Topology optimization for additive manufacturing: a front propagation-based approach

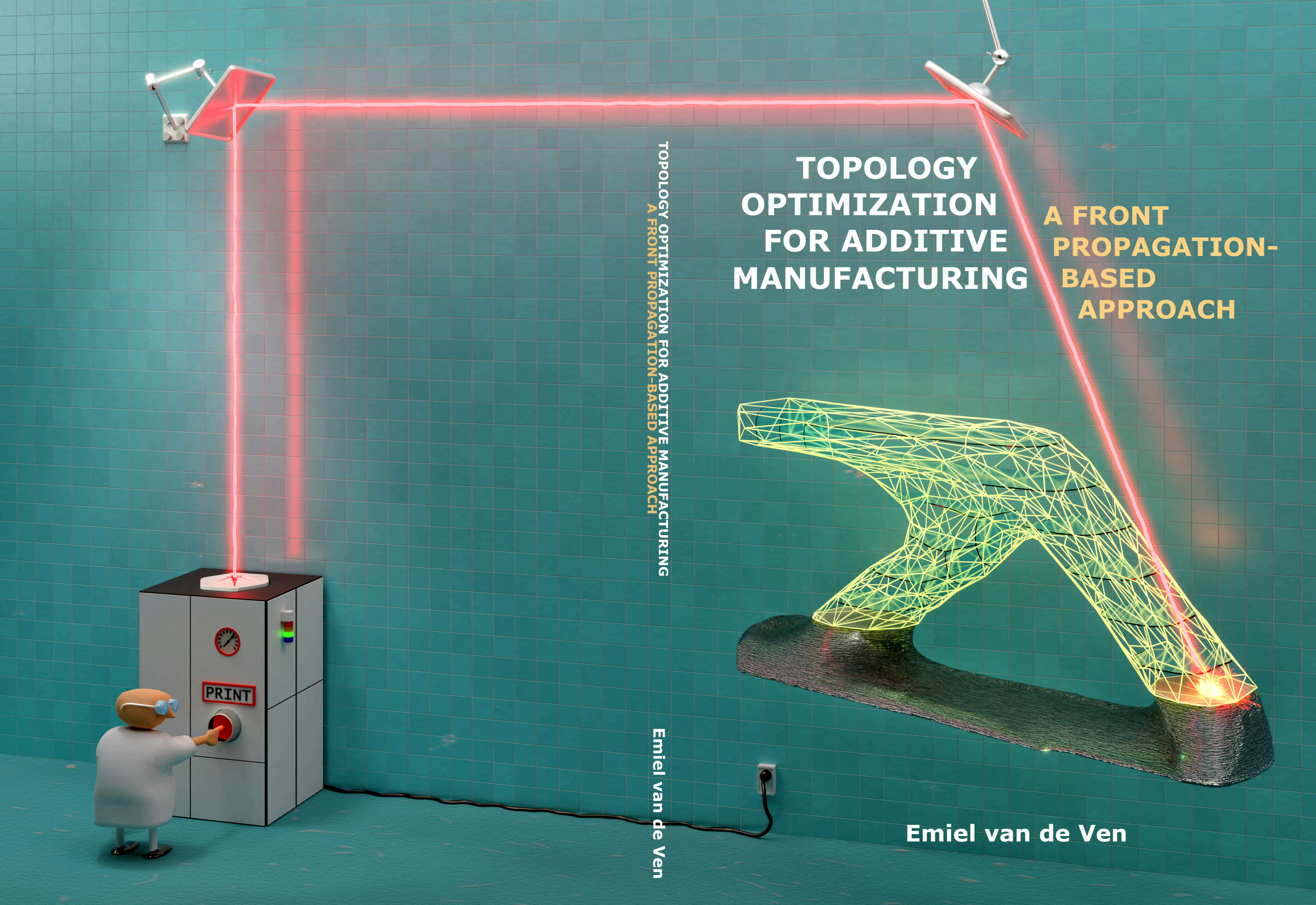
van de Ven, E.A.

DOI
10.4233/uuid:4fa895ca-e27a-4ada-8417-d50e673a528d

Publication date
2021

Document Version
Final published version

Citation (APA)
van de Ven, E. A. (2021). *Topology optimization for additive manufacturing: a front propagation-based approach*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:4fa895ca-e27a-4ada-8417-d50e673a528d



TOPOLOGY OPTIMIZATION FOR ADDITIVE MANUFACTURING

A FRONT PROPAGATION-BASED APPROACH

Emiel van de Ven

TOPOLOGY OPTIMIZATION FOR ADDITIVE MANUFACTURING
A FRONT PROPAGATION-BASED APPROACH
Emiel van de Ven

TOPOLOGY OPTIMIZATION FOR ADDITIVE MANUFACTURING: A FRONT PROPAGATION-BASED APPROACH

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
donderdag 2 december 2021 om 15:00 uur

door

Emiel Anton VAN DE VEN

werktuigkundig ingenieur,
Technische Universiteit Delft, Nederland
geboren te Grubbenvorst, Nederland.

Dit proefschrift is goedgekeurd door de promotoren.

Samenstelling promotiecommissie bestaat uit:

Rector magnificus,	voorzitter
Prof. dr. ir. A. van Keulen	Technische Universiteit Delft, promotor
Dr. ir. M. Langelaar	Technische Universiteit Delft, promotor
Dr. C. Ayas	Technische Universiteit Delft, copromotor
Onafhankelijke leden:	
Prof. dr. O. Sigmund	Danmarks Tekniske Universitet
Prof. dr. G. Allaire	École polytechnique
Prof. dr. ir. C. Vuik	Technische Universiteit Delft
Drs. R. Maas	Koninklijk Nederlands Lucht- en Ruimtevaartcentrum
Prof. dr. ir. J.L. Herder	Technische Universiteit Delft, reservelid

Keywords: Topology optimization, Additive manufacturing, Overhang constraint, Front propagation

Printed by: Gildeprint

Front & Back: Emiel van de Ven

ISBN 978-94-6366-480-6

Summary

Topology optimization for additive manufacturing: a front propagation-based approach

Topology optimization is becoming an increasingly popular design tool. No longer within the sole reach of academic community, it is now an integral part of most commercial finite element analysis packages, where it is available to engineers at the click of a mouse. Its popularity is further increased by the rise of additive manufacturing, which provides a means to translate the complex, intricate geometries typically resulting from topology optimization into real-world products. The goal of this thesis is to improve the integration between topology optimization and additive manufacturing, to further advance the adoption of both technologies.

Additive manufacturing is known for its tremendous form freedom. Due to the layer-wise fabrication of a part, geometrically complex components can be realized in contrast to conventional manufacturing techniques such as casting or milling. However, additive manufacturing is not free from manufacturing limitations. One of the most prominent limitations is the overhang constraint, which gives rise to the support structures often surrounding 3D-printed components along the overhanging (down-facing) surfaces.

In the first part of this thesis, a novel method is developed that prevents the topology optimized designs from requiring any supporting structures during fabrication. This can save cost, as supporting structures require machine time and consume material to print, while their removal after printing is labor intensive.

In Chapter 2 and Chapter 3, an overhang constraint is presented in 2D and 3D, respectively, based on a coarse approximation of the printing process using front propagation. The evolution of a part during the printing process can be seen as a front that propagates through space with every layer that is printed. This front propagation is performed in every iteration of the topology optimization and forms the basis of an overhang filter that checks whether the current geometry is printable without supports. Consequently, regions that are not printable are removed from the current state of the design. Sensitivities for the overhang filter constraint are derived and included in the optimization process. Therefore, if some region of the part is not printable but of high importance for the performance, it is not removed but rather the shape of the part is adapted to ensure printability of the design. Where possible the design will combine functionality to efficiently use material such that regions that are primarily supporting another region also have a load carrying function. The resulting designs are optimized for their original function as well as for fabrication by additive manufacturing. This is shown in extensive numerical cases in 2D and 3D where stiffness is maximized, and for a flow channel optimization and compliant mechanism design in Chapter 4.

It is shown that the front propagation based overhang constraint has several advantages compared to existing methods. First of all, it is applicable to structured as well as unstructured meshes, typically used in real engineering applications. Furthermore, the limiting overhang angle can be freely adjusted, to correspond to the overhang angle valid for the specific printing process and choice of material of the end user. Additionally, the overhang constraint is robust: there is no need for parameter tuning for different cases. Lastly, the overhang constraint is computationally efficient. On large unstructured 3D meshes, it is about 10 times faster than the finite element analysis required for the stiffness evaluation.

An in-depth comparison of the presented overhang constraint and a layer-by-layer overhang constraint for structured meshes is performed in Chapter 5, where the latter is formulated as a front propagation problem to reveal the true differences between both methods. This leads to a fundamental reason why the overhang constraint presented in this thesis can be discretized on an unstructured grid whereas the existing method cannot, and it is shown that some attractive features of the existing method can be incorporated to the front propagation based overhang constraint to further improve it.

Finally, an extension of the overhang constraint is proposed in Chapter 6, where the overhang constraint is only activated in regions where support material is difficult or impossible to remove. This is achieved by combining the overhang constraint with several Boolean operations on the geometry. The resulting filter is shown to increase the performance of the optimized geometries compared to a conventional overhang constraint as overhanging surfaces are allowed in places where the supports are easy to remove.

Samenvatting

Topologie optimalisatie en 3D printen: een aanpak gebaseerd op front propagatie

Topologie optimalisatie wordt steeds populairder als ontwerpmethodiek. Nu het bij de meeste commerciële eindige elementen pakketten standaard mee geleverd wordt, is het niet langer enkel een methode voor academici, maar gemakkelijk beschikbaar voor ingenieurs in het veld. De populariteit van topologie optimalisatie neemt verder toe door de opkomst van 3D printen, welke het mogelijk maakt de complexe structuren van topologie optimalisatie om te zetten in tastbare producten. Het doel van deze thesis is het verder verbeteren van de integratie tussen topologie optimalisatie en 3D printen, zodat beide technologieën nog wijder verspreid raken.

3D printen staat bekend om de grote ontwerpvrijheid van het proces. Doordat producten laag-voor-laag worden opgebouwd, kunnen complexere vormen gemaakt worden dan met conventionele maakprocessen zoals gieten of frezen. Ontwerpen voor 3D printen is echter niet volledig vrij van ontwerpregels. Een van de belangrijkste limitaties van 3D printen is de zogenaamde minimale overhang hoek, welke tevens de oorzaak is van de ondersteunende structuren die typisch meegeprint worden.

In het eerste deel van deze thesis wordt een methodiek gepresenteerd welke ervoor zorgt dat topologie geoptimaliseerde ontwerpen voldoen aan de minimale overhang hoek. Hierdoor kan bespaard worden op machinetijd en materiaal kosten voor het printen van de steunstructuren, en op arbeidskosten voor het verwijderen van deze structuren, wat een arbeidsintensieve klus kan zijn.

De methodiek die in Chapter 2 en Chapter 3 wordt gepresenteerd om de minimale overhang hoek regel mee te nemen, is gebaseerd op een benadering van het printproces met front propagatie: het printproces kan gezien worden als een front dat met elke geprinte laag door de ruimte propageert. Deze front propagatie wordt elke iteratie van de topologie optimalisatie uitgevoerd en functioneert als een filter welke bepaalt of de geometrie printbaar is zonder ondersteuning. Delen die niet printbaar zijn worden vervolgens verwijderd van de huidige geometrie. De afgeleiden van het filter zijn bepaald, waardoor deze meegenomen kunnen worden in de optimalisatie. Dit zorgt ervoor dat als een bepaald gebied niet printbaar is zonder ondersteuning maar belangrijk is voor het functioneren van het onderdeel, dit gedeelte van de structuur niet verwijderd wordt, maar materiaal wordt toegevoegd zodat het voldoende ondersteund is. Waar mogelijk zal de optimalisatie functionaliteit combineren zodat delen die voornamelijk een ondersteunende functie hebben ook bij zullen dragen aan de algehele prestatie van het onderdeel. De resulterende onderdelen zullen zeer optimaal zijn voor hun oorspronkelijke doel en voor fabricage door middel van 3D printen. Dit wordt gedemonstreerd aan de hand van enkele numerieke voorbeelden in 2D en 3D waar stijfheid wordt gemaximaliseerd, en aan de hand van een stromingsoptimalisatie en ontwerp van een flexibel

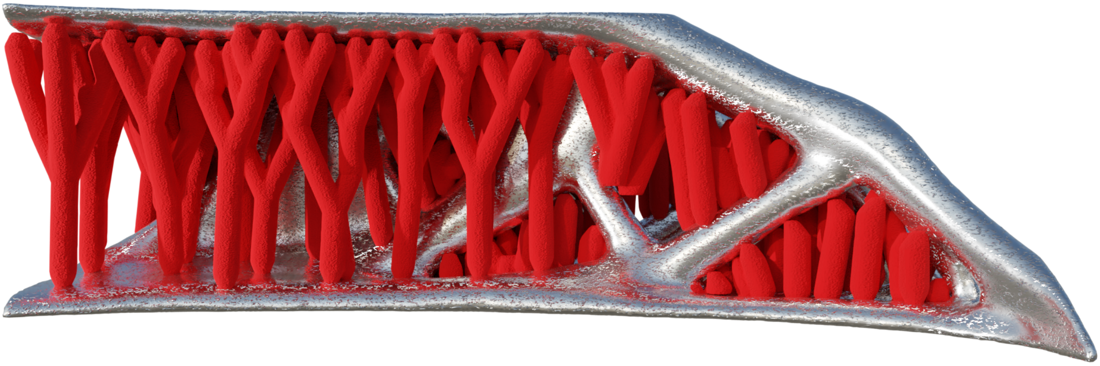
mechanisme in Chapter 4.

Verder laten we zien dat de front propagatie formulatie van het printproces meerdere voordelen heeft ten opzichte van bestaande methodieken. Ten eerste is de front propagatie methodiek toepasbaar op regelmatige en onregelmatige discretisaties van het ontwerpdomein. Daarbij kan de overhang hoek continu gevarieerd worden zodat het overeenkomt met het gekozen proces en materiaal. De formulatie is ook robuust: er hoeven geen parameters aangepast te worden voor verschillende optimalisaties. Ten slotte is de front propagatie efficiënt. De tijd die nodig is voor het berekenen van de front propagatie is ongeveer een factor 10 lager dan de tijd benodigd voor een lineaire eindige elementen analyse, zelfs voor grote 3D problemen.

In Chapter 5 wordt een gedetailleerde vergelijking gemaakt tussen een bestaande methode om te voldoen aan de minimale overhang hoek en de methode gepresenteerd in deze thesis. Door de bestaande methode te herscrijven als een front propagatie probleem worden de verschillen tussen beide methoden zichtbaar. Hierdoor kan worden vastgesteld waarom de bestaande methodiek niet op een onregelmatige discretisatie kan werken, terwijl dit wel mogelijk is voor de methode uit deze thesis. Daarbij blijkt dat een aantal gunsitge eigenschappen van de bestaande methode kunnen worden overgenomen om de voorgestelde methode robuster te maken.

Tot slot wordt in Chapter 6 een versoepeling van de minimale overhang hoek regel voorgesteld, zodat de resulterende ontwerpen alleen geen ondersteuning nodig hebben op plekken waar de ondersteuning achteraf moeilijk te verwijderen is. Dit wordt gedaan door een mix van de gepresenteerde methodiek met enkele discrete operaties op de geometrie. Door het toestaan van ondersteuning op makkelijk te verwijderen plekken neemt de prestatie van de onderdelen toe, omdat er minder materiaal voor ondersteuning gebruikt hoeft te worden.

Contents



1 INTRODUCTION

At the beginning of the 20th century, automobiles were produced by skilled craftsman in shops. Production of parts was scattered over small machine shops, and individual parts had to be worked to fit one another because of the lack of a single gauging system. The cars were produced in low volume, and in effect luxury items only available to the wealthy (Womack *et al.*, 1990).

This changed in 1908 with the introduction of the Ford Model T. Henry Ford introduced the innovative concept of mass production, and revolutionized, first, the production of automobiles, and later almost every other industry. Effectively, the Model T was the first car that was *designed for manufacturing*. Parts were specifically designed to be machine-made in high volumes, and for ease of assembly. This drastically reduced the price of the Model T, and allowed Ford to dominate the automotive industry for the next 50 years (Womack *et al.*, 1990). In the following decades, there are countless examples of how improved design for manufacture and assembly (DFMA) led to vast savings in manufacturing and assembling cost (Boothroyd *et al.*, 2011), and it is the central theme of this thesis.

In the late 1980s, a new design methodology and a new manufacturing method were being developed: topology optimization (TO; Bendsøe and Kikuchi, 1988), and additive manufacturing (AM; Hull, 1986). It would later turn out that both methods complement each other perfectly. As with any other manufacturing technique, AM comes with its own set of design-rules, and ignoring these during the design phase can lead to an increase in product cost, defects in the part produced or even failure of the manufacturing process. Therefore, it is the aim of this thesis to adopt the manufacturing constraints of AM into TO, to make TO a *design for manufacturing* tool, and to empower AM with a capable design tool.

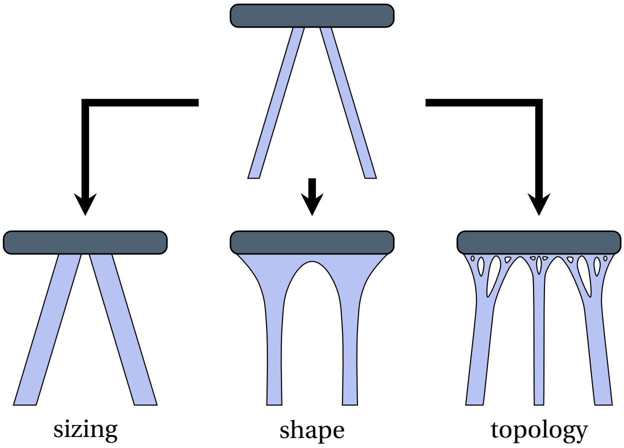


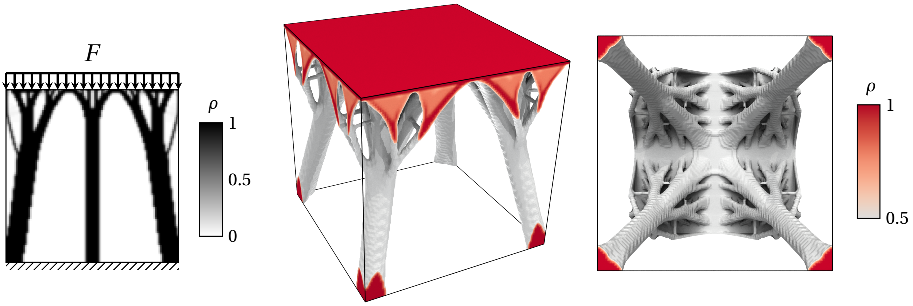
Figure 1.1: Optimization of the legs of a stool for stiffness with a volume constraint for three different classes of optimization: sizing, shape and topology optimization.

1.1. Topology Optimization

Topology optimization (TO) started its development in the late 1980s (Bendsøe and Kikuchi, 1988). It is a design methodology that optimizes the geometry of a component for an objective function, and possibly subject to constraints. The objective and constraints can vary widely. For example, TO can be used to obtain a part as light as possible, while not exceeding stress limits for prescribed loading configurations. A completely different objective could be to minimize the pressure drop of a fluid flow, or the thermal expansion of a component.

TO is one out of three classes of structural optimization methods: sizing, shape and topology optimization (Christensen and Klarbring, 2008). Consider the optimization of the legs of a stool, displayed in Figure 1.1. With a sizing optimization, only some prescribed geometrical parameters can be changed, for example the thickness of the legs. With shape optimization, the shape and position of the boundary is optimized. This allows for much larger geometrical changes compared to sizing optimization. However, TO is the least restricted optimization method, allowing change of shape and connectivity. Since TO has the most design freedom, designs generated with TO often outperform those generated with other methods, but also tend to be more complex as can be seen in Figure 1.1. Therefore, topology optimized design are typically difficult to manufacture.

For a sizing optimization, the design variables are obvious: it are those parameters that are to be optimized, such as the thickness, length or cross-section of certain members. With shape optimization, the surface of the geometry has to parameterized. The surface can be defined explicitly, e.g., as a series of polynomial surfaces such as a B-splines or Bézier surfaces (Schneider and Eberly, 2003), where the control points become the design variables of the optimization. However, such a description becomes difficult to manage when the shape transforms substantially and different surfaces intersect and/or merge. This is usually the case with TO where the whole geometrical layout is subject to change, and therefore the geometry is mostly defined implicitly. This can be



(a) Optimized stool in 2D, with 100×100 elements and a 35% volume constraint.

(b) Side and bottom view of the optimized stool in 3D, with 100×100×100 elements and an 8% volume constraint.

Figure 1.2: Topology optimization of a stool in 2D and 3D.

done, e.g., by using a level-set function (van Dijk *et al.*, 2013), or by defining a pseudo-density field over the domain, specifying for each location if it contains material.

The latter describes the most common method for TO, which is used throughout this thesis: the density-based approach (Bendsøe and Sigmund, 2004). In the design domain Ω, a density field $\rho(\mathbf{x}) \in \Omega$ defines for every location $\mathbf{x}$ if it is a material region ($\rho = 1$), or is void ($\rho = 0$). This results in a discrete optimization problem, which is difficult to solve. There are TO methods that handle this discrete optimization problem directly, such as evolutionary structural optimization (Huang and Xie, 2010; Xie and Steven, 1993), but in density-based methods the optimization problem is relaxed into a continuous form by also allowing intermediate densities such that $0 \leq \rho \leq 1$. This enables the use of powerful gradient-based optimization algorithms, which allows the TO to converge with a limited amount of design iterations compared to derivative-free optimization algorithms. This is of paramount importance since every design iteration requires the objective evaluation through a finite element analysis (FEA), which is usually the most time consuming step in the optimization procedure.

The focus of this work is on developing manufacturing constraints for TO, and the effectiveness and implications of the addition of such a constraint can be best studied on a simple, well-studied TO problem. Consequently, the methods developed in this thesis are, with the exception of Chapter 4, all demonstrated on a compliance minimization optimization subject to a maximum volume constraint, which is given as

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \mathbf{f}^T\mathbf{u} \\ \text{s.t.} \quad & \mathbf{K}\mathbf{u} = \mathbf{f}, \\ & V(\boldsymbol{\rho}) \leq V_{\text{lim}}, \\ & \mathbf{0} \leq \boldsymbol{\rho} \leq \mathbf{1}, \end{aligned} \tag{1.1}$$

where $\mathbf{f}$ and $\mathbf{u}$ are the discretized load and displacement vectors and $\mathbf{K}$ is the stiffness matrix which is a function of the discretized design variables $\boldsymbol{\rho}$. $V(\boldsymbol{\rho})$ represents the volume of the design directly defined by integrating $\boldsymbol{\rho}$ in the design domain and V_{lim} is the maximum allowable volume.

Examples of a compliance minimization in 2D and 3D are given in Figure 1.2, where a stool is topology optimized, similar to the problem depicted in Figure 1.1. A distributed load, mimicking a person sitting on the stool, is applied on the top surface, while displacements are prescribed to be zero at the bottom, and the volume is constraint to be at maximum 35% and 8% in 2D and 3D, respectively. For 2D optimizations, the complete density field is shown, as can be seen in Figure 2.1a, while in 3D the density field is thresholded at $\rho = 0.5$, showing only the densities with $\rho \geq 0.5$. Consequently, inside the domain the gray isosurface for $\rho = 0.5$ is shown, while on the edge of the domain density values above 0.5 can be present (e.g. the red areas in Figure 1.2b). It can be seen that in both cases the resulting geometries are complex. The 3D stool in Figure 1.2b has many surfaces that are not accessible for machining, and will be difficult and expensive to manufacture with conventional manufacturing techniques. However, it can be produced with AM.

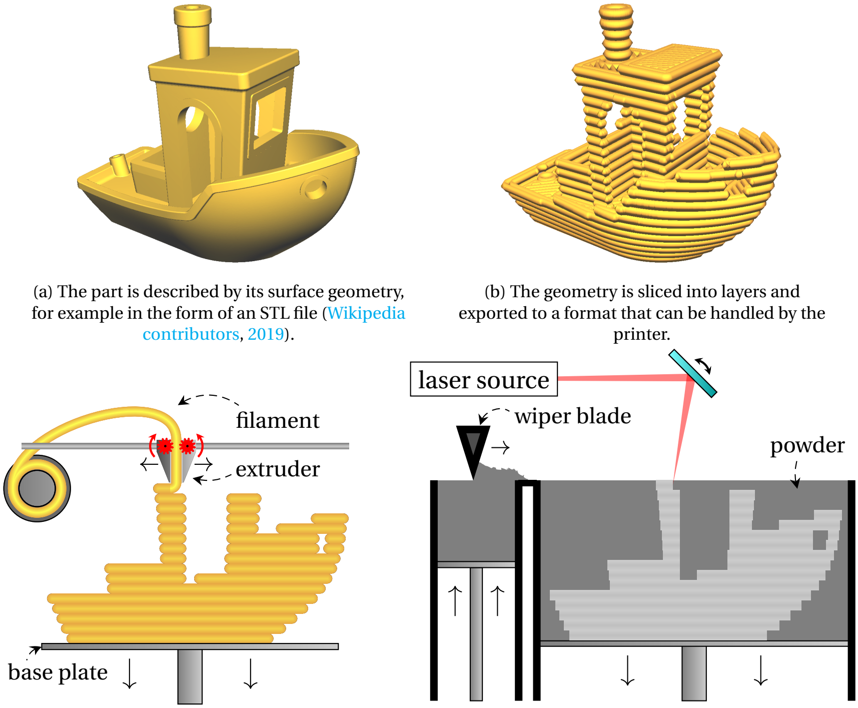
1.2. ADDITIVE MANUFACTURING

Additive manufacturing, or 3D printing, is a collective term for manufacturing processes that fabricate a product by a layer-wise addition of material. This is in contrast to conventional subtractive manufacturing processes such as milling, or formative manufacturing processes such as casting (Gebhardt and Hötter, 2016). The process originated as rapid-prototyping in the 1980s (Hull, 1986), and was mainly used to create mockups for, e.g., visualization, assembly studies or marketing, while the manufacturing of the final product was done using conventional techniques (Gibson *et al.*, 2015; Yan and Gu, 1996). Subsequently, the development of metal printing technologies such as selective laser melting (SLM), electron beam melting (EBM), and directed energy deposition (DED) enabled the production of structural parts with AM (Gibson *et al.*, 2015; Liu *et al.*, 2017).

The workflow of AM is similar to other computer numerical controlled (CNC) machining tools and is depicted in Figure 1.3. One starts with a computer file that describes the surface geometry of the part, e.g. an STL file as shown in Figure 1.3a (Wikipedia contributors, 2019). In an engineering context, this surface geometry is usually exported from computer-aided design (CAD) software. The surface geometry is then processed by 3D printing software that slices the object into layers and generates machine instructions (Figure 1.3b). In this step machine specific parameters can be set such as layer height, printing speed, laser power, etc. These instructions are transferred to the printer which will generate the part.

Two common printing processes are depicted in Figures 1.3c and 1.3d. Fused deposition modeling (FDM) is a printing technique that is mainly used to print thermoplastic polymers. In the most common variant, the material is provided as filament on a spool. This filament is fed through a nozzle where it is heated, gets molten, and deposited at the correct location with extrusion, reminiscent to a hot glue gun. After each layer is deposited, the base plate descends and the next layer is printed (Figure 1.3c). Due to the simplicity of the process, inexpensive low-end FDM machines are popular among hobbyists (Gibson *et al.*, 2015).

Selective laser melting (SLM) is a popular technique for metal AM, but can also used



(a) The part is described by its surface geometry, for example in the form of an STL file (Wikipedia contributors, 2019).

(b) The geometry is sliced into layers and exported to a format that can be handled by the printer.

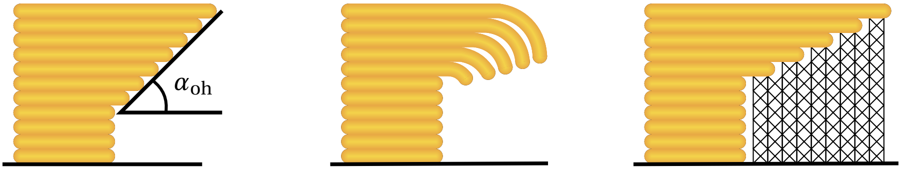
(c) In the FDM printing process, a continuous filament is fed to an extruder and heated, after which it is deposited on a previously printed layer.

(d) In the SLM printing process, material in powder form is deposited layer-by-layer, and melted selectively with a laser to fuse it to the previously printed layer.

Figure 1.3: Schematic overview of the processing and printing of the 3D printing benchmark #3DBenchy boat (Creative-Tools.com, 2019) using FDM (c) and SLM (d).

for polymer printing. Instead of providing the material with a filament, it is provided in powder form. First, a layer of metal powder is deposited onto the base plate. Then, a moving laser beam melts and fuses the powder in the selected locations where the part is to be built. Afterwards, the base plate descends and the process repeats (Figure 1.3d). This process is typically more expensive than FDM, and is mainly used in aerospace, automotive and medical industry as it enables printing in metal (Gibson *et al.*, 2015).

With conventional manufacturing techniques there is a clear correlation between manufacturing cost and product complexity: increased complexity usually requires more tooling steps, resulting in longer production time and higher costs. Moreover, beyond a certain geometric complexity it can be impossible to manufacture a part with conventional manufacturing techniques. For additive manufacturing, this is not the case. The production time is mainly determined by the part volume and height, and not by the shape of each layer. Furthermore, since a part is constructed in a layer-by-layer manner, tool accessibility or drafting angles are not required, resulting in a large design freedom. Therefore, the designer can now think about the most optimal configuration of material in the design space to meet the requirements for a part. As was stated by Hague *et al.* (2003): “with the unlimited geometry capability designers will therefore need to be



(a) The critical overhang angle α_{oh} is defined as the smallest manufacturable angle between a down-facing surface and the base plate.

(b) When the overhang angle is smaller than the critical overhang angle, the print will fail.

(c) In order to print small overhang angles properly, support structures are required. However, these can be costly to print and difficult to remove.

Figure 1.4: Definition of the critical overhang angle and support structures.

much more imaginative to make full use of the new manufacturing processes" (Hague *et al.*, 2003, p. 348). Instead, the designer can use TO to generate optimal designs.

1.3. THE OVERHANG CONSTRAINT

Although it is possible to manufacture highly complex components, AM does have some manufacturing constraints. One of the most prominent constraints is the overhang constraint, which is the focus of this thesis. With AM, each layer is deposited onto the previously built layer, and for most printing processes the distance with which a layer can extend over the edge of the previous layer is limited. The maximum extension distance is usually described by the critical overhang angle α_{oh}, which is the minimum angle between a down-facing surface and the base plate, as depicted in Figure 1.4a. In this thesis, surfaces that violate the critical overhang angle, i.e. have an overhang angle smaller than the critical overhang angle, are called *overhanging*. Designs that do not contain any overhanging surfaces are referred to as *self-supporting* or *overhang-free*. Violating the critical overhang angle can result in bad surface quality or part failure (Vandenbroucke and Kruth, 2007; Wang *et al.*, 2013). For example with FDM, the gravitational forces distort the part and drooping occurs, as can be seen in Figure 1.4b. With SLM, where the layers are resting on the powder bed, a lack of mechanical support and heat dissipation can distort the part (Cloots *et al.*, 2013; Mercelis and Kruth, 2006-10; Wang *et al.*, 2013). The actual value of the critical overhang angle is variable and depends on the process and the material used. Additionally, it is continually improved as the printing process is better understood and controlled. For SLM, the critical overhang angle is typically 45° (Wang *et al.*, 2013), while for FDM it can be much lower (Jiang *et al.*, 2018; Lee and Lee, 2017).

In order to print overhanging features, sacrificial support structures are required, as can be seen in Figure 1.4c. These scaffolds are printed with the original part, often in the same material, and have to be removed after the build. This adds printing and material cost, and removal is usually a manual job which can be time consuming. Furthermore, in complex parts, e.g. manifolds or heat exchangers, support structures might be required in internal cavities or channels from which supports cannot be removed. Therefore, to save printing time and cost, and to enable the manufacturing of complex components, designs should adhere to the overhang constraint. One option is to post-process the

(a) Front propagation methods can be used to simulate the wave front that originates from a rock thrown into a pond.

(b) The resulting arrival time field indicates for each location at what time the wavefront passed that location for the first time. Isolines of the arrival time field indicate the wavefront at a certain time instant.

Figure 1.5: Pond propagation.

topology optimized designs in order to adhere to overhang constraints. However, this is not always possible, and compromises the optimality of the design. It is also bad design practice, as manufacturing should be taken into account as early as possible in the design phase to save expenses (Boothroyd *et al.*, 2011). Consequently, in order to print a part without support structures, the overhang constraint should be included *into* the TO process, such that the geometries resulting directly from the TO do not require any support for printing.

In this thesis, the overhang constraint will be implemented as a filter into TO, and is technically an overhang *filter*. However, by applying the overhang filter the overhang is constrained, and we therefore use the terms overhang filter and overhang constraint interchangeably.

1.4. FRONT PROPAGATION

The methodology that is developed in this thesis to detect overhanging regions relies heavily on algorithms that have been developed to solve front propagation problems. The connection between front propagation problems and overhang detection is discussed in detail in Chapter 2 and 3, but in general it comes down to describing the generation of the part during 3D printing as a front that is propagated through space, tracing the material deposition with time. Some key concepts of front propagation are introduced to ease the understanding of the formulations given in the subsequent chapters.

Front propagation can be used in a wide variety of problems: path planning, image recognition, grid generation, and many more (Sethian, 1999). As an example one can think of the wavefront produced by a stone that is thrown into a pond, as shown in Figure 1.5a. The result of such a front propagation problem is the arrival time field $T(\mathbf{x})$, which describes the time at which the wavefront passed the location $\mathbf{x}$ (Figure 1.5b). The front is thus described implicitly by the arrival time field $T(\mathbf{x})$. At time t, the front equals the isoline in 2D, or the isosurface in 3D, that is described by $T(\mathbf{x}) = t$.

In order to obtain the arrival time field $T(\mathbf{x})$ for a front that propagates with equal speed in all directions, i.e. isotropic speed, the Eikonal equation (Sethian, 1996, 1999) is solved:

$$\begin{aligned} \|\nabla T(\mathbf{x})\| &= \frac{1}{F(\mathbf{x})}, \quad && \mathbf{x} \in \Omega, \\ T(\mathbf{x}) &= 0, && \mathbf{x} \in \partial\Omega_0, \end{aligned} \tag{1.2}$$

where Ω is the domain of propagation, $\partial\Omega_0$ is the initial position of the front, $\|...\|$ denotes the L^2 norm, and $F(\mathbf{x})$ is the speed function given as input, which defines the propagation speed of the front at location $\mathbf{x}$, and is strictly positive. The Eikonal equation is intuitive: the greatest increase in arrival time per unit distance, which is in the direction normal to the front, should be equal to the inverse of the propagation speed. In the case of the stone in the pond, if the propagation speed of the waves is assumed to be constant in the pond, the speed function reduces to a constant. However, if for example the wave speed is proportional to the depth of the pond, the speed function becomes a field quantity $F(\mathbf{x})$. In the subsequent chapters, a speed function is required that also is a function of propagation direction. By doing so, the speed function can be changed when the front propagates at an angle lower than the critical overhang angle, which can later be used to detect overhanging features. The speed function is then anisotropic, and a function of position $\mathbf{x}$ and the unit normal vector of the front $\mathbf{n} = \nabla T / \|\nabla T\|$, i.e. $F(\mathbf{x}, \mathbf{n})$.

The same problem can be viewed from a different perspective, which is of interest to optimal control theory (Sethian and Vladimirsky, 2001). Consider a vehicle at a certain position in time $\mathbf{y}(t)$, that moves in direction $\mathbf{a}(t)$, where $\mathbf{a}$ is a unit vector pointing in the direction of movement, with initial position $\mathbf{x}$, i.e. $\mathbf{y}(0) = \mathbf{x}$. Furthermore, the vehicle moves with speed $f(\mathbf{x})$ at location $\mathbf{x}$. The objective is to find the control $\mathbf{a}$ which minimizes the time at which each vehicle in the domain Ω arrives at $\partial\Omega_0$. In other words, find the fastest trajectory for each vehicle to reach the boundary $\partial\Omega_0$. This problem is governed by the Hamilton-Jacobi-Bellman (HJB) equation:

$$\begin{aligned} &\min_{\mathbf{a} \in S_{n-1}} \left\{ (\nabla T(\mathbf{x}) \cdot \mathbf{a}) f(\mathbf{x}) \right\} + 1 = 0, \qquad && \mathbf{x} \in \Omega, \\ &T(\mathbf{x}) = 0, && \mathbf{x} \in \partial\Omega_0, \end{aligned} \tag{1.3}$$

where $T(\mathbf{x})$ is the time it takes a vehicle at location $\mathbf{x}$ to reach $\partial\Omega_0$, n the dimensionality of the problem, and S_{n-1} is a unit ball ($S_{n-1} = \left\{\mathbf{a} \in \mathbb{R}^n : \|\mathbf{a}\| = 1\right\}$). The boundary condition is straightforward: the vehicles that are starting at the boundary have zero travel time. Then, for each location the travel direction is chosen that minimizes the increase in travel time (for a detailed derivation see Vladimirsky (2001)). Again, the speed can be a function of propagation direction as well: $f(\mathbf{x}, \mathbf{a})$.

Both the Eikonal and the HJB equation can describe the same problem, but the speed functions f and F are only equal if the they are independent of propagation direction: $F(\mathbf{x}) = f(\mathbf{x})$. When $f(\mathbf{x})$ does not depend on $\mathbf{a}$, the minimizer in Equation (1.3) is $\mathbf{a} = -\nabla T(\mathbf{x}) / \|\nabla T(\mathbf{x})\|$. Equation (1.3) can then be written as Equation (1.2) with $F(\mathbf{x}) = f(\mathbf{x})$. When the speed functions are dependent on propagation direction, $F(\mathbf{x}, \mathbf{n}) \neq f(\mathbf{x}, \mathbf{a})$, and the realation between Equation (1.2) and Equation (1.3) is less

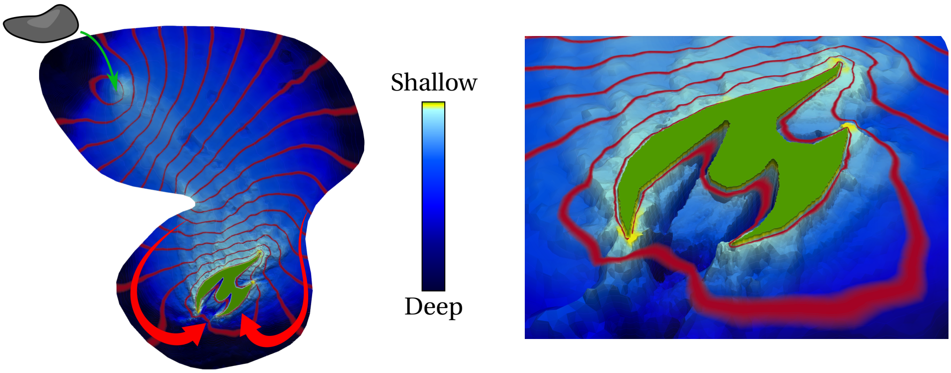


Figure 1.6: Due to the optimized depth profile of the pond, the waves (indicated by the red isolines) generated by throwing a rock at the spot indicated by the green arrow arrive simultaneously around the circumference of the TU Delft logo-shaped island (in green). On the side of the island facing the rock the pond is shallow, slowing the propagation. On the other side, as indicated by the red arrows, the front is sped up due to the increased depth allowing the front to arrive at the same time while traveling a larger distance. In the bay on the bottom of the island, there are two deep trenches to speed up the propagation, as can be seen in the right figure.

straightforward. For a detailed comparison the reader is referred to Vladimirsky (2001). In the subsequent chapters, the printing process is modeled using the HJB equation, as this simplifies the interpretation of the speed function. However, either perspective could have been used.

Although both the Eikonal and the HJB equation are non-linear equations, efficient numerical solution methods exist. These take advantage of the causality of the problem: the arrival time $T(\mathbf{x})$ is only influenced by locations with a shorter arrival time. Therefore, starting from the boundary, one can propagate the front by calculating the arrival times at nodes close to the current front, and advance the front by adding the node with the shortest arrival time to the front. For example, for an isotropic speed function, one can use the fast marching method, with a computational complexity of $O(N \log N)$ for a problem of N nodes (Sethian, 1996). For anistropic speed functions, the ordered upwind method can be used with similar computational complexity (Sethian and Vladimirsky, 2001). It is these efficient solution schemes that made it attractive to use front propagation to formulate an overhang constraint.

1.5. COUPLING FRONT PROPAGATION AND TOPOLOGY OPTIMIZATION

In order to use the front propagation together with topology optimization, there has to be a coupling between the two. In this thesis, that coupling is through the propagation speed. The speed $F(\mathbf{x})$ at which the front propagates is a function of the density field $\rho(\mathbf{x})$. Coming back to the example of the pond, one can for example imagine the density field to represent the depth of the pond: in deep regions the front propagates faster than in shallow regions. Now one can control the propagation of the front by setting the depth profile. For example, if we make an island in the pond in the shape of the flame of the TU Delft logo, we can optimize a depth profile such that the waves arrive simultaneously over the complete circumference of this island. A depth profile is then generated that slows the front down or speeds it up to give the desired arrival time at the shore of the island. The resulting depth profile is shown in Fig. 1.6. Compared to uniform propagation shown in Fig. 1.5b, the propagation is heavily distorted to wrap around the island. For the final depth profile, the average relative error of the arrival time over the circumference is only 0.05%.

In the remainder of this thesis the front propagation algorithm will be used in a more abstract sense, not directly related to a physical phenomenon as was the case for the pond example. However, the propagation speed will be controlled by the density field in exactly the same manner: in material regions ($\rho = 1$) the front propagates fast, while in void regions ($\rho = 0$), the front is slowed down.

