcement is needed and a rough estimate in what volumes.

As presented in Section 2.1.1 there are recommendations regarding the angles between struts and ties, as well as deviation angles of concentrated forces. These are not hard constraints and certain deviations from the recommendations can be acceptable depending on the model and the forces in the considered members. Therefore these angle limitations are not part of the optimisation procedure even though it could be possible to find a suitable gradient for them. It should also be noted that it is quite trivial for the user to check the angles in the final solution and decide whether they are acceptable or not.

In Section 2.1.2 certain cases are outlined when the compression capacity can be increased by 10% and the capacity can be increased quite a lot when a node is subjected to triaxial compression. Neither of these increases is considered in the application, mainly because it would be harder for the optimisation procedure to find a converging solution and to distinguish between all of the different cases. This will also result in solutions a bit more towards the conservative side.

The difference between the centroid of tapered struts and the system line in the STM (described in Section 2.3) and how that affects have not been investigated further in this thesis. It will generate a certain error in the results but the authors still deem it as an acceptable solution, but this is another argument that the application should only be used in a preliminary design situation. Other methods propose different ways how to expand the STM from 2D to 3D but they have not been implemented in our application. It would be interesting to implement another method and compare the solutions generated by them. The benefits of the chosen method are the simplicity of approximating the nodal zones as cuboids and the struts cross-sections as projections of these cuboids.

13.4 Optimisation

The truss is optimised using steepest descent (Section 5.1) which was largely chosen due to its simplicity, but that does lead to drawbacks where the system becomes preoccupied with following local variations, leading to a large number of steps. This can be clearly seen by optimising over the Rosenbrock function where it will zig-zag. The other part of this implementation is the Armijo step rule (Section 5.2) which approximates a good step length by taking trial steps. This is based on that evaluating a function is usually easier than taking its derivative. This is still true here, but the main computational effort is spent on the LDL decomposition, which only needs to be done once, and not when computing the derivative. This implies that

one could either use more information than just the function value during the trial step for relatively low extra effort or that some other scheme could be more suitable.

There are also other optimisation schemes to consider such as Newton's method which fits a second-order approximation and thus extracts more information and can take further step lengths. It does however rely on computing the inverse of the functions hessian matrix (on index notation $\frac{\partial^2 f}{\partial x_i \partial x_j}$) which has the same size as the stiffness matrix, but no guarantees on being sparse. Hence such a solution would need to both store a full matrix and solve another system of equations per iteration.

This manner of optimisation will find a local minimum meaning that there may exist some other configuration of the base topology which would produce a lower objective value. The strictly lowest objective value is however not necessarily what we are after, as the STM requires that the linear stress state can smoothly transform to the plastic stress state of which the truss approximates. By initialising the truss geometry from the linear stress field and finding a local optimum from there, the solution is closer to the linear state and there is greater confidence that the structure can handle all the intermediate states before reaching this point.

Setting up a model to run includes all the regular considerations regarding FEM with the addition that one must also constrain in which directions the initial structure is allowed to evolve and that the optimisation process might attempt to use stabilising constraints to carry the load as it would be more efficient. This process where one both must place displacement and location constraints which may affect each other is not as clear as it could be.

13.4.1 Strut-and-Tie Model

The results from the truss optimisation presented in Chapter 12 are very promising and it can be seen that the objective value is decreasing for the tested cases and that the penalties are dealt with. The ties and nodes do not exceed their capacity in the final STM check and the solution is, therefore, acceptable. However, it should again be noted that the application is only suitable in preliminary design since the exact reinforcement layout is not decided. It would be hard to automatically decide on the reinforcement layout since that depends on a lot of different parameters and it would be even harder to come up with a suitable gradient for different layouts and how they affect the solution as a whole.

The utilisation ratios obtained after optimisation are really good and indicate that the optimisation procedure is working well. The ties often become utilised close to 100% and the node checks usually stay below 90%. It is chosen that the utilisation ratio of the nodes should not exceed 90% since a ductile failure mode is wanted, initiated by the yielding of the reinforcement. The nodes sometimes have a very low utilisation ratio but that should not be seen as that there is a problem with the optimisation. That is because the volume of concrete is not changed during the iteration but stays the same as input by the user. However, it could indicate

that the entire concrete volume could be optimised by changing the geometry. Our application does not provide any direct suggestions on how to perform this change, that needs to be investigated by the user if such a change is wanted.

Since the optimisation procedure utilises a FE analysis it is important that the truss is considered stable and not a mechanism. This can be solved by introducing diagonals to the initial structure, but that should be done with caution since the optimisation procedure could then be prevented from finding the wanted solution since then some of these diagonals might be forced to become tensile and therefore preventing the wanted solution by increasing the tie volume. Instead, boundary conditions can be placed on certain nodes to make the truss stable. It should then be made sure that these boundary conditions do not affect the structure in any other way and that they do not carry any forces.

This is largely a problem in that any tensile force is now considered to require reinforcement, which is a faulty assumption in that concrete can take some tensile force and that due to minimum reinforcement demands no extra reinforcement would be added. But even disregarding that there is still a problem which can be illustrated by the pile cap in Section 11.2. Removing the diagonal ties would pass more load through the other ties, but the total reinforcement volume would presumably decrease. The model does however not have any way of making a member disappear as the geometry of the problem enforces that they are stretched. Since they are stretched they are ties and therefore, according to the model, must be thick enough that they do not break. Some mechanism to differentiate between members and stabilizers and ways for the model itself to mark them as such would be needed.

The true inclusion test for members (Section 9.5.2) happen in the extra constraint test described in Section 9.5. This means that the continuous optimisations only care if the nodes are within the volume. This is not a problem for convex shapes but could be problematic otherwise. It is however not certain that it will cause problems as can be seen in the corbel example 12.1 which is a concave shape.

The element containment test should also be developed as it currently can evaluate if it is intersecting well, but determining what size it should have is more difficult. That problem is largely in that it is not a local problem where the question to answer is what sizes should the nodes have to produce the most contained area for this element without disturbing all the other members connecting to them in a bad way. This is an extra bad problem if one considers that those elements may also be intersecting the boundary. It would frankly be an optimisation problem in itself only that the intersection checks are still difficult to deal with despite the simplifications and frameworks presented in Appendix A.4.

14
Conclusion

The use of voxels in the peridynamics workflow has proved useful in terms that each particle's neighbour list is implied and that the relative displacement vectors in the undeformed configuration, $\boldsymbol{\xi}$, can be pre-computed. By implementing the peridynamics solver so that it utilises the GPU the run-time of the simulation is drastically decreased on a personal computer. The GPU implementation also works well with the voxel concept.

The use of the implicit boundary conditions is working well and the results are promising. By using implicit boundary conditions edge effects close to the boundary are eliminated, which means that a coarser particle density can be used. Otherwise, the particle density might need to increase around boundaries which would make the simulation slower since the application only can handle a constant particle density within the structure. The boundary conditions are also easy for the user to apply and they can be changed during the run-time of the simulation and directly see how that would affect the structure.

From the principal stress field generated by the peridynamics solver, a reasonable initial truss topology can be created for most structures. The different tolerances for the generating procedure need to be chosen carefully since that will have a large impact on the final structure. It is also important that the generated truss topology is studied carefully to verify that it applies to the current structure and is a good approximation of its linear stress state.

To optimise a truss there is need to both define boundary conditions for the FEM model and set valid constraints for the optimisation model which can be difficult since both regard locations in different manners and their interaction can be subtle. The STM is considered in a simplified way in our application where the actual placement of the reinforcement bars is not considered. It still, however, indicates the position of the reinforcement ties and their required steel area. The truss is optimised with the steepest descent method where the step length is decided with the Armijo step rule. When applying the optimisation procedure it can be seen that the total reinforcement volume needed is decreasing. The utilisation of the ties is in general also very high, close to 100% and for the struts, it varies quite a lot but in general does not exceed 90%, as wanted. While the model as defined works, more work is needed to improve the model and incorporate finer parts of STM and this should largely be seen as a proof of concept. The resulting truss and distribution of the reinforcement will be useful in an early design process but since the exact

placement of the bars is not decided a more detailed analysis will be needed at a later stage.

Despite the simplifications described we still believe that the application can be very useful in the early stages of a design process to estimate where reinforcement is needed and approximately in what volumes. The application can also be very helpful in terms of understanding the structural behaviour and the flow of stresses within a structure. For example, simply tracing principal stress curves from an arbitrary point within a structure can be quite illuminating. The non-linear material behaviour investigated also gives some understanding regarding how plastic redistribution works but the material behaviour should not be directly translated to reinforced concrete. The concept of particle importance is also very interesting and is a measure which is easy to calculate and quickly indicates the force-flow within the structure. It seems to be very useful within the field of evolutionary structural optimisation. All in all, the application is a good tool to better understand the three-dimensional behaviour of reinforced concrete structures in the ultimate limit state.

15
Further Research

During the development of this thesis several different aspects have come up that it would be interesting to study further which are presented below.

Voxels:

- Variable size and distribution of particles within the voxel concept.
- Segmentation of the voxel cuboids to deal with slender structures which still has dense particle distribution within it.
- Use the SDF concept for a smoother boundary definition with few particles, see Figure A.2.