Since the optimization that is used is gradient based, sensitivities of arrival times with respect to the density variables are required. The influence of increasing or decreasing the depth on the front propagation has to be known such that the gradient descent optimization algorithm can make the required changes to the depth profile. These sensitivities are calculated with the adjoint method, which requires a sweep over all the nodes in reverse order as in which the front propagated. That is because the depth of a certain location only influences the arrival time of the front at all the locations that the front passes after it has passed that location. By going in reverse order, the influence of changing the depth is accumulated back towards the origin of the front. Because the front propagation algorithms that make the use of front propagation attractive contain a minimum operator which is non-differentiable, there are specific cases in which the sensitivities can be inconsistent. This can be prevented by using a smooth-minimum operator, but it was found that the increased accuracy of the sensitivities has little influence on the overall optimization, while increasing the complexity of the algorithm, and it is therefore discarded (see Section 5.4.3). Because the sensitivity analyses for the front propagation requires only a single loop over all the nodes, it is extremely efficient, and therefore well-suited for integration within a topology optimization algorithm.

1.6. Research aim

When this research project started, only a few papers addressed an overhang constraint for topology optimization. However, none of the solutions proposed were practical, which is qualified as follows. First of all, the overhang constraint should be able to handle an arbitrary critical overhang angle. The critical overhang angle is process and material specific and should be given as input parameter to the overhang constraint. Second, the overhang constraint should be applicable to unstructured meshes. Optimization problems in industry often deal with irregular domains defined by CAD files, and are meshed with an unstructured grid. If the overhang constraint can be applied to unstructured meshes it can readily use the existing grid instead of requiring a mapping to a structured grid. Third, the overhang constraint should be robust. The constraint should not require a continuation scheme which requires tweaking for different load cases and design domains, but work for a fixed set of parameters. Finally, the overhang constraint should be numerically efficient and applicable to large-scale high resolution problems in 3D.

Therefore, the computational burden upon enabling the overhang constraint should be less than that of the conventional topology optimization. This is quantified as such that the overhang constraint and its sensitivity analysis should not exceed the computational time required by the FEA. Furthermore, its computational time should scale similarly or better with respect to number of nodes compared to the FEA. Therefore, the aim of this research is:

> *Develop a practical overhang constraint for topology optimization.*

1.7. SCOPE OF THIS THESIS

The focus of this thesis is on the overhang constraint, and the remaining aspects of topology optimization are kept as standard as possible. As stated in Section 1.1, density-based TO is used. Every optimization problem encountered in this thesis is a compliance minimization where for the FEA linear isotropic elasticity is assumed. This does not mean that the overhang constraint methods developed are limited to compliance minimization; it can in principle be applied to any density-based TO problem.

The overhang constraint developed in this thesis implements the strict rule presented in Section 1.3, that a down-facing surface cannot have an overhang angle lower than the critical overhang angle α_{oh}. This is a purely geometrical rule of thumb. In practice, overhanging features can sometimes be printable, for example when they are small. In order to model this, a more detailed physical simulation is required. That is however not the scope of this thesis, as the resulting constraint will be computationally much more expensive.

An important aspect when considering overhang for AM is the orientation in which the part is printed. Parts that have many overhanging surfaces in one orientation can be printable without support by simply rotating it. This is however not considered in this thesis for two reasons. First of all, considering part orientation during the optimization requires a measure of overhang, which makes it more natural to first develop an overhang constraint, and in future research combine it with an orientation optimization. Second, often the orientation of a part is already determined by the fixed features of a part. Holes, for example, can dictate part orientation to a large extent, because they often have to be printed without internal supports. Therefore, in this thesis only an overhang constraint is developed, which in future research could be combined with an orientation optimization.

1.8. THESIS OUTLINE

The research performed for this thesis resulted in four journal publications, each presented in a separate chapter (Chapters 2, 3, 5 and 6), and a section of a book chapter (Chapter 4). There will be some overlap, as each chapter briefly introduces TO and AM. The contents are graphically summarized in Figure 1.7, and is as follows:

- In Chapter 2, a novel overhang constraint is presented in a 2D setting, which uses front propagation to identify overhanging regions. The constraint is formulated in a continuous setting, and it is shown how established algorithms from the front propagation field can be utilized with the specific propagation function required

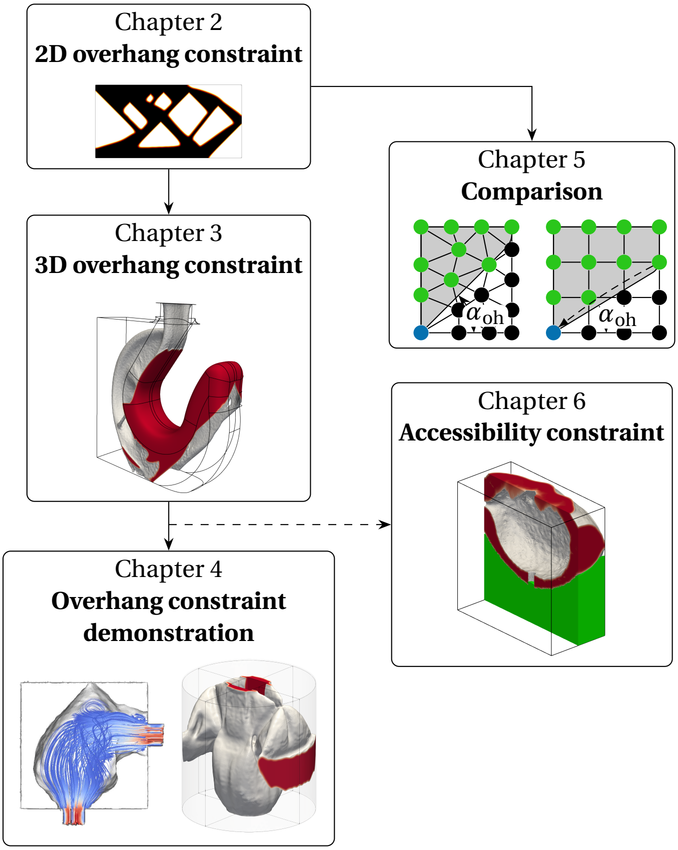


Figure 1.7: Graphical overview of the contents of this tehsis.

to detect overhang.

- The overhang constraint is extended to 3D in Chapter 3. The main challenges that are addressed are the efficient evaluation of the front propagation on tetrahedral meshes, and parallelization of the front propagation algorithm. The overhang constraint is demonstrated on large-scale problems on unstructured meshes.

- To demonstrate the wide range applicability of the 3D overhang constraint as presented in Chapter 3 on problems other than compliance minimization, it is applied to fluid flow optimization and compliant mechanism design in Chapter 4.

- In Chapter 5, the front propagation-based overhang constraint presented in Chapter 2 and Chapter 3 is compared to the overhang constraint presented by Langelaar (2017). Reformulation of the overhang constraint presented in Langelaar (2017) using front propagation leads to new insight that is used to improve the overhang constraint presented in Chapter 2 and Chapter 3.

- Finally, Chapter 6 presents a method to include the removability of supports. If supports are inaccessible, and thus cannot be removed after the printing process, the overhang constraint is activated, whereas supports that are accessible and thus can be easily removed are allowed. Although the overhang constraint that is presented in Chapter 2 and Chapter 3 is used, the methodology can be applied to any overhang constraint implementation.

REFERENCES

Bendsøe MP, Kikuchi N (1988) Generating optimal topologies in structural design using a homogenization method. Computer Methods in Applied Mechanics and Engineering 71(2):197–224

Bendsøe MP, Sigmund O (2004) Topology Optimization: Theory, Methods and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg

Boothroyd G, Dewhurst P, Knight WA (2011) Product design for manufacture and assembly, 3rd edn. CRC Press, Boca Raton, Fl

Christensen P, Klarbring A (2008) An Introduction to Structural Optimization, Solid Mechanics and Its Applications, vol 153. Springer Netherlands, Dordrecht

Cloots M, Spierings AB, Wegener K (2013) Thermomechanisches multilayer-modell zur simulation von eigenspannungen in SLM-proben. In: Hildebrand J (ed) Simlationsforum 2013 - Schweissen und Waermebehandlung, Weimar, Germany

Creative-Toolscom (2019) #3DBenchy. http://www.3dbenchy.com Accessed 11 April 2019

Gebhardt A, Hötter JS (2016) Additive Manufacturing: 3D Printing for Prototyping and Manufacturing. Carl Hanser Verlag, Munich

Gibson I, Rosen DW, Stucker B (2015) Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping and Direct Digital Manufacturing, 2nd edn. Springer, New York

Hague R, Mansour S, Saleh N (2003) Design opportunities with rapid manufacturing. Assembly Automation 23(4):346–356

Huang X, Xie YM (2010) Evolutionary Topology Optimization of Continuum Structures: Methods and Applications. Wiley, Chichester

Hull CW (1986) Apparatus for Production of Three-Dimensional Objects by Stereolithography. U.S. Patent 4575330A

Jiang J, Stringer J, Xu X, Zhong RY (2018) Investigation of printable threshold overhang angle in extrusion-based additive manufacturing for reducing support waste. International Journal of Computer Integrated Manufacturing 31(10):961–969

Langelaar M (2017) An additive manufacturing filter for topology optimization of print-ready designs. Structural and Multidisciplinary Optimization 55(3):871–883

Lee J, Lee K (2017) Block-based inner support structure generation algorithm for 3D printing using fused deposition modeling. The International Journal of Advanced Manufacturing Technology 89(5-8):2151–2163

Liu R, Wang Z, Sparks T, Liou F, Newkirk J (2017) 13 - Aerospace applications of laser additive manufacturing. In: Brandt M (ed) Laser Additive Manufacturing, Woodhead Publishing Series in Electronic and Optical Materials, Woodhead Publishing, pp 351–371

Mercelis P, Kruth J (2006-10) Residual stresses in selective laser sintering and selective laser melting. Rapid Prototyping Journal 12(5):254–265

Schneider PJ, Eberly DH (2003) Geometric Tools for Computer Graphics. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, Morgan Kaufmann Publishers, Amsterdam

Sethian JA (1996) A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Sciences 93(4):1591–1595

Sethian JA (1999) Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, 2nd edn. Cambridge Monographs on Applied and Computational Mathematics, Cambridge Univ. Press

Sethian JA, Vladimirsky A (2001) Ordered upwind methods for static Hamilton-Jacobi equations. Proceedings of the National Academy of Sciences 98(20):11,069–11,074

van Dijk NP, Maute K, Langelaar M, van Keulen F (2013) Level-set methods for structural topology optimization: A review. Structural and Multidisciplinary Optimization 48(3):437–472

Vandenbroucke B, Kruth JP (2007) Selective laser melting of biocompatible metals for rapid manufacturing of medical parts. Rapid Prototyping Journal 13(4):196–203

Vladimirsky AB (2001) Fast methods for static Hamilton-Jacobi Partial Differential Equations. PhD thesis

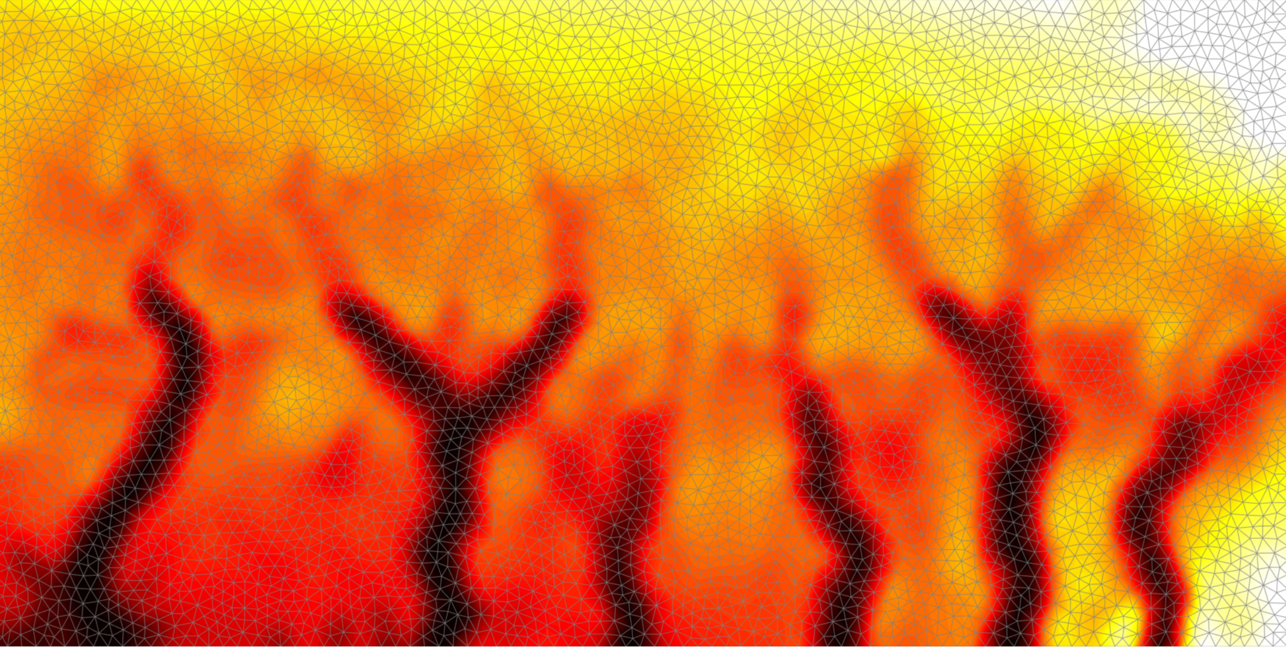
Wang D, Yang Y, Yi Z, Su X (2013) Research on the fabricating quality optimization of the overhanging surface in SLM process. The International Journal of Advanced Manufacturing Technology 65(9-12):1471–1484

Wikipedia contributors (2019) STL (file format) — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=STL_(file_format)&oldid=891935248 Accessed 11 April 2019

Womack JP, Jones DT, Roos D (1990) The Machine That Changed the World: Based on the Massachusetts Institute of Technology 5-Million Dollar 5-Year Study on the Future of the Automobile. Rawson Associates, New York

Xie Y, Steven G (1993) A simple evolutionary procedure for structural optimization. Computers & Structures 49(5):885–896

Yan X, Gu P (1996) A review of rapid prototyping technologies and systems. Computer-Aided Design 28(4):307–318



2

Front propagation-based overhang control in 2D

Additive manufacturing enables the nearly uncompromised production of optimized topologies. However, due to the overhang limitation, some designs require a large number of supporting structures to enable manufacturing. Because these supports are costly to build and difficult to remove, it is desirable to find alternative designs that do not require support. In this chapter, a filter is presented that suppresses non-manufacturable regions within the topology optimization loop, resulting in designs that can be manufactured without the need for supports. The filter is based on front propagation, can be evaluated efficiently, and adjoint sensitivities are calculated with almost no additional computational cost. The filter can be applied also to unstructured meshes and the permissible degree of overhang can be freely chosen. The method is demonstrated on several compliance minimization problems in which its computational efficiency and flexibility are shown. The current applications are in 2D, and the proposed method is readily extensible to 3D.

This chapter is published in *Structural and Multidisciplinary Optimization* 57(5), 2075–2091 (2018). A preliminary version where the overhang control is implemented as a constraint instead of a filter can be found in van de Ven *et al.* (2018).

2.1. INTRODUCTION

Topology optimized designs are often complex, containing many branches or small details. In most cases, the geometrical complexity of these designs cannot be accommodated with conventional manufacturing methods such as milling or casting. Additive manufacturing on the other hand, enables the production of complex parts, by creating a product layer upon layer. Although additive manufacturing offers greater form freedom, it also has manufacturing limitations, such as a minimum feature size, minimum slot distance, and a limitation on the inclination of downward facing surfaces, the overhang limitation (Thomas, 2009). The present study concentrates on the overhang limitation.

Most additive manufacturing processes, such as selective laser melting (SLM), fused deposition melting and stereo lithography, exhibit an overhang limitation. This is caused by the fact that each layer needs a certain amount of mechanical support or thermal conduction from the previously built layer, which limits the distance that a layer can extend unsupported over the layer underneath. Manufacturability is thus controlled by the angle between a down-facing surface and the base plate, the overhang angle, as defined in Figure 2.1b. The minimum overhang angle, α_{oh}, is the smallest manufacturable overhang angle. For SLM, this angle is mostly reported around 45° (Thomas, 2009), but varies for different process conditions (Cloots *et al.*, 2013; Wang *et al.*, 2013). Overhanging regions of a design with $\alpha < \alpha_{\text{oh}}$ can be built by adding support structures as displayed in Figure 2.1b. However, support structures increase the build time, add material cost, and their removal can be a difficult and costly task, especially for internal structures that are difficult to access.

Consequently, developing topology optimization methods that incorporate a minimum overhang angle became an active research topic. To the best of authors' knowledge, Brackett *et al.* (2011) were the first to investigate manufacturing constraints for additive manufacturing in a topology optimization context. They proposed a methodology to measure the overhang angle for evolutionary topology optimization, but this method has not been implemented. The first actual implementation was done by Gaynor *et al.* (2014), detailed in Gaynor and Guest (2016). A wedge shaped filter in combination with Heaviside projection was used to obtain self-supporting topologies. However, due to the non-linearity introduced by the overhang filter in combination with Heaviside projection, the number of iterations required for convergence was high. Subsequently, Langelaar (2017) presented an overhang restriction that evaluates the overhang angle on a

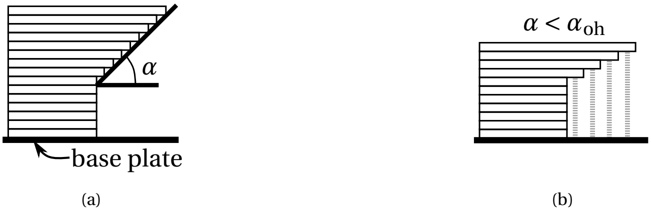


Figure 2.1: The overhang angle is defined as the angle α that a down-facing surface of the combined printing layers makes with respect to the base plate (a). Down-facing surfaces below the critical overhang angle need to be supported by support pillars, indicated with gray dashed lines (b).

structured mesh, where the amount of material below each element is used as a measure for overhang. Self-supporting designs were obtained in 2D as well as in 3D (Langelaar, 2016). However, the filter is only applicable to rectangular structured meshes, and α_{oh} can only be tuned by changing the element aspect ratio.

Both Gaynor and Guest (2016) and Langelaar (2016, 2017) evaluate the manufacturability in a global sense, following the layer by layer fashion of the manufacturing process. Other methods, that only constrain the overhang angle locally, have also been presented. Both Qian (2017) and Allaire *et al.* (2017a) proposed a geometrical overhang constraint by constraining the angle between the normal vector at the perimeter and the build direction. Qian (2017) uses density-based topology optimization in combination with a non-discreteness constraint to suppress intermediate densities, while in Allaire *et al.* (2017a) level-set-based topology optimization is used. Although both methods reduce the overhang, unmanufacturable sawtooth patterns are generated, due to the local nature of the methods. Finally, Guo *et al.* (2017) introduced an overhang constraint for moving morphable components and moving morphable voids. Although a large number of iterations is required, the resulting designs are overhang free. Unmanufacturable sawtooth patterns are avoided by preventing voids to overlap. Furthermore, the importance of build orientation is shown by including orientation in the optimization.

Besides the direct implementation of a minimum overhang angle as a design rule, other approaches have been proposed to limit the amount of support material required for manufacturing. Mirzendehdel and Suresh (2016) presented a constraint on the support structure volume. However, when no support was allowed, the discontinuous identification of overhanging surfaces seemed to result in a casting constraint, eliminating also the allowable overhanging surfaces. Although feasible, the results will most likely be sub-optimal. Finally, Allaire *et al.* (2017a,b) presented a constraint on the compliance of the intermediate shapes of a topology during the layer-by-layer manufacturing, which should constrain the overhang naturally. This is reflected in the results, where the amount of overhang is reduced. Although physics-based constraints have great potential by, e.g., predicting distortions during and after manufacturing, they tend to be computationally expensive, as the compliance of partly build designs has to be evaluated or approximated many times per iteration.

In order for a method to be of practical use in an industrial setting, it should meet the following requirements. First of all, the critical overhang angle should be adjustable, since this value varies according to the specific process conditions and the choice of material. Second, the overhang restriction should be able to work on unstructured meshes. In practical situations, the design domain is rarely rectangular and can contain holes and curved surfaces, which cannot be discritized with a structured mesh. Furthermore, the overhang restriction should be computationally inexpensive; its evaluation time and sensitivity analysis time should be of the same order, or lower, as the objective evaluation time, and should not add an excessive amount of iterations required to converge to the optimum layout. Finally, the overhang restriction should not contain parameters that need to be tuned for every optimization problem.

This article presents a method to control the angle of overhanging regions during topology optimization which addresses all the above mentioned requirements. Overhanging regions are identified by mimicking the layer upon layer manufacturing process.

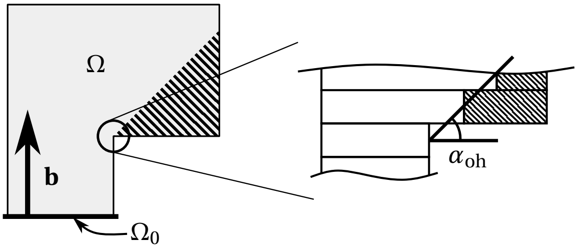


Figure 2.2: An example part which, when manufactured from the baseplate Ω_0 with build direction **b**, will have an overhanging region (shaded). The rate at which the layers can expand without failure defines the minimum overhanging angle α_{oh}.

Instead of adding discrete layers, the printing history is modelled as a continuous process with an advancing front. By employing efficient algorithms developed to solve front propagation problems in combination with adjoint sensitivities, the additional computation cost remains small. This method of overhang detection was first presented in van de Ven *et al.* (2018), where the overhang limitation was included as an additional constraint. In this paper, it is enforced through a filter, improving the robustness of the method. The formulation of the filter based on front propagation is dimension and mesh independent (Sethian, 1996), which allows for extension to 3D. For the sake of brevity and clarity of the discussion, the overhang restriction method and examples will be presented in a 2D setting only.

The next section introduces the overhang detection procedure, and the implementation thereof in topology optimization is discussed in Section 2.3. The numerical implementation is shown in Section 2.4, and numerical examples are presented in Section 2.5. Finally, conclusions are given in Section 2.6.

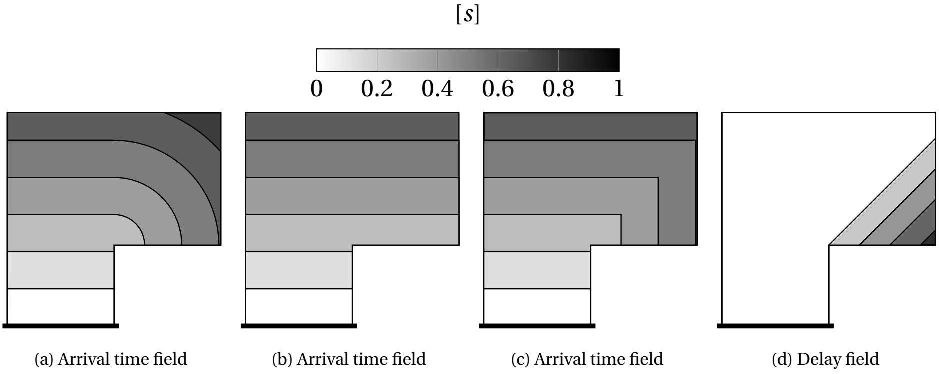
2.2. OVERHANG DETECTION

In this section, the overhang detection procedure based on front propagation, as presented in van de Ven *et al.* (2018), is presented. The resulting procedure is subsequently used in the topology optimization to eliminate overhanging regions.

2.2.1. OVERHANG DETECTION THROUGH FRONT PROPAGATION

Front propagation methods track an initial curve or surface Ω_0 as it evolves in space. This has a clear resemblance with the additive manufacturing process, where with every added layer, the boundary of the product advances. Instead of tracking the propagating front explicitly, the arrival time field of the propagation is calculated. The arrival time of a spatial point represents the time at which the front reaches that location. The front propagation can then be reconstructed by observing isolines of the arrival time field. How the front propagates, is ultimately determined by a speed function, which dictates the propagation of the front in each direction and location.

Consider the geometry given in Figure 2.2, that is to be printed on the base plate Ω_0, which coincides with the initial position of the front. When printed in the **b**-direction



(a) Arrival time field (b) Arrival time field (c) Arrival time field (d) Delay field

Figure 2.3: Contour plots of the arrival time fields for (a) an isotropic speed function, (b) an anisotropic speed function that gives equal arrival times per layer, and (c) an anisotropic speed function that delays the propagation in overhanging regions, and (d) for the delay field τ, from which the overhanging region can be identified.

with $\alpha_{\text{oh}} = 45°$, the shaded region will be overhanging, meaning that it will fail during printing. Although the complete extended region on the top-right is overhanging, from here on we will reserve the term overhang for regions that are not manufacturable due to the overhang limitation. The goal of the front propagation is to detect this region in a cost effective, robust, manner. When the front, initially at Ω_0, is propagated with an isotropic speed function within this geometry, it starts to curve around corners, as can be seen in Figure 2.3a. In order to obtain an arrival time field that represents the printing sequence of individual layers, the speed function is modified such that the front travels faster in directions deviating from the build direction. This increase in speed compensates for the larger distance to be traveled in the hanging region, so that the front stays parallel to base plate Ω_0 instead of curving, as illustrated in Figure 2.3b.

Finally, in order to detect overhang, the propagation speed is decreased when the front travels in a direction below α_{oh}, as shown in Figure 2.3c. The earliest possible arrival time, i.e. the minimum arrival time, for a point is the arrival time of a front that has traveled straight from the base plate towards that point, which is equal to the distance towards the base plate divided by the propagation speed:

$$T_{\min}(\mathbf{x}) = \frac{\mathbf{x} \cdot \mathbf{b}}{f(\mathbf{b})}, \tag{2.1}$$

where $\mathbf{b}$ is a unit vector parallel to the build direction, $f(\mathbf{b})$ is the propagation speed in that direction, and $\mathbf{a} \cdot \mathbf{b}$ denotes the inner product between vectors $\mathbf{a}$ and $\mathbf{b}$. In all non-overhanging regions, the arrival times are equal to the minimum arrival times, while in overhanging regions the arrival times exceed the minimum arrival time. Therefore, overhang is detected by observing the delay

$$\tau(\mathbf{x}) = T(\mathbf{x}) - T_{\min}(\mathbf{x}), \tag{2.2}$$

where $T(\mathbf{x})$ is the arrival time obtained through front propagation. When the delay $\tau = 0$, there is no overhang, and when $\tau > 0$ there is overhang (Figure 2.3d). This procedure can

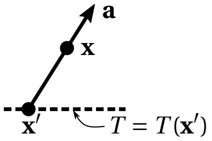


Figure 2.4: The calculation of arrival time for a point **x** from a known point **x**′.

be used on arbitrary geometries to detect overhanging regions as will be demonstrated in Section 2.5. In the following section the speed function required for the overhang detection will be proposed. This speed function is then used in the governing equations to obtain the arrival time field, as discussed thereafter in Section 2.2.3.

2.2.2. Anisotropic front propagation

As discussed in the previous section, the propagation speed is decreased when the front travels in directions below α_{oh}. This is done by making the speed function direction dependent. Consider a point $\mathbf{x}$, whose arrival time is to be calculated, as illustrated in Figure 2.4. The arrival time is updated from a given point $\mathbf{x}'$ on the front, where the arrival times are known. Finding $\mathbf{x}'$, from where $\mathbf{x}$ is to be updated, is covered in Section 2.2.3. The new arrival time can be calculated with

$$T(\mathbf{x}) = T(\mathbf{x}') + \frac{\|\mathbf{x}-\mathbf{x}'\|}{f(\mathbf{a})}, \tag{2.3}$$

where $\mathbf{a} = (\mathbf{x}-\mathbf{x}')/(\|\mathbf{x}-\mathbf{x}'\|)$ is a unit vector pointing from $\mathbf{x}'$ to $\mathbf{x}$, and $f(\mathbf{a})$ is the speed function, dependent on the direction of the update. $\|\ldots\|$ is used throughout the paper to denote the L^2 norm. The update direction is defined as $\alpha = \pi/2 - \arccos(\mathbf{a}\cdot\mathbf{b})$. Let us first consider a speed function that results in equal arrival times per layer as in Figure 2.3b. The time difference between two points should match the distance between the points projected on the build direction, divided by the propagation speed in the build direction f_0:

$$T(\mathbf{x}) - T(\mathbf{x}') = \frac{\left(\mathbf{x}-\mathbf{x}'\right)\cdot\mathbf{b}}{f_0}. \tag{2.4}$$

f_0 is a constant that simply scales the arrival time field and is set to 1m/s. By combining Equation (2.3) and Equation (2.4), the speed function becomes:

$$f_1(\mathbf{a}) = \frac{f_0\|\mathbf{x}-\mathbf{x}'\|}{(\mathbf{x}-\mathbf{x}')\cdot\mathbf{b}} = \frac{f_0}{\mathbf{a}\cdot\mathbf{b}}. \tag{2.5}$$

In order to obtain an arrival time field as shown in Figure 2.3c, the propagation is delayed in overhanging regions. This is achieved by decreasing the speed function whenever the update direction $\mathbf{a}$ is below the critical overhang angle, i.e. when $\alpha < \alpha_{\text{oh}}$ or $\alpha > \pi - \alpha_{\text{oh}}$, or equivalently $\mathbf{a}\cdot\mathbf{b} < \sin(\alpha_{\text{oh}})$. The speed can be decreased in numerous ways, and for numerical reasons detailed in Section 2.4, the speed function for propagation in directions below α_{oh} is chosen as:

$$f_2(\mathbf{a}, \alpha_{\text{oh}}) = \frac{f_0}{\tan(\alpha_{\text{oh}})\,\|\mathbf{Pa}\|}, \tag{2.6}$$

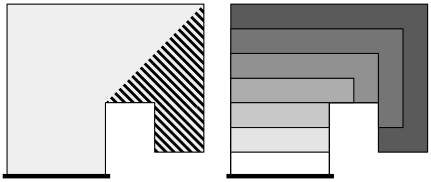


Figure 2.5: When geometries containing hanging sections, downward propagation is required. These areas are by definition overhanging.

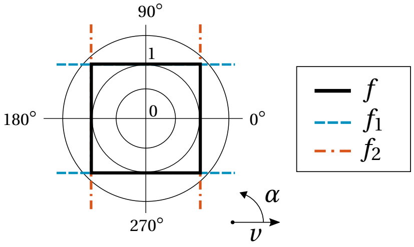


Figure 2.6: Polar plot of the speed function f for $\alpha_{\text{oh}} = 45°$, and its components f_1 and f_2. The tangential axis represents propagation direction, and the radial axis represents propagation speed.

where $\mathbf{P} = (\mathbf{I} - \mathbf{b} \otimes \mathbf{b})$, and $\mathbf{a} \otimes \mathbf{b}$ denotes the outer product between the vectors $\mathbf{a}$ and $\mathbf{b}$. It can be shown that $f_2 < f_1$ when $\mathbf{a} \cdot \mathbf{b} < \sin(\alpha_{\text{oh}})$ and $f_1 = f_2$ when $\mathbf{a} \cdot \mathbf{b} = \sin(\alpha_{\text{oh}})$, hence decreasing the speed for propagation below α_{oh}.

So far, only upwards updates, where $\mathbf{a} \cdot \mathbf{b} \geq 0$, have been considered. The speed function should also be defined when the direction of propagation is downwards, i.e. when $\mathbf{a} \cdot \mathbf{b} < 0$. Downward propagation might happen in hanging regions, which by definition are overhanging, as can be seen in Figure 2.5. The downward propagation speed can in principal be chosen freely as long as it is greater than zero, since there will always be a delay because the front has to cover additional distance to reach hanging areas from within the structure. For numerical convenience, the downward profile is chosen identical to the upward propagation profile. The speed function is then

$$f(\mathbf{a}, \alpha_{oh}) = \begin{cases} |f_1| & \begin{array}{l} \mathbf{a} \cdot \mathbf{b} \geq \sin(\alpha_{\text{oh}}) \text{ or} \\ \mathbf{a} \cdot \mathbf{b} \leq -\sin(\alpha_{\text{oh}}) \end{array} \\ f_2 & -\sin(\alpha_{\text{oh}}) < \mathbf{a} \cdot \mathbf{b} < \sin(\alpha_{\text{oh}}) \end{cases} . \tag{2.7}$$

This can be rewritten as

$$f(\mathbf{a}, \alpha_{oh}) = \frac{f_0}{\max(\tan(\alpha_{oh}) \|\mathbf{Pa}\|, |\mathbf{b} \cdot \mathbf{a}|)}. \tag{2.8}$$

This gives a speed function with a rectangular shape when displayed in polar coordinates, as can be seen in Figure 2.6, suitable for overhang detection. The effect of the minimum overhang angle on the speed function is displayed in Figure 2.7: lower minimum overhang angles widen the speed function, increasing the anisotropy.

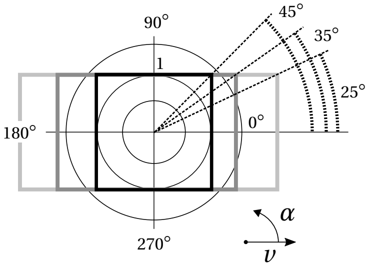


Figure 2.7: Polar plot of the speed function for α_{oh} equals 45°, 35°, and 25°. The tangential axis represents propagation direction, and the radial axis represents propagation speed.

2.2.3. GOVERNING EQUATIONS

Using Equation (2.3) and Equation (2.8), the arrival time at a point $\mathbf{x}$ can be calculated given a point $\mathbf{x}'$ with a known arrival time. In order to obtain the arrival time field, each point should be updated from the direction that results in the earliest arrival time (i.e., the direction from which the front reaches the point first). Therefore, Equation (2.3) is minimized for all directions $\mathbf{a} \in S_1$, $S_1 = \{\mathbf{a} \in \mathbf{R}^n \mid \|\mathbf{a}\| = 1\}$, where n is the dimensionality of the problem. By doing so and linearising around $\mathbf{x}$, the front propagation problem can be described as a boundary value problem governed by the Hamilton-Jacobi-Bellman equation, which is solved for T:

$$\begin{aligned} &\min_{\mathbf{a}\in S_1}\left\{(\nabla T(\mathbf{x})\cdot\mathbf{a})\, f(\mathbf{x},\mathbf{a})\right\}+1=0, && \mathbf{x}\in\Omega, \\ &T(\mathbf{x})=0, && \mathbf{x}\in\partial\Omega_0, \end{aligned} \tag{2.9}$$

where Ω is the interior of the domain and $\partial\Omega_0$ is the (partial) boundary of the domain at which the front is initiated. At the initial boundary, the arrival times are set to zero, and from there, arrival times can be progressively calculated throughout the domain, by which the front is advanced. Effectively, the front is advanced by calculating for every location the fastest path to the known front.

Instead of calculating the fastest path towards the front, another perspective is to expand the front and calculate the time it takes to reach each location. The front is expanded by the speed function $F(\mathbf{x},\mathbf{n})$, dependent on the normal direction of the front, which is determined by the gradient of the arrival times: $\mathbf{n} = \nabla T / \|\nabla T\|$. The norm of the gradient ∇T determines how fast the arrival time changes spatially, and has to be equal to the reciprocal of the speed function. This gives the governing equation

$$\begin{aligned} &\|\nabla T\| = \frac{1}{F(\mathbf{x},\mathbf{n})}, && \mathbf{x}\in\Omega, \\ &T(\mathbf{x})=0, && \mathbf{x}\in\partial\Omega_0. \end{aligned} \tag{2.10}$$

Note that the speed function F is generally not equal to the speed function f. For a detailed relation the reader is referred to Vladimirsky (2001), but F can interpreted as the speed of the front in the normal direction (semi-Lagrangian perspective) while f is the speed for an individual particle (Eulerian perspective), which do not coincide when the speed function is anisotropic.

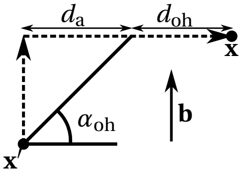


Figure 2.8: Given a non-overhang point $\mathbf{x}'$, the material in the next layer above $\mathbf{x}'$ is printable if the horizontal distance to $\mathbf{x}'$ is not larger than d_a. Overhang is measured by the distance to the closest manufacturable point in the same layer, indicated by d_{oh}.

Solving either Equation (2.9) or Equation (2.10) yields the same result, but for Equation (2.10) a root finding problem needs to be solved locally, while for Equation (2.9) this is a minimization problem. One or the other might be easier to solve depending on the speed functions f and F. For the speed function given in Equation (2.8), the local minimization problem can be solved efficiently as will be shown in Section 2.4. Therefore, this study will focus on solving the front propagation with the Hamilton-Jacobi-Bellman equation.

2.2.4. INTERPRETATION OF THE DELAY FIELD

With the speed function given in Equation (2.8), the resulting delay field has a physical interpretation. The delay at a point $\mathbf{x}$ is proportional to the distance to the closest non-overhanging, or manufacturable, point in the layer in which $\mathbf{x}$ is printed, i.e. in the direction orthogonal to $\mathbf{b}$. f^* is a speed that relates the delay to this distance:

$$\tau(\mathbf{x}) = f^* d_\mathrm{oh}(\mathbf{x}), \tag{2.11}$$

where d_oh is the distance from $\mathbf{x}$ to the closest manufacturable point in the same layer. For manufacturable points, the distance to the closest manufacturable point, i.e. to itself, $d_\mathrm{oh} = 0$, and the delay of manufacturable points is zero.

Now consider a point $\mathbf{x}$ that is updated from a manufacturable point $\mathbf{x}'$, as depicted in Figure 2.8. The delay $\tau(\mathbf{x})$ is calculated with Equations (2.1) to (2.3), and should be proportional to d_oh:

$$T(\mathbf{x}') + \frac{\|\mathbf{x}-\mathbf{x}'\|}{f(\mathbf{a})} - T_\mathrm{min}(\mathbf{x}) = f^* d_\mathrm{oh} \tag{2.12}$$

Assuming that $\mathbf{x}'$ is manufacturable gives $T(\mathbf{x}') = T_\mathrm{min}(\mathbf{x}') = \mathbf{x}'\cdot\mathbf{b}/f_0$ (Equation (2.1)). Furthermore, $T_\mathrm{min}(\mathbf{x}) = \mathbf{x}\cdot\mathbf{b}/f_0$. Simple trigonometry gives

$$d_\mathrm{oh} = \left\|\mathbf{P}(\mathbf{x}-\mathbf{x}')\right\| - d_\mathrm{a} \tag{2.13}$$

$$= \left\|\mathbf{P}(\mathbf{x}-\mathbf{x}')\right\| - \frac{\mathbf{b}\cdot(\mathbf{x}-\mathbf{x}')}{\tan(\alpha_\mathrm{oh})} \tag{2.14}$$

Then, by combining equations Equation (2.12) and Equation (2.13) the following expression for the speed function is obtained:

$$f(\mathbf{a}) = \frac{\|\mathbf{x}-\mathbf{x}'\|}{f^*\|\mathbf{P}(\mathbf{x}-\mathbf{x}')\| - f^*\frac{\mathbf{b}\cdot(\mathbf{x}-\mathbf{x}')}{\tan(\alpha_\mathrm{oh})} + \frac{\mathbf{b}\cdot(\mathbf{x}-\mathbf{x}')}{f_0}}. \tag{2.15}$$

By choosing $f^* = \tan(\alpha_{\text{oh}})/f_0$, this reduces to

$$f(\mathbf{a}) = \frac{f_0}{\tan(\alpha_{\text{oh}})\|\mathbf{Pa}\|}, \tag{2.16}$$

which is equal to the speed function for overhanging regions (Equation (2.6)). The delay of a point $\mathbf{x}$ is thus proportional to the distance of $\mathbf{x}$ to the closest manufacturable point in the layer in which $\mathbf{x}$ is printed.

2.3. INTEGRATION IN TOPOLOGY OPTIMIZATION

With the overhang detection procedure outlined in the previous section, an overhang filter for topology optimization is formulated. In van de Ven *et al.* (2018), this overhang detection procedure was used in an explicit overhang constraint. However, this required the introduction of several additional parameters and constraint aggregation, resulting in some constraint violations. Therefore in this work, α_{oh} will be enforced implicitly through a filter, as has been done for the overhang constraints in Gaynor and Guest (2016) and Langelaar (2017). The filter will be integrated in a density based topology optimization (Bendsøe and Sigmund, 2004). A schematic of the optimization flowchart is given in Figure 2.9a. First, the design variables ρ are filtered (Bruns and Tortorelli, 2001) to control length scale and to prevent checkerboarding (Sigmund and Petersson, 1998):

$$\rho^*_j = \sum_i \omega_{ij}\rho_i / \sum_i \omega_{ij}, \tag{2.17}$$

$$\omega_{ij} = \max(r - \|\mathbf{x}_i - \mathbf{x}_j\|, 0), \tag{2.18}$$

where ρ^*_j is the filtered density at position $\mathbf{x}_j$ and r is the filter radius. The filtered densities ρ^* define the geometry on which the overhang is detected with front propagation. This results in the printable densities ξ, which are used for the objective and constraint evaluation. Finally, the sensitivities are calculated and the design variables updated.

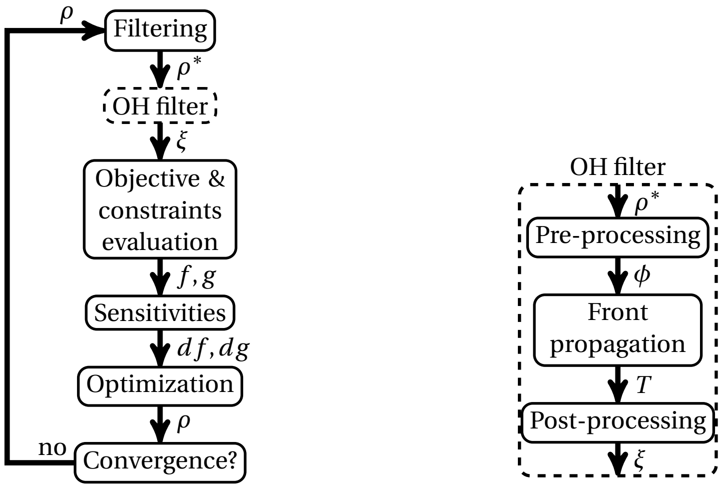
2.3.1. THE OVERHANG FILTER

The overhang filter, as indicated in Figure 2.9b, comprises of three steps. First, the filtered densities ρ^* are pre-processed. Then, the front propagation is performed which gives the arrival time field T. Lastly, the arrival times are post-processed to obtain the printable densities ξ (Figure 2.9b). These three steps are discussed in detail in the following sections.