Peridynamics:

- Extend the implicit boundary constraints to state-based peridynamics.
- Better field approximations from data points, specifically outside of the known domain, and see if that could be used to dynamically refine the number of particles.
- Investigate A Priori measure of particle resolution related to measured stress (divergence of stress tensor = applied load).
- Investigate other surface correction techniques.

Principal stress based truss generation:

- Implement the refinement of the truss topology in 3D (see Section 4.3) to better model the linear stress state.
- Possibility to handle multiple load cases.
- Potential combination of evolutionary structural optimisation and principal stress based design to generate more accurate truss topologies.

Optimisation:

- Gradient based optimisation methods for more general FE models.
- Other optimisation methods.
- Find an efficient formulation for the true node sizes during optimisation.
- Optimisation Element containment, such that it works during the continuous optimisation for non-convex shapes.
- Not using a direct solver for the FEM system while optimising as the separate solutions should be similar.
- Possibility to handle multiple load cases.
- Include the actual reinforcement layout in the STM checks.

Bibliography

[1] G. Edefors and D. Jonsson, *Interactive Design Using Peridynamics Development of a Design Tool for the Exploration of Structurally Efficient Geometries (MSc thesis)*, Gothenburg, 2021. [Online]. Available: https://hdl.handle.net/20.500.12380/303921

[2] J. Schlaich, K. Schaefer, and M. Jennewein, "TOWARD A CONSISTENT DESIGN OF STRUCTURAL CONCRETE." *PCI Journal*, vol. 32, no. 3, pp. 74–150, 1987. [Online]. Available: https://www.researchgate.net/publication/248122582_Toward_a_Consistent_Design_of_Structural_Concrete

[3] B. Engström, "Design and analysis of deep beams, plates and other discontinuity regions," Tech. Rep., 2015.

[4] J. Schlaich and K. S. S. Engineer, "Design and detailing of structural concrete using strut-and-tie models," *Structural Engineer*, vol. 69, no. 6, pp. 113–125, 1991. [Online]. Available: http://www.pgmec.ime.eb.br/~webde2/prof/ethomaz/bloco_sobre_estacas/biela_tirante.pdf

[5] "SVENSK STANDARD Eurokod 2: Dimensionering av betongkonstruktioner-Del 1-1: Allmänna regler och regler för byggnader Eurocode 2: Design of concrete structures-Part 1-1: General rules and rules for buildings," Tech. Rep., 2008. [Online]. Available: www.sis.se

[6] S. A. Silling, "Reformulation of elasticity theory for discontinuities and long-range forces," *Journal of the Mechanics and Physics of Solids*, vol. 48, no. 1, pp. 175–209, 1 2000.

[7] E. Madenci and E. Oterkus, *Peridynamic Theory and Its Applications*. Springer, 2014.

[8] R. A. Gingold and J. J. Monaghan, "Smoothed particle hydrodynamics: theory and application to non-spherical stars," *Monthly Notices of the Royal Astronomical Society*, vol. 181, no. 3, pp. 375–389, 12 1977. [Online]. Available: https://academic.oup.com/mnras/article/181/3/375/988212

[9] S. A. Silling, M. Epton, O. Weckner, J. Xu, and E. Askari, "Peridynamic States and Constitutive Modeling," *Journal of Elasticity 2007 88:2*, vol. 88, no. 2, pp. 151–184, 7 2007. [Online]. Available: https://link.springer.com/article/10.1007/s10659-007-9125-1

[10] M. Hillman, M. Pasetto, and G. Zhou, "Generalized reproducing kernel peridynamics: unification of local and non-local meshfree methods, non-local derivative operations, and an arbitrary-order state-based peridynamic formulation," 2019. [Online]. Available: https://doi.org/10.1007/s40571-019-00266-9

[11] J. Olsson, M. Ander, and C. J. K. Williams, "The Use of Peridynamic Virtual Fibres to Simulate Yielding and Brittle Fracture," *Journal of Peridynamics and Nonlocal Modeling*, vol. 3, no. 4, pp. 348–382, 12 2021.

[12] Y. M. Xie, X. Y. Yang, Q. Q. Liang, G. P. Steven, and O. M. Querin, *Evolutionary structural optimization.* American Society of Civil Engineers, 2002.

[13] T. H. Kwok, Y. Li, and Y. Chen, "A structural topology design method based on principal stress line," *Computer-Aided Design*, vol. 80, pp. 19–31, 11 2016.

[14] "Michell structures," 12 2021. [Online]. Available: https://en.wikipedia.org/wiki/Michell_structures

[15] G. Chantelot and A. Mathern, "Strut-and-tie modelling of reinforced concrete pile caps," Ph.D. dissertation, 2010.

[16] D. Bohman and N. Chaudhry, *Enhanced 3D strut-and-tie method for design of discontinuity regions Validation of the method through a case study (MSc thesis)*, 2021.

[17] Q. V. Le and F. Bobaru, "Surface corrections for peridynamic models in elasticity and fracture," *Computational Mechanics*, vol. 61, no. 4, pp. 499–518, 4 2018. [Online]. Available: https://link.springer.com/article/10.1007/s00466-017-1469-1

[18] F. Bobaru, J. T. Foster, P. H. Geubelle, and S. A. Silling, *Handbook of peridynamic modeling*, 2016.

[19] B. Nikolic, "Computational Methods of Physics." [Online]. Available: https://www.physics.udel.edu/~bnikolic/teaching/phys660/numerical_ode/node5.html

[20] A. S. Fallah, I. N. Giannakeas, R. Mella, M. R. Wenman, Y. Safa, and H. Bahai, "On the Computational Derivation of Bond-Based Peridynamic Stress Tensor," *Journal of Peridynamics and Nonlocal Modeling 2020 2:4*, vol. 2, no. 4, pp. 352–378, 7 2020. [Online]. Available: https://link.springer.com/article/10.1007/s42102-020-00036-9

[21] N. Andréasson, A. Evgrafov, M. Patriksson, and Holmbergs i Malmö AB, *An introduction to continuous optimization : foundations and fundamental algorithms*, 2016, vol. 3.

[22] "Cornell Virtual Workshop: Synchronization Overhead." [Online]. Available: https://cvw.cac.cornell.edu/parallel/synch

[23] "Graphics processing unit - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Graphics_processing_unit

[24] A. Munshi, "The OpenCL Specification," Tech. Rep., 2012.

[25] "NVIDIA OpenCL Best Practices Guide Version 1.0 NVIDIA OpenCL Best Practices Guide," 2009.

[26] M. Giles, "Lecture 3: control flow and synchronisation Warp divergence."

[27] "Flood fill - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Flood_fill#Span_Filling

[28] G. C. Ganzenmüller, S. Hiermaier, M. May, and F. Ernst-Mach, "Improvements to the Prototype Micro-Brittle Linear Elasticity Model of Peridynamics," 2013.

[29] "Trilinear interpolation," 6 2021. [Online]. Available: https://en.wikipedia.org/wiki/Trilinear_interpolation

[30] N. S. Ottosen and H. Petersson, *Introduction to the finite element method*. Prentice Hall, 1992.
[31] D. Nguyen, "Chapter 04.11 Cholesky and LDL T Decomposition," Tech. Rep.
[32] B. Boys, T. J. Dodwell, M. Hobbs, and M. Girolami, "PeriPy – A High Performance OpenCL Peridynamics Package," 5 2021. [Online]. Available: http://arxiv.org/abs/2105.04150http://dx.doi.org/10.1016/j.cma.2021.114085
[33] A. Kilic, "Peridynamic Theory for Progressive Failure Prediction in Homogeneous and Heterogeneous Materials Item Type text; Electronic Dissertation," Ph.D. dissertation, 2008. [Online]. Available: http://hdl.handle.net/10150/193658
[34] P. McMullen, "Volumes of projections of unit cubes," *Bulletin of the London Mathematical Society*, vol. 16, no. 3, pp. 278–280, 1984.

A

Appendix

A.1 Local Stiffness Matrix

The concept of a local stiffness matrix is that it is a linearisation of the stiffness around some deformed state. We are interested in the initial state where all displacements are zero. The local stiffness matrix for a particle is the sum of the local stiffness matrices for their bonds,

$$K_i = \sum_j \frac{\partial \mathbf{f_{ij}}}{\partial \eta_{ij}} \quad \text{where: } \eta = 0. \tag{A.1}$$

This should hold regardless of orientation and hence we choose to place our bond along the x-axis, turning it into a 1-dimensional problem,

$$\frac{\partial \mathbf{f}}{\partial x} = \frac{\partial}{\partial x}\left(\frac{(x - x_0) - L}{L} cv\right) = \frac{cv}{L}. \tag{A.2}$$

This can be placed back into the 3D problem with a stiffness matrix in local coordinates

$$K^e = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{cv}{L}. \tag{A.3}$$

Which in turn can be transformed to global coordinates with a transformation matrix

$$l_{ij} = [\hat{\mathbf{e}}'_i \cdot \hat{\mathbf{e}}_j] \tag{A.4}$$

where $\hat{\mathbf{e}}'_i$ is the ith basis vector in the transformed system and $\hat{\mathbf{e}}_j$ is the jth basis vector in the system to transform from.
And by applying it from local to global coordinates one gets

$$lK^e l^T = \begin{bmatrix} \hat{\mathbf{e}}'^{\,2}_{xx} & \hat{\mathbf{e}}'_{xx}\hat{\mathbf{e}}'_{xy} & \hat{\mathbf{e}}'_{xx}\hat{\mathbf{e}}'_{xz} \\ \hat{\mathbf{e}}'_{xy}\hat{\mathbf{e}}'_{xx} & \hat{\mathbf{e}}'^{\,2}_{xy} & \hat{\mathbf{e}}'_{xy}\hat{\mathbf{e}}'_{xz} \\ \hat{\mathbf{e}}'_{xz}\hat{\mathbf{e}}'_{xx} & \hat{\mathbf{e}}'_{xz}\hat{\mathbf{e}}'_{xy} & \hat{\mathbf{e}}'^{\,2}_{xz} \end{bmatrix} \frac{cv}{L}, \tag{A.5}$$

where we will note that only the basis vector which pointed along the bond is used, which is good as the other two are not defined. This can be written more compactly with index notation and using that the basis vector is along with the original bond:

$$cv\frac{\boldsymbol{\xi}_i \boldsymbol{\xi}_j}{|\boldsymbol{\xi}|^3}. \tag{A.6}$$

This is the local stiffness matrix for a single bond in the undeformed state.

A.2 Strain Tensor

The procedure to calculate the strain tensor at a particle in the domain is based on the stretch of each bond connected to that particle. It is quite trivial to calculate the stretch for the bonds, it is conducted during the iterations to come up with the solution. This will result in many more equations than necessary and an overly determined equation system. The least-square solution is used to calculate the final tensor.

The vector $\boldsymbol{\xi}$ is describing the relative position between the particles i and j in the undeformed state as seen in Equation 3.6. Similarly, the vector $\boldsymbol{\eta}$ is describing the relative displacement as seen in Equation 3.5. The strain (or stretch), ϵ, between these two particles can be calculated according to Equation 3.13,

$$\epsilon = \frac{|\boldsymbol{\xi} + \boldsymbol{\eta}| - |\boldsymbol{\xi}|}{|\boldsymbol{\xi}|}.$$

With $\boldsymbol{n}$ denoting the unitised direction of $\boldsymbol{\xi} + \boldsymbol{\eta}$ and where $\boldsymbol{E}$ is the strain tensor as in Equation A.7 (in 3D), then Equation A.8 describes the contribution to the strain tensor for a single bond,

$$\boldsymbol{E} = \begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} & \epsilon_{xz} \\ \epsilon_{yx} & \epsilon_{yy} & \epsilon_{yz} \\ \epsilon_{zx} & \epsilon_{zy} & \epsilon_{zz} \end{bmatrix}, \tag{A.7}$$

$$\boldsymbol{n}^T \boldsymbol{E} \boldsymbol{n} = \epsilon. \tag{A.8}$$

In Voigt notation this can be rewritten as

$$\begin{bmatrix} n_x^2 & n_y^2 & n_z^2 & 2n_y n_z & 2n_x n_z & 2n_x n_y \end{bmatrix} \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ \epsilon_{yz} \\ \epsilon_{xz} \\ \epsilon_{xy} \end{bmatrix} = \epsilon. \tag{A.9}$$

For particle i with bonds to neighbouring particles $j = 1, 2, 3, ..., m$ the following equation system can be set up,

$$\begin{bmatrix} n_{i1_x}^2 & n_{i1_y}^2 & n_{i1_z}^2 & 2n_{i1_y}n_{i1_z} & 2n_{i1_x}n_{i1_z} & 2n_{i1_x}n_{i1_y} \\ n_{i2_x}^2 & n_{i2_y}^2 & n_{i2_z}^2 & 2n_{i2_y}n_{i2_z} & 2n_{i2_x}n_{i2_z} & 2n_{i2_x}n_{i2_y} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ n_{im_x}^2 & n_{im_y}^2 & n_{im_z}^2 & 2n_{im_y}n_{im_z} & 2n_{im_x}n_{im_z} & 2n_{im_x}n_{im_y} \end{bmatrix} \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ \epsilon_{yz} \\ \epsilon_{xz} \\ \epsilon_{xy} \end{bmatrix} = \begin{bmatrix} \epsilon_{i1} \\ \epsilon_{i2} \\ \vdots \\ \epsilon_{im} \end{bmatrix}. \tag{A.10}$$

By calling the matrix with the direction components $\boldsymbol{N}$, the vector with the particle strains $\boldsymbol{A}$ and the vector with bond strains $\boldsymbol{S}$ then Equation A.10 can be rewritten as

$$\boldsymbol{N}\boldsymbol{A} = \boldsymbol{S}. \tag{A.11}$$

Since the number of bonds, m, is more than the number of unknowns the system is overdetermined and instead the least square problem is solved to obtain the strains for particle i,

$$\boldsymbol{A} = (\boldsymbol{N}^T\boldsymbol{N})^{-1}\boldsymbol{N}^T\boldsymbol{S}. \tag{A.12}$$

The strain tensor is not further utilised in our application since the initial truss topology is based on the principal stress field and the stresses are calculated independent of the strains, see Section 3.8. However, the strain tensor can still be of importance to verify the model.

A.3 On the Stiffness of Tapered Elements

The stiffness for a uniform rod is:

$$k = \frac{EA}{L}. \tag{A.13}$$

And one can get the stiffness of springs in a series by their inverse sum, and since each slice of the rod has a different area it is done an infinite amount of times,

$$\frac{1}{k} = \lim_{n\to\infty}\sum_{i=1}^{n}\frac{1}{k_i}. \tag{A.14}$$

This infinite sum can then be rewritten to an integral where the area is parameterised to vary linearly between the end areas,

$$\int_0^L \frac{dl}{E(A_s + \frac{l}{L}(A_e - A_s))} = \frac{L}{E}\frac{\ln\frac{A_e}{A_s}}{A_e - A_s}. \tag{A.15}$$

Leading to:

$$k = \frac{E}{L}\frac{A_e - A_s}{\ln\frac{A_e}{A_s}}. \tag{A.16}$$

This can be compared to averaging the end areas:

$$\frac{E}{L}\frac{A_s + A_e}{2}. \tag{A.17}$$

Meaning that for non-extreme relative areas the average is similar and only slightly stiffer.

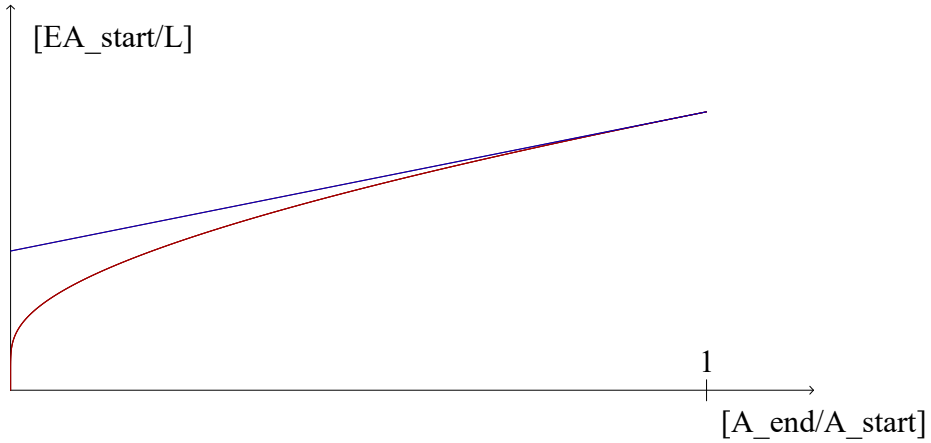


Figure A.1: The true (red) and approximated (blue) stiffness vs the proportions of the ends.

A.4 Signed Distance Field for Cuboids

To be able to perform both the node size optimisation and the element size check there is a need to be able to make evaluations about the space availability efficiently.

To facilitate this, the concept of a signed distance field is adapted from 3D graphics.

A.4.1 Signed Distance Fields

A signed distance field (SDF) is a scalar field where the values indicate the distance to a surface. It is signed in that it makes a difference in values inside and outside a volume. This can be used to represent smooth surfaces with a finite amount of values by having them as the iso-lines in the fields, as seen in Figure A.2. One can note that incomplete SDFs are common where distances further than some distance all have the same value as the information is only needed locally. More connected to the problems at hand, it is also used for intersection checks and ray-tracing.

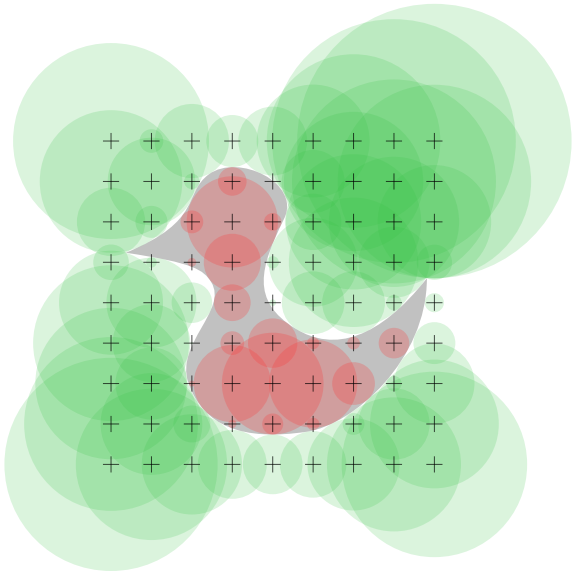


Figure A.2: A signed distance field with values at each cross represented as a circle, which together represents the shape of a duck. The red duck is produced from the field.[1]

A.4.2 Cuboid Modifications

An analogue to be used here is a field-defining the largest box that can be fitted around any point within the shape. The shape is defined by the active voxels. As the location of the cube is the evaluation point, the data to be stored are the dimensions of the box.