PRE-PROCESSING AND FRONT PROPAGATION

In density based topology optimization, the topology is defined by a pseudo-density field that determines the amount of material at every location. In order to capture this topology in the front propagation, the propagation speed is scaled by the local filtered density ρ^*. Furthermore, a lower bound for the scaling, v_{void}, is introduced to prevent infinite arrival times in void regions. For simplicity, a linear interpolation is used, which gives the speed scaling field

$$\phi(\mathbf{x}) = v_{\text{void}} + (1 - v_{\text{void}})\rho^*(\mathbf{x}), \tag{2.19}$$



(a)

(b)

Figure 2.9: Implementation of the overhang filter. The overhang filter (b) is added after the density filter, and all subsequent steps are performed on the printable densities ξ (a).

where typically $v_{\mathrm{void}} = 0.1$ is used. The speed function (Equation (2.8)) is, again for simplicity, linearly scaled by this field giving

$$f_s(\phi(\mathbf{x}), \mathbf{a}, \alpha_{oh}) = \phi(\mathbf{x}) f(\mathbf{a}, \alpha_{\mathrm{oh}}). \tag{2.20}$$

The front propagation is performed with this scaled speed function.

POST-PROCESSING

After the front propagation is conducted, the delay field (Equation (2.2)) can be constructed, given by

$$\tau(\mathbf{x}) = T(\mathbf{x}) - \frac{\mathbf{x} \cdot \mathbf{b}}{f(\mathbf{b})} = T(\mathbf{x}) - \frac{\mathbf{x} \cdot \mathbf{b}}{f_0}. \tag{2.21}$$

The delay is zero for manufacturable regions and larger than zero for overhanging regions. In order to compare printable densities with the original densities, a dimensionless field is required that is 1 for manufacturable regions and between 0 and 1 for overhanging regions. Therefore, the following function is used to map the arrival time delay to printability:

$$\xi(\mathbf{x}) = 2^{-k\tau(\mathbf{x})}, \tag{2.22}$$

where $k[\mathrm{s}^{-1}]$ controls how rapidly printability decreases with an increasing delay. It is defined as follows:

$$k = f_0 / \beta. \tag{2.23}$$

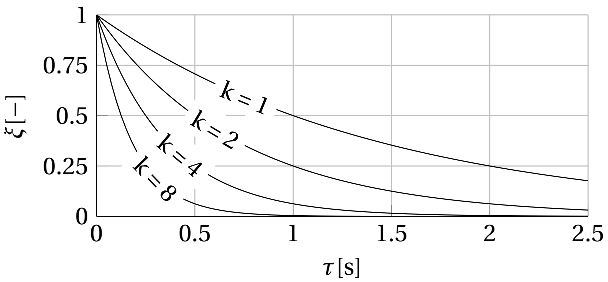


Figure 2.10: The relation between the delay field τ and the printable densities ξ for different values of k.

Because of the negative power of 2 in Equation (2.22), β[m] can be interpreted as the typical length after which the printable density of an overhanging part is halved. The relation between ξ and τ for different values of k is displayed in Figure 2.10. By increasing k, sharper edges and finer details are obtained, lower values of k can result in smoother convergence. β is typically chosen as $h/4$, where h is the typical element length.

INITIAL CONDITION FOR THE FRONT PROPAGATION

The arrival times are initialized at the base plate for the preferred building direction. Although the boundary on the base plate is manufacturable, as it is completely supported by the base plate, the arrival times are not initialized at 0, but with a slight offset T_0 proportional to the density:

$$T(\mathbf{x}) = (1 - \rho^*(\mathbf{x}))\,T_0, \qquad \mathbf{x} \in \partial\Omega_0. \tag{2.24}$$

Without this offset, the delay τ on the bottom layer will be 0, regardless of the density value. The printable densities ξ on the base plate will then be 1 (Equation (2.22)), resulting in a permanent layer of material on the base plate. With this offset, the initial arrival times are dependent on the local densities, and when the densities are 0, the delay $\tau = T_0$. T_0 is chosen such that a sufficiently small printable density is obtained when the local density is zero. For example, $T_0 = 8/k$, results in an acceptable $\xi = 0.0039$ when $\rho^* = 0$ (Equation (2.22)).

2.4. NUMERICAL IMPLEMENTATION

Efficient evaluation of the front propagation problem and its sensitivities is of paramount importance for its feasibility as an overhang filter. Fortunately, the directionality of the front propagation problem allows for a fast calculation of the arrival time field: because the arrival time at one location can only influence locations with a higher arrival time, the arrival times can be calculated using single-pass methods. These methods start at the boundary, and propagate the front by calculating arrival times in ascending order from the boundary. In principal, the arrival time at every location only needs to be evaluated once, hence the name single-pass. For the evaluation of an arrival time only a local problem is solved, resulting in a close to linear scaling of the algorithm with a computational complexity of $O(N \log N)$. For isotropic speed functions, the

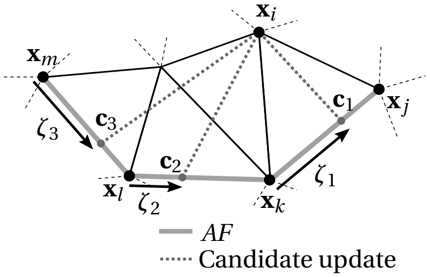


Figure 2.11: The arrival time of node $\mathbf{x}_i$ in a triangular mesh is updated from the *AF*. On each line segment on the *AF*, the point $\mathbf{c}$ that gives the fastest arrival time is determined, and the lowest arrival time resulting from the points $\mathbf{c}$ is accepted.

Fast Marching Method has been developed (Sethian, 1996), which is commonly used in, among others, the level-set method (Sethian and Wiegmann, 2000), but also in other optimization settings (e.g. van Keulen *et al.* (2008)). The Fast Marching Method has been expanded into the Ordered Upwind Method (OUM) (Sethian and Vladimirsky, 2003) for anisotropic speed functions. Furthermore, iterative methods have been developed, called fast-sweeping methods, and mixtures of marching and sweeping methods. Additionally, parallelized methods are available. However, since the performance of the OUM is sufficient and its implementation is straightforward, no alternatives have been considered.

2.4.1. ORDERED UPWIND METHOD

For the sake of completeness, the OUM will be briefly explained, following Sethian and Vladimirsky (2003). From here on we consider a 2D setting with triangular elements. Consider a discretized domain with N nodes. Node i is located at $\mathbf{x}_i$, and field quantities at node i are denoted with a subscript, e.g. ρ_i. In the OUM, each node is labelled as being either *Far*, *Considered* or *Accepted*. Initially, all the nodes are labelled *Far*, except for the initial boundary nodes which are labelled *Accepted*. Each iteration can be divided into three steps:

1. Move all the nodes that are in *Far* and adjacent to an *Accepted* node to *Considered*.
2. Evaluate the arrival times of the nodes in *Considered*, using the *Accepted* nodes.
3. Move the node in *Considered* with the earliest arrival time to *Accepted*.

This process is repeated until all the nodes are in *Accepted*. In Step 2, the arrival times of *Considered* nodes are calculated using the current front. The current front is defined in 2D as the set of line segments $\mathbf{x}_j\mathbf{x}_k$, for which $\mathbf{x}_j$ and $\mathbf{x}_k$ are adjacent to each other, in *Accepted*, but also adjacent to one or more *Considered* nodes. Nodes that fulfill these three requirements are said to be in the *AcceptedFront* (*AF*). In order to calculate the arrival time of a *Considered* node $\mathbf{x}_i$, the current front is scanned to see for which location on the front the travel time to the node in question is the shortest, as displayed in Figure 2.11. Only a small part of the current front, which is close enough to $\mathbf{x}_i$ to possibly provide the earliest arrival time, needs to be considered. This is the so-called *near front* (NF) of $\mathbf{x}_i$:

$$\begin{aligned} \mathrm{NF}(\mathbf{x}_i) = \{\mathbf{x}_j\mathbf{x}_k \in AF \,|\, \exists \tilde{\mathbf{x}} \text{ on } \mathbf{x}_j\mathbf{x}_k \\ \text{s.t.} \|\tilde{\mathbf{x}} - \mathbf{x}_i\| \le hG_2/G_1\}, \end{aligned} \tag{2.25}$$

where h is the typical mesh diameter, and G_1 and G_2 are the lower and upper bound of the speed function f_s, respectively. G_2/G_1 is a measure for the anisotropy of the speed function. The arrival time at $\mathbf{x}_i$ is updated from the segment of $\mathrm{NF}(\mathbf{x}_i)$ that gives the lowest arrival time:

$$T_i = \min_{\mathbf{x}_j\mathbf{x}_k \in \mathrm{NF}(\mathbf{x})} V_{\mathbf{x}_j\mathbf{x}_k}(\mathbf{x}_i), \tag{2.26}$$

where $V_{\mathbf{x}_j\mathbf{x}_k}(\mathbf{x}_i)$ is the upwind approximation of T_i when calculated from line segment $\mathbf{x}_j\mathbf{x}_k$. $V_{\mathbf{x}_j\mathbf{x}_k}(\mathbf{x}_i)$ can be evaluated from either the semi-Lagrangian (Equation (2.9)) or the Eulerian (Equation (2.10)) perspective. As stated in Section 2.2.3, the semi-Lagrangian perspective is used in this work, but similar results can be achieved using the Eulerian perspective. Following Sethian and Vladimirsky (2003), $V_{\mathbf{x}_j\mathbf{x}_k}(\mathbf{x}_i)$ is approximated with:

$$V_{\mathbf{x}_j,\mathbf{x}_k}(\mathbf{x}_i) = \min_{\zeta \in [0,1]} \left\{ \frac{\chi(\zeta)}{f_s(\phi_i, \mathbf{a}_\zeta, \alpha_{oh})} + \zeta T_j + (1-\zeta) T_k \right\}, \tag{2.27}$$

where $\chi(\zeta) = \left\| \mathbf{x}_i - \mathbf{c}_\zeta \right\|$, and $\mathbf{a}_\zeta = \left(\mathbf{x}_i - \mathbf{c}_\zeta \right) / \chi(\zeta)$. $\mathbf{c}_\zeta = \zeta \mathbf{x}_j + (1-\zeta)\mathbf{x}_k$, which is a point on the segment $\mathbf{x}_j\mathbf{x}_k$ determined by ζ. For example, in order to calculate T_i for a *Considered* point $\mathbf{x}_i$, as displayed in Figure 2.11, the lowest possible arrival time for each segment on the *AF* is determined by solving Equation (2.27). Then, the update that gives the earliest arrival time is accepted (Equation (2.26)). Due to the anisotropy of the speed function, this is often not the closest point and the update might even cross several elements (as is, for example, the case for the update from $\mathbf{c}_2$ and $\mathbf{c}_3$ in Figure 2.11). Nonetheless, the speed function f_s is assumed to be constant, as its only spatially varying argument, the speed scaling field ϕ, is only evaluated at the target location $\mathbf{x}_i$ (Equation (2.27)). No instabilities have been observed related to this approximation, but one could make a more precise approximation by interpolating ϕ over the update path. However, Equation (2.27) will become more difficult to solve, and the sensitivities will be less local as the arrival time will then depend on the densities of all the nodes from the elements that are crossed.

Although Equation (2.27) is evaluated for every segment in the near front, the eventual arrival time T_i will only depend on the earliest upwind approximation $V_{\mathbf{x}_j,\mathbf{x}_k}(\mathbf{x}_i)$. Therefore, T_i is only a function of the arrival times that appear in $V_{\mathbf{x}_j,\mathbf{x}_k}(\mathbf{x}_i)$, and the speed scaling field at $\mathbf{x}_i$, ϕ_i. For brevity, Equation (2.26) is written as

$$T_i = V(T_j, T_k, \phi_i) \overset{\text{def}}{=} V_i. \tag{2.28}$$

The minimization problem given in Equation (2.27) needs to be solved multiple times for every node that is updated. Therefore, solving it efficiently is crucial for a computationally fast overhang filter. The second and third term of Equation (2.27) are linear with ζ. With the speed function given in Equation (2.20), the first term of Equation (2.27) can be rewritten as

$$\frac{\chi(\zeta)}{f_s(\phi_i, \mathbf{a}_\zeta, \alpha_{oh})} = \chi(\zeta)\frac{\max\left(\tan(\alpha_{oh})\|\mathbf{P}\mathbf{a}_\zeta\|, |\mathbf{b}\cdot\mathbf{a}_\zeta|\right)}{\phi_i}, \tag{2.29}$$

$$= \frac{\max\left(\tan(\alpha_{oh})\|\mathbf{P}(\mathbf{x}-\mathbf{c}_\zeta)\|, |\mathbf{b}\cdot(\mathbf{x}-\mathbf{c}_\zeta)|\right)}{\phi_i}, \tag{2.30}$$

with only $\mathbf{c}_\zeta$ linearly dependent on ζ. In two dimensions, the arguments of the maximum function are piecewise linear functions of ζ. The other terms in Equation (2.27) are also linear functions of ζ. Consequently, Equation (2.27) is piecewise linear in ζ, and the minimum will be either at the edges of the domain ($\zeta = 0$ or $\zeta = 1$), or at the intersection points of two piecewise linear sections. This will only occur when the two arguments of the maximum function are equal:

$$\tan(\alpha_{oh})\|\mathbf{P}(\mathbf{x}-\mathbf{c}_\zeta)\| = |\mathbf{b}\cdot(\mathbf{x}-\mathbf{c}_\zeta)| \tag{2.31}$$

In two dimensions, this gives two possible intersection points:

$$\zeta_1 = \frac{-\tan(\alpha_{oh})\|\mathbf{P}(\mathbf{x}_k-\mathbf{x})\| - \mathbf{b}\cdot(\mathbf{x}_k-\mathbf{x})}{\tan(\alpha_{oh})\|\mathbf{P}(\mathbf{x}_j-\mathbf{x}_k)\| + \mathbf{b}\cdot(\mathbf{x}_j-\mathbf{x}_k)} \tag{2.32}$$

$$\zeta_2 = \frac{-\tan(\alpha_{oh})\|\mathbf{P}(\mathbf{x}_k-\mathbf{x})\| + \mathbf{b}\cdot(\mathbf{x}_k-\mathbf{x})}{\tan(\alpha_{oh})\|\mathbf{P}(\mathbf{x}_j-\mathbf{x}_k)\| - \mathbf{b}\cdot(\mathbf{x}_j-\mathbf{x}_k)} \tag{2.33}$$

Therefore, the minimum in Equation (2.27) is efficiently obtained by simply evaluating the minimization problem at $\zeta = (0, 1, \zeta_1, \zeta_2)$, and accepting the minimum value that lies on the interval (i.e. $0 \le \zeta \le 1$).

Summarizing, in each iteration of the algorithm, the node with the lowest arrival time is added to the set of *Accepted* nodes, and its neighboring nodes' arrival times are updated. The arrival time of a node is updated by scanning the front within a certain radius of that node for the shortest arrival time (Equation (2.26)). The minimum arrival time on a segment is found by solving Equation (2.27), which can be done by probing the line segment at four locations. In 3D, the front is represented by a surface, and the arrival time of a node is updated from a surface instead of a line. The minimization problem presented in Equation (2.26) is therefore more complex, and will be elaborated in a separate paper.

2.4.2. SENSITIVITIES

The sensitivities are derived from the descritized equations as outlined in Equations (2.25) to (2.32). By doing so, the computational effort for the sensitivities becomes negligible; only one loop over all the nodes is required as will become clear in Section 2.4.3. However, in specific cases this can result in erroneous sensitivities due to the non-differentiability of the minimum operator in Eq. (2.26). This can be circumvented by introducing a smooth minimum operator, but this is not implemented since in practice it has little effect on the algorithm's performance (see Section 5.4.3).

The sensitivities are given for a general function g which is a function of the printable densities ξ. This could be either the objective or a constraint function. The sensitivities with respect to the arrival times follow directly from the chain rule:

$$\frac{\partial g}{\partial T_i} = \frac{\partial g}{\partial \xi_i}\frac{\partial \xi_i}{\partial \tau_i}\frac{\partial \tau_i}{\partial T_i} = \frac{\partial g}{\partial \xi_i}\left(-k\ln(2)\xi_i\right). \tag{2.34}$$

In order to obtain the derivatives of the arrival times T with respect to the velocity field ϕ, the state equation (Equation (2.28)) is added to g for every node, multiplied by an adjoint field λ:

$$g^* = g + \sum_{j=1}^{N} \lambda_j \left(T_j - V_j\right). \tag{2.35}$$

Deriving with respect to the velocity field gives

$$\frac{\mathrm{d}g^*}{\mathrm{d}\phi_i} = \sum_{j=1}^{N}\left[\frac{\partial g}{\partial T_j}\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i}\right] + \sum_{j=1}^{N}\left[\lambda_j\left(\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i} - \frac{\mathrm{d}V_j}{\mathrm{d}\phi_i}\right)\right]. \tag{2.36}$$

The summation can be combined and the term $\mathrm{d}V_j/\mathrm{d}\phi_i$ expanded:

$$\frac{\mathrm{d}g^*}{\mathrm{d}\phi_i} = \sum_{j=1}^{N}\left[\frac{\partial g}{\partial T_j}\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i} + \lambda_j\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i} - \lambda_j\left(\frac{\partial V_j}{\partial \phi_i} + \sum_{k=1}^{N}\left[\frac{\partial V_j}{\partial T_k}\frac{\mathrm{d}T_k}{\mathrm{d}\phi_i}\right]\right)\right]. \tag{2.37}$$

Since the last term consists of two nested summations, both from 1 to N, the indices k and j can be swapped:

$$\frac{\mathrm{d}g^*}{\mathrm{d}\phi_i} = \sum_{j=1}^{N}\left[\frac{\partial g}{\partial T_j}\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i} + \lambda_j\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i} - \lambda_j\frac{\partial V_j}{\partial \phi_i} - \sum_{k=1}^{N}\left[\lambda_k\frac{\partial V_k}{\partial T_j}\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i}\right]\right]. \tag{2.38}$$

Now all the terms containing $\mathrm{d}T_j/\mathrm{d}\phi_i$ can be combined:

$$\frac{\mathrm{d}g^*}{\mathrm{d}\phi_i} = \sum_{j=1}^{N}\left[-\lambda_j\frac{\partial V_j}{\partial \phi_i} + \left(\frac{\partial g}{\partial T_j} + \lambda_j - \sum_{k=1}^{N}\left[\lambda_k\frac{\partial V_k}{\partial T_j}\right]\right)\frac{\mathrm{d}T_j}{\mathrm{d}\phi_i}\right]. \tag{2.39}$$

By choosing the adjoint such that the terms between brackets becomes zero, $\mathrm{d}T_j/\mathrm{d}\phi_i$ does not need to be evaluated. Therefore, the following condition has to be satisfied:

$$\frac{\partial g}{\partial T_j} + \lambda_j - \sum_{k=1}^{N}\left[\lambda_k\frac{\partial V_k}{\partial T_j}\right] = 0. \tag{2.40}$$

Finally, the sensitivities become

$$\frac{\mathrm{d}g^*}{\mathrm{d}\phi_i} = -\sum_{j=1}^{N}\lambda_j\frac{\partial V_j}{\partial \phi_i}. \tag{2.41}$$

However, $\partial V_j/\partial\phi_i$ is only nonzero when $j = i$, which simplifies the sensitivities to

$$\frac{\mathrm{d}g^*}{\mathrm{d}\phi_i} = -\lambda_i \frac{\partial V_i}{\partial \phi_i}. \tag{2.42}$$

The sensitivities with respect to the densities follow from the chain rule:

$$\frac{\mathrm{d}g^*}{\mathrm{d}\rho_i} = \frac{\mathrm{d}g^*}{\mathrm{d}\phi_j}\frac{\partial\phi_j}{\partial\rho_j^*}\frac{\mathrm{d}\rho_j^*}{\mathrm{d}\rho_i}, \tag{2.43}$$

where, from Equation (2.19), $\partial\phi_j/\partial\rho_j^* = (1-v_{\mathrm{void}})$, and $\mathrm{d}\rho_j^*/\mathrm{d}\rho_i$ is the derivative of the density filter. Note that for the nodes on $\partial\Omega_0$, the derivatives are slightly different due to the initialization (Equation (2.24)):

$$\frac{\mathrm{d}g^*}{\mathrm{d}\rho_i} = \lambda_j T_0 \frac{\mathrm{d}\rho_j^*}{\mathrm{d}\rho_i}, \qquad \mathbf{x}_i \in \partial\Omega_0. \tag{2.44}$$

2.4.3. EVALUATING THE ADJOINT

Equation (2.40) can be rearranged to obtain a recursive expression for the adjoint variables:

$$\lambda_j = \sum_{k=1}^{N}\left[\lambda_k \frac{\partial V_k}{\partial T_j}\right] - \frac{\partial g}{\partial T_j}. \tag{2.45}$$

The second term on the right-hand side can be evaluated directly with Equation (2.34). For the first term on the right-hand side, the adjoint variables λ_k of the nodes k whose arrival time has a dependence on the node j in consideration, i.e. when $\partial V_k/\partial T_j \neq 0$, must be known. By evaluating the adjoint in the opposite order as in which the arrival times have been calculated during the front propagation, it is guaranteed that the adjoint variables are evaluated before they appear in the first right-hand side term for another adjoint variable: clearly, $\partial V_k/\partial T_j \neq 0$ only when T_k is based on, and thus calculated after, T_j (note that the partial derivative $\partial V_k/\partial T_k = 0$). However, a variable number of arrival times T_k can depend on arrival time T_j. Therefore it is more convenient to index on which arrival times T_j depends, since every arrival time depends on exactly two other arrival times (Equation (2.28)), except for the nodes on $\partial\Omega_0$, whose arrival times depend only on the local density (Equation (2.24)).

Thus, during the front propagation, the order in which arrival times are accepted is stored in an array o, and the indices of the two nodes on which the accepted arrival times depend are registered in arrays c^1 and c^2. The adjoint variable can then be evaluated in a single loop, as outlined in Algorithm 1. Note that $\partial V/\partial T$ is a sparse matrix with two entries per row and $\partial V/\partial\phi$ has only entries on its diagonal.

2.5. RESULTS

In this section, the newly developed overhang filter is demonstrated on a given geometry, and on three cases where the compliance is minimized. The optimization problem reads

Algorithm 1 Adjoint evaluation

```
Input:
    $\partial g/\partial T, \partial V/\partial T, \partial V/\partial \phi, c^1, c^2, o, \mathbf{x}, N$
Initialize:
    $\lambda_i \leftarrow 0,\ i = 1, \dots, N$
for $k = N$ to 1 do
    $j \leftarrow o_k$
    $\lambda_j \leftarrow \lambda_j - \partial g/\partial T_j$
    $\mathrm{d}g^*/\mathrm{d}\phi_j \leftarrow -\lambda_j \partial V_j/\partial \phi_j$
    if $\mathbf{x}_j \notin \partial\Omega_0$ then
        $\lambda_{c_j^1} \leftarrow \lambda_{c_j^1} + \lambda_j \partial V_j/\partial T_{c_j^1}$
        $\lambda_{c_j^2} \leftarrow \lambda_{c_j^2} + \lambda_j \partial V_j/\partial T_{c_j^2}$
    end if
end for
```

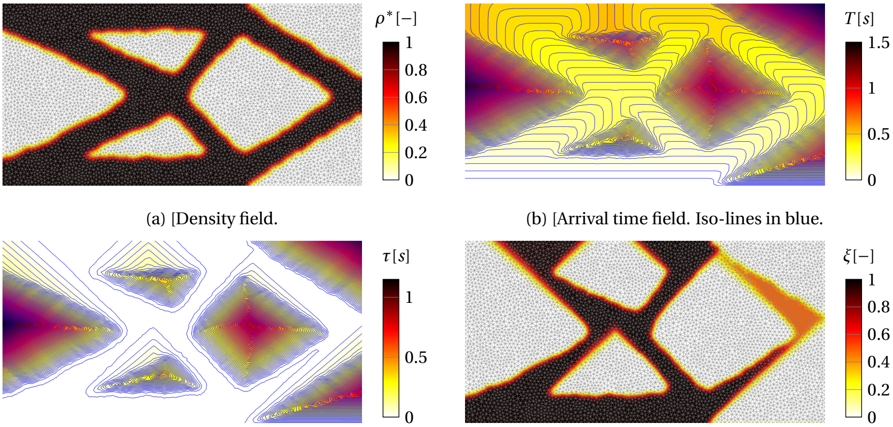
as follows:

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \mathbf{f}^T \mathbf{u} \\ \text{s.t.} \quad & \mathbf{K}(\boldsymbol{\xi})\mathbf{u} = \mathbf{f}, \\ & V(\boldsymbol{\xi})/V_{\text{lim}} - 1 \leq 0, \\ & \mathbf{0} \leq \boldsymbol{\rho} \leq \mathbf{1}. \end{aligned} \tag{2.46}$$

Here $\mathbf{f}$ and $\mathbf{u}$ denote the load and displacement vector, respectively. $\mathbf{K}(\boldsymbol{\xi})$ is the element stiffness matrix evaluated on the printable density field $\boldsymbol{\xi}$. $V(\boldsymbol{\xi})$ is the total volume, also evaluated on the printable densities, and V_{lim} denotes the maximum permitted volume. The sensitivities of the objective and constraint w.r.t. the printable densities can be found in e.g. Bendsøe and Sigmund (2004).

The first test case that is presented is the cantilever problem, which is well known and therefore allows for a clear interpretation of the results. The second case is a tensile test case, which has a sharp contrast between the objective and obtaining an overhang-free design. Finally, the capability to detect overhang on an unstructured mesh is demonstrated on the optimization of a crane hook. On the test cases, the filter is tested for several overhang angles, filter sizes and volume fractions. Finally, the computational cost is evaluated.

Unless stated otherwise, the following parameters are used in the upcoming examples. The Young's modulus E is set to 1 and $1 \cdot 10^{-6}\,\mathrm{N\,m^{-2}}$ for material and void, respectively, and the Poisson ratio $\nu = 0.3$. The applied force $F = 1\mathrm{N}$. RAMP penalization is used with $q = 10$ (Stolpe and Svanberg, 2001). The optimizations are run for 100 iterations with the Method of Moving Asymptotes (MMA) optimizer (Svanberg, 1987), in order to test different cases with roughly the same computational time, as this is often limiting factor in a practical environment. Standard increase and decrease parameters of respectively 1.2 and 0.7 are used. The Portable and Extendable Toolkit for Scientific Computing (PETSc) (Balay *et al.*, 1997, 2016) is used to parallelize the FEM assembly and solve (not the front propagation), in combination with the Multifrontal Massively Parallel sparse



(a) [Density field.

(b) [Arrival time field. Iso-lines in blue.

(c) Delay field. Iso-lines in blue.

(d) Printable density field.

Figure 2.12: The process of obtaining the printable densities (d) for a given topology (a), by performing a front propagation (b) and evaluating the delay field (c).

direct Solver MUMPS (Amestoy *et al.*, 2001, 2006). All the presented examples can be run on a single desktop, therefore, the implementation of PETSc is mainly intended for future 3D cases.

2.5.1. OVERHANG DETECTION

Consider a typical material distribution one might encounter during a topology optimization on an unstructured mesh, as given in Figure 2.12a. The overhang filter is applied as follows. First, front propagation with $\alpha_{\text{oh}} = 40°$ is performed on the given geometry, resulting in the arrival time field as shown in Figure 2.12b. Due to the numerical implementation of the front propagation, there is slight rounding of the corners of the arrival time field iso-contour lines. The rounding causes a small overestimation of the critical overhang angle, and can be reduced with mesh refinement. However, this is generally not necessary as the error is small, for this particular case in the order of 2°. From the arrival time field the delay field is calculated, as shown in Figure 2.12c. In this field the non-overhanging area with $\tau = 0$ (no delay) is already clearly visible. The printable densities are then evaluated with Equation (2.22), resulting in the material distribution given in Figure 2.12d. Compared to the original density field, the overhanging regions are removed, and the top-right member that is close to printable has intermediate densities. During the optimization, penalization of intermediate densities will limit the emergence of intermediate density values in the optimized topologies.

2.5.2. CANTILEVER TEST CASE

The overhang filter is first applied to the cantilever case, where compliance is minimized on a rectangular domain, as illustrated in Figure 2.13. The domain length $a = 1.0$m, and the domain is fully clamped on the left side and a vertical point force acts on the right side. The domain is discretized with a structured triangular mesh, comprised of 30

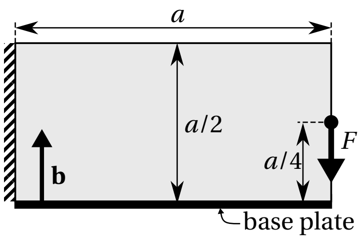


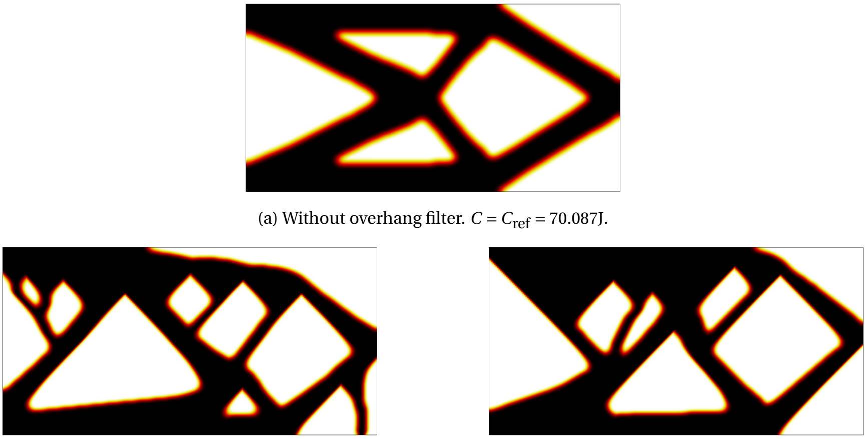
Figure 2.13: The cantilever test case, mechanically supported on the left side and with build direction **b**.

000 elements, with an average element edge length of 4.6mm. Furthermore, a density filter is applied with a filter radius of 20mm, and the volume is constrained at 50% of the design domain. The optimal design without overhang filter is depicted in Figure 2.14a. Its final objective, $C_{\text{ref}} = 70.087$J, is used as a reference for the overhang-free designs. Although this design is printable when rotated 90° counter-clockwise, the overhang filter is applied to make the designs printable when the build direction coincides with the y-axis, with $\alpha_{\text{oh}} = 45°$. The overhang filter parameters are chosen as $k = 500$, $T_0 = 0.02$ and $v_{\text{void}} = 0.1$. With overhang filter, a discrete, overhang-free design is obtained, as can be seen in Figure 2.14b. The initially overhanging members are supported, and most down-facing edges make a 45° angle with the base plate, lying exactly on the limit. The objective of the printable design is 12% higher than the conventional design, due to the added manufacturability filter. It can be observed that the edges of the filtered design are crisper than in the original design, which is controlled by the value of k. Lower values of k will decrease the crispness.

Compared to the constraint implementation presented in van de Ven *et al.* (2018), the cusps at the topside of the small holes depicted in Figure 2.14b are crisper, with almost no overhang present when the filter approach is implemented. With the constraint implementation, overhang was not completely eliminated in small holes (van de Ven *et al.*, 2018). The cost per iteration of both methods is roughly equal, since front propagation and the corresponding sensitivity calculation are identical in both approaches.

2.5.3. INITIAL CONFIGURATION, CONVERGENCE AND CONTINUATION

The optimization with overhang filter converges smoothly, and after 50 iterations the design hardly changes, as can be seen in Figure 2.15. The objective is autonomously low at the start as an initial density field of $\rho = 1$ is imposed, implying a completely filled domain and resulting in a violation of the volume constraint. After 10 iterations, when the volume constraint is satisfied, the objective decreases monotonically. The choice of a completely filled initial configuration is necessary to allow the optimizer to place material freely throughout the complete domain in the first few iterations. If the optimization starts with a density distribution in accordance with the volume constraint, i.e. $\rho = 0.5$, most of the domain is detected as overhanging and therefore does not contribute to the overall stiffness. Consequently, the design grows from the base-plate in the build direction, with slower convergence behavior and results in a far-from-optimal local minimum, as can be seen in Figure 2.16.



(a) Without overhang filter. $C = C_{\mathrm{ref}} = 70.087\mathrm{J}$.

(b) With overhang filter. $C = 1.12C_{\mathrm{ref}}$.

(c) With overhang filter and continuation. $C = 1.09C_{\mathrm{ref}}$.

Figure 2.14: Optimized designs for the cantilever case.

However, also with the completely filled initial configuration, like every nonconvex topology optimization problem, the optimization with overhang filter is susceptible to converge to inferior local optima. As can be seen in Figure 2.14b, not all the material contributes to the stiffness of the structure: the supporting leg on the bottom right has no mechanical function as the bottom of the domain is mechanically unconstrained. Although it is expected that enforcing printability decreases the overall performance, it seems that this member could have been placed under a 45° angle to add support as well as stiffness, instead of only support. From the optimization history it becomes clear that this member is formed early in the optimization to allow material around the point where the force acts, and is not repositioned later on.

A common method to avoid inferior local optima is to apply continuation. In order to activate the overhang constraint in a gradual manner, the physical densities ξ_c are linearly interpolated between the filtered densities ρ^* and the printable densities ξ:

$$\xi_c = \eta\xi + (1-\eta)\rho^*, \tag{2.47}$$

where the objective and constraint evaluations are now performed on the physical densities ξ_c, and $\eta \in [0,1]$ is the continuation parameter. In the remaining examples in this paper, when continuation is applied, η is continuously increased from 0 to 1 over the first 25 iterations of the optimization.

The resulting design with continuation is displayed in Figure 2.14c, and its convergence behaviour is plotted in Figure 2.15. When continuation is used, the initial configuration can be chosen as a uniform density distribution of $\rho = 0.5$, resulting in a higher initial objective as compared to the optimisation without continuation but satisfying the volume constraint. In the first 25 iterations the objective decreases, but not monotonically due to the ramp up of the continuation parameter η. Generally, when continuation is used an improvement is observed in the final objective, as compared to the value of the final objective attained without continuation.

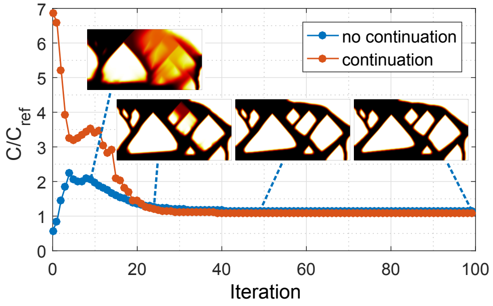


Figure 2.15: Convergence behavior for the cantilever case with and without continuation. The snapshots are taken at iteration 10, 25, 50 and 100, from the optimization without continuation. Note that without continuation, the optimization starts from a completely filled design, hence the high performance in the first few iterations when the volume constraint is not yet satisfied.

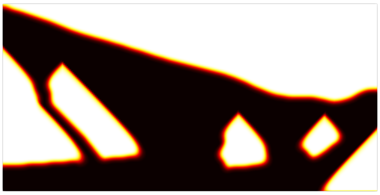


Figure 2.16: Design obtained with overhang filter using a conventional initial design. To obtain good results, starting with a fully solid design is recommended. $C = 1.69C_{\text{ref}}$.

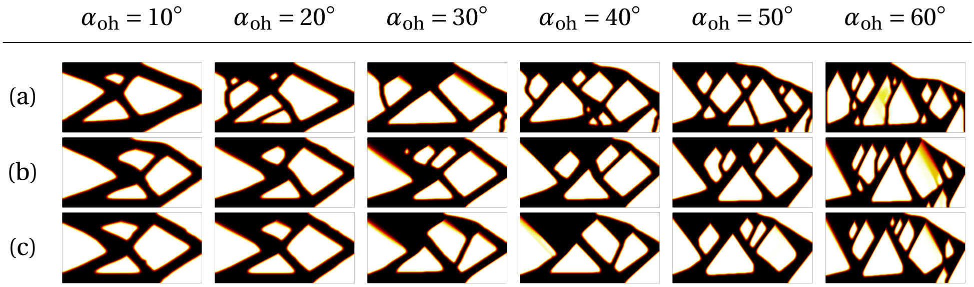

$\alpha_{\text{oh}} = 10°$
$\alpha_{\text{oh}} = 20°$
$\alpha_{\text{oh}} = 30°$
$\alpha_{\text{oh}} = 40°$
$\alpha_{\text{oh}} = 50°$
$\alpha_{\text{oh}} = 60°$
(a)
(b)
(c)

Figure 2.17: Resulting designs for the cantilever case with various minimum overhang angles optimized (a) without continuation, (b) with continuation over 25 iterations, and (c) with continuation over 100 iterations.

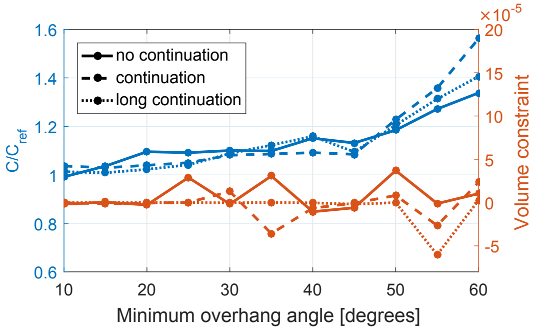


Figure 2.18: The final objective and volume constraint values as a function of the minimum overhang angle. The objective increases with the overhang angle, as more material has to be used for supporting purposes. Furthermore, it can be seen that continuation does not always lead to a lower objective. The volume constraint is overall satisfied.

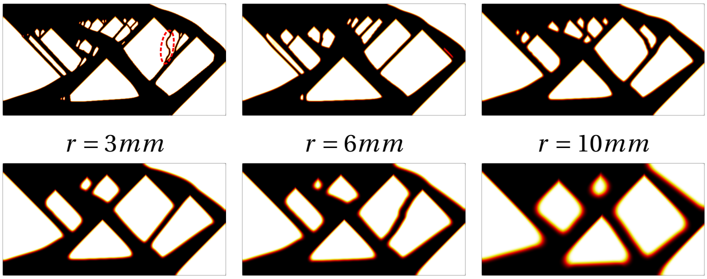
2.5.4. VARIABLE OVERHANG ANGLE

The novel overhang detection method based on front propagation can filter out overhangs with any value of α_{oh}. However, for very large angles ($\alpha_{\text{oh}} > 60^\circ$), the optimisation does not converge well as it becomes harder to find topologies that meet such a stringent manufacturing constraint. Since such high overhang angles are usually printable with modern printers, a parameter study for $10^\circ \leq \alpha_{\text{oh}} \leq 60^\circ$ is performed. For every angle three calculations are performed: without continuation, with continuation as described in Section 2.5.3, and with extra long continuation where η is continuously increased from 0 to 1 over the course of 100 iterations and the optimization is run for 400 iterations. The results are shown in Figure 2.17, and the final objective values are plotted in Figure 2.18.

As expected, lower overhang angles result in designs similar to designs obtained without activating the overhang filter as shown in Figure 2.14a. For higher overhang angles, more material is required for support, and the objective increases. Furthermore, as observed in the previous sections, the optimizations without continuation contain a higher fraction of material that does not contribute to the stiffness, but is only in place to satisfy the overhang limitation. Except for $\alpha_{\text{oh}} = 60^\circ$, the extra long continuation does not seems to contribute to better designs. This can also be seen in the final objective values, which are plotted in Figure 2.18. Interestingly, although the designs with continuation look visually more appealing than the designs without continuation, their overall objective values are only slightly lower for several overhang angles.

2.5.5. FILTER SIZE

The final parameter to be investigated on the cantilever case, is the density filter radius. For this parametric study, the optimisations are performed on a finer mesh comprised of 180 000 elements, in order to accommodate smaller radii. The average edge length is 1.9mm, and the filter is varied from 3mm to 40mm. The resulting designs are displayed in Figure 2.19. It is clear that the filter radius has an effect on the feature size, as smaller features appear for smaller radii. For these smaller radii, supporting structures hardly cost any volume. Therefore, the main structure can resemble the original design



$r = 3mm$ $r = 6mm$ $r = 10mm$

$r = 15mm$ $r = 20mm$ $r = 40mm$

Figure 2.19: The influence of density filter radius on the resulting topology. Smaller filter radii allow thin supports, reducing the cost of the overhang filter on the objective. For $r = 3mm$, a zigzagging support can be observed (encircled in red).

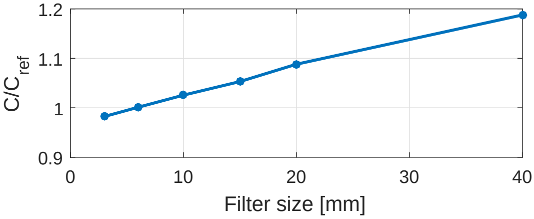


Figure 2.20: The final objective as a function of filter radius. Lower radii allow for thinner supports, and consequently result in lower objective values.

closely, resulting in a lower objective value, as can be seen in Figure 2.20, where the final objective values are plotted. Although oscillatory boundaries develop under the main structural beams for small density filter radii, the presence of these detailed features is not a manifestation of the sawtooth patterns observed when the overhang is only locally evaluated (Allaire *et al.*, 2017a; Qian, 2017): in our results the cusps of any sawtooth are at all times sufficiently supported, and thus manufacturable.

Exact control over the length scale is lost, since members can be positioned such that they are partially overhanging, resulting in thinner members in the overhang filtered design. In order to impose an exact minimum feature size, one should apply an additional filter after the overhang filter. Because a linear weighted average filter would reintroduce overhang in sharp corners, a dilate filter could be used (Sigmund, 2007).

2.5.6. TENSILE TEST CASE

An extreme test for the overhang filter is the tensile test case. The case is similar to the cantilever case except that the force acts in the horizontal direction and is applied closer to the top side, as displayed in Figure 2.21. Without overhang filter, the optimal design is a beam connecting the force to the fixed side. For the purpose of testing our algorithm, we disregard the possibility to translate the beam to the base plate. The bottom side of this beam will be completely overhanging, and therefore supports need to be generated to connect the base plate to the beam. These supports will have no mechanical function,

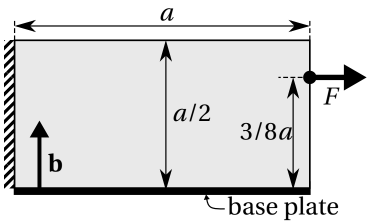


Figure 2.21: The tensile test case case, mechanically supported on the left side and with build direction **b**.

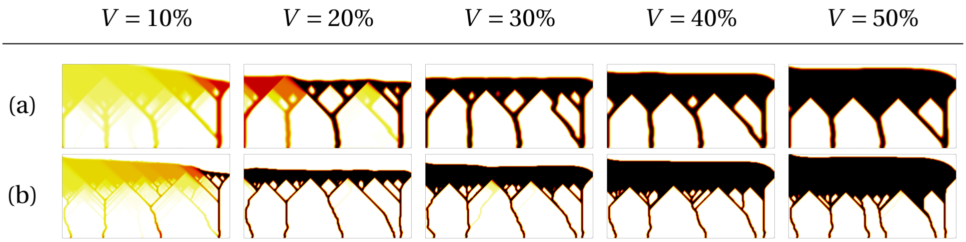


Figure 2.22: Results for the tensile test case for various volume fractions and for (a) a 20mm filter radius, and (b) a 7.5mm filter radius. For small volume fractions, dependent on the filter radius, not enough material is available to support the design, and the optimizations fail to converge to a black and white design.

and thus completely counteract the objective with volume constraint. Therefore, it is a good test to see if the overhang filter is able to generate fully dense supports, that have no function other than supporting the design.

The tensile test case is optimized for several volume fractions, ranging from 10% to 50%, and for a 20mm and a 7.5mm filter radius, as displayed in Figure 2.22, without the use of continuation. It can be seen that for volume fractions of 30% and higher, fully dense supports are created for both filter sizes. With decreasing volume fraction, the material available to increase the stiffness diminishes. Consequently, for 20%, the larger filter size is unable to converge to a black and white design, and for 10%, neither converges to a black and white design.

Furthermore, it can be seen that there are some supports that “zigzag” downwards, instead of a more volume efficient straight line. This behavior can also be seen in Figure 2.19, for $r = 3mm$. However, the influence on the objective is usually minute, as this is mostly observed for thinner supports.

2.5.7. CRANE HOOK CASE

For the final case, the compliance of a crane hook is minimized in order to demonstrate the overhang filter on a domain that is not easily meshed with a structured mesh, as is often the case in industrial practice. The domain and boundary conditions are displayed in Figure 2.23. The domain is mechanically clamped at the top and a vertical distributed load of 1N/m is applied on the inside of the hook. The compliance is minimized subject to a 40% volume constraint. The domain is meshed with 4000 elements with an average edge length of 46mm, and a density filter with a 75mm radius is applied and continua-

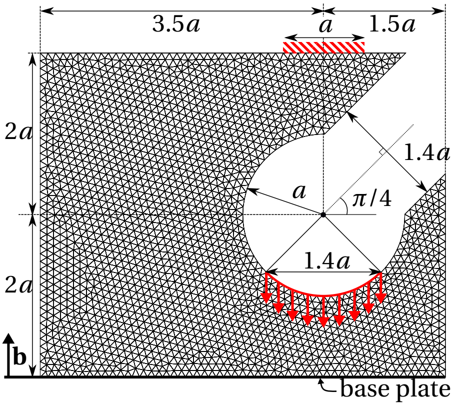


Figure 2.23: The crane hook case with unstructured mesh. The domain is clamped at the shaded region on the top, and a distributed load is applied as indicated by the red arrows, representing a hoist load. The overhang filter is applied with build direction **b**.

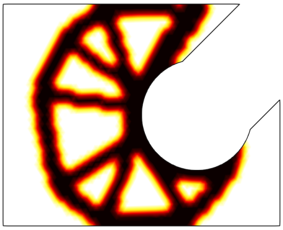
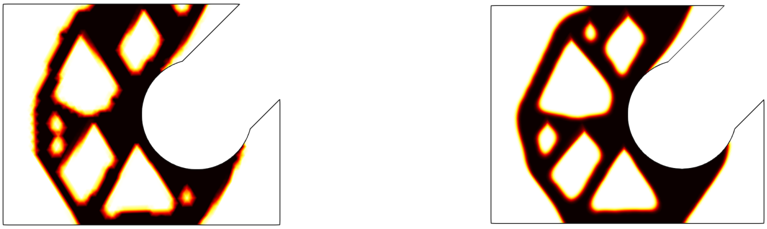


Figure 2.24: The optimized design for the crane hook without overhang constraint.

tion on the overhang filter is used. Without overhang filter, the resulting design resembles a typical hook, as displayed in Figure 2.24. When the overhang filter with $\alpha_{\mathrm{oh}} = 45°$ is applied, the design changes as can be seen in Figure 2.25a. A clear, overhang free, design is obtained. Due to the relatively coarse mesh, the final design contains some rough edges. With mesh refinement, this disappears as can be seen in Figure 2.25b, where the domain is meshed with 16 000 elements.

2.5.8. COMPUTATIONAL EFFICIENCY

Since the computational complexity of the OUM used by the overhang filter is of $O(N \log N)$ worst case, it is expected that the computational cost is small as compared to the objective and sensitivity evaluation for a compliance problem. The scaling of the computational cost of the compliance evaluation (excluding the time spent on the overhang filter), the overhang filter, and the full sensitivity analysis related to the overhang filter, with respect to element number is plotted in Figure 2.26. A power function is fitted to the measured CPU times, which are given for a single core computation on a 3.4Ghz Xeon E3-1240 V2. Compared to the compliance evaluation, which is primarily dominated by solving the system of linear equations, the overhang filter is significantly faster, and scales close to linear with the number of DOFs. Furthermore, it can be seen that the



(a)

(b)

Figure 2.25: Overhang free designs for the crane hook case on (a) the mesh as displayed in Figure 2.23, and (b) a 4x finer mesh.

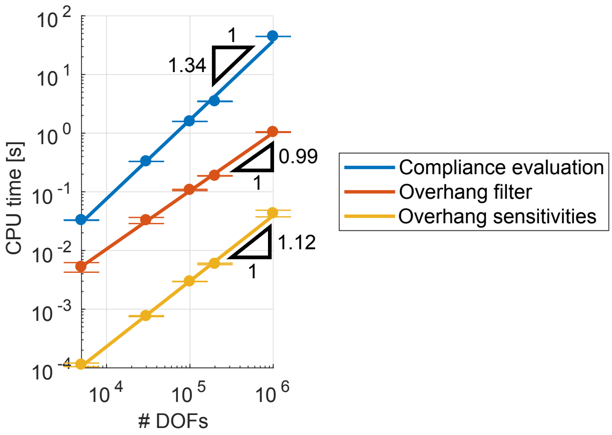


Figure 2.26: Average computational cost of the overhang filter and corresponding sensitivities w.r.t. the compliance evaluation (excluding the overhang filter) for a single core calculation. The errorbars indicate ± the standard deviation of the CPU times.

sensitivity calculation for the overhang filter is negligible in terms of computational cost.

Note that although the overhang filter sensitivity analysis only requires a single loop over all the nodes, it does not scale linearly. Because there are only few calculations in each iteration, the sensitivity calculation is memory bandwidth bound instead of compute-bound. In every iteration, non-contiguous entries of several arrays are accessed (see Algorithm 1), making it difficult for the compiler to load the correct part of the array to cache. Careful ordering of the arrays and prefetching are therefore important to control the scaling of the sensitivity analysis.

2.6. CONCLUSIONS

In this work a novel overhang filter based on front propagation is presented. Front propagation proves capable of detecting overhanging regions in a density-based topology by the use of an anisotropic speed function. By delaying the propagation in overhanging directions, a delay field can be constructed where overhanging regions have positive delay time while printable regions have 0 delay. This overhang detection procedure is incorporated as a filter into the topology optimization loop, and adjoint sensitivities are derived consistently. As such, the optimization algorithm can correct for unsupported regions by either removing or supporting them.

The Ordered Upwind Method is used to perform the front propagation, as it is computationally efficient and handles propagation with anisotropic speed functions. Furthermore, adjoint sensitivities are evaluated with a single loop over the elements, at low computational cost. The presented numerical results show various aspects of the overhang filter. It is shown that the overhang filter works for an arbitrary minimum overhang angle, that fully dense supports are generated when the volume constraint permits, and that the filter can handle unstructured meshes. In order to avoid inferior local optima, continuation is used. It is also observed that the supports that are generated are not always the shortest possible supports but sometimes “zigzag”. This is most likely related to the path of the sensitivities in the front propagation, and is a topic of further research.

Overall, the overhang filter performs well for the demonstrated 2D examples, and the front propagation is extensible to 3D as its formulation is mostly dimension independent. Although the specifics of the front propagation (Section 2.4.1) require adaptation for 3D, the Ordered Upwind Method will have the same computational complexity and hence the same scaling as the 2D algorithm evaluated in Section 2.5.8 (c.f. Figure 2.26). In a practical setting, the complete removal of overhanging regions might not be necessary, but only in inaccessible locations. This also remains a topic of further research.

Finally, this paper introduces a new way to use front propagation algorithms within topology optimization. Because of the computational efficiency of the front propagation, it is an attractive algorithm to include in additional constraints or filters, if they can be modeled by a propagating front. Further research will focus on the use of front propagation to model more aspects of the printing process, and to include these into topology optimization.

ACKNOWLEDGEMENTS

This research was carried out under project number S12.7.14549a in the framework of the Metals for Additive Manufacturing Partnership Programme of the Materials innovation institute M2i (www.m2i.nl) and the Netherlands Organisation for Scientific Research (www.nwo.nl). The authors thank Krister Svanberg for use of the MMA optimizer.

REFERENCES

Allaire G, Dapogny C, Estevez R, Faure A, Michailidis G (2017a) Structural optimization under overhang constraints imposed by additive manufacturing technologies. Journal of Computational Physics 351:295 – 328

Allaire G, Dapogny C, Faure A, Michailidis G (2017b) Shape optimization of a layer by layer mechanical constraint for additive manufacturing. Comptes Rendus Mathematique 355(6):699 – 717

Amestoy PR, Duff IS, L’Excellent JY, Koster J (2001) A fully asynchronous multifrontal solver using distributed dynamic scheduling. SIAM Journal on Matrix Analysis and Applications 23(1):15–41

Amestoy PR, Guermouche A, L'Excellent JY, Pralet S (2006) Hybrid scheduling for the parallel solution of linear systems. Parallel Computing 32(2):136–156

Balay S, Gropp WD, McInnes LC, Smith BF (1997) Efficient management of parallelism in object oriented numerical software libraries. In: Arge E, Bruaset AM, Langtangen HP (eds) Modern Software Tools in Scientific Computing, Birkhäuser Press, pp 163–202

Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Eijkhout V, Gropp WD, Kaushik D, Knepley MG, McInnes LC, Rupp K, Smith BF, Zampini S, Zhang H, Zhang H (2016) PETSc Users Manual. Tech. Rep. ANL-95/11 - Revision 3.7, Argonne National Laboratory

Bendsøe MP, Sigmund O (2004) Topology Optimization: Theory, Methods and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg

Brackett D, Ashcroft I, Hague R (2011) Topology optimization for additive manufacturing. In: Proceedings of the solid freeform fabrication symposium, Austin, TX, pp 348–362

Bruns TE, Tortorelli DA (2001) Topology optimization of non-linear elastic structures and compliant mechanisms. Computer Methods in Applied Mechanics and Engineering 190(26):3443 – 3459

Cloots M, Spierings A, Wegener K (2013) Assessing new support minimizing strategies for the additive manufacturing technology SLM. In: 24th International SFF Symposium - An Additive Manufacturing Conference, SFF 2013, pp 631–643

Gaynor AT, Guest JK (2016) Topology optimization considering overhang constraints: Eliminating sacrificial support material in additive manufacturing through design. Structural and Multidisciplinary Optimization 54(5):1157–1172

Gaynor AT, Meisel NA, Williams CB, Guest JK (2014) Topology Optimization for Additive Manufacturing: Considering Maximum Overhang Constraint. In: AIAA Aviation, American Institute of Aeronautics and Astronautics

Guo X, Zhou J, Zhang W, Du Z, Liu C, Liu Y (2017) Self-supporting structure design in additive manufacturing through explicit topology optimization. Computer Methods in Applied Mechanics and Engineering 323:27 – 63

van Keulen F, Hirai Y, Tabata O (2008) Objective function and adjoint sensitivities for Moving-Mask Lithography. In: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, p 5934

Langelaar M (2016) Topology optimization of 3D self-supporting structures for additive manufacturing. Additive Manufacturing 12:60–70

Langelaar M (2017) An additive manufacturing filter for topology optimization of print-ready designs. Structural and Multidisciplinary Optimization 55(3):871–883

Mirzendehdel AM, Suresh K (2016) Support structure constrained topology optimization for additive manufacturing. Computer-Aided Design 81:1–13

Qian X (2017) Undercut and overhang angle control in topology optimization: A density gradient based integral approach. International Journal for Numerical Methods in Engineering 111(3):247–272, nme.5461

Sethian JA (1996) A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Sciences 93(4):1591–1595

Sethian JA, Vladimirsky A (2003) Ordered upwind methods for static Hamilton–Jacobi equations: Theory and algorithms. SIAM Journal on Numerical Analysis 41(1):325–363

Sethian JA, Wiegmann A (2000) Structural boundary design via level set and immersed interface methods. Journal of Computational Physics 163(2):489 – 528

Sigmund O (2007) Morphology-based black and white filters for topology optimization. Structural and Multidisciplinary Optimization 33(4):401–424

Sigmund O, Petersson J (1998) Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima. Structural optimization 16(1):68–75

Stolpe M, Svanberg K (2001) An alternative interpolation scheme for minimum compliance topology optimization. Structural and Multidisciplinary Optimization 22(2):116–124

Svanberg K (1987) The method of moving asymptotes – a new method for structural optimization. International journal for numerical methods in engineering 24(2):359–373

Thomas D (2009) The Development of Design Rules for Selective Laser Melting. PhD thesis, University of Wales

van de Ven E, Ayas C, Langelaar M, Maas R, van Keulen F (2018) A PDE-Based Approach to Constrain the Minimum Overhang Angle in Topology Optimization for Additive Manufacturing. In: Schumacher A, Vietor T, Fiebig S, Bletzinger KU, Maute K (eds) Advances in Structural and Multidisciplinary Optimization: Proceedings of the 12th World Congress of Structural and Multidisciplinary Optimization (WCSMO12) , Springer, Cham, pp 1185–1199

Vladimirsky AB (2001) Fast methods for static Hamilton-Jacobi Partial Differential Equations. PhD thesis, University of California

Wang D, Yang Y, Yi Z, Su X (2013) Research on the fabricating quality optimization of the overhanging surface in SLM process. The International Journal of Advanced Manufacturing Technology 65(9-12):1471–1484

3

FRONT PROPAGATION-BASED OVERHANG CONTROL IN 3D

It is attractive to combine topology optimization (TO) with additive manufacturing (AM), due to the design freedom provided by AM, and the increased performance that can be achieved with TO. One important aspect is to include the design rules associated with the process restrictions of AM to prevent the requirement of relatively large support volumes during printing. This paper presents a TO filter that enforces a minimum overhang angle, resulting in an optimized topology that is printable without the need for support structures. The filter is based on front propagation, which, as it is described by a PDE, allows for a straightforward application on unstructured meshes, to enforce an arbitrary overhang angle. Efficient algorithms developed for front propagation are used in combination with adjoint sensitivities, in order to have a minor influence on the total computational cost. The focus of this work is on the implementation of the filter for high resolution 3D cases, which requires development of the front propagation for tetrahedral elements, and its parallelization.

This chapter is published in *Computer Methods in Applied Mechanics and Engineering* 369, 113169 (2020).

3.1. Introduction

Additive manufacturing (AM) offers tremendously more design freedom compared to conventional manufacturing techniques, and geometric complexity has a much lesser relative impact on production cost. Therefore, it is frequently linked to topology optimization (TO), a computational design method that generates optimized designs for given objective and constraints, but often results in complex, organically shaped designs, which were difficult to manufacture until the emergence of AM (Deaton and Grandhi, 2014; Sigmund and Maute, 2013).

However, in practice, AM is not completely free of manufacturing constraints. Many studies have been performed to identify design rules for AM (Adam and Zimmer, 2014; Kranz *et al.*, 2015; Kruth *et al.*, 2005; Thomas, 2009; Thompson *et al.*, 2016). Design for manufacturing practice states that ignoring manufacturing constraints during the design process will lead to extra costs during manufacturing (Boothroyd *et al.*, 2011). When a topology optimized design is modified after optimization to incorporate the manufacturing constraints, optimality is mostly compromised. Therefore, the AM constraints should be included into the topology optimization to retain optimality while adhering to the manufacturing constraints.

The AM constraint that is the focus of this study is the overhang limitation. This constraint arises from the fact that each consecutive layer that is printed requires a certain amount of overlap with the previous layer, in order to have sufficient mechanical support and/or heat dissipation (Vandenbroucke and Kruth, 2007). Therefore, the distance with which each layer can "overhang" the previous layer is limited. The degree of overhang is usually measured as the angle between a downward-facing surface and the base plate. The minimum allowable overhang angle α_{oh} depends on the type of process and material, hence a general overhang constraint should be able to handle a range of minimum overhang angles (Wang *et al.*, 2013). In the remainder of this paper, surfaces are termed overhanging if they have an overhang angle smaller than the minimum overhang angle.

Incorporation of AM constraints into topology optimization has recently become an active field of study. Various papers have presented 2D approaches (Brackett *et al.*, 2011; Gaynor and Guest, 2016; Guo *et al.*, 2017; Langelaar, 2017; van de Ven *et al.*, 2018), but relevance to industrial practice requires methods to be highly effective in general 3D settings. Therefore we shall focus our discussion on the studies that show a 3D implementation of the overhang constraint. For a more comprehensive review the reader is referred to Liu *et al.* (2018). Generally, the overhang constraints can be classified into three categories: local boundary angle control, geometrical AM process modelling, and physics-based AM modelling.

Constraints in the first category attempt to detect the topological boundary and constrain the overhang angle locally. This has been applied in Qian (2017), Mirzendehdel and Suresh (2016) and Allaire *et al.* (2017). The geometrical AM process modelling constraints also enforce a given overhang angle, but do so by scanning the topology in the print order to detect overhanging areas - crudely mimicking the printing process (Hoffarth *et al.*, 2017; Johnson and Gaynor, 2018; Langelaar, 2016; Wang *et al.*, 2018). Finally, the physics-based constraints incorporate a more elaborate model of the printing process, by modelling the manufacturing, e.g., as a series of self-weight loads (Allaire *et al.*, 2017; Amir and Mass, 2018).

The local boundary angle control methods usually converge to sub-optimal local minima, generating saw-tooth like structures that are not manufacturable, whereas the physics-based constraints, although potentially providing more detail, are numerically expensive since they generally involve one or more finite element analyses to model the printing process. Therefore, this study presents an overhang constraint of the geometrical AM process modelling type. Compared to the existing methods, it can handle unstructured meshes and variable overhang angles naturally, as opposed to the work of Langelaar (2016) which would require an additional mapping as detailed in Langelaar (2018). Furthermore, there is no additional non-linearity introduced by a heaviside projection filter (Johnson and Gaynor, 2018), and no additional filtering is required to prevent floating supports (Hoffarth *et al.*, 2017). Finally, it can be implemented in any density based topology optimization as opposed to Wang *et al.* (2018) which uses a custom topology description.

This paper presents the extension to 3D of the front propagation based overhang filter presented in van de Ven *et al.* (2018). While our earlier 2D implementation validated the concept of detecting and eliminating overhang using front propagation, the 3D implementation allows true 3D high-resolution design for AM. The non-trivial steps required in expanding from 2D to 3D are (i) efficiently propagating the front on the element level, and (ii) parallelizing the front propagation. For these challenges novel solutions are proposed. Furthermore, improvements have been made to control the length scale of the overhang filtered design. The paper is organized as follows. In Section 3.2 the method of overhang detection and the incorporation of the filter in TO is explained. Section 3.3 provides the details of the numerical implementation in 3D. Numerical results are presented in Section 3.4, and Section 3.5 concludes this paper.

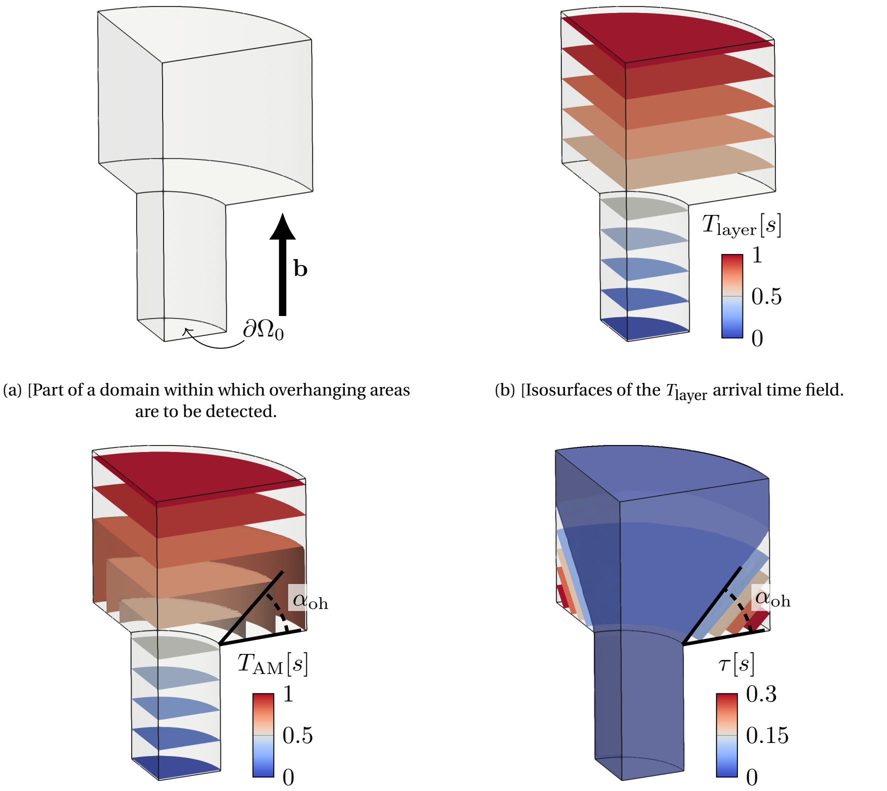
3.2. Method

3.2.1. Detecting overhang using front propagation

The idea of using front propagation for overhang detection originates from the printing process itself, which can be seen as a front evolving with each printed layer. Similar to the real printing process, the front propagation is initiated at the base plate, defined by a boundary $\partial\Omega_0$. The front propagation can be described by an arrival-time field $T(\mathbf{x})$, which represents the pseudo-time at which the front reaches to location $\mathbf{x}$. The core idea of our method is to construct an arrival-time field in a 3D design that distinguishes printable regions from overhanging ones.

Consider the structure given in Figure 3.1a on which the overhanging region should be identified, when printed in the build direction indicated by $\mathbf{b}$. The first arrival-time field that is required for overhang detection is termed the layer arrival-time field, T_{layer} (Figure 3.1b). It describes the sequence of printing, as each isosurface of this fields represents a printing layer. The arrival time of each layer is a measure of distance to the base plate, as can be seen in Figure 3.1b.

Subsequently, a second arrival-time field, T_{AM}, is constructed that includes information on the overhang limitation of an actual AM process (Figure 3.1c). It is constructed by using a front propagation that results in $T_{\text{AM}} = T_{\text{layer}}$ in printable regions, but delays the propagation when the direction of propagation is lower than a given overhang angle



(a) [Part of a domain within which overhanging areas are to be detected.

(b) [Isosurfaces of the T_{layer} arrival time field.

(c) Isosurfaces of the T_{AM} arrival time field.

(d) Isosurfaces of the delay field τ, with indication of the minimum overhang angle α_{oh}.

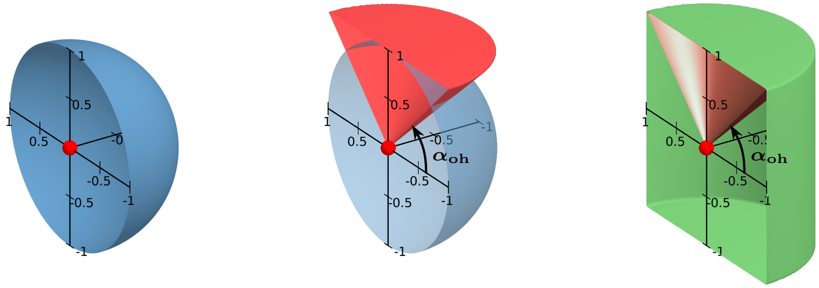
Figure 3.1: Overhang can be detected in an arbitrary geometry by calculating the delay τ of the T_{AM} field w.r.t. the T_{layer} field. The regions with zero delay are not overhanging (the blue region in Figure 3.1d).

α_{oh} (i.e. violating the overhang limitation). By doing so, the isosurfaces of T_{AM} are not flat as T_{layer}, but bend around corners, as displayed in Figure 3.1c.

The final step to detect the overhanging regions is to compare the two arrival-time fields T_{layer} and T_{AM}. The delay field τ of the T_{AM} field is defined as

$$\tau(\mathbf{x}) = T_{AM}(\mathbf{x}) - T_{\text{layer}}(\mathbf{x}). \tag{3.1}$$

When $\tau(\mathbf{x}) > 0$, the AM front is delayed compared to the layer front, which implies that the AM front has propagated in a direction lower than α_{oh}. Consequently, overhanging regions are identified as those regions where $\tau(\mathbf{x}) > 0$. This can be seen in Figure 3.1d, where the region $\tau(\mathbf{x}) = 0$ is manufacturable, and the rest is overhanging and, thus, can not be manufactured. Furthermore, the value of $\tau(\mathbf{x})$ is a measure for the distance to the closest manufacturable region, giving an indication of how much material it would require to support a certain location. This continuous measure of overhang is beneficial for gradient-based optimization used in topology optimization.



(a) Spherical speed function.

(b) Conic speed function (red), with spherical speed function overlay (blue).

(c) Cylindrical speed function (green), with conic speed function overlay (red).

Figure 3.2: The depicted surfaces indicate the speed with which the front propagates in each direction from the origin (red dot). The surfaces are cut in half for visualization purposes, but are in fact rotationally symmetric about the vertical axis.

3.2.2. CONSTRUCTING THE ARRIVAL-TIME FIELDS

The construction of T_{layer} is straightforward. Since the arrival time is a measure of distance to the base plate, this field is defined as:

$$T_{\text{layer}}(\mathbf{x}) = \frac{\mathbf{x} \cdot \mathbf{b}}{f_0}, \tag{3.2}$$

where $\mathbf{b}$ is a unit vector pointing in the build direction, f_0 the propagation speed, which can be interpreted as the printing rate, and assuming that the origin of the coordinate system is on the base-plate.

For the second arrival-time field, T_{AM}, the front propagation can be written as a boundary value problem and is governed by the Hamilton-Jacobi-Bellman equation:

$$\begin{aligned} &T(\mathbf{x}) = T_0(\mathbf{x}), && \mathbf{x} \in \partial\Omega_0, \\ &\min_{\mathbf{a}\in S_1} \left\{ (\nabla T(\mathbf{x}) \cdot \mathbf{a}) f_s(\mathbf{x}, \mathbf{a}, \alpha_{oh}) \right\} + 1 = 0, && \mathbf{x} \in \Omega, \end{aligned} \tag{3.3}$$

where Ω is the domain under consideration, $T(\mathbf{x})$ is the arrival time at location $\mathbf{x}$, fixed at T_0 at the boundary $\partial\Omega_0$, $\mathbf{a}$ is a unit vector indicating the local propagation direction: $\mathbf{a} \in S_3$, $S_3 = \left\{\mathbf{a} \in \mathbf{R}^3 \mid \|\mathbf{a}\| = 1\right\}$, and $f_s(\mathbf{x}, \mathbf{a}, \alpha_{oh})$ is a speed function, giving the propagation speed for a given location, propagation direction and overhang angle. The speed function is decomposed in a part dependent on the location, and a part dependent on direction of propagation:

$$f_s(\mathbf{x}, \mathbf{a}, \alpha_{oh}) = \phi(\mathbf{x}) f(\mathbf{a}, \alpha_{\text{oh}}). \tag{3.4}$$

Let us for now ignore the location dependence, which is detailed in Section 3.2.3, and focus on the speed function $f(\mathbf{a}, \alpha_{oh})$. In order to be able to detect overhang, the speed function f should be chosen with care. The simplest speed function, used for isotropic front propagation (e.g. wave propagation in isotropic media), is $f = c$, where c represents a constant propagation speed irrespective of the propagation direction. This is depicted in Figure 3.2a, where the distance from the origin (the red dot) to the surface gives the propagation speed in each direction. However, it is difficult to detect overhang with this speed function, as no information on the minimum overhang angle is governed by this speed function.

For a front that mimics the printing process, a conic propagation profile, as depicted in Figure 3.2b, is suitable. It can be seen that the front propagation speed reduces to zero below the minimum overhang angle, and the propagation speed increases when the front propagates in directions other then the build direction, to compensate for the larger distance it travels to reach the next layer. Basically, the cone represents the region that can be built when starting from a single point. Propagating a front with this speed function would indicate the printable regions in a topology. Unfortunately, this speed function is numerically difficult to propagate, as it has a large anisotropy and a zero sideways velocity, which would result in infinitely large arrival times.

For numerical convenience, the conic speed function is changed to a cylindrical speed function as displayed in Figure 3.2c. Compared to the conic speed function it has the same profile for propagation directions above the minimum overhang angle. For other directions, the speed is not set to zero, but to a finite value governed by the surface of the cylinder. This means the front can still propagate in non-manufacturable regions, albeit at a slower speed, and hence it will be delayed. Next to being numerically more tractable compared to the conic speed profile, the cylindrical speed function also has the benefit of yielding information on the severity of the overhang in terms of the delay, which will be utilized during the optimization. The cylindrical speed function for a given overhang angle α_{oh} and propagation direction $\mathbf{a}$ is described by the following equation (see van de Ven *et al.* (2018)):

$$f(\mathbf{a}, \alpha_{oh}) = \frac{f_0}{\max\left(\tan(\alpha_{oh})\,\|\mathbf{Pa}\|\,, |\mathbf{b}\cdot\mathbf{a}|\right)}, \tag{3.5}$$

where $\mathbf{P}$ is the projection on the plane orthogonal to $\mathbf{b}$, defined as $\mathbf{P} = (\mathbf{I} - \mathbf{b}\otimes\mathbf{b})$, with $\mathbf{x}\otimes\mathbf{y}$ denoting the outer product between the vectors $\mathbf{x}$ and $\mathbf{y}$. Propagating a front with this cylindrical speed function gives the required arrival-time field T_{AM}, as shown in Figure 3.1c.

3.2.3. Incorporation in topology optimization

Thus far, the front propagation was carried out on a given part domain. However, with topology optimization the geometry is implicitly defined. In this study, density-based topology optimization is used (Bendsøe and Sigmund, 2004), where the topology is defined by a pseudo-density field $\rho(\mathbf{x})$, which indicates for each location if it is either void ($\rho = 0$), contains material ($\rho = 1$), or has an intermediate value ($0 < \rho < 1$). The topology optimization algorithm with overhang filter is schematically depicted in Figure 3.3. In the first step, the pseudo-density field ρ is filtered to control length scale and to prevent checkerboarding (Sigmund and Petersson, 1998). The filtered densities ρ^* are thus a weighted average of the densities in their surrounding region up to the filter radius r (Bruns and Tortorelli, 2001).

The filtered densities ρ^* are input to the overhang filter, which results in the printable density field ξ, a density field similar to ρ^* but with the overhanging regions removed.

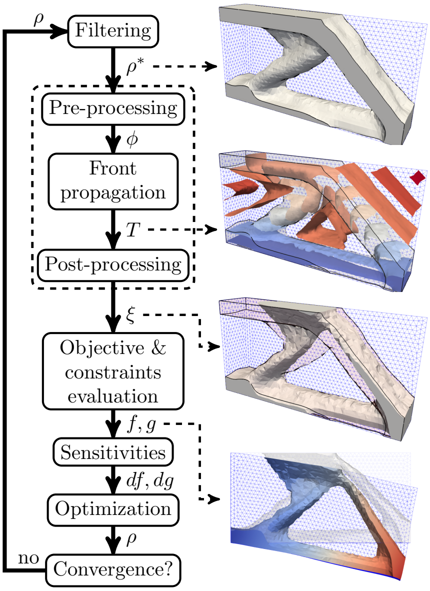


Figure 3.3: Schematic of the TO algorithm with overhang filter, indicated by the dashed rectangle. The fields ρ^*, T, and ξ have been depicted for a typical TO result. Note that the objective and constraints are evaluated on the printable design ξ, as shown in the bottom figure.

The objective and constraint evaluation is then carried out on ξ, instead of the filtered densities ρ^* one would normally use.

The overhang filter comprises three steps. First, the filtered densities are processed to serve as a scaling field ϕ for the speed of the front propagation. The purpose of the speed scaling field ϕ is to only allow the front to propagate through material regions, and to delay it in void regions. With this step, the geometry given by the density field ρ^* is coupled into the front propagation; otherwise, the front propagation would not be influenced by the density field at all. For simplicity a linear relation is chosen, and the speed scaling field ϕ is defined as

$$\phi(\mathbf{x}) = v_{\text{void}} + (1 - v_{\text{void}})\rho^*(\mathbf{x}), \tag{3.6}$$

where v_{void} $(0 < v_{\text{void}} < 1)$ is the scaling of the propagation speed in void regions, which is typically chosen as 0.5. As stated in Equation (3.4), the speed function for the front propagation (Equation (3.5)) is scaled linearly with ϕ:

$$f_s(\mathbf{x}, \mathbf{a}, \alpha_{oh}) = \phi(\mathbf{x}) f(\mathbf{a}, \alpha_{\text{oh}}). \tag{3.7}$$

Then, in the second step, the front propagation is performed with the scaled speed function f_s, to obtain the arrival-time field T_{AM}.

In the last step of the overhang filter, the arrival-time field T_{AM} is post-processed to obtain the printable density field ξ. First, the delay is evaluated (Equation (3.1)):

$$\tau(\mathbf{x}) = T_{\text{AM}}(\mathbf{x}) - T_{\text{layer}}(\mathbf{x}), \tag{3.8}$$

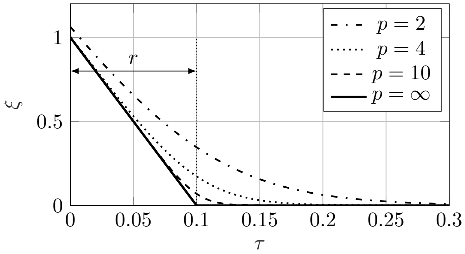


Figure 3.4: Relation between the delay τ and the printable density field ξ, for different p with a filter radius $r = 0.1$.

which is transformed into the printable density field ξ by a function h:

$$\xi(\mathbf{x}) = h(\tau(\mathbf{x})). \tag{3.9}$$

Since manufacturable regions are defined as those regions where the delay $\tau = 0$ (Equation (3.1)), h should be such that $h(0) = 1$. Regions with a delay $\tau > 0$ are not manufacturable or void in the original design, and therefore h should decrease towards zero for increasing values of τ. The choice of h dictates the transition of material to void in the printable density field. In order to retain the original length scale of the density filter with a filter radius of r, the following relation has been chosen:

$$h(\tau(\mathbf{x})) = \mathrm{smax}_p\left(0, 1 - \frac{\tau(\mathbf{x})\, v_{\mathrm{void}}}{r}\right), \tag{3.10}$$

$$\mathrm{smax}_p(a, b) = \frac{1}{p}\ln\big(\exp(pa) + \exp(pb)\big), \tag{3.11}$$

where smax_p is a smooth maximum operator, and p determines the smoothness. The relation between ξ and τ is displayed in Figure 3.4. In the manufacturable regions where $\tau = 0$, the printable density becomes $\xi = 1$, and for higher values of p, ξ decreases linearly towards 0 with increasing delay, comparable to a density filter (Bruns and Tortorelli, 2001). As can be seen, p should not be chosen too small, as that will result in printable densities significantly larger than one for $\tau = 0$. In this study, p is chosen as $p = 10$.

The relation specified in Equation (3.10) implies that when the front is propagating at the void speed v_{void}, the transition from $\xi = 1$ to $\xi = 0$ will take place within the original filter radius, as depicted in Figure 3.4. However, this is not true for every transition from $\xi = 1$ to $\xi = 0$. First of all, due to the original density filter, the propagation speed will not decrease to v_{void} instantaneously, but gradually ramp off, and second, the propagation speed is not equal in the build direction and in directions orthogonal to the build direction if $\alpha_{\mathrm{oh}} \neq 45°$. Therefore, the length scale can be influenced by the application of the overhang filter. If an exact length scale is required, the overhang filter can be applied before the density filter. However, this will reintroduce some overhang due to rounding of the density filter, and therefore the density filter is applied first in this work.

With h specified in Equation (3.10), the prescribed values T_0 for the front propagation at the boundary can be determined (Equation (3.3)). Since material points at the base plate $\partial\Omega_0$ are printable, it should hold that

$$\xi(\mathbf{x}) = \rho^*(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega_0. \tag{3.12}$$

At the base plate, $T_{\text{layer}} = 0$, and thus $\xi = h(\tau) = h(T_0)$. To satisfy Equation (3.12), the initial arrival time at the boundary is chosen as

$$T_0(\mathbf{x}) = h^{-1}(\rho^*(\mathbf{x})), \quad \mathbf{x} \in \partial\Omega_0, \tag{3.13}$$

$$= \frac{-r}{v_{\text{void}}} \left(\ln\left(\exp(\rho^*(\mathbf{x})p)\right)/p - 1\right), \tag{3.14}$$

where h^{-1} is the inverse function of h such that $h^{-1}(h(\rho^*(\mathbf{x}))) = \rho^*(\mathbf{x})$. Note that due to the logarithm, $T_0 = \infty$ for $\rho^* = 0$. If infinite values cannot be used, T_0 should be set to an arbitrary high value wherever $\rho^* = 0$.

3.3. NUMERICAL IMPLEMENTATION

The two main challenges of the 3D implementation of the overhang filter are (i) parallelizing the front propagation problem to achieve the desired scalability with problem size, and (ii) efficiently propagating the front on element level, i.e. the local arrival time update, which is a key component in the numerical front propagation scheme. This section is intended to aide with the numerical implementation of the method and mainly concerns implementation details. Readers who want to obtain a general overview of the front propagation-based overhang filter can choose to continue with the numerical results in Section 3.4.

3.3.1. UPDATING LOCAL ARRIVAL TIME IN 3D

Most front propagation algorithms contain a function where within an element the arrival time of one node is calculated based on the known arrival time of the other nodes. This local update is used to propagate the front throughout the domain, as with each newly calculated node, arrival times for other unknown nodes can now be computed. This update is performed multiple times for each node from different directions, up to the number of elements that contain the node. Eventually, the update resulting in the lowest arrival time is accepted. Because of the multiple evaluations per node, it is the function that is called the most times in the propagation algorithm and thus it is essential that the update is numerically inexpensive. The local update is the only part of the algorithm that is different from the 2D implementation presented in van de Ven *et al.* (2018), although a similar approach is taken to evaluate the local update.

In the following, the local update is detailed for a tetrahedron, as any other polygonal element can be constructed from it. It is possible to define the local update for other element types as well. Consider a tetrahedron $\mathbf{x}_i\mathbf{x}_j\mathbf{x}_k\mathbf{x}_l$ as displayed in Figure 3.5, where the arrival time T_i at node $\mathbf{x}_i$ is unknown, and the arrival times T_j, T_k, and T_l on the remaining nodes are known. The triangle $Q = \mathbf{x}_j\mathbf{x}_k\mathbf{x}_l$ spanned by the nodes with known arrival times. If these arrival times are equal, Q is the front of the propagation. A point $\mathbf{x_c}$ on this triangle can be defined in parametric form:

$$\mathbf{x_c}(\zeta_1,\zeta_2) = \mathbf{x}_j + \zeta_1(\mathbf{x}_k - \mathbf{x}_j) + \zeta_2(\mathbf{x}_l - \mathbf{x}_j), \tag{3.15}$$

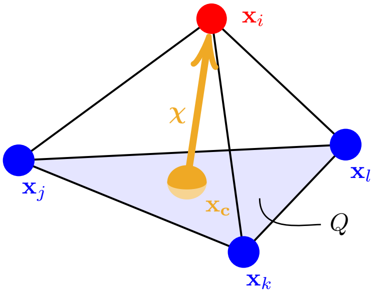


Figure 3.5: In the local update, the arrival time of a node $\mathbf{x}_i$ is calculated from three nodes with known arrival times, by finding a location $\mathbf{x_c}$ on the triangle Q spanned by these nodes, that minimizes the arrival time at $\mathbf{x}_i$.

with $0 \leq \zeta_1 \leq 1$, $0 \leq \zeta_2 \leq 1$ and $\zeta_1 + \zeta_2 \leq 1$. The arrival time at $\mathbf{x_c}$ on the front is a linear interpolation between the known nodes and is defined as

$$T_{\mathbf{c}}(\zeta_1, \zeta_2) = T_j + \zeta_1 (T_k - T_j) + \zeta_2 (T_l - T_j). \tag{3.16}$$

Now, the arrival time T_i, assuming it is updated from $\mathbf{x_c}$, can be calculated as the distance $\|\boldsymbol{\chi}\|$ between both points divided by the propagation speed f_s, plus the arrival time at $\mathbf{x_c}$, $T_{\mathbf{c}}$:

$$T_i = \frac{\|\boldsymbol{\chi}\|}{f_s(\phi_i, \mathbf{a}, \alpha_{oh})} + T_{\mathbf{c}}, \tag{3.17}$$

where $\boldsymbol{\chi} = \mathbf{x}_i - \mathbf{x_c}$, $\mathbf{a}$ is the direction of propagation defined as $\mathbf{a} = \boldsymbol{\chi} / \|\boldsymbol{\chi}\|$ (Figure 3.5), and $f_s(\phi_i, \mathbf{a}, \alpha_{oh})$ is the speed function as defined in Equation (3.7). Note that the propagation speed is only dependent on the speed scaling value ϕ_i, and thus density ρ_i, of the target node. This is done to simplify the front propagation and sensitivity evaluation, as it introduces less dependencies as compared to interpolating the speed scaling values ϕ of all the nodes in the element.

What remains is to determine from which point $\mathbf{x_c}$ on the triangle Q the arrival time of $\mathbf{x}_i$ should be updated. Consequently, the local update is essentially a minimization problem where a point $\mathbf{x_c}$ on triangle Q is sought that minimizes the arrival time T_i at the unknown node. Following Sethian and Vladimirsky (2003), the equation for the local update for node $\mathbf{x}_i$ as updated from triangle Q is given by

$$T^i_{jkl} = \min_{\zeta_1, \zeta_2 : \mathbf{x_c} \in Q} \left\{ \frac{\|\boldsymbol{\chi}\|}{f_s(\phi_i, \mathbf{a}, \alpha_{oh})} + T_{\mathbf{c}} \right\}, \tag{3.18}$$

where T^i_{jkl} is the arrival time at node i when evaluated from nodes j, k, and l. The final arrival time of node is the minimum of its local updates. The remainder of this section will detail the solving of the minimization problem posed in Equation (3.18). First, a list of potential locations that minimize Equation (3.18) is determined, by carefully examining the possible scenarios in the minimization problem posed by the maximum and absolute function present in the speed function f_s (Equation (3.7)). Then, the arrival times resulting from these locations are evaluated and the minimum arrival time is selected.