First, a definition of the largest box will be needed: let the box grow at an equal rate in each direction, and halt the directions separately once they meet an obstacle. Then we will note that the gradient for each direction is constrained to

$$|\frac{\partial F_i}{\partial i}| \leq 2, \tag{A.18}$$

as the dimension of a point centered box is twice the distance from an edge and $\frac{\partial x}{\partial x} = 1$. The absolute sign is there as one can approach an edge from both directions and the $\leq$ is as 3D effects will disturb this a bit.

From this, we can construct an initial heuristic for the field at any point by extrapolating from adjacent points and keeping the intersection. If we have passed a corner we will need to re-symmetrise the box and achieve this by constraining the largest dimension of the box to the smallest dimension where the middle of a face is not on a surface. Else the initial heuristic will introduce directional artefacts where it maintains the previous elongations. Then the brute force method of attempting to expand the box in each direction is made by checking the whole face in each

[1] made by user Drummyfish, Creative Commons CC0 1.0, https://en.wikipedia.org/wiki/File:Signed_distance_field_duck.svg

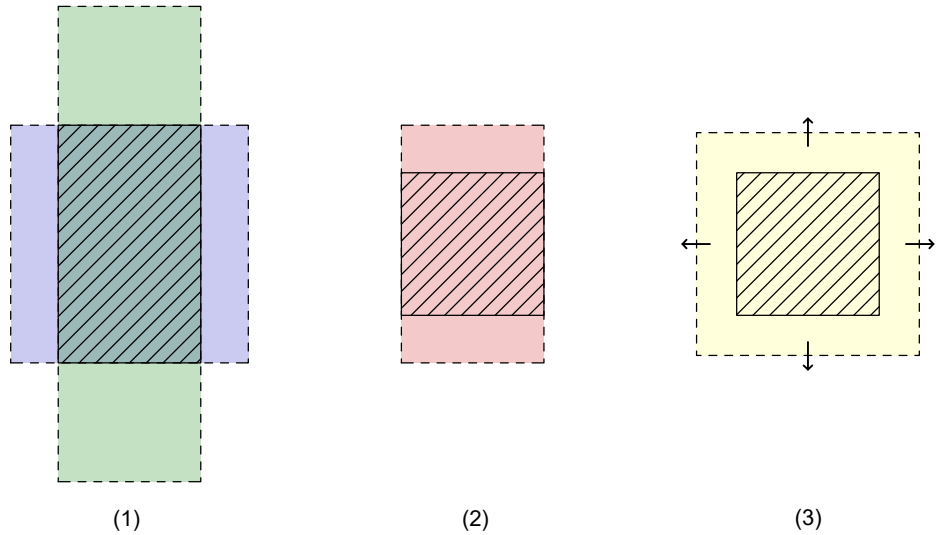


Figure A.3: (1) Use neighbouring cell values to construct a lower bound. (2) Reduce further if there are free edges close to the centre. (3) Finally expand by brute force until the maximum is found.

direction to see if it can expand, see Figure A.3. These three steps are done for each voxel in the domain in sequence, such that the exact box produced in the last step is available for the next box in step one.

A.4.3 Interpolation

With this, we have values at each voxel, but we will be needing values at intermediate points. Only that linear interpolation will lead to boxes extending outside the domain as 3D effects disturb things, as in Figure A.4.
Thus we will enforce Equation A.18 on the linear interpolation by setting the maximum value for any box dimension to $\min(x) + 2\Delta$. This enforces a lower bound on the boxes with the geometric constraints such that they remain within the domain.

A.5 Projected Area of Cuboids

The projected area of a cuboid can be acquired by noting that a projection is a linear transformation, as such it will treat areas much the same as it treats line segments. The projected length of a line is represented as a vector a on a unit vector n and is the absolute value of the scalar product $|a \cdot n| = |\|a\|\|n\|cos\theta|$. Similarly taking the scalar product between a unit vector and a vector with a magnitude equal to the area aligned on the normal of the face represents the signed area of its projection along the vector. This can be done for all faces of the cuboid and the areas of the projections will overlap, but since a cuboid is convex we know that any line intersecting the shape will cross the boundary twice, ergo the surface area's projected area is twice that of the projection. Using the cuboid symmetry means that we need only consider

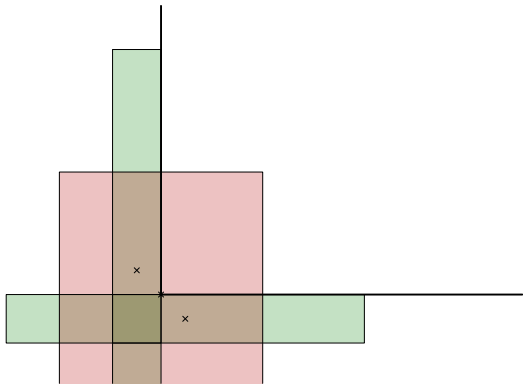


Figure A.4: A naive interpolation between the two green rectangles results in the red rectangle which extends beyond the domain and no longer represents a value in an SDF.

the positive sides of the cuboid as they will have the same result. [34]

$$\text{Proj}_A = \mathbf{n} \cdot (\mathbf{e_x} A_x) + \mathbf{n} \cdot (\mathbf{e_y} A_y) + \mathbf{n} \cdot (\mathbf{e_z} A_z) \tag{A.19}$$

$$\text{Proj}_A = \mathbf{n} \cdot \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} \tag{A.20}$$

$$\text{Proj}_A = \mathbf{n}_x yz + \mathbf{n}_y xz + \mathbf{n}_z xy \tag{A.21}$$

$\mathbf{n}$ = positive unit vector to project along,

A_x = Surface area on the x-side of the cuboid,

$\mathbf{e_x}$ = basis vector in the x-direction,

x, y, z = width, depth, height respectively.

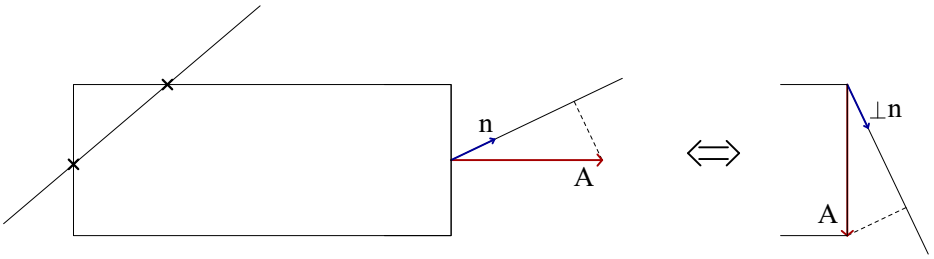


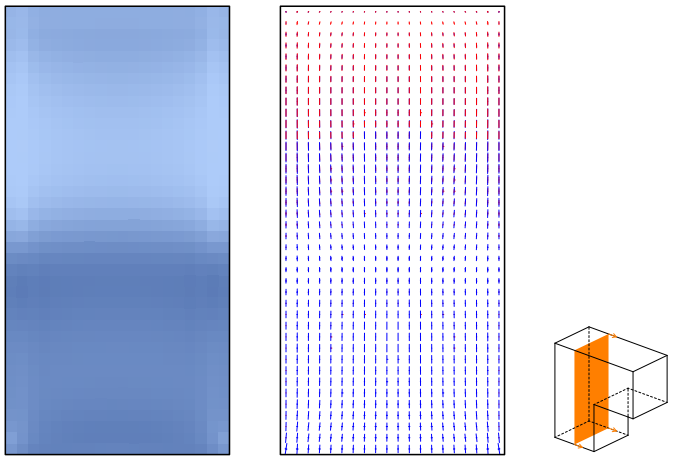
Figure A.5: Any line passing through a convex shape will cross the border twice, and a geometric illustration of why a normal with the magnitude of the area A projected on the projection vector n results in the projected area.

B

Auxiliary Figures for Results

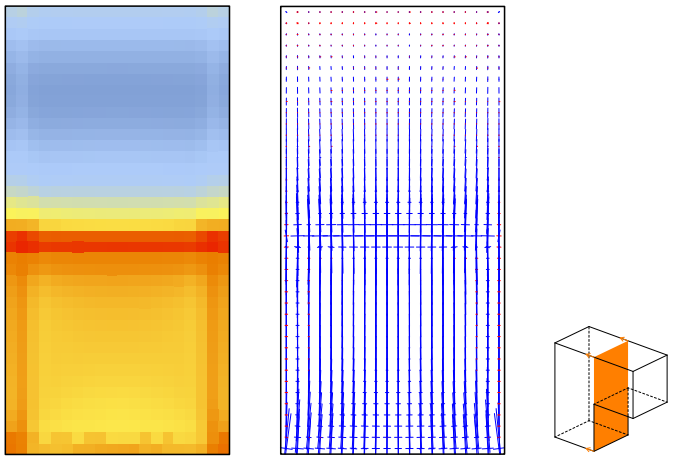
It should be noted that even though sections of the structures are presented, the von Mises stresses and the principal stress field are calculated in 3D. For the display of the principal stresses it means that the principal directions are projected onto the plane of the section.