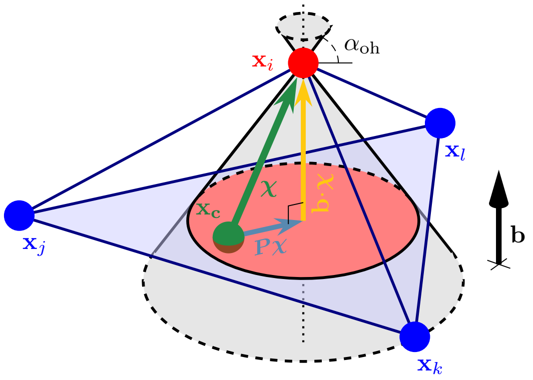


Figure 3.6: Given a point $\mathbf{x_c}$ on the triangle $Q = \mathbf{x}_j\mathbf{x}_k\mathbf{x}_l$, the arrival time at $\mathbf{x}_i$ is dependent on either the distance $|\mathbf{b}\cdot\boldsymbol{\chi}|$ inside the given cone, or $\|\mathbf{P}\boldsymbol{\chi}\|$ outside this cone, due to the maximum operator between the two (Equation (3.19)).

A CLOSER LOOK AT THE LOCAL UPDATE FUNCTION

The minimization problem posed by Equation (3.18) is first solved on the plane that contains the triangle Q. It is later evaluated if the minimum is actually inside Q (i.e. $0 \leq \zeta_1 \leq 1$, $0 \leq \zeta_2 \leq 1$ and $\zeta_1 + \zeta_2 \leq 1$). When the speed function f_s as defined in Equation (3.7) is substituted in Equation (3.18), it can be rewritten as

$$T^i_{jkl} = \min_{\zeta_1,\zeta_2:\mathbf{x_c}\in Q}\left\{\frac{\max\left(\tan(\alpha_{oh})\,\|\mathbf{P}\boldsymbol{\chi}\|\,,\left|\mathbf{b}\cdot\boldsymbol{\chi}\right|\right)}{f_0\phi_i} + T_{\mathbf{c}}\right\}. \tag{3.19}$$

Here $|\mathbf{b}\cdot\boldsymbol{\chi}|$ represents the length of $\boldsymbol{\chi}$ projected on the build direction $\mathbf{b}$, and $\|\mathbf{P}\boldsymbol{\chi}\|$ represents the length of $\boldsymbol{\chi}$ projected on the plane orthogonal to the build direction. This can be seen as a right-angled triangle, where $\boldsymbol{\chi}$ is the hypotenuse, and $|\mathbf{b}\cdot\boldsymbol{\chi}|$ and $\|\mathbf{P}\boldsymbol{\chi}\|$ are the lengths of the legs, as displayed in Figure 3.6.

Next, the maximum function in Equation (3.19) can be interpreted as follows. Let C be a double cone with vertex $\mathbf{x}_i$, its axis parallel to the build direction $\mathbf{b}$ and its aperture such that the generatrices of the cone make an angle α_{oh} with the base plate (see Figure 3.6). The following relations apply:

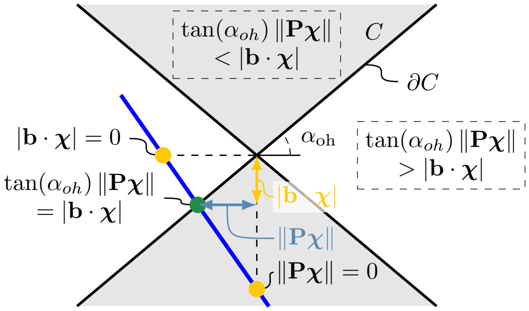
$$\tan(\alpha_{oh})\,\|\mathbf{P}\boldsymbol{\chi}\| = |\mathbf{b}\cdot\boldsymbol{\chi}|, \qquad \mathbf{x_c} \in \partial C \tag{3.20}$$

$$\tan(\alpha_{oh})\,\|\mathbf{P}\boldsymbol{\chi}\| < |\mathbf{b}\cdot\boldsymbol{\chi}|, \qquad \mathbf{x_c} \in C \tag{3.21}$$

$$\tan(\alpha_{oh})\,\|\mathbf{P}\boldsymbol{\chi}\| > |\mathbf{b}\cdot\boldsymbol{\chi}|, \qquad \mathbf{x_c} \notin C. \tag{3.22}$$

This implies that inside the cone (the red area in Figure 3.6), $|\mathbf{b}\cdot\boldsymbol{\chi}|$ is dominant, and outside the cone $\tan(\alpha_{oh})\,\|\mathbf{P}\boldsymbol{\chi}\|$ is dominant. This is also drawn for 2D in Figure 3.7, where it can be seen that for any plane, indicated with the blue line, that does not contain $\mathbf{x}_i$, $|\mathbf{b}\cdot\boldsymbol{\chi}|$ is minimized outside the cone, and $\|\mathbf{P}\boldsymbol{\chi}\|$ is minimized inside the cone. The maximum function is thus minimized on the cone edge ∂C when $\tan(\alpha_{oh})\,\|\mathbf{P}\boldsymbol{\chi}\| = |\mathbf{b}\cdot\boldsymbol{\chi}|$.

Besides the maximum term, Equation (3.19) also contains the interpolation of the known arrival times $T_{\mathbf{c}}$, which adds a linear field to the maximum term. Since the linear field $T_{\mathbf{c}}$ has no interior optimum (Equation (3.16)), the minimum of Equation (3.19)



$\tan(\alpha_{oh})\,\|\mathbf{P}\chi\| < |\mathbf{b}\cdot\chi|$
C
∂C
$|\mathbf{b}\cdot\chi| = 0$
α_{oh}
$\tan(\alpha_{oh})\,\|\mathbf{P}\chi\| > |\mathbf{b}\cdot\chi|$
$\tan(\alpha_{oh})\,\|\mathbf{P}\chi\| = |\mathbf{b}\cdot\chi|$
$|\mathbf{b}\cdot\chi|$
$\|\mathbf{P}\chi\|$
$\|\mathbf{P}\chi\| = 0$

Figure 3.7: For any plane, indicated in blue, that does not contain $\mathbf{x}_i$, both $|\mathbf{b}\cdot\chi|$ and $\|\mathbf{P}\chi\|$ are minimized at a location where they are not dominant in the maximum term of Equation (3.19). The maximum term is thus minimized at the cone edge ∂C.

will either remain on the cone edge ∂C when the gradient of this linear field is small, or the minimum will be at the bounds of the domain when the gradient of the linear field dominates. Based on these observations, the minimization problem is examined for two possibilities:

1. The maximum term is dominant. The minimum occurs at the cone edge ∂C.
2. The arrival time interpolation term is dominant. The minimum occurs outside of the cone C, on the edge of the triangle Q.

Both scenarios are discussed subsequently.

Minimum on cone edge On the cone edge, both terms that appear on the maximum function in Equation (3.19) are equal, and either one can be chosen to minimize over the cone edge. Since $|\mathbf{b}\cdot\chi|$ is piecewise linear, it is preferred over the quadratic function $\|\mathbf{P}\chi\|$. First, this function is minimized on the plane defined by the triangle Q. The update can be cast in the following form (see Appendix 3.A for details)

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{g}^T\mathbf{y} \\ \text{s.t.} \quad & \mathbf{y}^T\mathbf{K}\mathbf{y} + \mathbf{l}^T\mathbf{y} + c_6 = 0, \end{aligned} \tag{3.23}$$

where $\mathbf{g}$ contains the gradients of the update functions inside the cone, $\mathbf{K}$, $\mathbf{l}$ and c_6 are parameters defined by the geometry of the cone-plane intersection, and $\mathbf{y} = \begin{bmatrix}\zeta_1 & \zeta_2\end{bmatrix}^T$. This formulation minimizes a linear function on a cone-plane intersection, and can be solved explicitly using Lagrange multipliers (Appendix 3.A).

If the minimum found by solving Equation (3.23) is not inside the triangle Q, the minimum might occur on the intersection of an edge of Q with the cone. In order to find the edge-cone intersections, an algorithm described in Schneider and Eberly (2003) is used, which solves the following quadratic equation for a given edge $\mathbf{x}_j\mathbf{x}_k$ and unknown $\mathbf{x}_i$:

$$c_2\gamma^2 + c_1\gamma + c_0 = 0, \tag{3.24}$$

with $c_2 = \bar{\mathbf{a}}^T M \bar{\mathbf{a}}$, $c_1 = 2\bar{\mathbf{a}}^T M \bar{\mathbf{b}}$, $c_0 = \bar{\mathbf{b}}^T M \bar{\mathbf{b}}$, $M = \mathbf{b} \otimes \mathbf{b} - \sin^2(\alpha_{oh}) I$, $\bar{\mathbf{a}} = \mathbf{x}_k - \mathbf{x}_j$, and $\bar{\mathbf{b}} = \mathbf{x}_j - \mathbf{x}_i$. This gives two solutions for γ, from which the minimum locations are calculated with $\mathbf{x_c} = \gamma \mathbf{x}_j + (1-\gamma)\mathbf{x}_k$.

Minimum outside cone If the minimum lies outside of the cone, the following equation is to be minimized on each edge of triangle Q:

$$T^i_{jk} = \min_{\gamma \in [0,1]} \left\{ \frac{\tan(\alpha_{oh}) \left\| \mathbf{P}\boldsymbol{\chi} \right\|}{f_0 \phi_i} \right. \\ \left. + \gamma T_j + (1-\gamma) T_k \right\}. \tag{3.25}$$

This equation can be rewritten as

$$T^i_{jk} = \min_{\gamma \in [0,1]} \left\{ \hat{d}\sqrt{\gamma^2 \hat{a} + \gamma \hat{b} + \hat{c}} + \gamma \hat{e} + \hat{f} \right\} \tag{3.26}$$

with $\hat{a} = \hat{\mathbf{a}} \cdot \hat{\mathbf{a}}$, $\hat{b} = 2\hat{\mathbf{a}} \cdot \hat{\mathbf{b}}$, $\hat{c} = \hat{\mathbf{b}} \cdot \hat{\mathbf{b}}$, $\hat{d} = \tan(\alpha_{oh})/(f_0\phi_i)$, $\hat{e} = T_j - T_k$, $\hat{f} = T_k$, $\hat{\mathbf{a}} = \mathbf{P}(\mathbf{x}_k - \mathbf{x}_j)$, and $\hat{\mathbf{b}} = \mathbf{P}(\mathbf{x}_i - \mathbf{x}_k)$. The stationary points are found at

$$\gamma = \frac{\hat{b}\hat{e}^2 - \hat{a}\hat{b}\hat{d}^2 \pm \hat{e}\sqrt{(\hat{a}\hat{d}^2 - \hat{e}^2)(-\hat{b}^2 + 4\hat{a}\hat{c})}}{2\hat{a}(\hat{a}\hat{d}^2 - \hat{e}^2)}. \tag{3.27}$$

This again gives two solutions for γ, from which the minimum locations are calculated with $\mathbf{x_c} = \gamma \mathbf{x}_j + (1-\gamma)\mathbf{x}_k$.

Finding the minimum Each of the minimization problems posed in the previous two paragraphs returns a number of potential minimizers $\mathbf{x_c}$ for Equation (3.18). Since the minimum might not be inside the triangle, and not on the edges, the three corners of Q are also appended to the list of potential minimizers. Finally, the true minimum can be found by evaluating T_i with Equation (3.17) for each potential minimizer, and accepting the one with the lowest value.

Note that it is possible to exclude potential minimizers by looking at the second derivative, however because the evaluation of the minimum is extremely cheap, this did not result in a speed up.

3.3.2. Parallel front propagation

The local update described in the previous section can propagate the front from three nodes with a known arrival time to a fourth node. Another aspect of a front propagation algorithm is the order in which the nodes are updated. This is independent of the spatial dimension of the design domain, and in principle one could use the Ordered Upwind Method (OUM) (Sethian and Vladimirsky, 2003), similar to the 2D overhang filter as presented in van de Ven *et al.* (2018). From this work it followed that in 2D, the computational time of the overhang filter scaled roughly linear with number of DOFs, and was about one to two orders of magnitude smaller, compared to the computational time of the finite element analysis (FEA) for a stiffness optimization. In 3D, the computation times continue to scale favourably for the front propagation. However, the FEA associated with the objective evaluation is commonly parallelized, reducing its computation time proportional to the number of processors available. It is therefore paramount to parallelize the front propagation as well, such that the overhang filter does not become the main computational burden of the topology optimization.

Sequential Ordered Upwind Method

For the sake of completeness, we briefly discuss the sequential OUM detailed in Sethian and Vladimirsky (2003). The OUM is an extension of the Fast Marching Method (FMM) (Kimmel and Sethian, 1998; Sethian, 1996) that can also handle anisotropic speed functions. In the OUM, nodes are labelled either *Far*, *Considered* or *Accepted*. The *Accepted* nodes have their final arrival times, i.e. are not subject to further change. The *Considered* nodes have been updated at least once from the *Accpeted* nodes, but their arrival time values might still change. Finally, the *Far* nodes have not yet been updated and their arrival time are initialised at ∞. Furthermore, *Accepted* nodes can have the additional label *AcceptedFront* when they are adjacent to at least one *Considered* or *Far* node. The *AcceptedFront* is the current frontier of the propagation. When a node is updated, its arrival time is calculated from the nodes on the *AcceptedFront* within a radius $\bar{h}(F_2/F_1)$, where $\bar{h}$ is the maximum element edge length, and F_1 and F_2 the minimum and maximum of the direction dependent part of speed function given in Equation (3.5). Thus, F_2/F_1 represents the anisotropy ratio of the speed function, and for $\alpha_{\text{oh}} = 45°$ this is $F_2/F_1 = \sqrt{2}$.

The domain is initialized with all nodes labeled *Far*, except for the boundary nodes on $\partial\Omega_0$, which are labeled *Accepted*. Next, the *Far* nodes adjacent to the boundary are updated from the *AcceptedFront* nodes, using the local update described in the previous section, and become *Considered*. The algorithm then proceeds as described in Algorithm 2. When the algorithm terminates, all the nodes are in *Accepted* with the correct arrival times.

Algorithm 2 Front propagation

```
1: Move the Considered node with the lowest arrival time to Accepted.
2: Compute arrival times for the Far and Considered nodes within the radius $\bar{h}(F_2/F_1)$
   from the latest Accepted node, with adjacent nodes in the AcceptedFront, using the
   local update (Section 3.3.1). If the computed arrival time is smaller than the current
   value, update the arrival time. Label the nodes as Considered if they were previously
   labelled as Far.
3: If Considered is not empty, move to Step 1.
```

Parallel Ordered Upwind Method

Several parallel implementations of the FMM have been presented (e.g. Breuß *et al.* (2011); Herrmann (2003); Tugurlan (2008); Weber *et al.* (2008); Yang and Stern (2017)). However, to the best of our knowledge, no parallel implementation of the OUM has been published. We therefore present the parallel implementation of the OUM by adapting the algorithm presented in Yang and Stern (2017) to accommodate the anisotropic front propagation.

The parallelization of the FMM presented in Yang and Stern (2017) is based on a domain decomposition. At the basis of the parallel implementation of the OUM lies the sequential front propagation described in Algorithm 2, but within a subset of the entire domain, i.e. the narrow band. That implies that Algorithm 2 is terminated either if *Considerd* is empty, or if the node in *Considered* with the lowest arrival time has a value higher than a given band limit. As such, the front is only propagated a limited distance, instead of through the entire domain. Furthermore, for anisotropic speed functions, nodes are not necessarily updated from adjacent nodes. Therefore, there must be an overlapping region between adjacent domains, termed the ghost region, at least as wide as the update radius $\bar{h}(F_2/F_1)$. Finally, the front propagation must be redone for all the *Accepted* nodes that have an arrival time higher than the updated ghost nodes, because these *Accepted* nodes can possibly get a lower arrival time from the ghost nodes updated from adjacent domains. An outline of the parallel OUM algorithm that is executed in every parallel domain, is listed in Algorithm 3. The algorithm mainly consists of a loop which performs a front propagation, updates the domain boundary with parallel domains, rolls back the front propagation, and performs a corrective front propagation with the updated ghost nodes.

In the domain decomposition strategy as proposed by Yang and Stern (2017), each processor has a fixed part of the domain. This has the benefit that only boundary information needs to be communicated to other processors. However, it also results in a large amount of idle time, as the front will only propagate briefly trough each decomposed domain. We therefore implemented a shared memory approach per machine, where every processor can work on every domain. By having many more domains than processors, there is little idle time as each processor can pick a domain in which the front needs to be propagated. The downside is that between each machine, not only the boundary nodes, but all the information of every domain that has been updated needs to be shared, in order to allow every processor to work on every domain. This large amount of data communication could be avoided by assigning a fixed domain to every machine, and then further dividing these domains locally. This is however left for future work.

Algorithm 3 Parallel front propagation

```
1: Perform a front propagation up to the band limit.
2: For each ghost node determine the domain containing the minimum value, and scatter that value to the other domains.
3: Exit if there are no updated nodes and Considered is empty.
4: Move Accepted nodes that have an arrival time higher than or equal to the updated ghost node with the lowest arrival time to Considered, and add these nodes to the heap.
5: Perform a front propagation up to the band limit.
6: Increase the band limit, move to Step 1.
```

SENSITIVITIES

The sensitivities of the front propagation problem can be calculated efficiently in an adjoint sense, as shown in van de Ven *et al.* (2018). Note that due to the discrete minimum operator, the algorithm is not fully differentiable, which in specific cases can lead to erroneous sensitivities, although in practice it seems to have little influence on the optimization process. This is discussed in more detail in Section 5.4.3. The adjoint is calculated backwards, in exactly the opposite order as the front propagation. However, when the front propagation is performed in parallel, the order of operations is no longer clearly defined, as several nodes are accepted simultaneously in different domains. Fortunately, for every node it is known from which nodes its arrival time has been calculated. Assume that for every node i, the array $dependence(i)$ contains the indices of the nodes from which the arrival time of node i is calculated. In order to calculate the adjoint in the correct order we therefore propose the strategy outlined in Algorithm 4.

The sensitivity calculation can also be parallelized, where essentially this algorithm is executed in every domain, until the queue is empty, then the dependency and adjoint values of the ghost nodes are exchanged, and Algorithm 4 is restarted. This is repeated until all the sensitivities have been calculated.

Algorithm 4 Parallel front propagation sensitivities

```
Set up a vector $dep\_count$, such that $dep\_count(i)$ contains the number of nodes
  whose arrival time is calculated from node $i$
Add the nodes $i$ with $dep\_count(i) = 0$ to queue L
while L is not empty do
    $i = \textsc{Pop}(L)$
    Calculate adjoint and sensitivities for node $i$
    for all $j \in dependence(i)$ do
        Propagate the adjoint to node $j$
        $dep\_count(j)--$
        if $dep\_count(j) == 0$ then
            Add $j$ to queue L
        end if
    end for
end while
```

3.4. NUMERICAL EXAMPLES

The proposed front-propagation-based overhang filter is demonstrated on three cases, presented in the following, using the optimization scheme presented in Section 3.2.3. In all three cases compliance is minimized, and the optimization problem reads

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \mathbf{f}^T\mathbf{u} \\ \text{s.t.} \quad & \mathbf{K}(\boldsymbol{\xi})\mathbf{u} = \mathbf{f}, \\ & \int_\Omega \xi \mathrm{d}\Omega / V_{\mathrm{lim}} - 1 \leq 0, \\ & \boldsymbol{\rho}_{\min} \leq \boldsymbol{\rho} \leq \mathbf{1}, \end{aligned} \tag{3.28}$$

where **f** and **u** represent the load and displacement vectors, respectively. $\mathbf{K}(\boldsymbol{\xi})$ is the stiffness matrix dependent on the printable densities ξ. V_{lim} is the maximum allowed volume fraction and $\rho_{\min} = 0.01$. For more details on the compliance optimization problem and its sensitivities, the reader is referred to Bendsøe and Sigmund (2004).

In all cases, a density filter is used with a radius of roughly $2\bar{h}$, where $\bar{h}$ is the average edge length of the elements. The domains are discretized with linear tetrahedral elements using Gmsh (Geuzaine and Remacle, 2009). Furthermore, the material properties are chosen as $E = 1.0\,\mathrm{N/m^2}$ and $\nu = 0.3$, and the SIMP-model is used to interpolate the Young's modulus with $p = 3$ (Bendsøe, 1989). The topology optimization code utilizes the Portable, Extensible Toolkit for Scientific Computing (PETSc), which is used to parallelize the density filter and the FEA, and to handle the unstructured mesh (Balay *et al.*, 1997, 2016; Karypis and Kumar, 1998). The optimization is performed with the Method of Moving Asymptotes (MMA) (Svanberg, 1987), with default asymptote increase and decrease parameters of 1.2 and 0.7, respectively. For this, the PETSc based MMA class of the code presented in Aage *et al.* (2015) has been used (Aage and Lazarov, 2013). The topologies are visualised by displaying the printable density field for $\xi \geq 0.5$. This gives an isosurface of $\xi = 0.5$ inside the domain, and $\xi \geq 0.5$ at the bounds of the domain, as can be seen in Figure 3.8. The topologies are displayed without any smoothing, and are obtained using ParaView (Ahrens *et al.*, 2005).

Furthermore, similar to the 2D implementation, it is preferred to initially deactivate and then gradually activate the overhang filter, in order to apply the overhang filter to a better initial design. This is achieved with a linear interpolation between the filtered densities and the printable densities:

$$\hat{\xi} = (1-c)\rho^* + c\xi, \tag{3.29}$$

where c is the continuation variable. The objective and constraints are then evaluated on $\hat{\xi}$ instead of ξ. The continuation scheme applied is as follows: $c = 0$ for the first 10 iterations, then c is increased by 0.1 every iteration until $c = 1$ at iteration 20.

Finally, all optimizations are terminated after 300 iterations. It was found that for the larger 3D cases, it can easily take up to 1000 iterations before conventional convergence criteria are met, such as the maximum change in design variables smaller than $1 \cdot 10^{-2}$, or the relative objective change below $1 \cdot 10^{-6}$. However, after 300 iterations the change in design is minute, and the largest improvement in objective observed by allowing the optimization to run for 1000 iterations instead of 300 was around 0.5%. Therefore, in order to save computational time, and also to mimic a practical engineering application where computational time will be the limiting factor, the number of iterations has been limited to 300, instead of waiting for convergence.

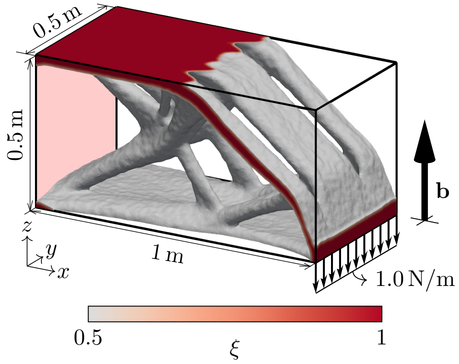


Figure 3.8: The cantilever beam design domain and boundary conditions. The light red surface at $x = 0$ is fully clamped, while a distributed load is acting on the front rib, and the build direction is equal to the positive z-axis. The displayed topology is optimized without overhang filter. The density field is displayed for $\xi \geq 0.5$. This gives an isosurface of $\xi = 0.5$ inside the domain, and $\xi \geq 0.5$ at the bounds of the domain.

3.4.1. Case 1: the cantilever beam

The overhang filter is first demonstrated on the well-known cantilever beam problem, for several different overhang angles. The design domain is a block with aspect ratio 2:1:1, which is fully clamped on one side and a distributed load is acting on a rib on the opposite side, as displayed in Figure 3.8. The resulting design is to be printed in the positive z-direction, as indicated by the build direction **b**, and the base plate coincides with the bottom of the domain ($z = 0$). The filter radius is chosen to be 15 mm, the volume constraint is set at 20%, and the domain is discretized with an unstructured tetrahedral mesh, consisting of $4.3 \cdot 10^6$ elements and $756 \cdot 10^3$ nodes, resulting in an average edge length of 8 mm.

The resulting designs are shown in Figure 3.9. It can be seen that the design without overhang filter contains a large overhanging surface at the top. With an increasing allowable minimum overhang angle, the design changes towards a topology with two separated columns, connected with a plate to the bottom. A similar design can be observed in Langelaar (2016). In Figure 3.9c the topology is coloured by the angle between the surface and the base plate, with $\alpha = 0°$ implying a surface parallel to the base plate. The view is from underneath the domain, with the bottom part cut off to allow a view on the downward facing surfaces. It can be seen that for the topologies optimized with overhang filter, there are a lot of green areas, which are surfaces that lie exactly at the minimum overhang angle. Overall the overhang constraint is satisfied, except in some small regions when two surfaces meet, or some individual elements due to the mesh roughness. These small localized overhanging regions are not an issue in AM practice, as they can be printed without support.

Convergence and objective

An interesting aspect of applying an overhang filter is the effect on the convergence behavior and the final objective. The convergence graphs for the cantilever beam problem are displayed in Figure 3.10. For the first 10 iterations, the overhang filter is inactive and

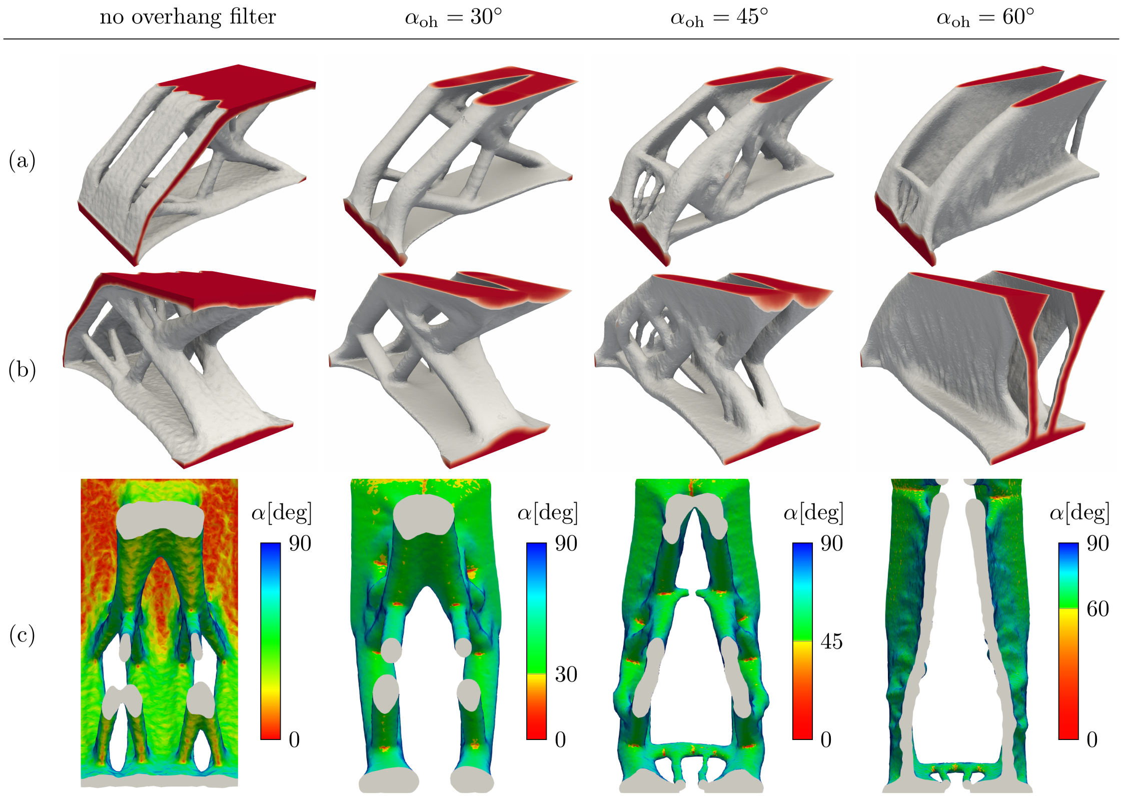


Figure 3.9: Resulting topologies for the cantilever beam with a 20% volume constraint, without (left), and with overhang filter with overhang angles of 30°, 45°, and 60°. Rows (a)-(b) show the designs from different angles, while row (c) shows the designs from the bottom, with the bottom plate cut off. For rows (a)-(b), the same color scheme as in Figure 3.8 has been used. In row (c), the colors represent the angle α between the base plate and the isosurface, chosen such that yellow-red colors indicate overhanging surfaces for the given overhang constraint. It can be seen that all designs are practically free of overhang.

thus all optimizations have the same objective values. During iterations 10-20, the overhang filter is activated gradually as descried in Section 3.2.3, and there is a significant increase in objective for $\alpha_{\text{oh}} = 45°$ and $\alpha_{\text{oh}} = 60°$. Then, the designs adapt to the overhang filter, and the objective steadily decreases until the optimization is stopped at 300 iterations.

As expected, the optimization without overhang filter has the lowest objective. However, the performance of the overhang filtered optimization is counter-intuitive: the optimization with the lowest minimum overhang angle has the worst performance, while the largest overhang angle performs best. The reason for this behavior is the fact that the optimizer can use the overhang filter to make smaller features on overhanging surfaces than the filter radius. This effect increases with increasing minimum overhang angle, as larger minimum overhang angles have a lower sideways propagation speed (see Figure 3.2c, for larger overhang angles the height of the cylinder remains the same, but the radius decreases). This means that the delay increases faster over distance, which produces sharper features. Therefore, the $\alpha_{\text{oh}} = 60°$ setting can make the sharpest edges, and achieves a lower objective. This can be prevented by placing the density filter after the overhang filter, however, this will reintroduce some overhang due to rounding. Therefore, we choose to place the overhang filter after the density filter. Furthermore, this behavior only emerges in cases that are relatively insensitive to the design, such as this cantilever beam.

PARALLEL SPEEDUP

The cantilever beam problem is used to evaluate the parallel performance of the overhang filter. The average wall clock time over the first 10 iterations of the FEA, the overhang filter, and the corresponding sensitivities is plotted in Figure 3.11, for various numbers of processors. It can be seen that the overhang filter is roughly a factor 10 faster than the FEA. The speedup is defined as T_{np}/T_1, where T_1 is the wall clock time for the serial process, and T_{np} is the wall clock time for np processors. The speedup of the overhang filter is close to linear for a low number of processors, and drops for larger number of processors, comparable to the FEA. The speedup is tested up to 20 processors, which was the largest number of cores available for a single computational node. Because the

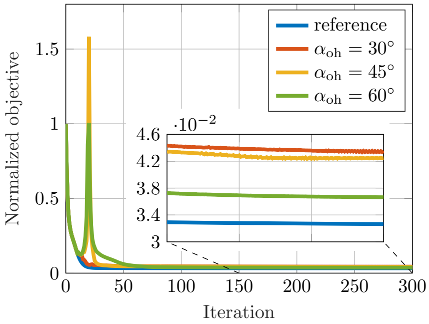


Figure 3.10: Convergence graph for the cantilever beam problem, normalized with the first objective evaluation. The reference is optimized without overhang filter.

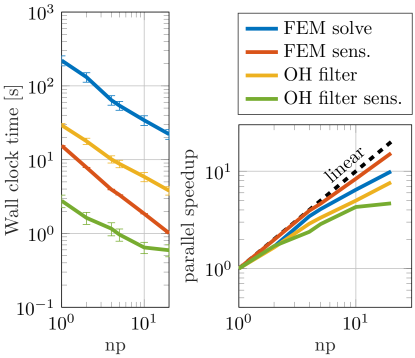


Figure 3.11: Wall clock time and speedup, defined as T_{np}/T_1, over number of processors (np) of the overhang filter compared to the FEA, and the sensitivity computations. Error bars indicate the standard deviation.

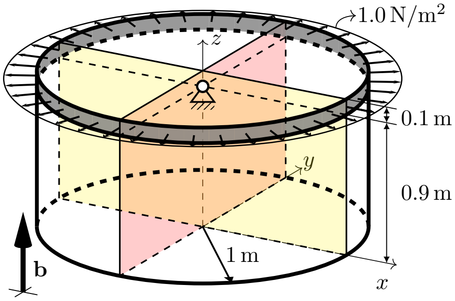


Figure 3.12: The tension cylinder design domain with boundary conditions. The tension load acts on the gray ring, while the red surface is clamped in the x-direction, and the yellow surface is clamped in the y-direction. One point on the z-axis is fully clamped, as indicated. The build direction is equal to the positive z-axis, and the build plate is the $z = 0$ plane

available cluster did not feature a fast interconnect, the speedup of the overhang filter did not increase when an extra node was added to the system. Therefore, a hybrid implementation between a complete domain decomposed approach as presented in Yang and Stern (2017), and a shared memory approach adopted in this work (see Section 3.3.2) would be faster, as it would require only boundary node communication.

3.4.2. CASE 2: TENSION CYLINDER

The second case on which the overhang filter is tested is a cylinder under tensional loading. This is a critical test for the overhang filter, as for the lower volume fractions, most of the material that is needed for support is not contributing to the structural stiffness. The design domain is a cylinder, with a distributed load acting on the top part of the outer boundary, as displayed in Figure 3.12. The filter radius is 20 mm, and the domain is meshed with $17 \cdot 10^6$ elements and $2.9 \cdot 10^6$ nodes, with in an average edge length of 12 mm.

Without overhang filter, the resulting design will roughly be a solid disk starting from

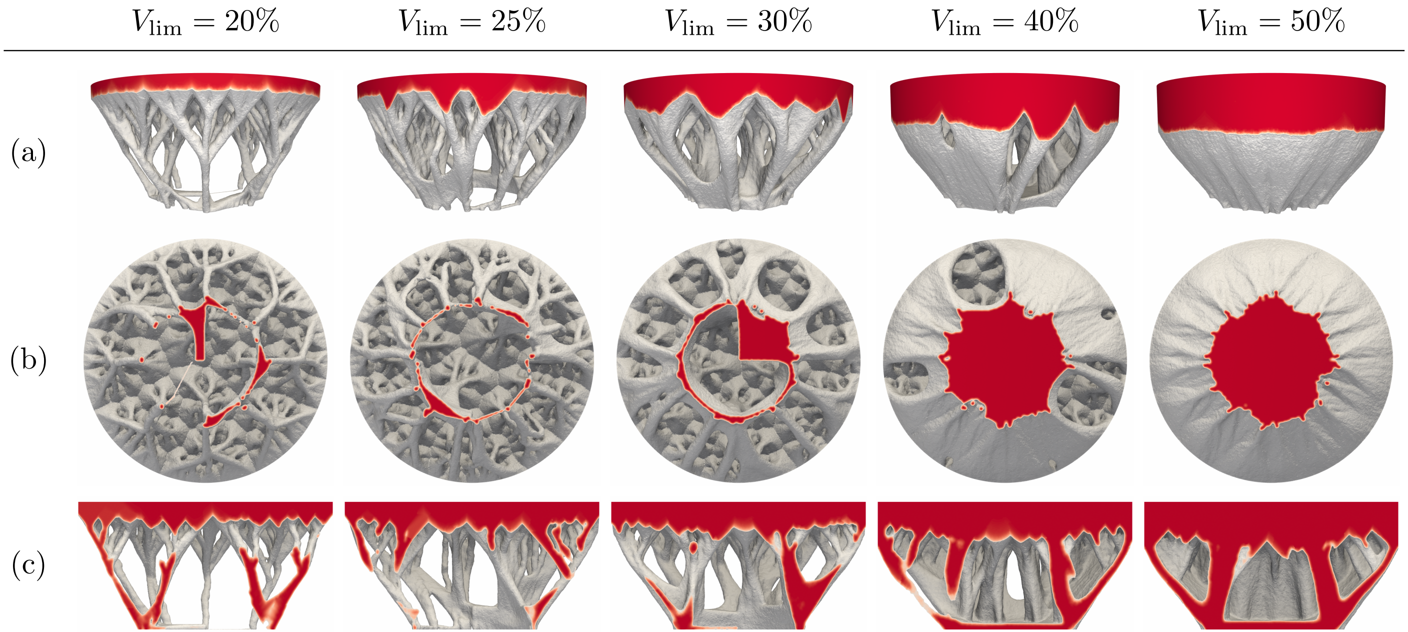

$V_{\mathrm{lim}} = 20\%$
$V_{\mathrm{lim}} = 25\%$
$V_{\mathrm{lim}} = 30\%$
$V_{\mathrm{lim}} = 40\%$
$V_{\mathrm{lim}} = 50\%$
(a)
(b)
(c)

Figure 3.13: Resulting designs for the tension cylinder case for different volume fractions viewed (a) from the side, (b) from the bottom, and (c) from the side while cut in half. The same color scheme as in Figure 3.8 has been used.

the top of the cylinder, with a height equal to the allowed volume fraction. The resulting topologies for different volume fractions with $\alpha_{\mathrm{oh}} = 45°$ are displayed in Figure 3.13. It can be seen that even for the lowest volume fraction, fully dense supports are generated, although they do not contribute to the stiffness. The supports fan out in a tree-like fashion to span an as large as possible area with the least amount of material. As the volume fraction increases, the topology resembles an inverted dome. For the higher volume fractions, the supports are also used to increase the stiffness, as they become interconnected.

3.4.3. CASE 3: CRANE HOOK

The final case on which the overhang constraint is demonstrated is the stiffness optimization of a crane hook. Due to the capability to detect overhang on an unstructured mesh, it is straightforward to optimize a geometry generated in a CAD environment, with overhang constraint. The geometry of the crane hook is displayed in Figure 3.14. It is a complex domain containing multiple curved surfaces, which would be difficult to accurately represent with at structured mesh. The design domain measures 455 × 491 × 118 mm (width × height × depth). A density filter with a radius of 3 mm is applied, and the elements have an average edge length of 1.5 mm. The volume constraint is set at 20%, and the build direction is in the vertical direction as displayed in Figure 3.14. The build plate is at the bottom of the domain.

The domain is fully clamped at the top, with a vertical distributed load applied on the hook, representing a lifting force. Furthermore, this test case features a fixed solid region at the loaded surface and near the tip of the hook (the red and blue surfaces in Figure 3.14). If the densities of the blue region would not be fixed, the material will be removed since it does not contribute to the stiffness. However, this region should be retained in the final topology to prevent the load from slipping off the hook. By simply fixing the design variables, the fixed regions will still be removed by the overhang filter when it is overhanging. We therefore added the following constraint:

$$\int_{\Gamma_f} (1-\xi)\ \mathrm{d}\Gamma_f \leq 0, \tag{3.30}$$

where Γ_f is the fixed density region. This constraint is only satisfied when the printable densities $\xi = 1$ on the fixed region, and this successfully fixed the printable densities of the fixed region.

The crane hook is the largest test case in the set, with the final mesh containing $50 \cdot 10^6$ elements and $8.8 \cdot 10^6$ nodes. The problem was solved using 160 cores divided over 8 nodes with two 10 core 2.2Ghz Intel Xeon CPUs (E-2630v4) each. The average wall clock time for the FEA was 193 seconds, while the average time for the overhang filter was 115 seconds. The computational time for the sensitivities was negligible compared to the forward solve, 1.9 seconds for the adjoint sensitivities of the FEA, and 17 seconds for the overhang filter. Although the computational time required for the overhang filter is considerable at 60% of the FEA, one has to keep in mind that the overhang filter is only running at 1 node, compared to 8 nodes for the FEA.

The optimized design is displayed in Figure 3.15. It resembles a common crane hook, with a curved stiffener at the back (the leftmost part in Figure 3.15a) to counteract the

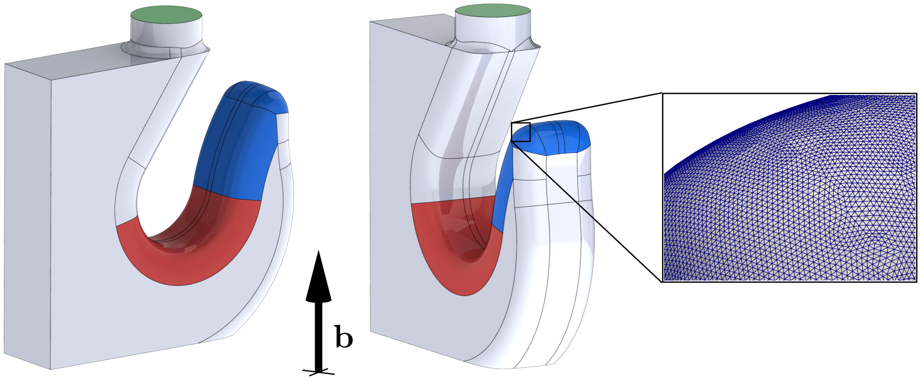
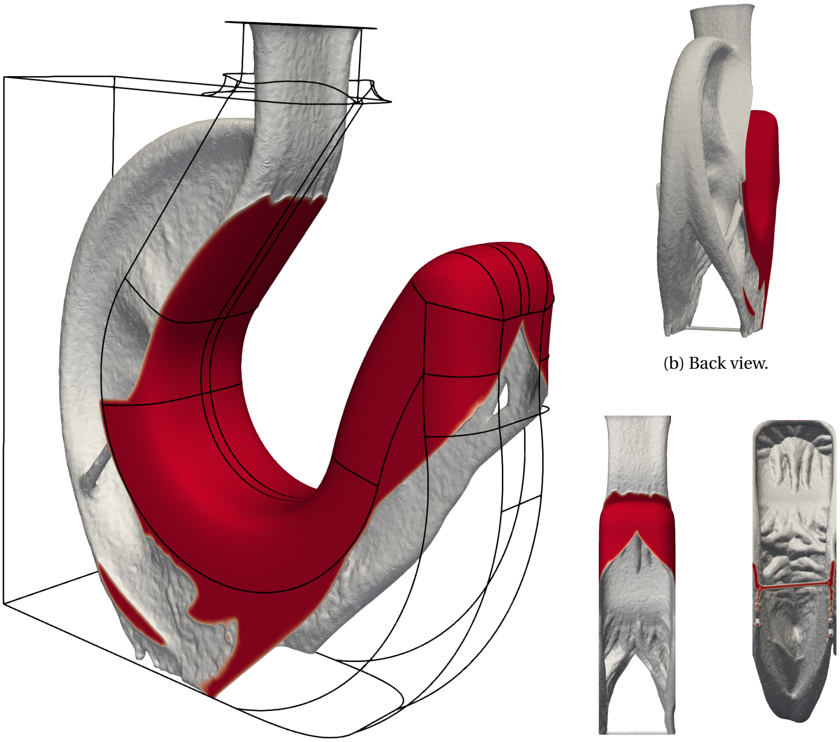


Figure 3.14: Design domain and tetrahedral mesh for the crane hook case. The green surface is fully clamped, a vertical distributed load is applied on the red surface, and the red and blue surfaces are constrained at full density.