B.1 Corbel



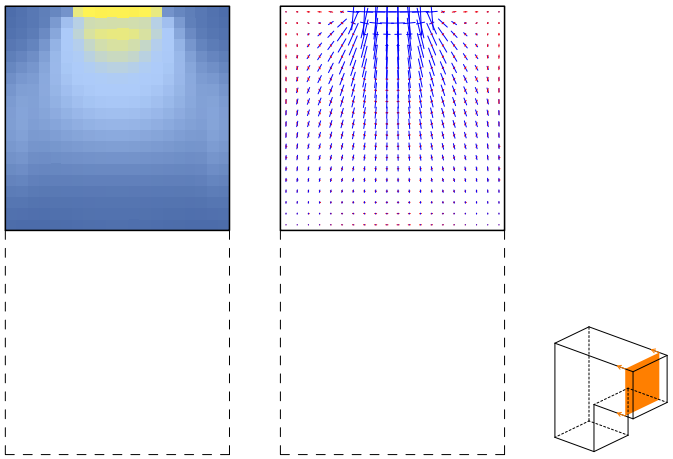
((a)) ((b))

Figure B.1: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 2.5.



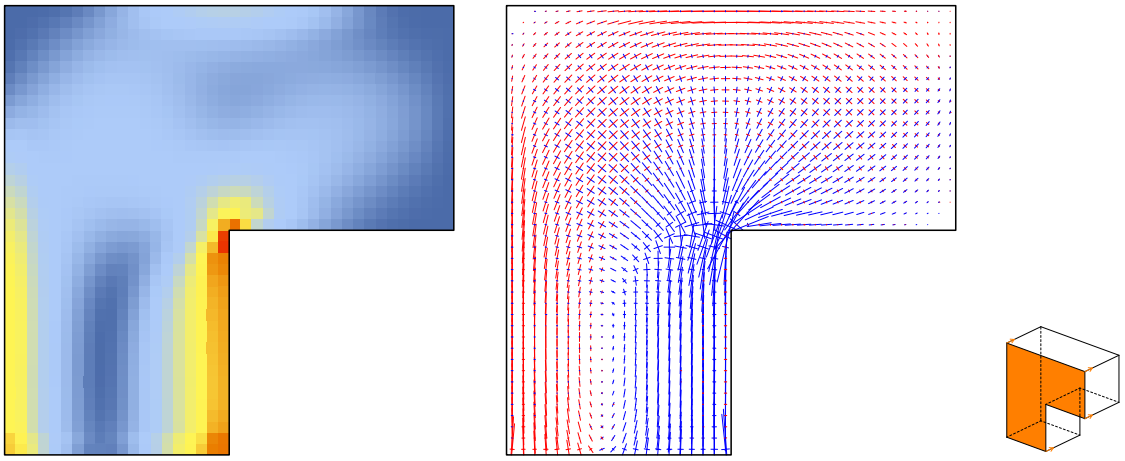
((a)) ((b))

Figure B.2: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 5.



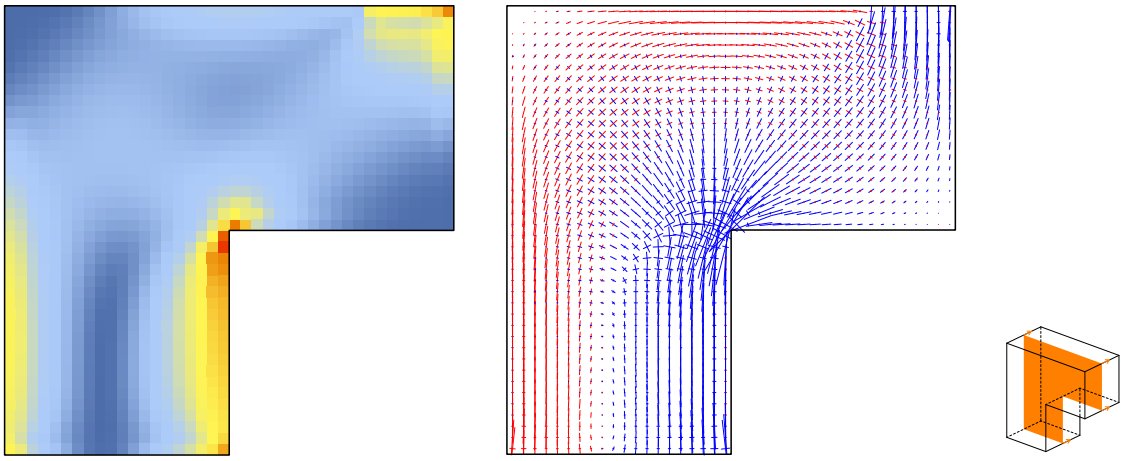
((a)) ((b))

Figure B.3: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 9.



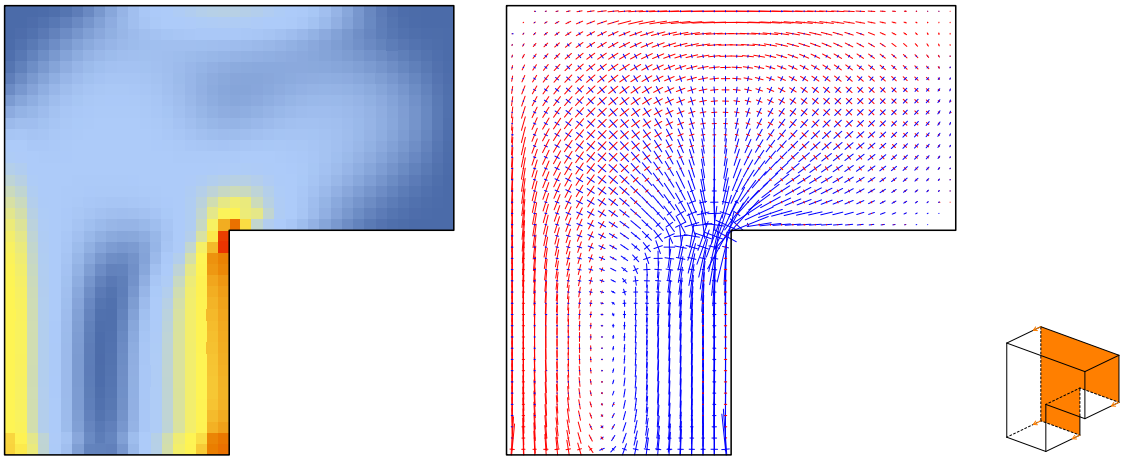
((a)) ((b))

Figure B.4: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 0.



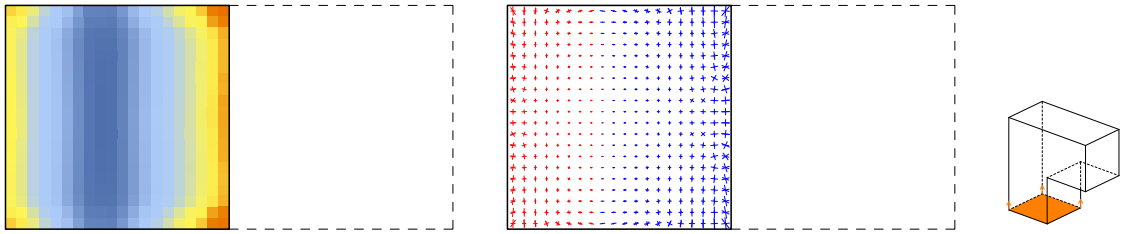
((a)) ((b))

Figure B.5: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 2.5.



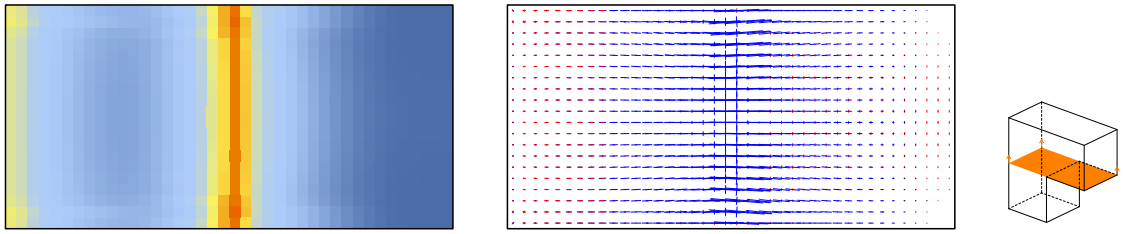
((a)) ((b))

Figure B.6: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 5.



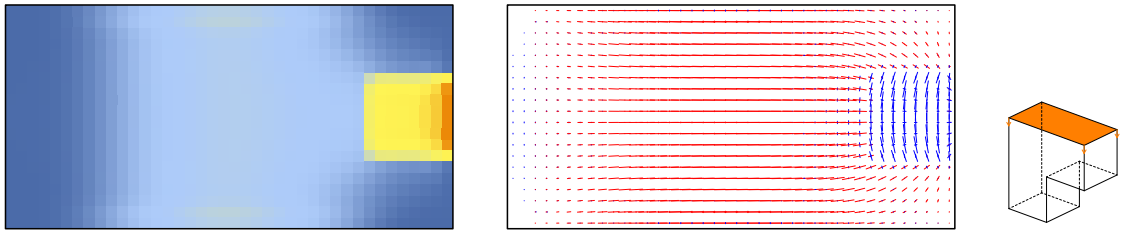
((a)) ((b))

Figure B.7: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 0.



((a)) ((b))

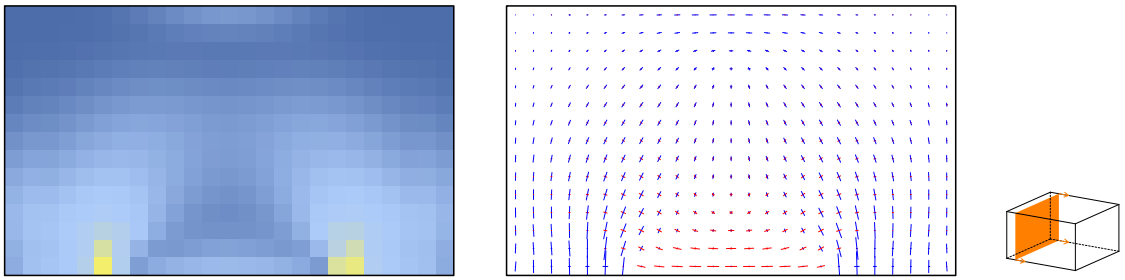
Figure B.8: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 5.



((a)) ((b))

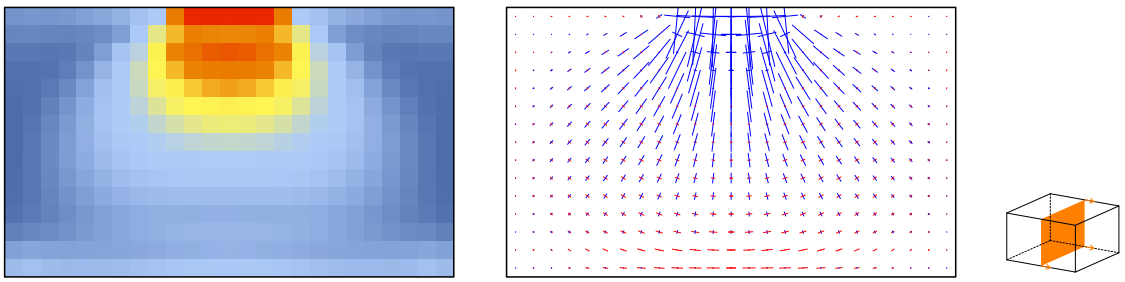
Figure B.9: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 10.

B.2 Pile Cap



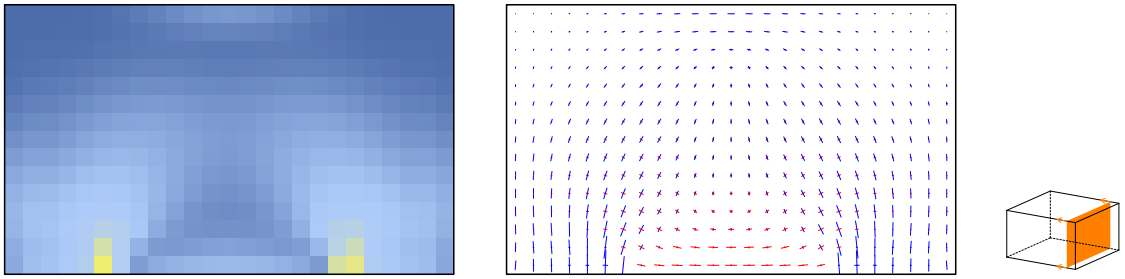
((a)) ((b))

Figure B.10: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 0.625.



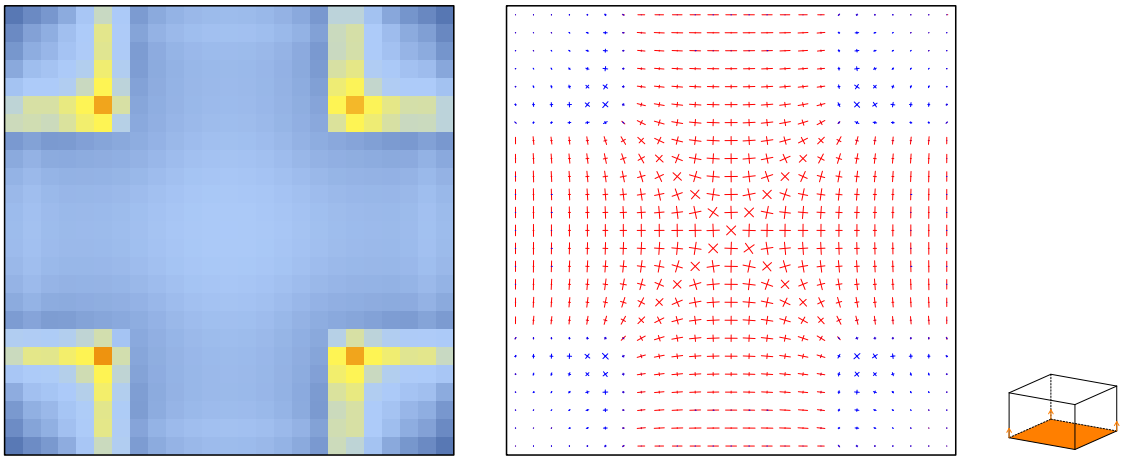
((a)) ((b))

Figure B.11: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 2.5.



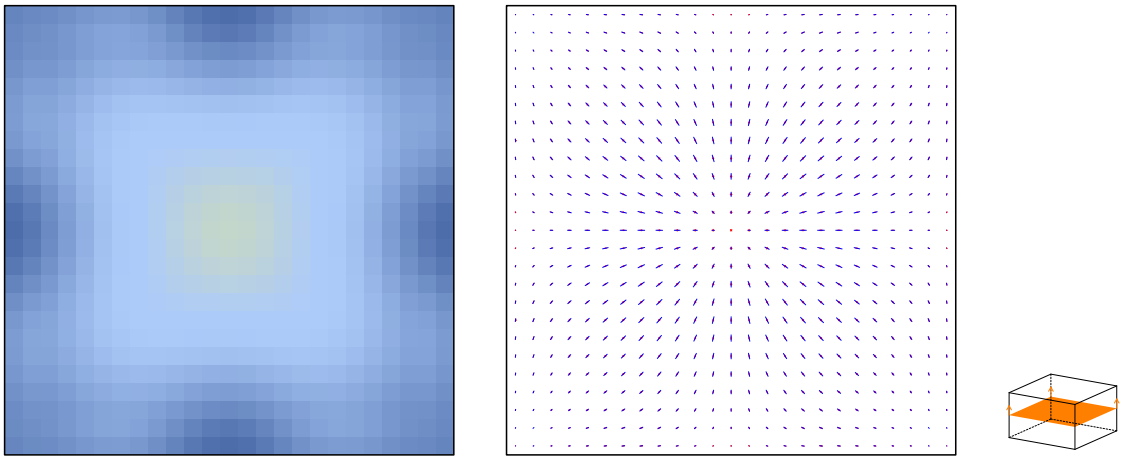
((a)) ((b))

Figure B.12: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 4.375.



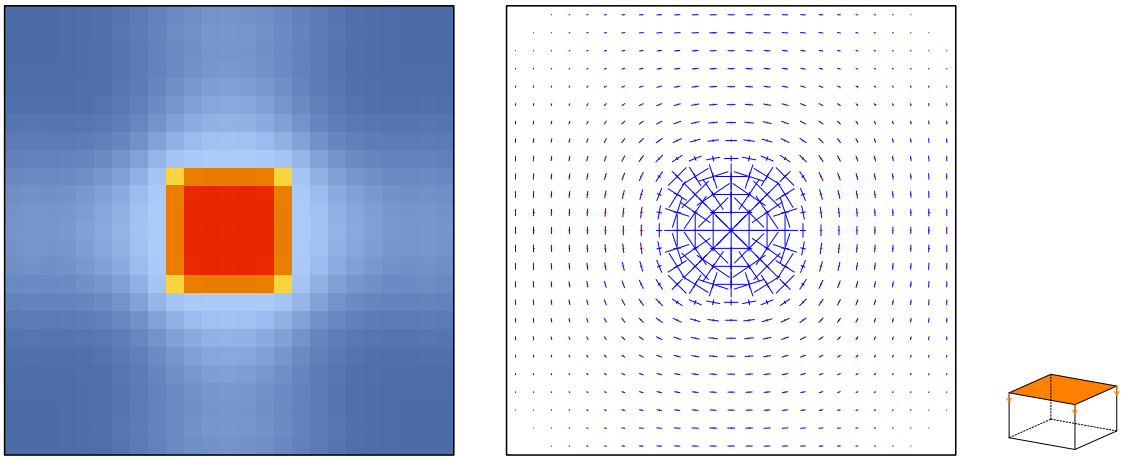
((a)) ((b))

Figure B.13: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 0.