(a) Side view.

(b) Back view.

(c) Front view.

(d) Bottom view.

Figure 3.15: Results of the crane hook case. The same color scheme as in Figure 3.8 has been used.

torque on the hook due to the asymmetric geometry. The stiffener contains some small hollow sections to save material. Also the fixed region at the tip is made as thin as possible, for the most part it is only a shell. The design is completely printable, as can be seen on the bottom in Figures 3.15b to 3.15d, where the members in the middle join under a 45° angle. The decrease in performance compared to an optimization without overhang constraint is 14%.

3.5. CONCLUSION

In this work, the front propagation based overhang constraint as presented for 2D in van de Ven *et al.* (2018) has been successfully extended to 3D. The two main challenges of the extension were the efficient propagation in tetrahedral elements, as opposed to triangles in 2D, and the parallelization of the front propagation in order to keep the computational times in the same order as the FEA, which is usually parallelized for 3D cases.

It is shown that also for 3D front propagation, a minimization problem can be set up for each tetrahedral element that can be solved by probing a limited set of cheaply obtainable locations. This enables the fast propagation through an unstructured mesh, for arbitrary overhang angle, contrary to minimum overhang angle filters that work on structured meshes.

The paralellization of the ordered upwind method was achieved by following the same strategy that has been presented for the closely related fast marching method (Yang and Stern, 2017). However, instead of providing each processor with its own fixed domain, a different approach was taken where every processor can work on any domain. Although this reduces processor idle time and provides speedup for relatively small meshes (on the order of $1 \cdot 10^6$ nodes), the approach presented in Yang and Stern (2017) is likely to give better performance on increasing mesh sizes. Nevertheless, the parallellized overhang filter is about a factor 10 faster than the parallel FEA, when executed on a single machine.

The overhang filter is demonstrated on three problems to test the performance of the filter. It is shown that the filter handles arbitrary overhang angles, generates solid supports even when there is no benefit for the objective, and works on large unstructured meshes. With the framework for parallel 3D front propagation, further research will focus on different applications of front propagation for topology optimization, for example for accessibility of supports: the complete removal of overhang is often not in the designer's best interest, and we thus would like to activate the overhang constraint only in areas where supports are difficult to remove.

ACKNOWLEDGEMENTS

This research was carried out under project number S12.7.14549a in the framework of the Metals for Additive Manufacturing Partnership Program of the Materials innovation institute M2i (www.m2i.nl) and the Netherlands Organization for Scientific Research (www.nwo.nl).

3.A. SOLVING THE CONE PROBLEM

Following Schneider and Eberly (2003), on the cone edge, the following equality holds:

$$(\mathbf{b}\cdot\boldsymbol{\chi})^2 - \sin(a_{\mathrm{oh}})^2\|\boldsymbol{\chi}\|^2 = 0, \tag{3.31}$$

with $\boldsymbol{\chi} = \mathbf{x}_i - \mathbf{c} = \zeta_1\tilde{\mathbf{a}} + \zeta_2\tilde{\mathbf{b}} + \tilde{\mathbf{c}}$, and $\tilde{\mathbf{a}} = (\mathbf{x}_j - \mathbf{x}_k)$, $\tilde{\mathbf{b}} = (\mathbf{x}_j - \mathbf{x}_l)$, and $\tilde{\mathbf{c}} = (\mathbf{x}_i - \mathbf{x}_j)$. By defining $\tilde{a} = \mathbf{b}\cdot\tilde{\mathbf{a}}$, $\tilde{b} = \mathbf{b}\cdot\tilde{\mathbf{b}}$, and $\tilde{c} = \mathbf{b}\cdot\tilde{\mathbf{c}}$, Eq. (3.31) can be rewritten as

$$c_1\zeta_1^2 + 2c_2\zeta_1\zeta_2 + c_3\zeta_2^2 + c_4\zeta_1 + c_5\zeta_2 + c_6 = 0 \tag{3.32}$$

with

$$\begin{aligned}
c_1 &= \tilde{a}^2 - \sin(a_{\mathrm{oh}})^2\tilde{\mathbf{a}}\cdot\tilde{\mathbf{a}}, & c_4 &= 2\tilde{a}\tilde{c} - 2\sin(a_{\mathrm{oh}})^2\tilde{\mathbf{a}}\cdot\tilde{\mathbf{c}},\\
c_2 &= \tilde{a}\tilde{b} - \sin(a_{\mathrm{oh}})^2\tilde{\mathbf{a}}\cdot\tilde{\mathbf{b}}, & c_5 &= 2\tilde{b}\tilde{c} - 2\sin(a_{\mathrm{oh}})^2\tilde{\mathbf{b}}\cdot\tilde{\mathbf{c}},\\
c_3 &= \tilde{b}^2 - \sin(a_{\mathrm{oh}})^2\tilde{\mathbf{b}}\cdot\tilde{\mathbf{b}}, & c_6 &= \tilde{c}^2 - \sin(a_{\mathrm{oh}})^2\tilde{\mathbf{c}}\cdot\tilde{\mathbf{c}}.
\end{aligned}$$

Eq. (3.32) can be rewritten as

$$\mathbf{y}^T\mathbf{K}\mathbf{y} + \mathbf{l}^T\mathbf{y} + c_6 = 0, \tag{3.33}$$

with $\mathbf{K} = \begin{bmatrix} c_1 & c_2 \\ c_2 & c_3 \end{bmatrix}$, $\mathbf{l} = \begin{bmatrix} c_4 & c_5 \end{bmatrix}^T$, $\mathbf{y} = \begin{bmatrix} \zeta_1 & \zeta_2 \end{bmatrix}^T$.

Since on the cone edge, $\tan(\alpha_{oh})\left\|\mathbf{P}\boldsymbol{\chi}\right\| = \left|\mathbf{b}\cdot\boldsymbol{\chi}\right|$, the minimization problem posed in Eq. (3.19) simplifies on the cone edge to

$$\begin{aligned}
\min_{\mathbf{y}} \quad & \left\{\frac{\left|\mathbf{b}\cdot\boldsymbol{\chi}\right|}{f_0\phi_i} + (1-\zeta_1-\zeta_2)T_j + \zeta_1 T_k + \zeta_2 T_l\right\},\\
\text{s.t.} \quad & \mathbf{y}^T\mathbf{K}\mathbf{y} + \mathbf{l}^T\mathbf{y} + c_6 = 0,
\end{aligned} \tag{3.34}$$

which can be rewritten as

$$\begin{aligned}
\min_{\mathbf{y}} \quad & \mathbf{g}^T\mathbf{y},\\
\text{s.t.} \quad & \mathbf{y}^T\mathbf{K}\mathbf{y} + \mathbf{l}^T\mathbf{y} + c_6 = 0,
\end{aligned} \tag{3.35}$$

with $\mathbf{g} = (\pm\tilde{\mathbf{d}}/(f_0\phi_i) + \tilde{\mathbf{T}})$, $\tilde{\mathbf{d}} = \begin{bmatrix} \tilde{a} & \tilde{b} \end{bmatrix}^T$ and $\tilde{\mathbf{T}} = \begin{bmatrix} T_k - T_j & T_l - T_j \end{bmatrix}^T$. Note that in Eq. (3.35), the constant terms have been dropped as they do not influence the minimization, and the absolute function is omitted by probing both the positive and the negative variant of $\tilde{\mathbf{d}}$. The constraint in Eq. (3.35) can be added to the objective with a Lagrange multiplier:

$$\mathscr{L}(\mathbf{y},\lambda) = \mathbf{g}^T\mathbf{y} + \lambda\left(\mathbf{y}^T\mathbf{K}\mathbf{y} + \mathbf{l}^T\mathbf{y} + c_6\right), \tag{3.36}$$

whose stationary points can be found by solving

$$\frac{\partial\mathscr{L}}{\partial\mathbf{y}} = \mathbf{g} + \lambda\left(2\mathbf{K}\mathbf{y} + \mathbf{l}\right) = 0, \tag{3.37}$$

$$\frac{\partial\mathscr{L}}{\partial\lambda} = \mathbf{y}^T\mathbf{K}\mathbf{y} + \mathbf{l}^T\mathbf{y} + c_6 = 0. \tag{3.38}$$

Now, λ can be found by solving Eq. (3.37) for **y** and substituting the result into Eq. (3.38). This gives

$$\lambda = \pm\sqrt{\frac{\mathbf{g}^T\mathbf{K}^{-1}\mathbf{g}}{\mathbf{l}^T\mathbf{K}^{-1}\mathbf{l} - 4c_6}}. \tag{3.39}$$

Substituting this result back into Eq. (3.37) gives

$$\mathbf{y} = -\frac{1}{2}\mathbf{K}^{-1}\left(\frac{\mathbf{g}}{\lambda} + \mathbf{l}\right). \tag{3.40}$$

Note that in total four potential minimum locations are found. Two resulting from the omission of the absolute sign by probing both possibilities, and two from the square root when calculating λ.

REFERENCES

Aage N, Lazarov BS (2013) Parallel framework for topology optimization using the method of moving asymptotes. Structural and multidisciplinary optimization 47(4):493–505

Aage N, Andreassen E, Lazarov BS (2015) Topology optimization using PETSc: An easy-to-use, fully parallel, open source topology optimization framework. Structural and Multidisciplinary Optimization 51(3):565–572

Adam GA, Zimmer D (2014) Design for Additive Manufacturing-Element transitions and aggregated structures. CIRP Journal of Manufacturing Science and Technology 7(1):20 – 28

Ahrens J, Geveci B, Law C (2005) 36 - ParaView: An End-User Tool for Large-Data Visualization. In: Hansen CD, Johnson CR (eds) Visualization Handbook, Butterworth-Heinemann, Burlington, pp 717–731

Allaire G, Dapogny C, Estevez R, Faure A, Michailidis G (2017) Structural optimization under overhang constraints imposed by additive manufacturing technologies. Journal of Computational Physics 351:295 – 328

Amir O, Mass Y (2018) Topology optimization for staged construction. Structural and Multidisciplinary Optimization 57(4):1679–1694

Balay S, Gropp WD, McInnes LC, Smith BF (1997) Efficient management of parallelism in object oriented numerical software libraries. In: Arge E, Bruaset AM, Langtangen HP (eds) Modern Software Tools in Scientific Computing, Birkhäuser Press, pp 163–202

Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Eijkhout V, Gropp WD, Kaushik D, Knepley MG, McInnes LC, Rupp K, Smith BF, Zampini S, Zhang H, Zhang H (2016) PETSc Users Manual. Tech. Rep. ANL-95/11 - Revision 3.7, Argonne National Laboratory

Bendsøe MP (1989) Optimal shape design as a material distribution problem. Structural optimization 1(4):193–202

Bendsøe MP, Sigmund O (2004) Topology Optimization: Theory, Methods and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg

Boothroyd G, Dewhurst P, Knight WA (2011) Product design for manufacture and assembly, 3rd edn. CRC Press, Boca Raton, Fl

Brackett D, Ashcroft I, Hague R (2011) Topology optimization for additive manufacturing. In: Proceedings of the solid freeform fabrication symposium, Austin, TX, pp 348–362

Breuß M, Cristiani E, Gwosdek P, Vogel O (2011) An adaptive domain-decomposition technique for parallelization of the fast marching method. Applied Mathematics and Computation 218(1):32–44

Bruns TE, Tortorelli DA (2001) Topology optimization of non-linear elastic structures and compliant mechanisms. Computer Methods in Applied Mechanics and Engineering 190(26):3443 – 3459

Deaton JD, Grandhi RV (2014) A survey of structural and multidisciplinary continuum topology optimization: Post 2000. Structural and Multidisciplinary Optimization 49(1):1–38

Gaynor AT, Guest JK (2016) Topology optimization considering overhang constraints: Eliminating sacrificial support material in additive manufacturing through design. Structural and Multidisciplinary Optimization 54(5):1157–1172

Geuzaine C, Remacle JF (2009) Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. International journal for numerical methods in engineering 79(11):1309–1331

Guo X, Zhou J, Zhang W, Du Z, Liu C, Liu Y (2017) Self-supporting structure design in additive manufacturing through explicit topology optimization. Computer Methods in Applied Mechanics and Engineering 323:27 – 63

Herrmann M (2003) A domain decomposition parallelization of the fast marching method. In: Annual Research Briefs, Center for Turbulence Research, Stanford University, Stanford, CA, pp 213–225

Hoffarth M, Gerzen N, Pedersen C (2017) ALM overhang constraint in topology optimization for industrial applications. In: Proceedings of the 12th world congress on structural and multidisciplinary optimisation, Braunschweig, Germany

Johnson TE, Gaynor AT (2018) Three-dimensional projection-based topology optimization for prescribed-angle self-supporting additively manufactured structures. Additive Manufacturing 24:667–686

Karypis G, Kumar V (1998) A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. Journal of Parallel and Distributed Computing 48:71–85

Kimmel R, Sethian JA (1998) Computing geodesic paths on manifolds. Proceedings of the National Academy of Sciences 95(15):8431–8435

Kranz J, Herzog D, Emmelmann C (2015) Design guidelines for laser additive manufacturing of lightweight structures in TiAl6V4. Journal of Laser Applications 27(S1):S14,001

Kruth JP, Vandenbroucke B, Van Vaerenbergh J, Mercelis P (2005) Benchmarking of different SLS/SLM processes as Rapid Manufacturing techniques. IEEE Electron Device Letters 10

Langelaar M (2016) Topology optimization of 3D self-supporting structures for additive manufacturing. Additive Manufacturing 12:60–70

Langelaar M (2017) An additive manufacturing filter for topology optimization of print-ready designs. Structural and Multidisciplinary Optimization 55(3):871–883

Langelaar M (2018) Combined optimization of part topology, support structure layout and build orientation for additive manufacturing. Structural and Multidisciplinary Optimization 57(5):1985–2004

Liu J, Gaynor AT, Chen S, Kang Z, Suresh K, Takezawa A, Li L, Kato J, Tang J, Wang CCL, Cheng L, Liang X, To AC (2018) Current and future trends in topology optimization for additive manufacturing. Structural and Multidisciplinary Optimization 57(6):2457–2483

Mirzendehdel AM, Suresh K (2016) Support structure constrained topology optimization for additive manufacturing. Computer-Aided Design 81:1–13

Qian X (2017) Undercut and overhang angle control in topology optimization: A density gradient based integral approach. International Journal for Numerical Methods in Engineering 111(3):247–272, nme.5461

Schneider PJ, Eberly DH (2003) Geometric Tools for Computer Graphics. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, Morgan Kaufmann Publishers, Amsterdam

Sethian JA (1996) A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Sciences 93(4):1591–1595

Sethian JA, Vladimirsky A (2003) Ordered upwind methods for static Hamilton–Jacobi equations: Theory and algorithms. SIAM Journal on Numerical Analysis 41(1):325–363

Sigmund O, Maute K (2013) Topology optimization approaches: A comparative review. Structural and Multidisciplinary Optimization 48(6):1031–1055

Sigmund O, Petersson J (1998) Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima. Structural optimization 16(1):68–75

Svanberg K (1987) The method of moving asymptotes – a new method for structural optimization. International journal for numerical methods in engineering 24(2):359–373

Thomas D (2009) The Development of Design Rules for Selective Laser Melting. PhD thesis, University of Wales

Thompson MK, Moroni G, Vaneker T, Fadel G, Campbell RI, Gibson I, Bernard A, Schulz J, Graf P, Ahuja B, Martina F (2016) Design for additive manufacturing: Trends, opportunities, considerations, and constraints. CIRP Annals 65(2):737 – 760

Tugurlan MC (2008) Fast marching methods-parallel implementation and analysis. PhD thesis, Louisiana State University

Vandenbroucke B, Kruth JP (2007) Selective laser melting of biocompatible metals for rapid manufacturing of medical parts. Rapid Prototyping Journal 13(4):196–203

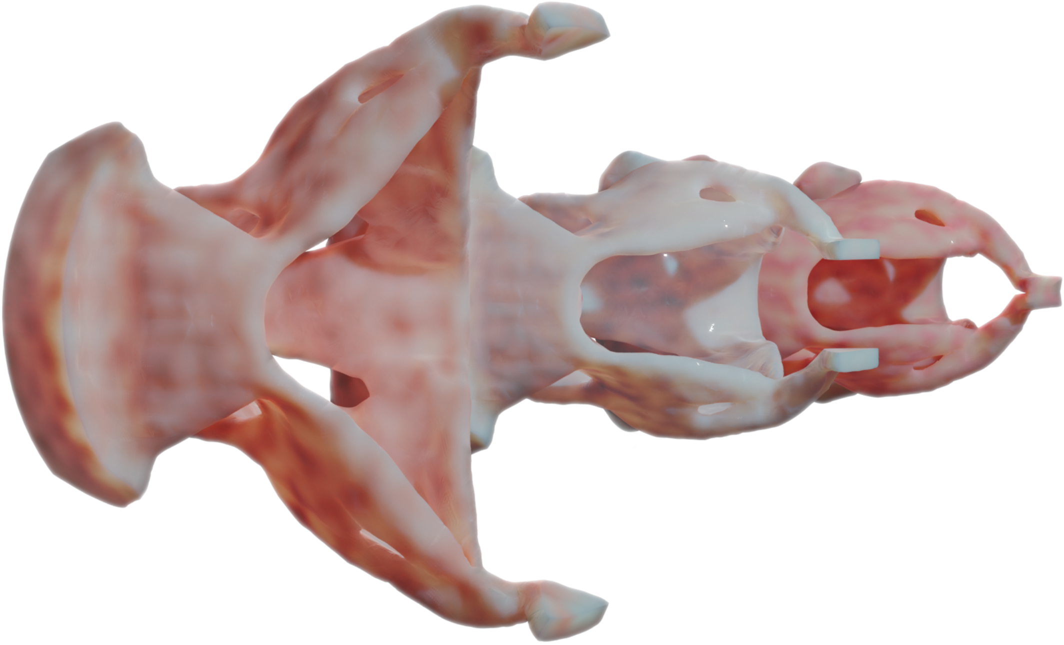
van de Ven E, Maas R, Ayas C, Langelaar M, van Keulen F (2018) Continuous front propagation-based overhang control for topology optimization with additive manufacturing. Structural and Multidisciplinary Optimization 57(5):2075–2091

Wang D, Yang Y, Yi Z, Su X (2013) Research on the fabricating quality optimization of the overhanging surface in SLM process. The International Journal of Advanced Manufacturing Technology 65(9-12):1471–1484

Wang W, Liu YJ, Wu J, Tian S, Wang CCL, Liu L, Liu X (2018) Support-Free Hollowing. IEEE Transactions on Visualization and Computer Graphics 24(10):2787–2798

Weber O, Devir YS, Bronstein AM, Bronstein MM, Kimmel R (2008) Parallel Algorithms for Approximation of Distance Maps on Parametric Surfaces. ACM Transactions on Graphics 27(4)

Yang J, Stern F (2017) A highly scalable massively parallel fast marching method for the Eikonal equation. Journal of Computational Physics 332:333–362



4

Overhang constraint demonstration: fluid flow optimization and compliant mechanism design

For some components produced with additive manufacturing, the use of support structures is cumbersome but acceptable. However, for high-potential applications of additive manufacturing such as conformal cooling, hydraulic manifolds, and compliant mechanisms, support structures in internal structures and channels are extremely difficult to remove, and often cannot be removed at all after printing. These are also applications that are well suited to be designed by topology optimization. In this chapter the front propagation-based overhang filter presented in the previous chapters is applied to fluid flow optimization and compliant mechanism design.

The compliant mechanism work presented in this chapter is based on a conference paper presented at the *2018 ASPE and euspen Summer Topical Meeting: Advancing Precision in Additive Manufacturing* (van de Ven *et al.*, 2018). The fluid flow optimization work is based on work presented at the *1st ECCOMAS Thematic Conference on Simulation for Additive Manufacturing (SIM-AM) 2017* and a book chapter in *3D Printing for Energy Systems* (in press).

In the previous chapters, the overhang constraint is demonstrated on compliance minimization problems. However, for an overhang constraint to be practically applicable, it has to function on any type of optimization problem. Therefore, we show in this chapter that the overhang constraint works just as well on, first, fluid flow optimization, and thereafter, compliant mechanism design.

4.1. Fluid flow optimization

On of the exciting possibilities enabled by additive manufacturing (AM) is the integration of fluid channels into components. For example, internal cooling channels that form intricate patterns or hydraulic manifolds with complex layouts can be manufactured by metal AM. In this section we want to demonstrate improvements that can be realized by topology optimization (TO) of a hydraulic manifold for AM.

Manifolds are components that facilitate fluid flow from an inlet to an outlet at predefined positions. Such components are found in numerous applications, e.g. oil and gas, hydropower, and the process and aerospace industry. Conventional manifolds often consist of straight cylindrical channels realized by drilling, meeting at right angles. This results in significant drag forces in the fluid due to the suboptimal flow configuration, and hence pressure drop which in turn increases power requirements for pumps. We here explore the potential to minimize this power dissipation over a manifold, by exploiting the design freedom that 3D printing offers. For that purpose a TO-based computational design approach combined with overhang constraint control is developed. Generating self-supporting designs is critical in this application, because support material cannot be removed from inaccessible regions such as the inner surface of channels. The complexity of the manifolds in the following examples is not comparable to industrial cases, and should be seen as a preliminary study to demonstrate the combination of fluid flow optimization and AM constraints.

4.1.1. Fluid flow topology optimization

Topology optimization of fluid problems started with the work of Borrvall and Petersson (2003), in which they optimized Stokes flow problems by modeling the design domain as a porous medium. The density field in TO is coupled to the flow problem by scaling the permeability of the medium: high density regions are considered impermeable, while regions with a low density are considered porous. Since then, optimization of Stokes flow became an active research topic (Aage *et al.*, 2008; Evgrafov, 2005; Guest and Prévost, 2006), and has been extended to laminar Navier-Stokes (Deng *et al.*, 2011; Gersborg-Hansen *et al.*, 2005; Kreissl *et al.*, 2011; Olesen *et al.*, 2006a; Pingen *et al.*, 2007), and later to the turbulent flow regime (Dilgen *et al.*, 2018; Kontoleontos *et al.*, 2013; Othmer, 2008; Zymaris *et al.*, 2009). Here we do not focus on the fluid flow aspects of the problem but instead demonstrate the combination of fluid flow TO with 3D printing constraints to arrive at a manufacturable design. Therefore, although some publications show complete differentiation of the Navier-Stokes equations and turbulence model (Dilgen *et al.*, 2018; Kontoleontos *et al.*, 2013; Zymaris *et al.*, 2009), we will follow the simplified “frozen turbulence” approach outlined in Othmer *et al.* (2007), where the turbulence model is not differentiated with respect to the design variables. For this purpose, we used the Open-

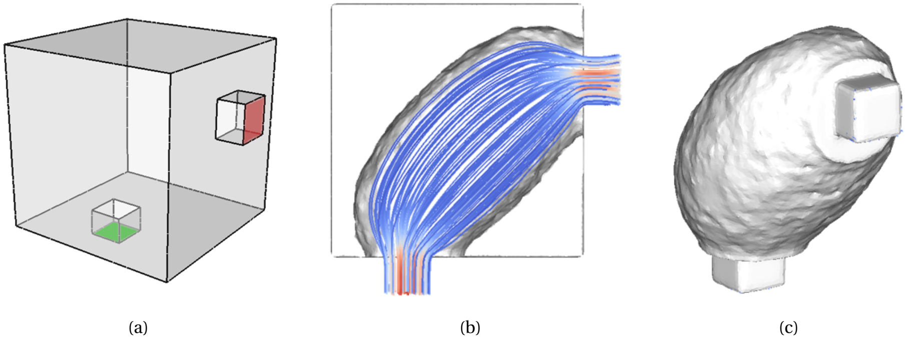


Figure 4.1: Topology optimization of a pipe bend. The design domain with inlet (red) and outlet (green) is shown in (a), and the resulting design with flow lines in (b), where a blue and red color represent a low and high flow speed, respectively. The fluid domain is shown in (c), which is the inverse of the design in figure (b).

FOAM framework (OpenCFD). Before including the 3D-printing constraint, we will first investigate conventional fluid optimization.

Assuming steady-state incompressible flow and adopting the porous medium interpretation of the density field, the Reynolds-averaged Navier-Stokes (RANS) equations which describe the motion of the fluid are given by

$$\begin{aligned} \rho_0 (\mathbf{v}\cdot\nabla)\,\mathbf{v} &= -\nabla p + \mu\Delta\mathbf{v} - \alpha\mathbf{u}, \\ \nabla\cdot\mathbf{v} &= 0, \end{aligned} \tag{4.1}$$

where ρ_0 and μ are the fluid density and viscosity assumed to be constant, $\mathbf{v}$ and p are the mean velocity and pressure fields. $\alpha\mathbf{u}$ is the Darcy term, where α is a function of the pseudo-density field $\rho \in [0,1]$, and varies between 0 (void) and a maximum value $\alpha_{\max}$ (impermeable):

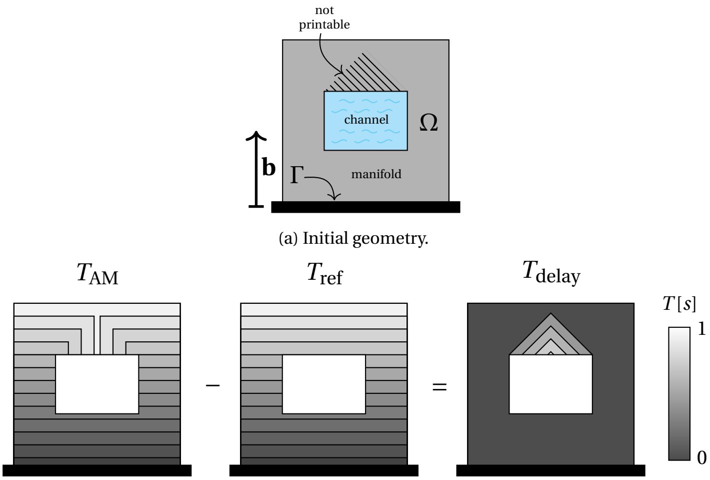
$$\alpha = \alpha_{\max}\rho. \tag{4.2}$$

The value of $\alpha_{\max}$ is dependent on the fluid and domain properties, and can be determined using Darcy's law (Olesen *et al.*, 2006b). By prescribing the inflow velocity and outlet pressure gradient, the fluid flow can be solved for any given manifold layout using Equation (4.1).

In order to optimize a manifold for minimum power loss, the following minimization problem is solved:

$$\begin{aligned} \min_{\rho\in[0,1]} \quad & -\int_\Gamma \left(p + \frac{1}{2}v^2\right)\mathbf{v}\cdot\mathbf{n}\mathrm{d}\Gamma \\ \text{s.t.} \quad & \rho_0 (\mathbf{v}\cdot\nabla)\,\mathbf{v} = -\nabla p + \mu\Delta\mathbf{v} - \alpha_{\max}\rho\mathbf{u}, \\ & \nabla\cdot\mathbf{v} = 0, \\ & \int_\Omega 1 - \rho\mathrm{d}\Omega \le V_{\mathrm{lim}}, \end{aligned} \tag{4.3}$$

where the power loss is calculated as the net inward flux of energy (Othmer, 2008), and V_{lim} is the maximum allowable volume of the channel. The optimization problem is



(a) Initial geometry.

(b) Procedure of OH detection. Regions where $T_{\text{delay}} = 0$ are printable.

Figure 4.2: Schematic overview of the front propagation-based overhang filter.

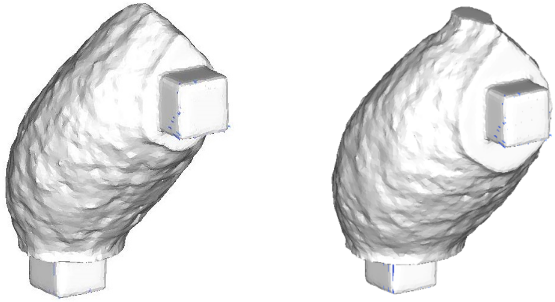
solved using gradient-based optimization algorithms, which require sensitivity information of the objective function with respect to the design variables. The sensitivities are obtained in a continuous manner by solving the adjoint RANS equations. For the optimization algorithm, a so-called one-shot approach is used, where the optimization updates the design before the SIMPLE-algorithm (Moukalled *et al.*, 2016) that is used to solve the RANS equations is converged.

The fluid TO is first demonstrated on a pipe-bend example. In this example, the flow enters the domain from the inlet, and has to make a 90° turn to reach to the outlet, as is shown in Figure 4.1a. The optimal design is shown in Figure 4.1b, where a channel is formed from inlet to outlet. From here on, we will visualize the fluid domain instead of the optimized manifold, as shown in Figure 4.1c, for ease of interpretation when the 3D-printing constraint is introduced.

Although the optimized channel shown in Figure 4.1c has a simple geometry, it cannot be printed without the addition of support structures inside the channel, which are difficult to remove afterwards. Therefore, in order to avoid using support structures, a post-processing step can be performed on the optimized domain to make the channel printable. For example, the shape of the channel can be changed from round to droplet-like, as shown in Verboom (2017) and Thomas (2009). For simple geometries this is a feasible approach. However, for more complex geometries and systems with multiple adjacent channels this might be impossible. Furthermore, the optimality will be compromised to an unknown degree. In order to obtain an optimal and printable design, the overhang limitation has to be included into the TO.

4.1.2. Front propagation-based 3D printing constraint

In order to obtain an self-supporting design, the front propagation-based overhang filter presented in Chapter 2 and Chapter 3 is included into the TO loop. This filter results in a



(a) Optimal self-supporting design.

(b) self-supporting design by post-processing.

Figure 4.3: Topology optimization of a pipe bend with overhang filter.

density field ρ_p from which areas that cannot be printed without supports are removed. A brief recap of the overhang filter is given here.

Consider the domain Ω, which represents a manifold with an internal channel, as shown in Figure 4.2a. It is to be printed in the given build direction $\mathbf{b}$, on the base plate Γ. In order to detect regions that are not printable, a front propagation is performed. The front is initialized on the base plate Γ, and propagated throughout the domain Ω. The propagation speed is reduced when the propagation direction is below the allowable overhang angle. This for instance happens right above the channel, as can be seen in the arrival time field T_{AM} in Figure 4.2b. The arrival time field T_{AM} is compared to a reference field T_{ref}, where the arrival time of a point is its distance to the base plate Γ. The difference $T_{\text{AM}} - T_{\text{ref}}$ depicted in Figure 4.2b reveals where the front propagation was delayed, and thus where the overhang limitation is violated. The field $T_{\text{delay}} = T_{\text{AM}} - T_{\text{ref}}$ is used to construct the printable densities ρ_p, by:

$$\rho_p(\mathbf{x}) = 2^{-\beta T_{\text{delay}}(\mathbf{x})}, \tag{4.4}$$

where β is a parameter that influences the length of transitions from material to void regions in the design (see Equation (2.22) for details). With the relation given in Equation (4.4), regions that are not printable are removed: wherever $T_{\text{delay}} = 0$ (printable), $\rho_p = 1$ (material), and for $T_{\text{delay}} \gg 0$ (not printable), $\rho_p \approx 0$ (void).

4.1.3. FLUID TO WITH 3D PRINTING CONSTRAINT

It remains to incorporate the above described overhang filter to the TO scheme and optimize the same pipe-bend case (Figure 4.1) as previously optimized without overhang filter. As can be seen in Figure 4.3a, the shape of the thereby obtained channel is printable, reminiscent of a droplet shape. If overhanging parts are removed as a post-processing step of the result shown in Figure 4.1c, one arrives at the design shown in Figure 4.3b. This is not a satisfactory result, as the volume of the channel increases, and the channel touches the boundary of the design domain at the top. These problems are avoided when the overhang filter is active during the optimization. Consequently, the volume constraint remains satisfied, and the solution is optimal. The convergence graph for this

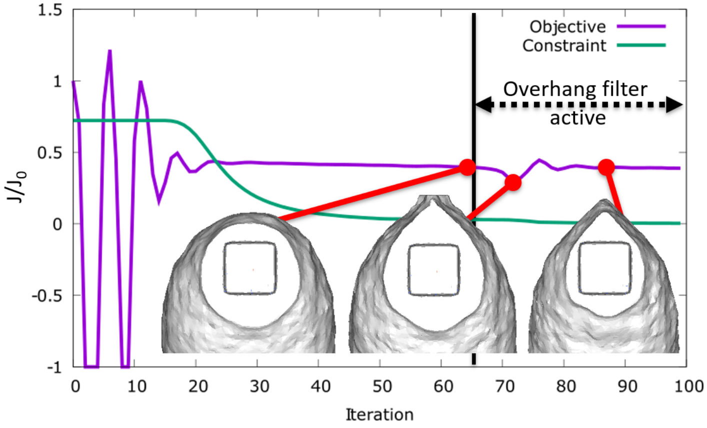


Figure 4.4: Convergence graph for the pipe band case. The overhang constraint is activated after 65 iterations.

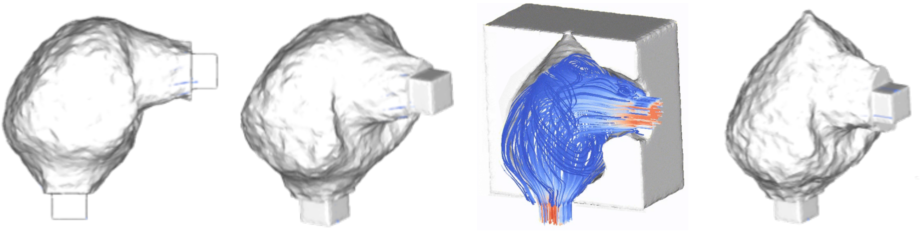
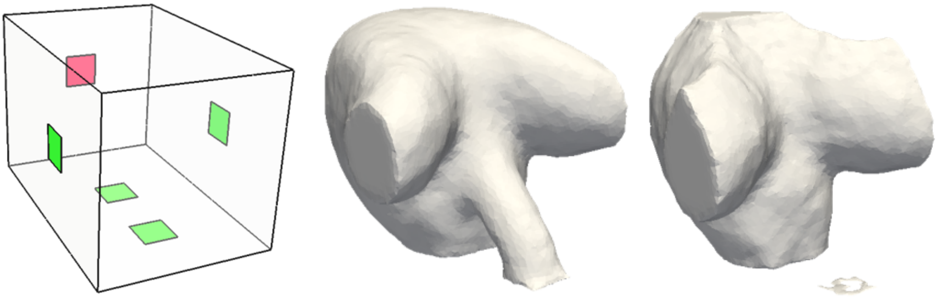


Figure 4.5: Topology optimization of a pipe bend without (a-b) and with (b-c) overhang filter for higher Reynolds numbers.



(a) Design domain with inlet (red) and outlets (green).

(b) Optimized design using all outlets.

(c) Optimized overhang-free design, using 3 out of 4 outlets.

Figure 4.6: Topology optimization of a simple manifold.

optimization is given in Figure 4.4. First of all, it is clearly visible by the large oscillations of the objective in the first 20 iterations that the design is updated before convergence of the Navier-Stokes equations. In order to avoid pronounced non-linearity in the first iterations, the overhang constraint is activated after 65 iterations. It can be seen that as the overhang filter is activated first the channel size increases, but after a few iterations the channel size is reduced while maintaining printability.

The pipe-bend case is also optimized for a more turbulent flow case, as shown in Figure 4.5. With increasing the Reynolds number, due to the more turbulent nature of the flow, the channel with the lowest power loss is no longer the shortest channel from inlet to outlet, but a more intricate structure that guides the flow towards the outlet, as shown in Figure 4.5b. A large part of the top surface of this channel is not printable without supports. With overhang constraint, a new design is generated, as shown in Figure 4.6b, with a conic shape in the top-middle section and teardrop shape channel at the inlet to provide printability. If the channel would instead have been post-processed to ensure printability, it would have required a much larger design space, and likely lose much of its optimality.

Another insightful example where the change in design is more prominent is given in Figure 4.6. Again the power dissipation is minimized for the manifold, but now there are four outlets. Without overhang filter, the inlet is connected to all the outlets as can be seen in Figure 4.6b. However, in the presence of the overhang filter, the outlet that is the furthest from the inlet is disconnected. With the limited volume that is available and due to the extra volume required for the droplet-like shaped channels, it is apparently more beneficial to connect the inlet to the three closest outlets. Such a change in design would be extremely difficult to predict and perform in a post-processing setting, but happens naturally when the overhang limitation is included during the optimization phase.

4.2. COMPLIANT MECHANISM DESIGN

Another field of application that can benefit from the combination of TO and AM is compliant mechanism design. Both techniques are already being used in compliant mechanism design and fabrication, and an overhang constraint will only strengthen the applicability of TO as a design tool for compliant mechanisms.

4.2.1. PROBLEM DEFINITION

The objective of the case on which the overhang constraint is demonstrated is a mechanism to transform a vertical force into a horizontal motion, in order to generate a gripping motion inside a cylindrical domain. The domain and boundary conditions for this case are displayed in Figure 4.7. A gripper resulting from this case could for example be used to pick and manipulate small objects in production equipment, or for instruments used in minimal invasive surgery.

During the optimization, the normal displacement of the gripping surfaces due to a load $\mathbf{f}_1$ applied on the bottom surface (depicted by the red arrow) is maximized. A relatively small penalty on the volume is added to the objective, in order to remove non-functional material from the final topology. Because compliant mechanisms tend to form weak links between structural members to mimic pin joints, a constraint that lim-

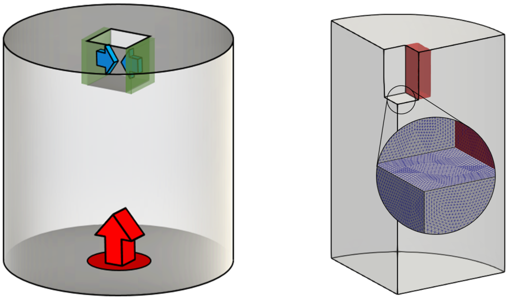


Figure 4.7: Compliant mechanism test case. A force is applied on the bottom with the objective to move the two green surfaces in the top cavity inwards, to provide for a gripping motion. The side of the cylinder is mechanically clamped. One quarter of the domain is simulated, as displayed on the right. Furthermore, A detail of the numerical mesh is displayed in the right image.

its the maximum compliance of the mechanism is included. For this constraint, the load $\mathbf{f_2}$ is a combination of the original load $\mathbf{f_1}$, and a normal force on the gripper surface indicted by the blue arrows. As such, the constraint guarantees a minimum stiffness between the gripper surfaces, the loading surface and the boundary of the domain. This leads to the following minimization problem

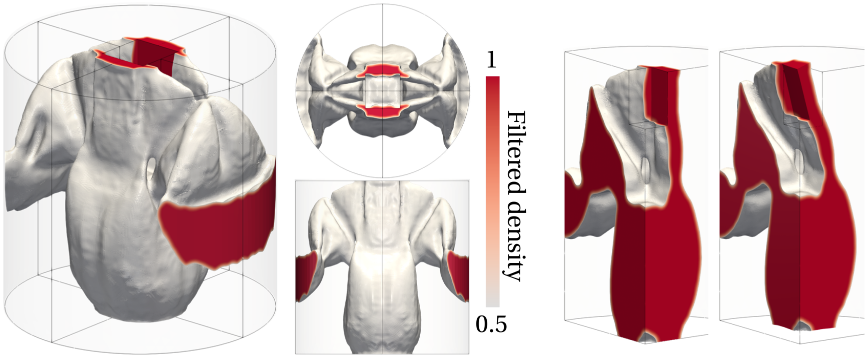
$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & \frac{1}{f_0}\int_{\Gamma_1} -\mathbf{u}_1\cdot\mathbf{n}\mathrm{d}A + \frac{\alpha}{V_{\mathrm{tot}}}\int_{\Omega}\rho\mathrm{d}V, \\ \text{s.t.} \quad & \mathbf{K}\mathbf{u}_1 = \mathbf{f}_1, \\ & \mathbf{K}\mathbf{u}_2 = \mathbf{f}_2, \\ & \int_{\Omega}\mathbf{f}_2\cdot\mathbf{u}_2\mathrm{d}V \le c_{\max}, \\ & \mathbf{0}\le\boldsymbol{\rho}\le\mathbf{1}, \end{aligned} \tag{4.5}$$

where f_0 is a normalization factor, V_{tot} the total volume, Γ_1 the gripper surface, and penalty factor α is chosen equal to 1.

Elements with $\rho = 1$ are given a Young's modulus of $\mathrm{E} = 1\mathrm{GPa}$, and for void elements this is $\mathrm{E} = 10^{-4}\mathrm{GPa}$. Furthermore, a density filter is used with a radius of $5h$, where h is a typical element size. The Method of Moving Asymptotes (Svanberg, 1987) is used to optimize the topology, and the FEM problem is solved in parallel using PETSc (Balay *et al.*, 2021). Due to the symmetry of the case, only a quarter of the cylinder is simulated, as shown in Figure 4.7. The domain is meshed with roughly 9×10^6 tetrahedral elements containing 2×10^6 vertices.

4.2.2. OPTIMAL DESIGN WITH AND WITHOUT OVERHANG FILTER

In the absence of the overhang filter, the mechanism displayed in Figure 4.8a is obtained. The working principle of the mechanism is visualized in Figure 4.9. The gripping surface is attached to the domain boundary by two rigid bodies connected with leaf spring like structures. When the gripping surface is pushed upwards by the actuator, this leaf spring



(a) Optimized gripper topology without overhang filter. The color represents the filtered density values, and the isosurface for $\rho = 0.5$ is given. A similar color scale and isosurface is used for the remaining images.

(b) Quarter section of the topology undeformed (left) and deformed (right).

Figure 4.8: Optimized compliant mechanism design without overhang constraint.