((a)) ((b))

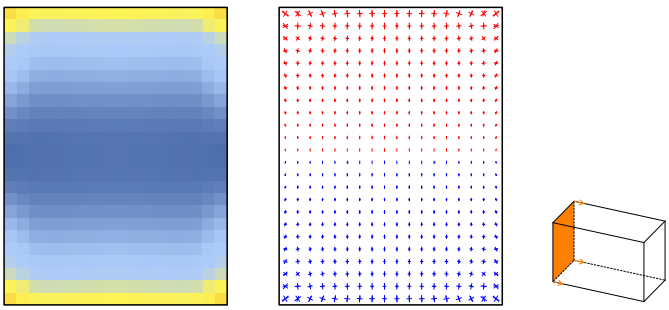
Figure B.14: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 1.5.



((a)) ((b))

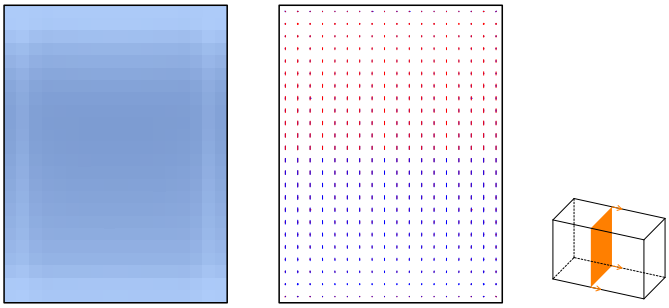
Figure B.15: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 3.

B.3 Cantilever



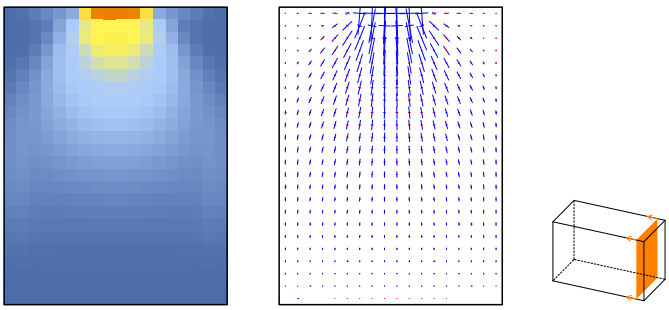
((a)) ((b))

Figure B.16: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 0.



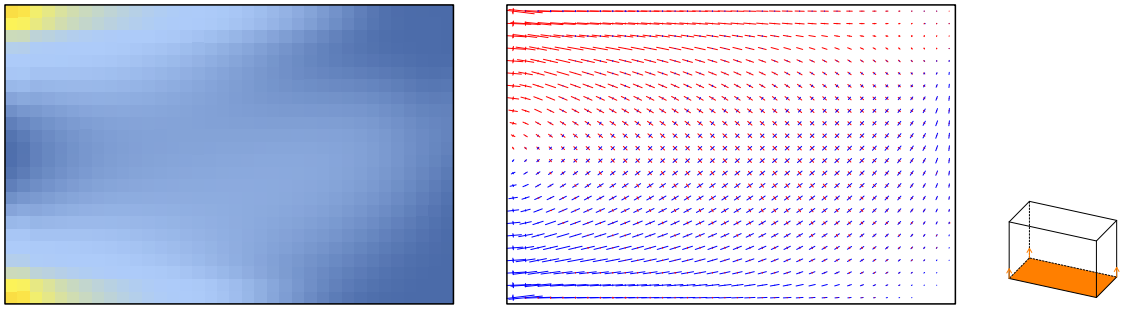
((a)) ((b))

Figure B.17: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 2.5.



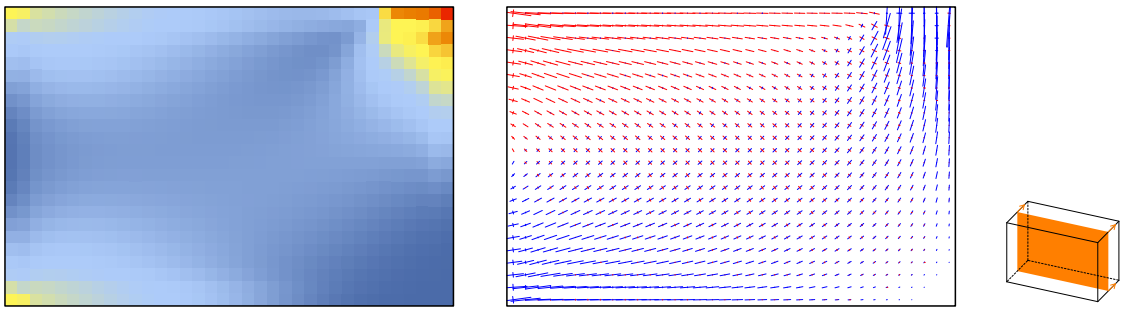
((a)) ((b))

Figure B.18: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 5.5.



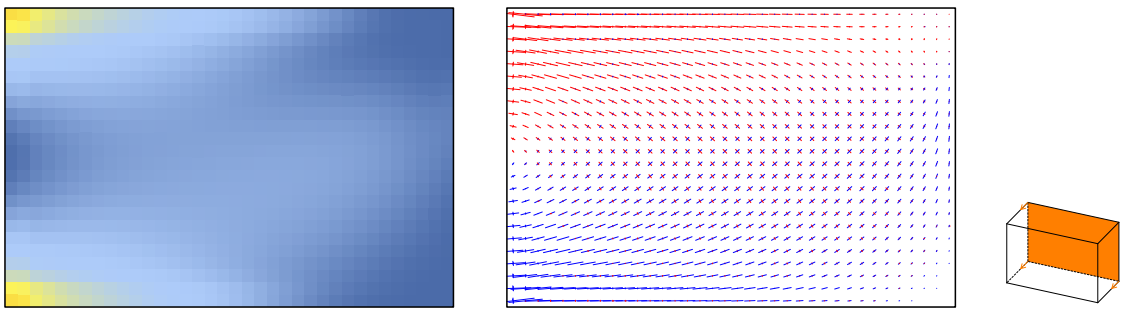
((a)) ((b))

Figure B.19: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 0.



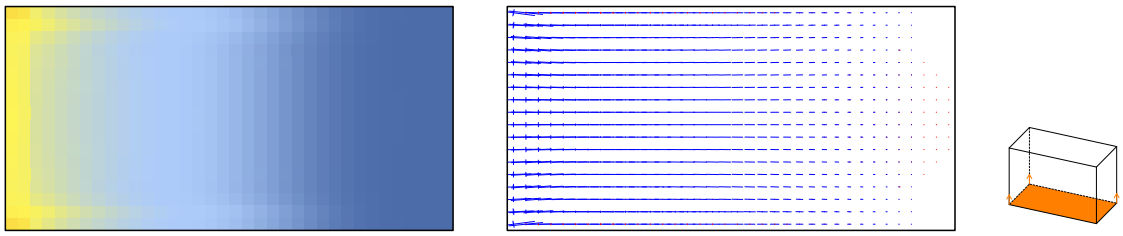
((a)) ((b))

Figure B.20: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 1.5.



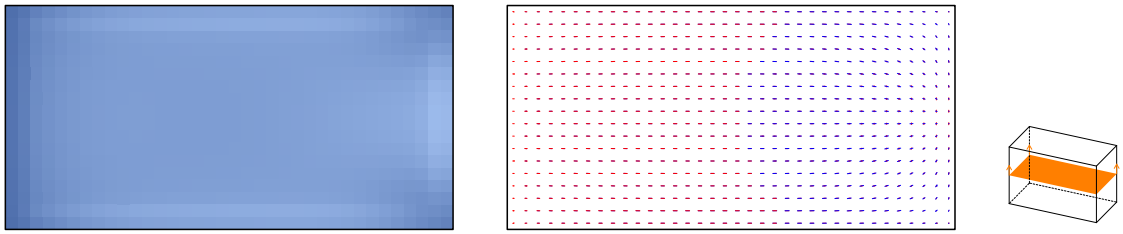
((a)) ((b))

Figure B.21: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 3.



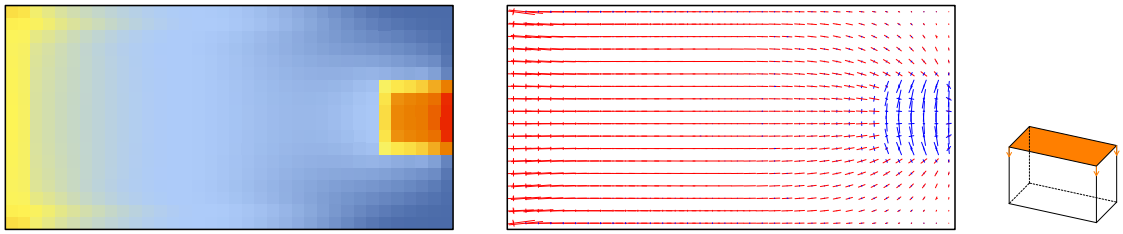
((a)) ((b))

Figure B.22: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 0.