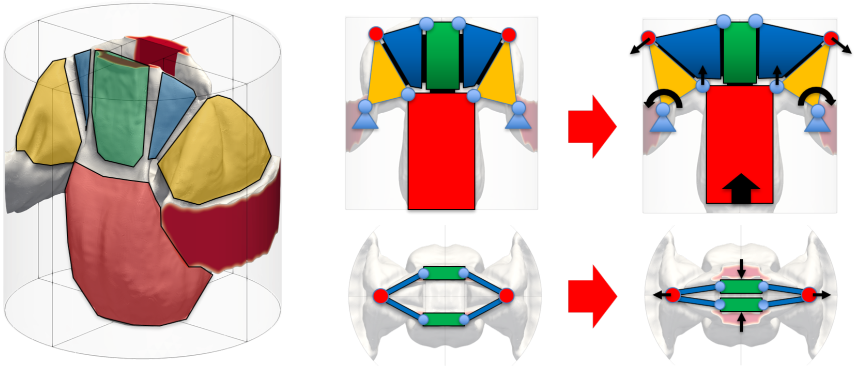


Figure 4.9: Working principle of the compliant mechanism. The mechanism can be seen as a combination of rigid bodies (each given a different color) and flexures between them. When the red bottom part is pushed upwards, the yellow members rotate outwards, closing the green gripper at the top.

structure pulls the gripper inwards, providing a gripping motion, which is also shown in Figure 4.8b.

Unfortunately, this topology is not printable without a substantial amount of support structures, as shown in Figure 4.10. The supports are costly to manufacture, and also difficult to remove as some of the supports are in small cavities. Therefore, the same case was optimized with the overhang filter proposed in Chapter 3 with $\alpha_{\text{oh}} = 45^\circ$. Similar to Chapter 3, the overhang filter is gradually introduced from iteration 10 to iteration 20, by interpolating between the filtered density field and the printable density field. The only parameter that is changed compared to the cases from Chapter 3 is related to the mesh size. The propagation of the front through the domain is visualized in Figure 4.10, and the resulting design is displayed in Figure 4.11a. In order to prevent supports underneath the attachment to the boundary, a slightly different structure is developed, and a connecting beam is added to make this structure printable without supporting struc-

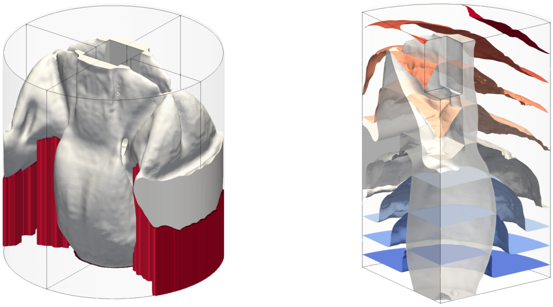
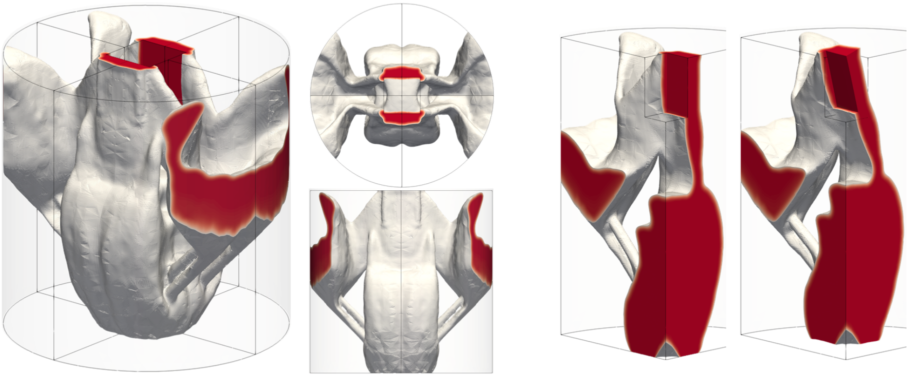


Figure 4.10: Left: the required supports to print the optimized topology in red. Right: front propagation through the optimized topology. In the printable regions of the topology, the front stays level with the base plate.



(a) Optimized gripper topology including overhang filter. The topology is printable in the displayed configuration without the need for supports.

(b) Quarter section of the printable topology undeformed (left), and deformed (right).

Figure 4.11: Optimized compliant mechanism design with overhang constraint.

tures. The resulting deformation when applying $\mathbf{f}_1$ is shown in Figure 4.11b. The overall objective decreases by 15% compared to the non-overhang-constrained TO. Generally, the addition of the overhang filter reduces the performance, dependent on how close to printable the original design is.

4.3. Conclusion

In this chapter it is demonstrated that the overhang constraint is applicable to different optimization problems besides compliance minimization. For the fluid flow optimization, the overhang constraint was implemented into a completely different framework in OpenFOAM, while for the compliant mechanism the PETSc framework as used in Chapter 3 has been used with multiple load cases. In both cases, the overhang constraint was able to provide designs which could be readily printed.

References

Aage N, Poulsen TH, Gersborg-Hansen A, Sigmund O (2008) Topology optimization of large scale stokes flow problems. Structural and Multidisciplinary Optimization 35(2):175–180

Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Dener A, Eijkhout V, Gropp WD, Karpeyev D, Kaushik D, Knepley MG, May DA, McInnes LC, Mills RT, Munson T, Rupp K, Sanan P, Smith BF, Zampini S, Zhang H, Zhang H (2021) PETSc Web page. https://www.mcs.anl.gov/petsc Accessed 9 May 2021

Borrvall T, Petersson J (2003) Topology optimization of fluids in Stokes flow. International Journal for Numerical Methods in Fluids 41(1):77–107

Deng Y, Liu Z, Zhang P, Liu Y, Wu Y (2011) Topology optimization of unsteady incompressible Navier–Stokes flows. Journal of Computational Physics 230(17):6688–6708

Dilgen CB, Dilgen SB, Fuhrman DR, Sigmund O, Lazarov BS (2018) Topology optimization of turbulent flows. Computer Methods in Applied Mechanics and Engineering 331:363–393

Evgrafov A (2005) The Limits of Porous Materials in the Topology Optimization of Stokes Flows. Applied Mathematics and Optimization 52(3):263–277

Gersborg-Hansen A, Sigmund O, Haber R (2005) Topology optimization of channel flow problems. Structural and Multidisciplinary Optimization 30(3):181–192

Guest JK, Prévost JH (2006) Topology optimization of creeping fluid flows using a Darcy–Stokes finite element. International Journal for Numerical Methods in Engineering 66(3):461–484

Kontoleontos EA, Papoutsis-Kiachagias EM, Zymaris AS, Papadimitriou DI, Giannakoglou KC (2013) Adjoint-based constrained topology optimization for viscous flows, including heat transfer. Engineering Optimization 45(8):941–961

Kreissl S, Pingen G, Maute K (2011) Topology optimization for unsteady flow. International Journal for Numerical Methods in Engineering 87(13):1229–1253

Moukalled F, Mangani L, Darwish M (2016) The finite volume method in computational fluid dynamics. Fluid mechanics and its applications, Springer, Cham, oCLC: 921234552

Olesen LH, Okkels F, Bruus H (2006a) A high-level programming-language implementation of topology optimization applied to steady-state Navier-Stokes flow. International Journal for Numerical Methods in Engineering 65(7):975–1001

Olesen LH, Okkels F, Bruus H (2006b) A high-level programming-language implementation of topology optimization applied to steady-state Navier-Stokes flow. International Journal for Numerical Methods in Engineering 65(7):975–1001

OpenCFD (2019) OpenFOAM® - Official home of The Open Source Computational Fluid Dynamics (CFD) Toolbox. Accessed: 2019-09-25

Othmer C (2008) A continuous adjoint formulation for the computation of topological and surface sensitivities of ducted flows. International Journal for Numerical Methods in Fluids 58(8):861–877

Othmer C, de Villiers E, Weller H (2007) Implementation of a Continuous Adjoint for Topology Optimization of Ducted Flows. In: 18th AIAA Computational Fluid Dynamics Conference, American Institute of Aeronautics and Astronautics, Miami, Florida

Pingen G, Evgrafov A, Maute K (2007) Topology optimization of flow domains using the lattice Boltzmann method. Structural and Multidisciplinary Optimization 34(6):507–524

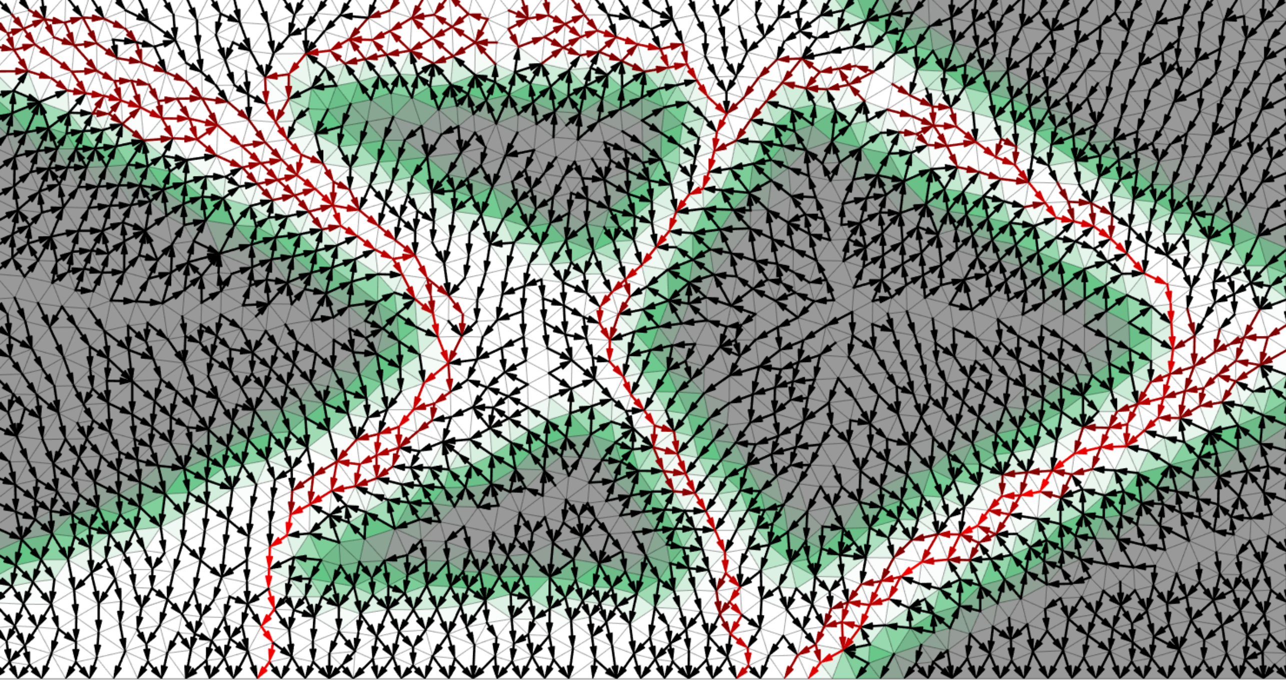
Svanberg K (1987) The method of moving asymptotes – a new method for structural optimization. International journal for numerical methods in engineering 24(2):359–373

Thomas D (2009) The Development of Design Rules for Selective Laser Melting. PhD thesis, University of Wales

van de Ven E, Ayas C, Langelaar M, Maas R, van Keulen F (2018) Topology optimization for additive manufacutring: fully printable compliant mechanisms. In: Advancing Precision in Additive Manufacturing, Berkeley, CA, pp 40–44

Verboom JM (2017) Design and Additive Manufacturing of Manifolds for Navier-Stokes Flow: A Topology Optimisation Approach. Master's thesis, TU Delft, Delft

Zymaris A, Papadimitriou D, Giannakoglou K, Othmer C (2009) Continuous adjoint approach to the Spalart–Allmaras turbulence model for incompressible flows. Computers & Fluids 38(8):1528–1538



5

Comparison of continuous front propagation-based and discrete layer-by-layer overhang control

Although additive manufacturing (AM) allows for a large design freedom, there are some manufacturing limitations that have to be taken into consideration. One of the most restricting design rules is the minimum allowable overhang angle. To make topology optimization suitable for AM, several algorithms have been published to enforce a minimum overhang angle. In this work, the layer-by-layer overhang filter proposed by Langelaar (2017), and the continuous, front propagation-based, overhang filter proposed by van de Ven et al. *(2018b) are compared in detail. First, it is shown that the discrete layer-by-layer filter can be formulated in a continuous setting using front propagation. Then a comparison is made in which the advantages and disadvantages of both methods are highlighted. Finally, the continuous overhang filter is improved by incorporating complementary aspects of the layer-by-layer filter: continuation of the overhang filter and a parameter that had to be user-defined are no longer required. An implementation of the improved continuous overhang filter is provided.*

This chapter is published in *Structural and Multidisciplinary Optimization* 64, 761-778 (2021).

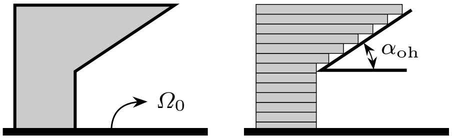


Figure 5.1: In order to print a given structure (left), it is discretized into layers (right). The overhang angle α_{oh} is defined as the angle between a down-facing surface and the base plate Ω_0.

5.1. Introduction

Additive manufacturing (AM) is widely recognized for its capability to manufacture complex components. As the resulting designs of topology optimization (TO) are frequently geometrically complex, the combination of both methods has received significant interest. This interest is even further increased with the advent of metal AM, which opens the possibility to realize optimal functional components with high strength and toughness.

One of the most active research topics on combining TO and AM, is the minimization of support structures required during manufacturing. Since AM is a layer-by-layer manufacturing process, in the majority of industrially relevant processes each layer requires a certain amount of support from the previously constructed layers, for a variety of reasons. For example, in fused deposition modeling, the support is mainly required because a new layer cannot be printed on air, and needs to be at least partly supported by the previous layer (Jiang *et al.*, 2018). For metal powder bed fusion methods, each layer is built on either the already existing part or metal powder. Heat accumulation can be a major issue as the powder has low conductivity (Sih and Barlow, 2004), obstructing the path for heat towards the base-plate which acts as a heat sink. Here supports provide heat conduction as well as mechanical support to, among others, prevent distortions (Cloots *et al.*, 2013; Mercelis and Kruth, 2006; Wang *et al.*, 2013).

Since the production and removal of supports subsequent to printing is costly and time consuming, design rules have been set up to prevent the need for supports (Adam and Zimmer, 2014; Kranz *et al.*, 2015; Thomas, 2009). For most AM processes, a critical overhang angle α_{oh} has been defined. The overhang angle is the angle between a down facing surface and the base plate, as shown in Fig. 5.1. Surfaces with an overhang angle lower than α_{oh} are termed *overhanging*, and require supports. In order to attain support free optimized structures, numerous studies have proposed various methodologies to incorporate the minimum overhang angle as a design rule in TO. Based on the principle of overhang detection, these can be classified into three categories: overhang detection by (i) a processing of the geometry in printing sequence (e.g. Gaynor and Guest, 2016; Langelaar, 2016; van de Ven *et al.*, 2018b); (ii) inspection of the boundary orientation (e.g. Allaire *et al.*, 2017; Guo *et al.*, 2017; Qian, 2017; Zhang *et al.*, 2019); and (iii) simulating the (simplified) physics of the printing process (e.g. Allaire *et al.*, 2017; Amir and Mass, 2018; Ranjan *et al.*, 2018). For a comprehensive overview, the reader is referred to Liu *et al.* (2018).

The focus of this study is on the methods in the first category: the filters that follow the printing sequence. Most methods in this category are presented as discrete filters, defined on a discretized geometry. However, the front propagation-based filter presented in van de Ven *et al.* (2018b) is continuous in nature. This study investigates the

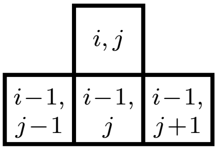


Figure 5.2: Element i, j can be supported by the three elements directly below it.

differences and similarities between the discrete, layer-by-layer methods presented in Langelaar (2016, 2017), and the continuous, front propagation method presented in van de Ven *et al.* (2018b). Due to similarities in implementation, it was suspected that the discrete layer-by-layer filter could be formulated using front propagation. In this paper it is shown that this is indeed the case. From this, it is concluded that the layer-by-layer filter cannot be used on unstructured grids as is. Furthermore, the front propagation-based filter is improved by using aspects of the layer-by-layer filter, which is enabled by the front propagation-based formulation of the layer-by-layer filter.

This paper is structured as follows. First, a brief overview of both methods is given in Section 5.2. Then, it is shown that the discrete method can also be formulated with front propagation discretized on a structured grid (Section 5.3). Next, the differences between both filters are examined (Section 5.4), and in Section 5.5 the continuous overhang filter is improved by using aspects of the layer-by-layer filter. Finally, the paper is concluded in Section 5.6. For clarity this paper focuses on 2D formulations, but the principles apply to the 3D case as well.

5.2. Overhang detection methods

For the sake of completeness, the overhang detection methods described in Langelaar (2017) and van de Ven *et al.* (2018b) are summarized in this section. In the remainder of this work, the methods described in Langelaar (2017) and van de Ven *et al.* (2018b) will be referred to as the layer-by-layer overhang filter, and the continuous overhang filter, respectively. Both methods are designed for density-based topology optimization, where the geometry is defined by a pseudo-density field ρ where $0 \leq \rho \leq 1$. Void regions are indicated with $\rho = 0$, and regions of material with $\rho = 1$ (Bendsøe and Sigmund, 2004). The overhang filter is typically applied directly after the density filter, to convert the density field ρ into a printable density field ξ. The overhang filter removes the overhanging regions: the geometry described by the printable density field ξ is thus directly printable without support structures. This field is then used evaluate the objective and constraints of the topology optimization. Note that the overhang filter can also be implemented as a constraint instead of a filter as demonstrated in van de Ven *et al.* (2018a), which gives the possibility to relax the overhang constraint and balance printability with the impact on performance.

5.2.1. The layer-by-layer overhang filter

The layer-by-layer overhang filter described in Langelaar (2017) is defined on a structured rectangular grid. In a 2D setting, any element of a structured mesh can be identified by its row and column i, j. The filter can be summarized in two statements: for an element $e_{i,j}$, its supporting elements are the three elements $e_{i-1,j-1}$, $e_{i-1,j}$ and $e_{i-1,j+1}$ directly below it (see Fig. 5.2), and, the element's printable density $\xi_{i,j}$ cannot exceed the maximum printable density ξ of the elements that support $e_{i,j}$, i.e.

$$\Xi_{i,j} = \max(\xi_{i-1,j-1}, \xi_{i-1,j}, \xi_{i-1,j+1}), \tag{5.1}$$

$$\xi_{i,j} = \min(\Xi_{i,j}, \rho_{i,j}), \tag{5.2}$$

where the printing direction is assumed to be from the bottom to the top row. The filter can be propagated through a domain by initializing the bottom row of elements with $\xi_{1,j} = \rho_{1,j}$, and then evaluating Eqs. (5.1) and (5.2) layer-by-layer in the printing direction. Evaluation of sensitivities is performed in the reverse order. For continuous differentiability and the use of gradient-based optimization, the nonsmooth min and max operators are replaced by differentiable counterparts.

Note that the overhang angle is inherently linked to the grid. For instance, a square element implies $\alpha_{\text{oh}} = \pi/4$. When a suitable structured finite element mesh is used, this filter can thus be applied directly to the elemental densities. In other cases, a mapping between FE mesh and a separate overhang grid must be used (Langelaar, 2018). Alternatively, Hoffarth *et al.* introduce a search cone to define support relations on arbitrary meshes, instead of a mapping (Hoffarth *et al.*, 2017).

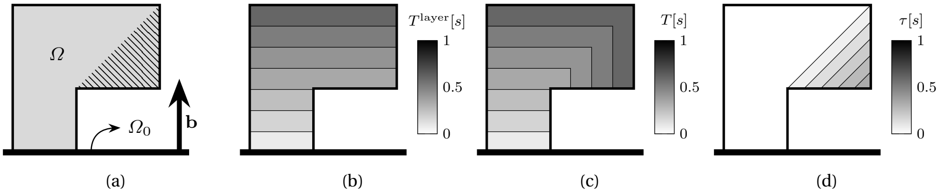
5.2.2. Continuous overhang filter

The overhang filter presented in van de Ven *et al.* (2018b) utilizes front propagation to detect overhanging regions. In a front propagation scheme, a front is initialized at a boundary, and propagated throughout the domain. The output is an arrival time field $T(\mathbf{x})$, which indicates for each location the time at which the front reaches that location.

First, the concept of detecting overhang with front propagation is demonstrated on a given geometry, ignoring the dependence on the density field ρ. In order to detect overhang in a given geometry, such as the one shown in Fig. 5.3a, two arrival time fields are required. In the first arrival time field, T^{layer}, the arrival times are proportional to the height from the base plate Ω_0, and isosurfaces thus represent the printing sequence, as displayed in Fig. 5.3b. Assuming that the base plate coincides with the origin, it is given as

$$T^{\text{layer}}(\mathbf{x}) = \mathbf{b} \cdot \mathbf{x} / f_0, \tag{5.3}$$

where $\mathbf{b}$ is a unit vector defining the build direction, and f_0 represents the default propagation speed in this build direction. The exact value of f_0 is not important as it is a



(a) (b) (c) (d)

Figure 5.3: The different steps required to detect overhang in the geometry given in Fig. 5.3a with build direction $\mathbf{b}$, using front propagation.

scale factor whose effect is canceled, and is chosen as $1\,\mathrm{m\,s^{-1}}$. For the second arrival time field, $T(\mathbf{x})$, a front propagation is performed in the design domain Ω, starting from the base plate Ω_0. This time, the propagation speed is chosen such that the arrival times are equal to $T^{\text{layer}}(\mathbf{x})$, except when the front travels in a direction *below* the minimum overhang angle α_{oh}, i.e. when a region is not printable. This is achieved using a an anisotropic speed function, and the resulting arrival time field can be seen in Fig. 5.3c. Consequently, a delay field $\tau(\mathbf{x})$ between the two arrival time fields can be calculated as

$$\tau(\mathbf{x}) = T(\mathbf{x}) - T^{\text{layer}}(\mathbf{x}). \tag{5.4}$$

In regions where $\tau(\mathbf{x}) > 0$, the propagation has been delayed, indicating that the structure is overhanging, as can be seen in Fig. 5.3d. Finally, the printable density field $\xi(\mathbf{x})$ is attained from the delay field $\tau(\mathbf{x})$ with the delay-density relation h:

$$\xi(\mathbf{x}) = h(\tau(\mathbf{x})), \tag{5.5}$$

where h is a monotonically decreasing differentiable function with $h(0) = 1$, such that zero delay results in full density, and $0 \leq h(\tau(\mathbf{x})) < 1$ for $\tau(\mathbf{x}) > 0$. In van de Ven *et al.* (2018b), h is defined as

$$h(\tau(\mathbf{x})) = 2^{-k\tau(\mathbf{x})}, \tag{5.6}$$

where k is a numerical parameter related to the element size: a high value for k will result in a short transition from full density to void as small delays will already result in low values for ξ. Therefore, for fine meshes, k is set to a higher value than for coarse meshes.

The first arrival time field, $T^{\text{layer}}(\mathbf{x})$, can be obtained without performing an actual front propagation, as the arrival times are equal to the distance between the base plate Ω_0 and the point of interest $\mathbf{x}$. However, in order to obtain the second arrival time field $T(\mathbf{x})$, a front propagation is performed, which is governed by the Hamilton-Jacobi-Bellman equation:

$$\min_{\mathbf{a}\in S_1}\left\{(\nabla T(\mathbf{x})\cdot\mathbf{a})\, f_s(\rho(\mathbf{x}),\mathbf{a},\alpha_{\text{oh}})\right\} = 1, \quad \mathbf{x}\in\Omega, \tag{5.7}$$

where f_s anisotropic is the speed function that dictates the propagation speed, and $\mathbf{a}$ is a unit vector determining the direction of propagation: $\mathbf{a} \in S_1$, $S_1 = \{\mathbf{a} \in \mathbf{R}^2 | \|\mathbf{a}\| = 1\}$. It is in the speed function f_s where the original density field ρ is coupled to the arrival times $T(\mathbf{x})$ and thus printable densities $\xi(\mathbf{x})$. It is later shown that the speed function is simply scaled with the densities. Since all the densities are printable at the base plate Ω_0, it is required that

$$\xi(\mathbf{x}) = \rho(\mathbf{x}), \quad \mathbf{x}\in\Omega_0. \tag{5.8}$$

From this relation and Eq. (5.5), the initial condition for the front propagation can be derived as

$$T(\mathbf{x}) = h^{-1}(\rho(\mathbf{x})), \quad \mathbf{x}\in\Omega_0, \tag{5.9}$$

where h^{-1} is the inverse function of the delay-density relation h, such that $h(h^{-1}(\rho(\mathbf{x}))) = \rho(\mathbf{x})$. For example, elements with $\rho = 1$ at the base plate are initialized at $T = T_{\text{layer}}$.

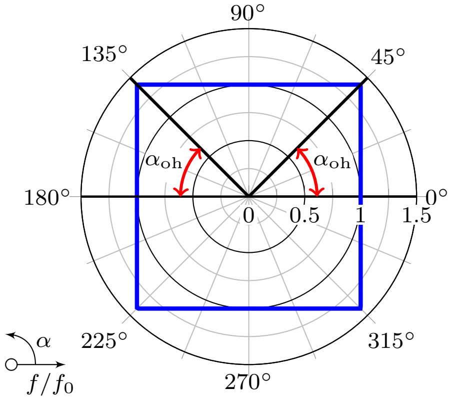


Figure 5.4: Polar plot of the rectangular speed function for $\alpha_{\mathrm{oh}} = 45°$, with propagation speed on the radial axis, and propagation direction on the tangential axis. A propagation direction of $\alpha = 90°$ coincides with the build direction **b**.

THE SPEED FUNCTION

For the sake of simplicity, the anisotropic speed function was scaled linearly with the density value ρ. As such, the speed function f_s can be decomposed into a part that depends on the density field, and a part that depends on the direction of propagation **a** and minimum overhang angle α_{oh}:

$$f_s(\rho(\mathbf{x}), \mathbf{a}, \alpha_{\mathrm{oh}}) = g(\rho(\mathbf{x}))\, f(\mathbf{a}, \alpha_{\mathrm{oh}}). \tag{5.10}$$

The function $f(\mathbf{a}, \alpha_{\mathrm{oh}})$ relates the direction of propagation to speed. The speed function as used in van de Ven *et al.* (2018b) is displayed in Fig. 5.4, where α represents the propagation direction defined by **a**. The top part of the speed function, where $\alpha_{\mathrm{oh}} \leq \alpha \leq 180° - \alpha_{\mathrm{oh}}$, is defined by $f = f_0 / \sin\alpha$. When the propagation direction is equal to the build direction ($\alpha = 90°$), the propagation speed is equal to that of the reference field T^{layer}, i.e. $f = f_0$. Moreover, in order to maintain a front parallel to the base plate, the propagation speed is increased to compensate for the larger distance travelled to the next printing layer. For example, for $\alpha = 45°$, $f = \sqrt{2} f_0$. However, when the propagation direction is below α_{oh}, i.e. when $\alpha < \alpha_{\mathrm{oh}}$ or $\alpha > 180° - \alpha_{\mathrm{oh}}$, the propagation speed is lowered, thereby a non zero value of $\tau(\mathbf{x})$ is attained. The anisotropic speed function capable of creating a delay in the overhanging regions is not unique. In van de Ven *et al.* (2018b) a function resembling a rectangle is used, which is numerically efficient. The aspect ratio of the rectangle shown in Fig. 5.4 designates the minimum overhang angle α_{oh}.

Finally, the speed function f_s also contains a geometry dependence in the speed-density relation $g(\rho(\mathbf{x}))$. In the example shown in Fig. 5.3, the front propagated only through the structure. However, with topology optimization, this structure is determined by the density field ρ. In van de Ven *et al.* (2018b), the propagation speed is linearly scaled with the density:

$$g(\rho(\mathbf{x})) = v_{\mathrm{void}} + (1 - v_{\mathrm{void}})\rho(\mathbf{x}), \tag{5.11}$$

where v_{void} is the minimum propagation speed in the void regions, chosen such that $0 < v_{\text{void}} < 1$. Consequently, the front will be delayed in void regions, leading to $\tau(x) > 0$, and thus also void in the printable density field ξ (Eq. (5.5)).

NUMERICAL IMPLEMENTATION

The layer-by-layer overhang filter by Langelaar (2017) is defined on a discretized domain, hence its numerical implementation is straightforward (see (Langelaar, 2017) for details). Here, the numerical implementation of the continuous filter is briefly described in order to make a comparison between the two.

The front propagation can be efficiently executed using the Ordered Upwind Method (OUM) (Sethian and Vladimirsky, 2003), on any mesh type in 2D as well as 3D. The necessary steps are briefly explained, while some details are omitted for brevity. In the OUM, nodes are labelled either *Far*, *Candidate*, or *Accepted*. All of the nodes are initialized as *Far*, with $T = \infty$. Then, the arrival times of nodes at the fixed boundary Ω_0 are initialized and are labeled *Accepted*. Finally, the nodes within a distance d of these *Accepted* nodes are labeled *Candidate*. Then, the following algorithm is executed:

1. Calculate the arrival times of the *Candidate* nodes.
2. Label the *Candidate* node with the lowest arrival time as *Accepted*, and label the nodes within a distance d of that node, that are not *Accepted*, as *Candidate*.
3. If there are any *Candidate* nodes left, go to Step 1.

The distance d is defined as

$$d = zF_2/F_1, \tag{5.12}$$

where z is a typical element length, and F_1 and F_2 are the maximum and minimum values of $f(\mathbf{a}, \alpha_{\text{oh}})$, respectively. Thus, the more anisotropic the speed function, the larger this distance.

A crucial part of the algorithm is the calculation of the arrival times of *Candidate* nodes. Let us first consider the calculation of the arrival time of node i from a given point $\mathbf{x}_\gamma$. This is simply the distance divided by the speed, plus the arrival time at point $\mathbf{x}_\gamma$, defined as T_γ:

$$T^i_\gamma = \frac{\|\mathbf{x}_i - \mathbf{x}_\gamma\|}{f_s(\rho(\mathbf{x}_i), \mathbf{a}, \alpha_{\text{oh}})} + T_\gamma, \tag{5.13}$$

where $\mathbf{a} = (\mathbf{x}_i - \mathbf{x}_\gamma)/\|\mathbf{x}_i - \mathbf{x}_\gamma\|$, and T^i_γ denotes the arrival time at $\mathbf{x}_i$ when calculated from $\mathbf{x}_\gamma$. In reality, the arrival time of the *Candidate* node i is calculated from, in 2D, two *Accepted* nodes, j and k. The point $\mathbf{x}_\gamma$ is then a point on the segment $\mathbf{x}_j\mathbf{x}_k$, as displayed in Fig. 5.5. The position and arrival time are linearly interpolated between nodes j and k, defined by parameter γ:

$$\mathbf{x}_\gamma = \mathbf{x}_j(1-\gamma) + \mathbf{x}_k\gamma, \tag{5.14}$$

$$T_\gamma = T_j(1-\gamma) + T_k\gamma. \tag{5.15}$$

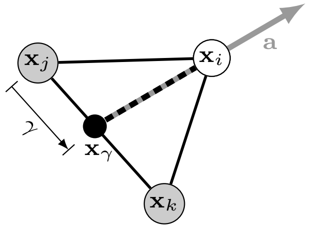


Figure 5.5: Update of a node $\mathbf{x}_i$ from $\mathbf{x}_\gamma$ on the line segment $\mathbf{x}_j\mathbf{x}_k$. The position of $\mathbf{x}_\gamma$ is determined by $\gamma \in [0,1]$. The update direction is given by the unit vector $\mathbf{a}$.

The point from which $\mathbf{x}_i$ is updated, is the point that results in the lowest arrival time:

$$T^i_{jk} = \min_{\gamma\in[0,1]} T^i_\gamma = \min_{\gamma\in[0,1]} \frac{\|\mathbf{x}_i - \mathbf{x}_\gamma\|}{f_s(\rho(\mathbf{x}_i), \mathbf{a}, \alpha_{\text{oh}})} + T_\gamma, \tag{5.16}$$

where T^i_{jk} denotes the arrival time at $\mathbf{x}_i$ when calculated from the segment $\mathbf{x}_j\mathbf{x}_k$. Note that there is a dependency between the speed function f_s and γ through the propagation direction $\mathbf{a}$. The final arrival time at $\mathbf{x}_i$ is the minimum arrival time that can be obtained from all the segments between adjacent *Accepted* nodes within a distance d:

$$T_i = \min_{\mathbf{x}_j\mathbf{x}_k \in \text{NF}(\mathbf{x}_i)} T^i_{jk}, \tag{5.17}$$

where NF($\mathbf{x}_i$) is the set of *Accepted* nodes that are within the distance d of $\mathbf{x}_i$.

In 3D the algorithm is unchanged, except that the propagation is based on tetrahedrons instead of triangles: the arrival time for a node i is calculated from three other nodes instead of two, where the point $\mathbf{x}_\gamma$ is now in the triangle spanned by these three nodes instead of a line segment.

Analogous to the layer-by-layer filter, the sensitivities are evaluated by following the reverse order in which the arrival times are evaluated. Since the order of the front propagation is already known, this reduces to a trivial loop over all the elements, as detailed in van de Ven *et al.* (2018b).

5.3. FORMULATING THE LAYER-BY-LAYER FILTER WITH FRONT PROPAGATION

Because the continuous filter is described in a continuous setting, it can be readily used in unstructured meshes while the maximum overhang angle can be adjusted independent of the mesh by simply modifying the speed function. However, in the previous section it became apparent that there is a considerable difference in complexity between the two methods: the layer-by-layer filter can be described by two equations while the continuous filter is much more involved. This is caused for a large part by the fact that the layer-by-layer filter is a discrete formulation, while the front propagation filter is continuous, which makes it difficult to compare both methods. Therefore, in this section, the layer-by-layer filter is cast into in an equivalent continuous formulation based on front propagation. To this end, the speed function (Eq. (5.10)) and the speed-density relation

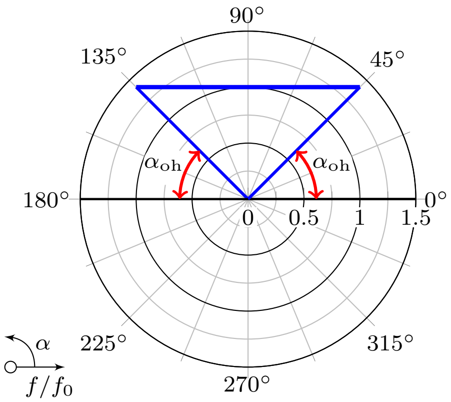


Figure 5.6: Polar plot of the wedge-shaped speed function, with propagation speed on the radial axis, and propagation direction on the tangential axis, for $\alpha_{\mathrm{oh}} = 45°$. A propagation direction of $\alpha = 90°$ coincides with the build direction **b**.

(Eq. (5.11)) have to be changed compared to the continuous filter, which is shown in the following.

5.3.1. SPEED FUNCTION FOR THE CONTINUOUS LAYER-BY-LATER FILTER

In order to represent the layer-by-layer filter with front propagation, first, the speed function is adapted. The speed function utilized in van de Ven *et al.* (2018b) allows propagation in all directions (Fig. 5.4). However, in the layer-by-layer filter, the printable density of an element only depends on either of the three elements directly below it (Eq. (5.1)). Therefore, the wedge-shaped speed function is chosen as displayed in Fig. 5.6 (its 3D equivalent is a cone). With this speed function, the propagation speed below the overhang angle is set to zero, and is given by

$$f(\mathbf{a}, \alpha_{\mathrm{oh}}) = \begin{cases} \dfrac{f_0}{\mathbf{b}\cdot\mathbf{a}} & \mathbf{b}\cdot\mathbf{a} \geq \sin(\alpha_{\mathrm{oh}}) \\ 0 & \mathbf{b}\cdot\mathbf{a} < \sin(\alpha_{\mathrm{oh}}) \end{cases}, \tag{5.18}$$

where, similar to the previous section, **a** and **b** are the propagation and build direction, respectively.

This speed function can be propagated on a structured grid with $\Delta y/\Delta x = \tan(\alpha_{\mathrm{oh}})$, where Δx and Δy are element width and height, and where the build direction coincides with the vertical axis, as displayed in Fig. 5.7. Because the propagation speed in Eq. (5.18) is zero for propagation directions below α_{oh}, a node can only be updated from nodes in a wedge below it, as indicated with the shaded green region in Fig. 5.7 where node i can only be updated from the red and yellow nodes. The same holds for the three red nodes directly below $\mathbf{x}_i$, which can be updated from nodes in the region shaded red. Due to the overlap of the red and green regions, it can be seen that any yellow node that can update node i, can also update at least one of the three red nodes. Since the red nodes are closer to the yellow nodes than node i is, the front will pass the red nodes first. In

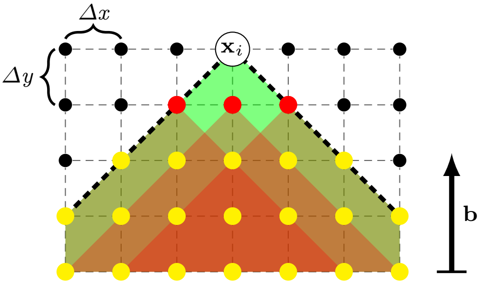


Figure 5.7: The node $\mathbf{x}_i$ can be updated from any node in the green area. The red nodes can be updated from nodes in the red area, covering all the yellow nodes. Due to the overlap, $\mathbf{x}_i$ only has to be updated by the red nodes.

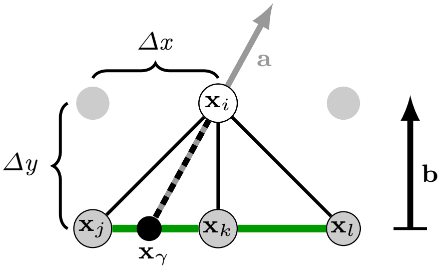


Figure 5.8: Update of a node $\mathbf{x}_i$ on a structured grid can occur only from the segments $\mathbf{x}_j\mathbf{x}_k$ and $\mathbf{x}_k\mathbf{x}_l$.

order to propagate the speed function in Fig. 5.6, it is therefore sufficient to only check for updates from the three nodes directly below a given node.

With only the three nodes directly below the node of interest required for the arrival time calculation, Eq. (5.17) simplifies to:

$$T_i = \min\left\{T^i_{jk}, T^i_{kl}\right\}, \tag{5.19}$$

where node i is the node of interest, and nodes j, k, and l are the three nodes below i, as displayed in Fig. 5.8. This means that node i is updated from a point on either $\mathbf{x}_j\mathbf{x}_k$ or $\mathbf{x}_k\mathbf{x}_l$. Since for any position $\mathbf{x}_\gamma$ on both line segments $\mathbf{b}\cdot\mathbf{a} \geq \sin(\alpha_{\text{oh}})$, the speed function in Eq. (5.18) reduces to $f(\mathbf{a},\alpha_{\text{oh}}) = f_0/(\mathbf{b}\cdot\mathbf{a})$. Substituted into Eq. (5.13) this gives

$$T^i_\gamma = \frac{\mathbf{b}\cdot(\mathbf{x}_i - \mathbf{x}_\gamma)}{g(\rho(\mathbf{x}_i))\, f_0} + T_\gamma. \tag{5.20}$$

Since the build direction $\mathbf{b}$ is a unit vector and $\mathbf{x}_\gamma$ moves perpendicular to $\mathbf{b}$ with changing γ, $\mathbf{b}\cdot(\mathbf{x}_i - \mathbf{x}_\gamma) = \Delta y$ (Fig. 5.8). This implies that the changing distance between $\mathbf{x}_i$ and $\mathbf{x}_\gamma$ has no influence on the final arrival time. Looking at the shape of the speed function in Fig. 5.6, this makes sense. As $\mathbf{x}_\gamma$ moves from $\mathbf{x}_k$ to $\mathbf{x}_j$ in Fig. 5.8, the distance $\mathbf{x}_i\mathbf{x}_\gamma$ increases, but the speed increases proportionally, as the direction of propagation changes.

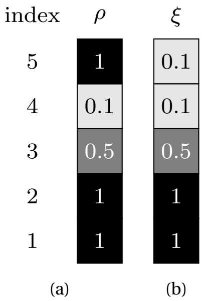


Figure 5.9: Demonstration of the conversion of densities ρ (a) to printable densities ξ (b) by the conventional layer-by-layer filter on a single column of elements.

This reduces Eq. (5.20) to

$$T_\gamma^i = \frac{\Delta y}{g(\rho(\mathbf{x}_i))f_0} + T_\gamma. \tag{5.21}$$

In order to find the point $\mathbf{x}_\gamma$ that results in the lowest arrival time T_γ^i, this equation is differentiated w.r.t. γ:

$$\frac{\partial T_\gamma^i}{\partial \gamma} = T_k - T_j. \tag{5.22}$$

As the right hand side of Eq. (5.22) is a constant term, the minimum lies at the bounds of each segment. This reduces Eq. (5.19) to

$$T_i = \min\left\{T_j^i, T_k^i, T_l^i\right\}, \tag{5.23}$$

where T_X^i represents the arrival time at node i when calculated from node X, as shown in Eq. (5.21).

5.3.2. Delay-density and speed-density relation

The change of speed function was the first step to represent the layer-by-layer filter with a front propagation formulation. It can be seen that Eq. (5.23) already resembles Eq. (5.1), where the arrival time value of a node depends on the three nodes below it. The next step is to choose the speed-density relation (Eq. (5.11)) and the delay-density relation (Eq. (5.5)) such that the behaviour of the layer-by-layer filter is obtained. For this part, we ignore without loss of generality that each element can be supported by three elements, as this is now inherent to the specific speed function chosen, and we will be looking at single columns of elements only, as depicted in Fig. 5.9a.

Let us first examine how the printable densities are related to original densities in the layer-by-layer filter. For a single column of elements, such as the one displayed in Fig. 5.9a, Eq. (5.1) and Eq. (5.2) reduce to

$$\xi_i = \min(\xi_{i-1}, \rho_i), \tag{5.24}$$
$$= \min(\rho_1, \rho_2, \ldots, \rho_i), \tag{5.25}$$

where i is the index of each element, with $i = 1$ being the bottom element. Thus, the printable density of element i is equal to the lowest density encountered in the elements 1 to i. This ensures that an element can only have full printable density if all the elements that support it also have full density. This can be seen in Fig. 5.9b, where the densities given in Fig. 5.9a have been processed according to Eq. (5.25). Here, e.g. the elements 4 and 5 have $\xi = 0.1$, as $\rho_4 = 0.1$ is the lowest density in the set. The density of element 5 is reduced from 1 to 0.1 as it is not properly supported.