((a)) ((b))

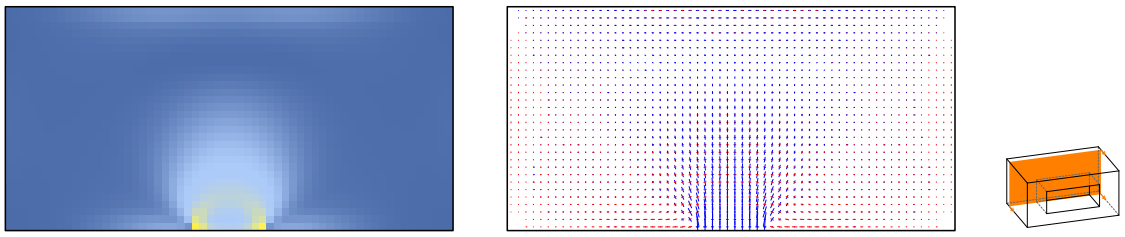
Figure B.23: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 2.



((a)) ((b))

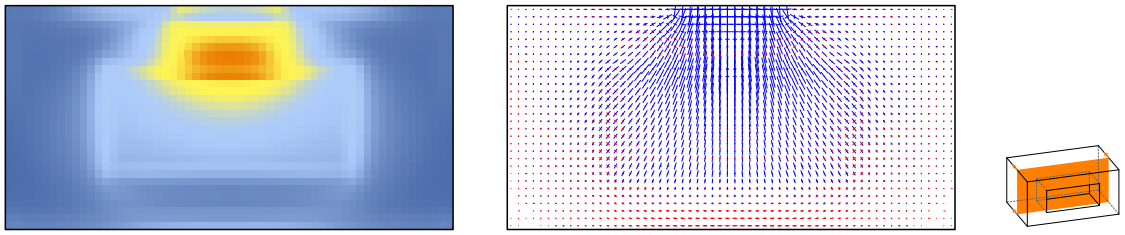
Figure B.24: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 4.

B.4 Structure with Cavity



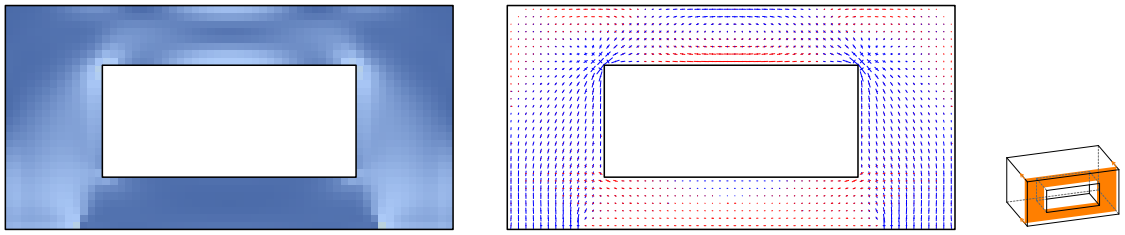
((a)) ((b))

Figure B.25: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 0.5.



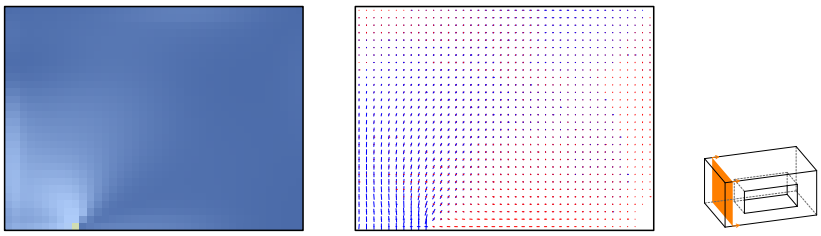
((a)) ((b))

Figure B.26: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 2.



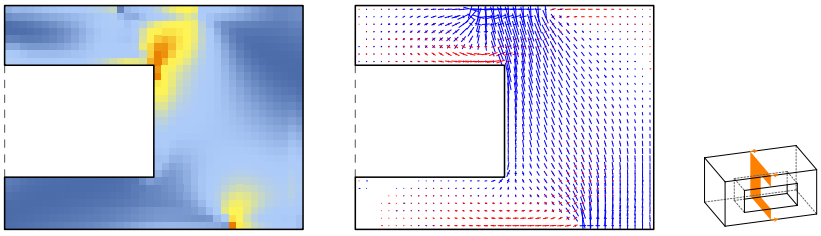
((a)) ((b))

Figure B.27: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at X = 3.5.



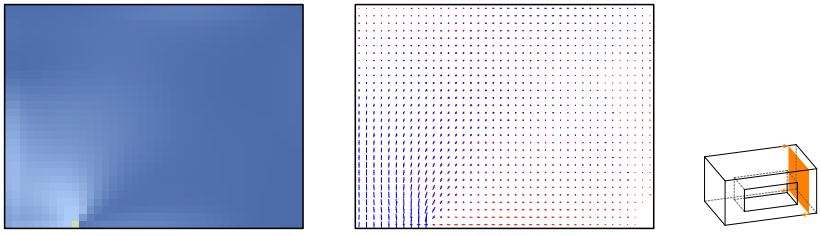
((a)) ((b))

Figure B.28: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 0.5.



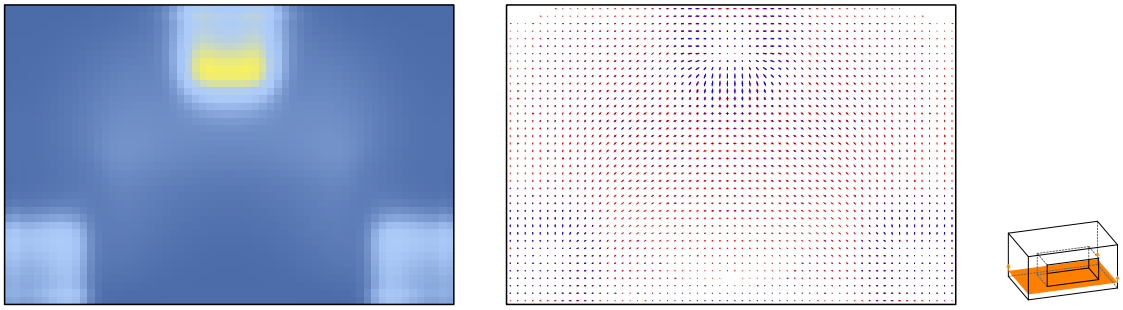
((a)) ((b))

Figure B.29: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 3.



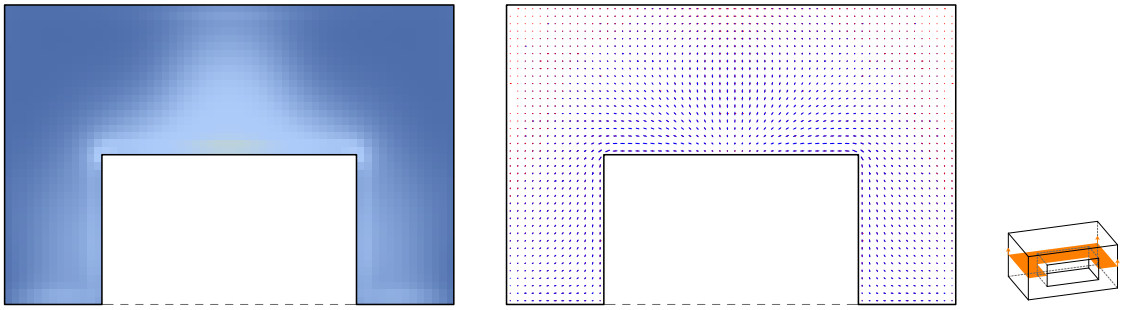
((a)) ((b))

Figure B.30: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Y = 5.5.



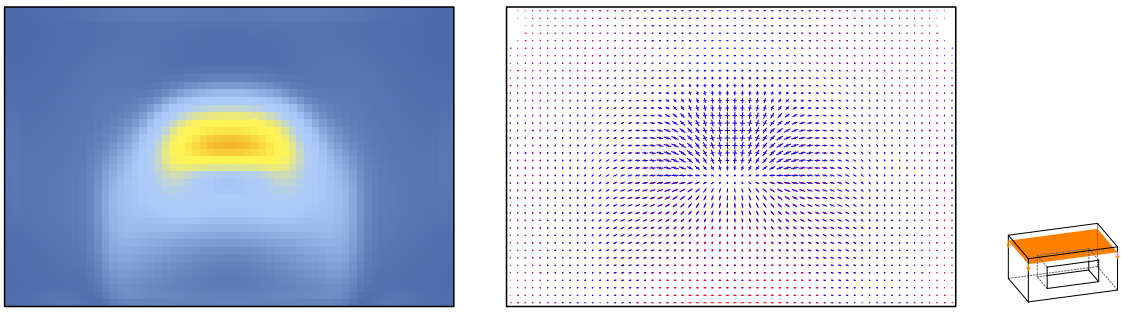
((a)) ((b))

Figure B.31: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 0.375.



((a)) ((b))

Figure B.32: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 1.5.



((a)) ((b))

Figure B.33: Distribution of the von Mises stresses in **(a)** and principal stress field in **(b)** for the section at Z = 2.625.