In order to describe the layer-by-layer filter with front propagation, the speed-density and delay-density relations need to be chosen such that Eq. (5.24) is obtained. Recall that with the front propagation formulation, the printable density is defined as (Eq. (5.5))

$$\xi(\mathbf{x}) = h(\tau(\mathbf{x})) = h\left(T(\mathbf{x}) - T^{\text{layer}}(\mathbf{x})\right). \tag{5.26}$$

In a discrete single column setting as in Fig. 5.9, and substituting Eq. (5.21) for the arrival time $T(\mathbf{x})$, the printable density for an element i (Eq. (5.26)) reduces to

$$\xi_i = h\left(T_{i-1} + \frac{\Delta y}{g(\rho_i) f_0} - T_i^{\text{layer}}\right). \tag{5.27}$$

The terms between brackets are: the arrival time of the previous element, the distance to the current element divided by the propagation speed, and the arrival time of the reference field. In order to obtain the layer-by-layer filter, the speed-density relation g is chosen such that Eq. (5.24) is obtained from Eq. (5.27):

$$h\left(T_{i-1} + \frac{\Delta y}{g f_0} - T_i^{\text{layer}}\right) = \min(\xi_{i-1}, \rho_i), \tag{5.28}$$

where the arguments of the speed-density relation g are omitted for brevity. The arrival time of the previous element, T_{i-1}, can be substituted by its printable density using Eq. (5.26):

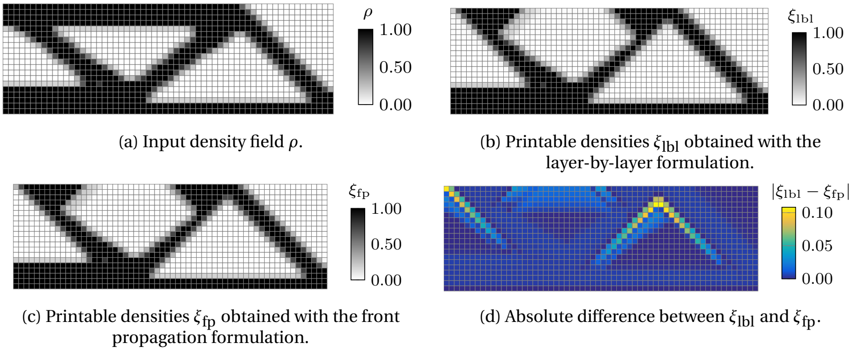
$$h\left(h^{-1}(\xi_{i-1}) + T_{i-1}^{\text{layer}} + \frac{\Delta y}{g f_0} - T_i^{\text{layer}}\right) = \min(\xi_{i-1}, \rho_i). \tag{5.29}$$

Using $T_i^{\text{layer}} - T_{i-1}^{\text{layer}} = \Delta y / f_0$ (Eq. (5.3)), the following speed-density relation is derived:

$$g = \frac{\Delta y}{f_0\left(h^{-1}\left(\min(\xi_{i-1}, \rho_i)\right) - h^{-1}(\xi_{i-1})\right) + \Delta y}. \tag{5.30}$$

This can be interpreted as follows. If $\rho_i > \xi_{i-1}$, the arrival time should be such that $\xi_i = \xi_{i-1}$, or $h(\tau_i) = h(\tau_{i-1})$. In other words, the delay should not increase, which is achieved when the front propagates with the same speed as with which T^{layer} increases over distance, i.e. when $g = 1$. This corresponds to Eq. (5.30), which reduces to 1 when $\rho_i > \xi_{i-1}$.

For $\rho_i < \xi_{i-1}$, the arrival time should ensure that $\xi_i = \rho_i$. Because $\rho_i < \xi_{i-1}$, the delay should increase to lower the printable density ξ_i as compared to ξ_{i-1}. With the speed-density relation given in Eq. (5.30), the speed is lowered exactly such that the new delay will result in $\xi_i = \rho_i$.



(a) Input density field ρ.

(b) Printable densities ξ_{lbl} obtained with the layer-by-layer formulation.

(c) Printable densities ξ_{fp} obtained with the front propagation formulation.

(d) Absolute difference between ξ_{lbl} and ξ_{fp}.

Figure 5.10: Application of the layer-by-layer and front propagation formulation on the density field given in (a). The differences occur only in the intermediate density regions, as different minimum approximation are used (d).

5.3.3. Comparison of front propagation and layer-by-layer formulation

With the speed-density relation in Eq. (5.30) and the speed function given in Eq. (5.18), the layer-by-layer AM filter can be described in a continuous setting with front propagation. For comparison of these formulations, both the layer-by-layer formulation as provided in Langelaar (2017) and its front propagation implementation described in this section are applied to a given density field displayed in Fig. 5.10a. The resulting printable density fields from the layer-by-layer filter and the front propagation implementation are shown in Fig. 5.10b and Fig. 5.10c, respectively. Although they appear visually similar, there are some differences, displayed in Fig. 5.10d. As can be seen, there is hardly a difference in printable regions ($\xi = 1$), but for intermediate density regions the printable density values differ. This is due to the different minimum approximations that are used. For the front propagation formulation, the minimum approximation are chosen as close as possible to the default settings given in Langelaar (2017). When the smooth minimum/maximum approximation are chosen more accurate and less smooth, the error between both implementations vanishes, and the layer-by-layer filter can be exactly reproduced with front propagation.

5.4. Comparison of layer-by-layer and front propagation based overhang filter

In the previous section, it is shown that the layer-by-layer filter by Langelaar (2017) can be cast into a front propagation formulation. The front propagation formulation of the layer-by-layer filter enables a detailed comparison between the layer-by-layer overhang filter and the front propagation-based overhang filter presented in van de Ven *et al.* (2018b). To avoid confusion of both methods, the front propagation implementation of the layer-by-layer filter, as shown in Section 5.3, is referred to as simply the layer-by-layer overhang filter. The original continuous, front propagation based, overhang filter, as introduced in Section 5.2.2, is referred to as the continuous overhang filter.

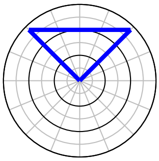
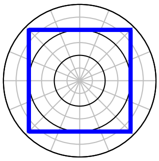
	layer-by-layer filter	continuous filter
$f(\mathbf{a}, \alpha_{\text{oh}})$		
$g(\rho_i, \xi_{i-1})$	$\dfrac{\Delta y}{f_0\left(h^{-1}\left(\min(\xi_{i-1}, \rho_i)\right) - h^{-1}(\xi_{i-1})\right) + \Delta y}$	$v_{\text{void}} + (1 - v_{\text{void}})\rho_i$

Figure 5.11: Differences in speed function and speed-density relation between the layer-by-layer and the continuous overhang filter.

The two elements of the front propagation that differ between both filters are summarized in Fig. 5.11. First, the difference in the speed function $f(\mathbf{a}, \alpha_{\text{oh}})$ is discussed, followed by the differences in speed-density relation. Finally, the numerical implementations of the various minimum/maximum operators and their effect on the sensitivities are analysed.

5.4.1. DIFFERENCES IN SPEED FUNCTION

The speed functions of both implementations share a similar profile for propagation directions above the minimum allowable overhang angle, i.e. when $\mathbf{b} \cdot \mathbf{a} \geq \sin(\alpha_{\text{oh}})$ (Eq. (5.18)). However, for other directions of propagation, the speed functions are different, as can be seen in Fig. 5.11 (both speed functions are displayed in more detail in Fig. 5.4 and Fig. 5.6). The rectangular speed function of the continuous filter allows propagation in all directions, even opposite to the build direction, and is only slightly anisotropic, with an anisotropy ratio, i.e. maximum speed divided by the minimum speed, of $\sqrt{2}$ for $\alpha_{\text{oh}} = 45°$. The triangular speed function of the layer-by-layer filter on the other hand, only allows propagation when $\mathbf{b} \cdot \mathbf{a} \geq \sin(\alpha_{\text{oh}})$ while the propagation speed is zero for other directions. It therefore has an infinite anisotropy ratio.

The layer-by-layer speed function has a close resemblance to the AM process, as it can only propagate in directions that are printable from its current position. It is therefore a more natural choice, and was investigated for the continuous filter as well, before adopting the rectangular speed function for numerical reasons further detailed in the following.

For practical applicability, one of the requirements for the continuous filter was that it should be applicable to unstructured meshes. On unstructured meshes, it is difficult to propagate a front with a highly anisotropic speed function. As mentioned in Section 5.2.2, a crucial part of the front propagation algorithm is the calculation of the arrival time for a node when updated from two other nodes. This update is executed several times per node, and is therefore the dominant factor in computational cost. Reducing the number of updates greatly reduces the computational cost.

The number of updates required per node depends on the distance d from which a node can be updated, which in turn depends on the anisotropy ratio of the speed func-

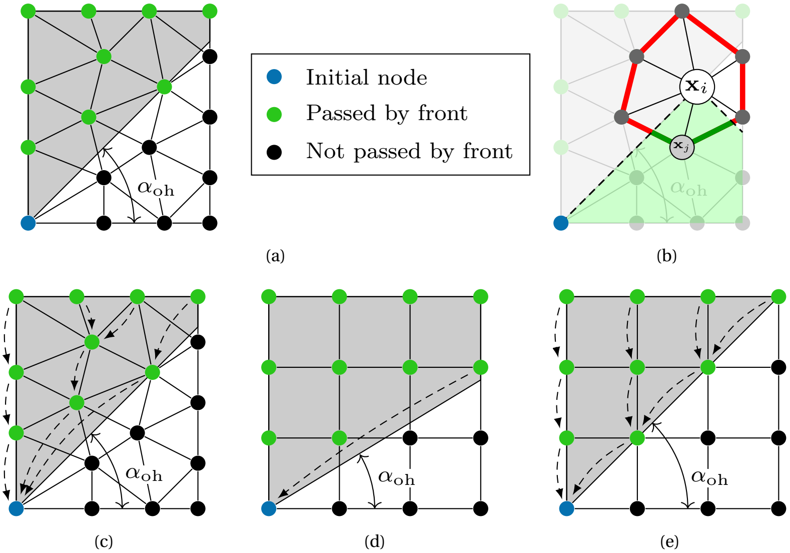


Figure 5.12: Front propagation from a single node (blue) with a speed function as displayed in Fig. 5.6 for unstructured and structured meshes. Nodes in the gray domain are reachable by the front, while black nodes are not.

tion (Eq. (5.12)). Ideally, one would update a node only from its direct neighbours, as is the case for an isotropic speed function. However, for the speed function of the layer-by-layer filter, as the minimum speed equals zero, $d = \infty$, meaning a node can be updated from every other node in the mesh. This is demonstrated on the unstructured mesh given in Fig. 5.12a. Here, a front is propagated from the blue node with the wedge-shaped speed function as given in Fig. 5.6. If the front is propagated correctly, all the nodes in the gray area should be reached by the front. In order to do so, some nodes must be updated from a large distance. For example, due to the nature of the speed function, node $\mathbf{x}_i$ in Fig. 5.12b can only be updated from locations in the transparent green area. The only node that can provide the update is the blue node itself. Consequently, this front can only be propagated correctly when nodes are updated from a much larger distance than only the direct neighbours. This is shown in Fig. 5.12c, where the arrows indicate the closest node from which a node can be updated, such that the direction of the update is above the overhang angle.

For structured meshes, on the other hand, the wedge-shaped speed function can be propagated efficiently. If the overhang angle does not align with the mesh, as in Fig. 5.12d, the situation is similar to unstructured meshes, and updates from a large distance might be required, as illustrated. However, when $\tan(\alpha_{\text{oh}}) = \Delta y/\Delta x$, where dy and dx are the element height and width, a node only has to be updated from its direct neighbours, as displayed in Fig. 5.12e. That is because if a node i can be updated from a far away node j, at least one of the direct neighbours below i can also be updated from node j (Fig. 5.7). For the speed function as given in Eq. (5.18), it is therefore sufficient to only consider the three nodes directly below the node of interest.

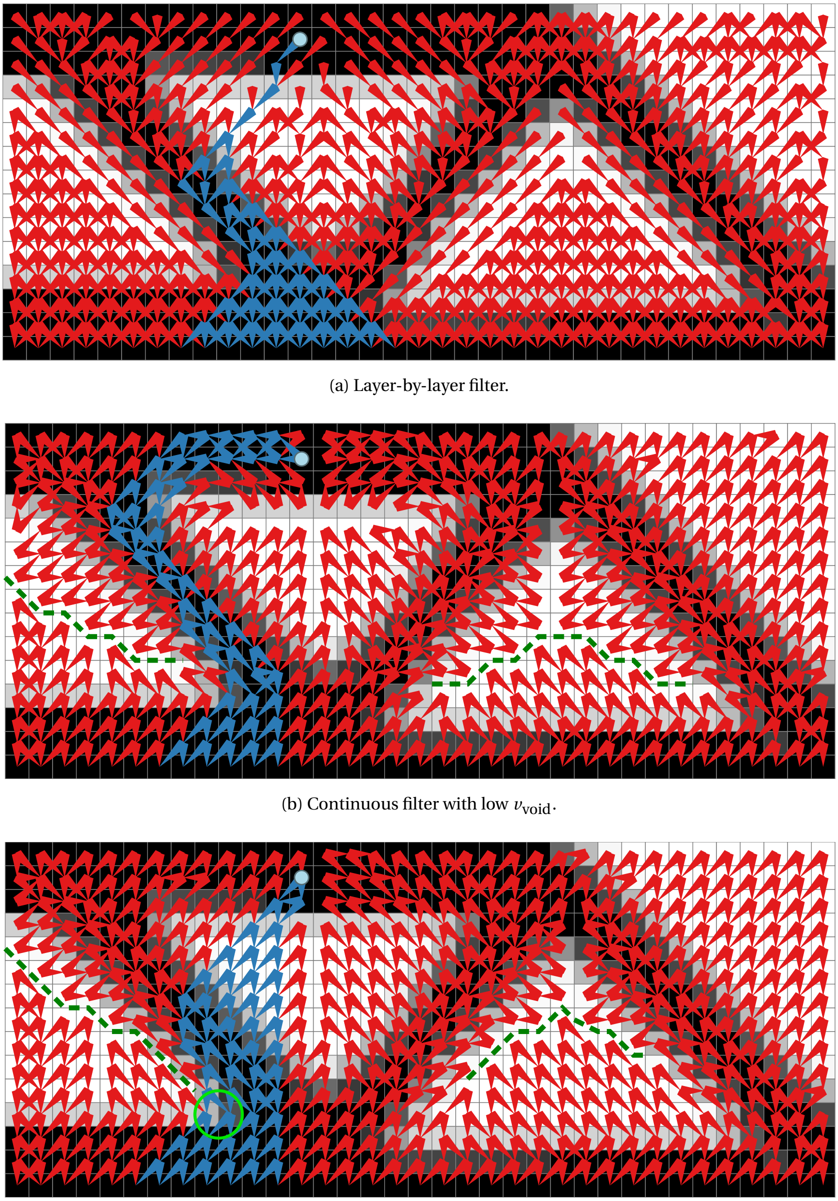
Altogether, if the filter is required to function on an unstructured mesh, the anisotropic speed function used in the layer-by-layer filter cannot be used. Therefore, a different speed function is chosen that instead of having a zero propagation speed in the direction that are not printable, has a non-zero propagation speed in those directions. This requires a different processing of the resulting arrival times of the front propagation, which is discussed in Section 5.4.2.

Effect of the speed function on sensitivities

The speed function influences the direction in which the front propagates, which determines the order in which the nodes nodes meet with the propagating front. This is important for the resulting sensitivities, as the sensitivity information spreads through the domain in exactly the opposite order as in which the front is propagated. This is shown in Fig. 5.13, where the layer-by-layer filter and the continuous filter have been applied to the same geometry shown in black-white. For every element, it is displayed from which other elements its arrival time has been calculated, indicated by the arrows. For one element, indicated with a light-blue circle, all the nodes that could have contributed to this node arrival time are highlighted with blue arrows.

For the layer-by-layer filter (Fig. 5.13a), each element can only be updated from the three elements of the underlying layer (Eq. (5.1)). Since a smooth maximum is used to determine from which of the three lower elements it is updated, it can depend on upto three elements. This is usually the case when all the three underlying elements have the same printable density, for example for all the elements of the second row. On the other hand, in the implementation of the continuous filter as in van de Ven *et al.* (2018b), no smooth minimum operator is used to select from which elements the arrival time is calculated (Eq. (5.17)). Therefore, each element depends strictly on two other elements: the two elements that define the line segment from which the element was updated (Eq. (5.16)), which can be seen in Fig. 5.13b. When multiple update directions result in a similar arrival time, a single direction is chosen. This can be seen in, for example, the elements on the second row. Most of them are updated from the elements below-left and below. An update from the elements below-right and below would have resulted in the same arrival time, but this is neglected due to the strict minimum operator. Furthermore, contrary to the layer-by-layer filter, the continuous filter allows propagation in all directions. Therefore, one can find elements that have been updated from elements in the same or even higher layers, as is the case for the element indicated with the light-blue circle.

As the sensitivities are propagated in opposite direction of the front propagation, these differences have a major effect on the sensitivities, albeit the resulting arrival times are similar. In Fig. 5.13, all the elements that have a blue arrow are in some way involved in the calculation of the arrival time of the element indicated with the light-blue circle. In other words, in order to increase the printable density of the indicated element, the density of the blue-arrowed elements should be increased. For the layer-by-layer filter, this results in the creation of a support column through the void region in the middle. For the continuous filter, the column to left of the indicated element will be expanded towards this element. In this case, the layer-by-layer filter is more likely to form support columns, while the continuous filter will expand or shift existing column for support.



(a) Layer-by-layer filter.

(b) Continuous filter with low v_{void}.

(c) Continuous filter with high v_{void}.

Figure 5.13: The arrows pointing from each element indicate which elements are used to calculate its arrival time. This indicates to which elements sensitivity information is passed. For one element, highlighted by a blue circle, all the dependencies of elements that are involved in the calculation of its arrival time are coloured blue.

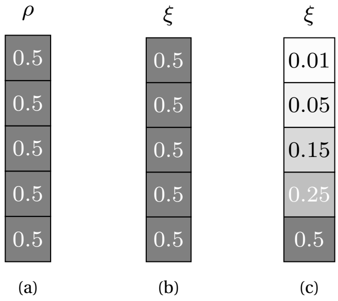


Figure 5.14: Processing of a uniform intermediate density field (a), a typical initial condition of density based topology optimization, by the layer-by-layer filter (b), and the continuous filter (c).

For the continuous filter, the speed with which the front propagates through void can be increased, as can be seen in Fig. 5.13c. The higher void speed shifts the region for which it is faster to travel directly to the base upwards (i.e. where elements are updated from below, as in the layer-by-layer filter), as indicated by the dashed green line in Fig. 5.13c and Fig. 5.13b. This will result in a more similar behaviour to the layer-by-layer filter. Finally, a higher void speed also allows the front to ”cut corners”, as can be seen by the corner indicated with the green circle, which can be beneficial for the sensitivities, as these corner elements are usually the only elements whose density can be further increased, as can be seen in Fig. 5.13b for the blue-arrowed elements.

5.4.2. Differences in speed-density relations

One of the most attractive features of the layer-by-layer filter as compared to the continuous filter, is that an element can be supported by an underlying element with a similar or higher density. For example, when the layer-by-layer filter is applied to a uniform density field of $\rho = 0.5$, the resulting printable density field is identical: $\xi = 0.5$, as displayed in Figs. 5.14a and 5.14b. With the continuous filter, the front will travel at half speed through the domain, and the delay will increase with increasing height, resulting in a printable density field that decays with height, as can be seen in Fig. 5.14c. Since only material at the bottom of the domain is usually an unfavourable starting condition, it is proposed in van de Ven *et al.* (2018b) to activate the continuous filter after 10 iterations.

The layer-by-layer filter achieves its behaviour by a smart choice of speed-density relation. As explained in Section 5.3.2, the propagation speed is only lowered when the current element has a lower density than the printable density of the supporting element. Otherwise, $g = 1$, maintaining a propagation speed equal to f_0, the propagation speed used for the reference field T^{layer}. If the propagation speed is equal to f_0, the delay does not increase, and the printable density is equal to that of the supporting element, resulting in the desired behaviour described by Eq. (5.2).

For the continuous filter, the propagation speed is simply scaled with the element's density: when the element has full density, $g = 1$, and for intermediate densities $g < 1$. This results in a stricter penalization of intermediate densities: for every intermediate density element, $g < 1$ and the delay increases.

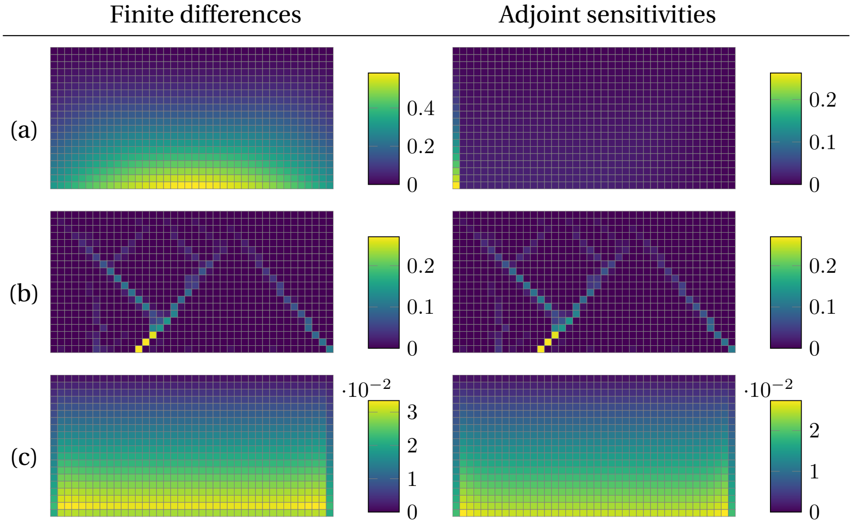


Figure 5.15: Comparison of finite differences and adjoint sensitivities for (a) the continuous filter applied to a uniform density field, (b) the continuous filter applied to a typical MBB beam result, and (c) the smoothed continuous filter applied to a uniform density field. The values on the color bars represent the magnitude of the derivative of the volume fraction with respect to the densities [-].

Certainly, the layer-by-layer speed-density relation is preferable for better convergence. In Section 5.5 it is investigated if the speed function used in the layer-by-layer filter can be transferred to the continuous filter to improve its performance.

5.4.3. Smooth min/max approximations

The final difference between both methods that is compared in this work, is the implementation of the minimum/maximum operators of both methods to select which underlying element is the supporting element. For the layer-by-layer filter, this is the element with the highest printable density (Eq. (5.1)), and for the continuous filter the element with the lowest arrival time (Eq. (5.17)).

In Langelaar (2017), the discrete maximum operator is implemented as a penalized p-norm, to provide a smooth differentiable function. This enables the calculation of correct sensitivities when the arguments of the maximum operator are close to each other. In van de Ven *et al.* (2018b), the minimum operator that selects the lowest arrival time is implemented as a discrete operator: only the lowest arrival time is used for the front propagation, and in the sensitivity analysis the adjoint is propagated only to the elements that provided the lowest arrival time. This leads to incorrect sensitivity information when the arrival time calculated from different nodes is equal, which is especially visible in the first iteration, where one starts with a uniform density field.

In order to investigate the effect of the discrete operator on the sensitivities, consider the volume constraint

$$f_1 = \sum_{i=1}^{N} \xi_i V_{\text{el}} / V_{\text{tot}} - V_{\text{frac}}, \tag{5.31}$$

where V_{el} and V_{tot} are the element and domain volume, respectively, and N is the total number of elements. For the volume constraint on a structured mesh, the derivatives to the printable densities are equal for every element:

$$\frac{\mathrm{d}f_1}{\mathrm{d}\xi_i} = V_{\text{el}} / V_{\text{tot}}, \quad \forall i. \tag{5.32}$$

The sensitivities towards the densities $\frac{\mathrm{d}f_1}{\mathrm{d}\rho_i}$ for a uniform density field are displayed in Fig. 5.15a. The finite differences show the expected result, where densities towards the bottom have a higher sensitivity value, since more printable densities (and thus a large volume) are influenced by the lower elements than by the higher elements, as the front propagates bottom to top.

Unfortunately, the adjoint sensitivity analysis is not able to capture the sensitivity behaviour accurately, as can be seen in Fig. 5.15a. Although for every element the three underlying elements provide equal arrival times in the uniform density field, the sensitivity information is propagated to only one of the underlying elements because of the discrete minimum operator. In this case, the sensitivities are transferred to the lower left element because it has the lowest index. This results in an aggregation of sensitivities in the leftmost column, as displayed in Fig. 5.15a.

Fortunately, this is a problem that only occurs for a perfectly uniform density field. For example, when a tiny random perturbation with a magnitude of 1×10^{-5} is added to the density field, the adjoint sensitivities and finite differences are virtually identical. This can be seen in Fig. 5.15b, where instead of a uniform density field, a density field resulting from an MBB-beam optimization is used, similar to the density field shown in Fig. 5.10a. As can be seen, there is a good correspondence between the finite differences and adjoint sensitivities, with a maximum relative difference of less than 0.1%.

It is however possible to formulate the front propagation with a smooth minimum operator. The implementation is mainly an exercise in bookkeeping as, contrary to the layer-by-layer filter, it is not known a-priori from how many elements an element is updated. This is determined during the front propagation, based on the order in which the elements are updated. Therefore, one needs to keep track of the number of updates an element receives, from which elements these updates are done, the resulting arrival times of these updates, and the corresponding sensitivity information. The resulting adjoint sensitivities are much closer to the finite differences for a uniform field, as shown in Fig. 5.15c. As the smoothed minimum operator is tedious to implement and did not result in a significant improvement in convergence, it was not implemented in van de Ven *et al.* (2018b).

5.5. Improved continuous overhang filter

In the previous section, a detailed comparison between the layer-by-layer filter and the continuous filter was performed, enabled by the front-propagation formulation of the layer-by-layer filter. In this section, an improved version of the continuous front propagation filter is proposed, by incorporating elements of the front propagation-formulation of the layer-by-layer filter.

In Section 5.4.1 it was shown that the wedge-shaped speed function used by the layer-by-layer filter can only be used on a structured grid, and is therefore not eligible for use in the continuous filter. However, the speed-density relation g (Eq. (5.30)) can be applied to the rectangular speed function of the continuous filter. For this, only one alteration is required. For the layer-by-layer filter, the distance between two elements projected on the build direction is constant and appears as Δy in the speed-density relation (Eq. (5.30)). For the continuous filter, an element i is update from a location $\mathbf{x}_\gamma$ on a line segment between two other elements j and k, as displayed in Fig. 5.5. Therefore, the speed-density relation is formulated as:

$$g = \frac{|\mathbf{b}\cdot(\mathbf{x}_i - \mathbf{x}_\gamma)|}{f_0\left(h^{-1}\left(\min(\xi_\gamma, \rho_i)\right) - h^{-1}(\xi_\gamma)\right) + |\mathbf{b}\cdot(\mathbf{x}_i - \mathbf{x}_\gamma)|}, \tag{5.33}$$

with

$$\xi_\gamma = h(T_\gamma - T_\gamma^{\text{layer}}). \tag{5.34}$$

In principle, this speed-density relation can be interpreted similar to the layer-by-layer version, as described in Section 5.3.2. Compared to the old speed density relation, this speed function is not strictly anisotropic. When $\xi_\gamma \leq \rho_i$ on the whole line segment $\mathbf{x}_j\mathbf{x}_k$, the speed-density relation reduces to 1, and is isotropic (except for the edge case when $|\mathbf{b}\cdot(\mathbf{x}_i - \mathbf{x}_\gamma)| = 0$, in which case the speed function should be defined as $g = 1$). However, when $\xi_\gamma > \rho_i$, the speed-density relation tries to increase the delay such that $\xi_i = \rho_i$, which makes it dependent on $\mathbf{x}_\gamma$ and thus anisotropic. Therefore, one cannot use the same update algorithm as proposed in van de Ven *et al.* (2018b). For this study, the minimization problem that has to be solved to find the minimal arrival time over the line segment $\mathbf{x}_j\mathbf{x}_k$ (Eq. (5.16)) is solved with brute force by probing 10 locations along the line segment and accepting the minimum value. For practical applications, a similar solution as in van de Ven *et al.* (2018b) should be constructed to speed up the front propagation.

Another effect of using the speed-density relation as given in Eq. (5.33), is that the choice of the delay-density relation h (Eq. (5.34)), as long as it is monotonically decreasing, is not relevant: the speed-density relation g will reduce the equation for printable densities to a minimum formulation (Eq. (5.24)), independent of h. Therefore, the parameter k that had to be set for the continuous overhang filter in van de Ven *et al.* (2018b) (Eq. (5.6)), is no longer required. This simplifies the formulation and makes it more robust, as a user-defined parameter is eliminated.

5.5.1. Numerical results

First, the layer-by-layer filter and the conventional and improved continuous filter are compared on a 2D problem, after which the improved continuous filter is demonstrated in 3D.

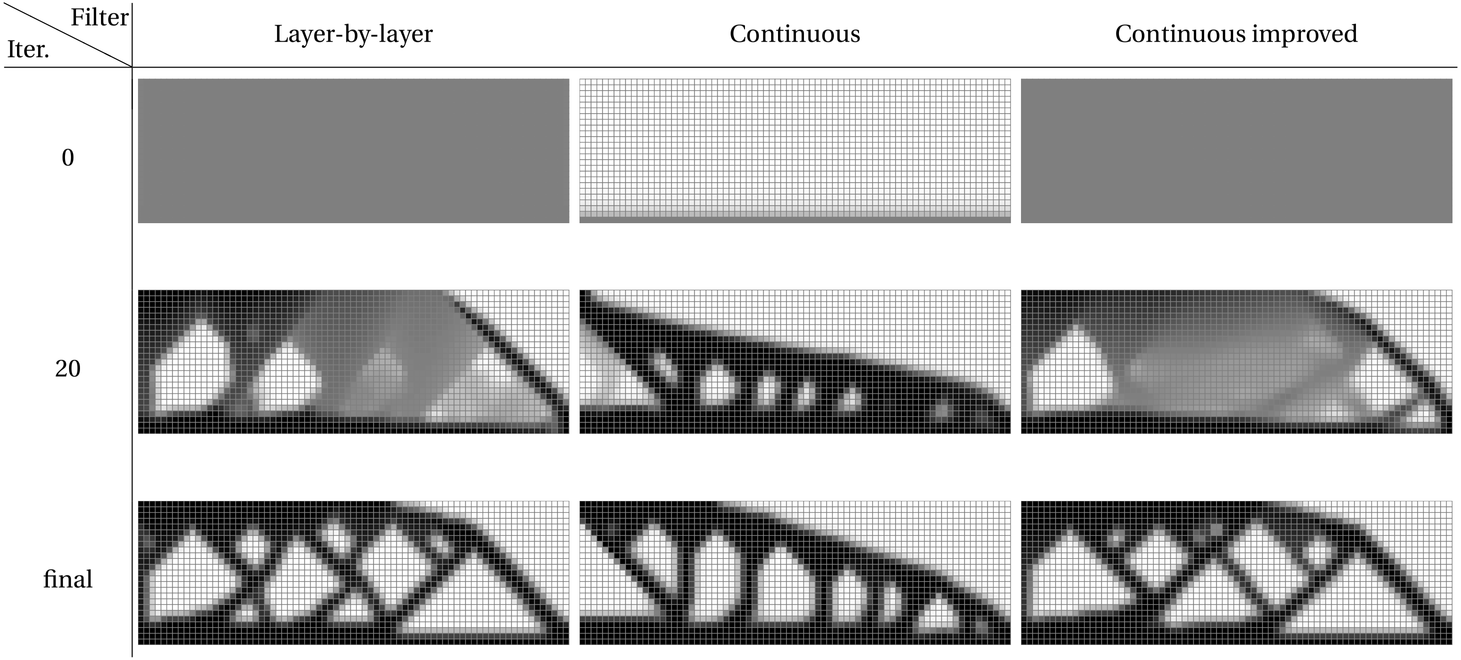


Figure 5.16: Initial, intermediate and final designs of the different overhang filters for the MBB beam optimization problem.

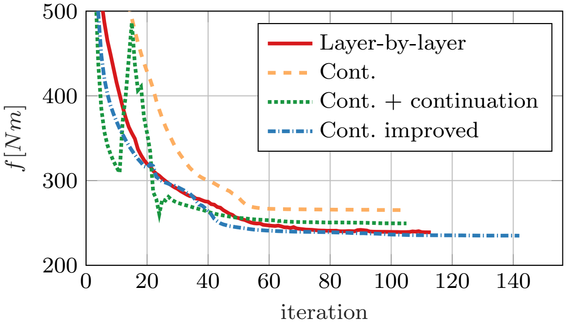


Figure 5.17: Convergence behaviour of the different overhang filters.

2D CASE - COMPARISON

The different overhang filters are compared on the compliance optimization of the MBB-beam with 3:1 aspect ratio, with a volume constraint set to 50%. A density filter (Bruns and Tortorelli, 2001; Sigmund and Petersson, 1998) with a filter radius $r = 2\Delta x$ is applied, intermediate densities are penalized with SIMP penalization (Bendsøe, 1989; Bendsøe and Sigmund, 1999) of $p = 3$, and the Method of Moving Asymptotes (Svanberg, 1987) is used as optimization algorithm. The optimizations are terminated when the maximum change of printable densities is smaller than 0.01.

The printable densities at iteration 0, 20, and the final design for the three methods are displayed in Fig. 5.16. The initial condition is a uniform density field with $\rho = 0.5$. It can be seen that both the layer-by-layer filter and the improved continuous filter allow the stacking of intermediate densities, and thus have a resulting initial field of $\xi = 0.5$. The conventional implementation of the continuous filter does not allow this and therefore only the bottom layers remain. This is an unfavourable starting condition, as the design now has to grow from the bottom, as can be seen in the following snapshots. Eventually the optimization terminates in an suboptimal local minimum, as can be seen in Fig. 5.17. This behaviour is very similar to the overhang filter presented in Gaynor and Guest (2016).

The layer-by-layer filter and the improved continuous filter do not suffer from the poor initial condition, and can straight away put material at the top of the domain, as can be seen in Fig. 5.16. Both methods finalize in similar designs with similar objective values, as displayed in Fig. 5.17. Finally, the convergence behaviour of the conventional continuous filter with gradual activation after 10 iterations, as proposed in van de Ven *et al.* (2018b), is also plotted in Fig. 5.17. As the filter is activated, the objective spikes, but converges to a lower objective than without continuation. However, the final objective is still higher than the layer-by-layer filter and the improved continuous filter.

3D CASE - BICYCLE PULL BRAKE

To fully demonstrate the capabilities of the improved continuous filter, it is demonstrated on a compliance case with a more challenging design domain, such as one might encounter in real-life problems. For the 3D case a linear pull brake for a bicycle is optimized for stiffness against a 10% volume constraint. The case will be optimized with-

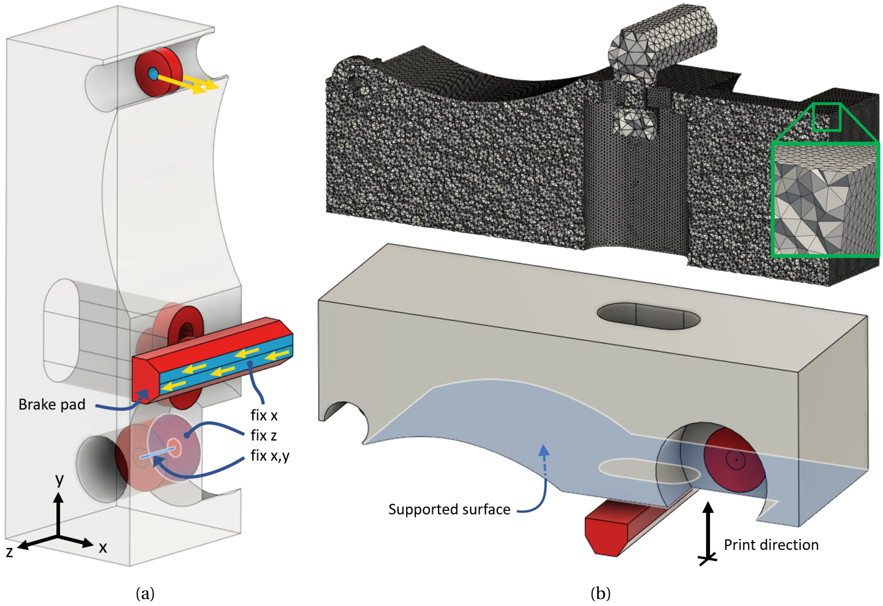


Figure 5.18: The bicycle pull brake case. The fixed regions are shown in red, while the design domain is shown in gray. Applied forces are indicated by yellow arrows and fixed regions are indicated in (a). The domain is meshed with tetrahedral elements, of which a clipped section is shown in (b) and is to be printed in the orientation as shown in (b).

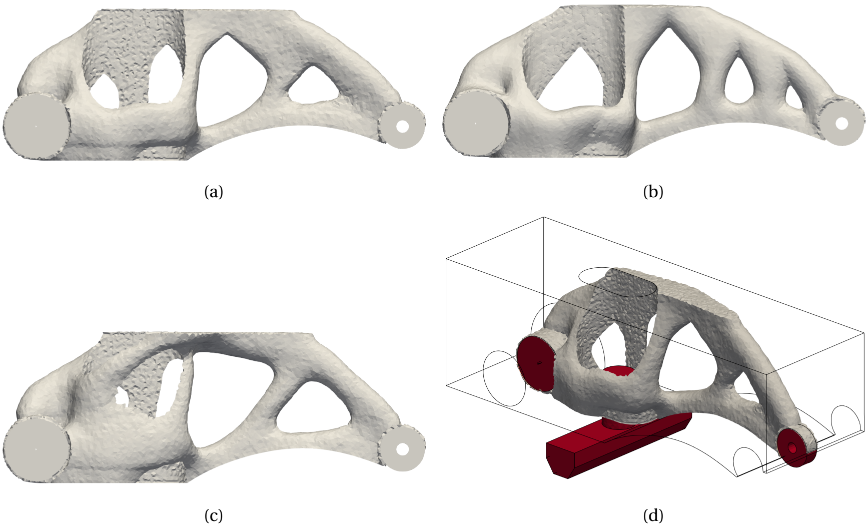


Figure 5.19: Optimized geometries of the bicycle pull brake with: improved continuous filter (a), the old continuous filter with continuation (b), and without overhang filter (c). The geometry of the improved continuous filter is also shown with outlines of the original design domain, where the fixed density regions are shown in red (d).

out an overhang constraint and with a 45° overhang constraint. The 3D optimization is performed with an optimization code based on the Portable, Extensible Toolkit for Scientific Computing (PETSc) (Balay *et al.*, 1997, 2017; Karypis and Kumar, 1998), and visualised with ParaView (Ahrens *et al.*, 2005). Isosurfaces of the printable density field at $\xi = 0.5$ are depicted for visualization. Finally, the PETSc-based MMA implementation presented in Aage and Lazarov (2013); Aage *et al.* (2015) has been used as optimization algorithm.

The design domain for the pull brake is shown in Fig. 5.18. The red regions are fixed regions, while the gray region is the design domain. The pull brake is mounted to the bicycle with a bolt at the lowest fixed region in Fig. 5.18a. This is simulated by fixing the center line of the bolted region in x and y-direction, allowing rotation around the z-axis, and by fixing the back side in z-direction as indicated. When the brake is applied, a clamping force of $F_{\text{clamp}} = 200\,\text{N}$ is exerted on the top fixed region, indicated by the two yellow arrows in x-direction. This will push the brake pad against the wheel. Therefore, the brake pad is fixed in x-direction, and the brake force exerted by the tire on the brake pad is applied in z-direction, indicated by the yellow arrows in z-direction. The force on the brake pad is proportional to the clamping force: due to the geometry of the lever mechanism, the clamping force is amplified by a factor 3.7, which gives $F_{\text{brake}} = 3.7\mu F_{\text{clamp}}$, where μ is the friction coefficient assumed to be 0.5. The mechanism is to be printed (without the brake pad) in the orientation shown in Fig. 5.18b. A small amount of support will be required to manufacture the curved section of the supported surface, but since metal components are generally already printed on a small layer of supports, this is not introducing a large amount of additional support material.

The irregular design domain of the pull brake shows the advantage of being able to restrict overhanging features on unstructured meshes. Standard meshing algorithms can be used to create a tetrahedral mesh, and fixed regions such as the brake pad can be meshed with a lower resolution to reduce the total number of nodes, as shown in Fig. 5.18b. In this case, the domain is meshed using Gmsh (Geuzaine and Remacle, 2009), with 2.7×10^5 nodes and 1.5×10^6 elements.

The optimized geometries are shown in Fig. 5.19. It can be seen that the geometries optimized with overhang filter can be printed in the given orientation without support material, while the geometry without overhang constraint has some overhanging members. The optimizations ran for 300 iterations, however, after 90 iterations there is little change in the design and the objective values of all three optimizations are within 1% of their final values. The convergence behavior for the first 90 iterations is shown in Fig. 5.20. The convergence curve is smooth for the improved continuous filter, as there is no need for continuation. For the continuous filter which requires continuation there is some non-smooth behavior between iteration 10 and 20 where the continuation is applied, but the impact on the final result is negligible. It can however happen that the continuation is too aggressive, and the design has to be built up from the bottom of the domain, as was the case in Fig. 5.16 for the continuous filter without continuation. Therefore, the improved filter is a more stable version of the continuous filter because it does not require continuation. The final objective values for the optimizations with the continuous improved filter and continuous filter with continuation are 0.6% and 1.0% higher than without overhang filter, respectively. It is however difficult to compare these

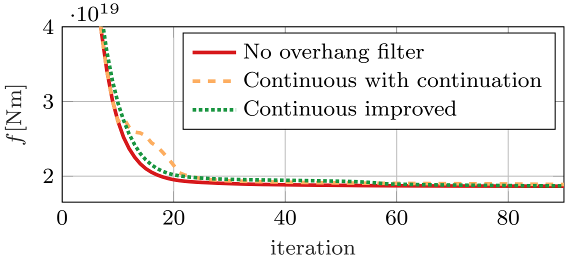


Figure 5.20: Convergence behavior for the first 90 iterations of the 3D case.

values as the overhang filter slightly alters the length scale introduced by the density filter.

5.6. Conclusion

In this paper, a detailed comparison is made between the layer-by-layer overhang filter as proposed in Langelaar (2017), and the continuous overhang filter as proposed in van de Ven *et al.* (2018b). It is demonstrated that the layer-by-layer filter, originally formulated in a discrete setting, can be formulated in a continuous setting using front propagation. Similar to the continuous filter, the printable densities are based on a delay between two arrival time fields. As such, the layer-by-layer filter can be formulated using front propagation by only changing the speed function, and the speed-density relation.

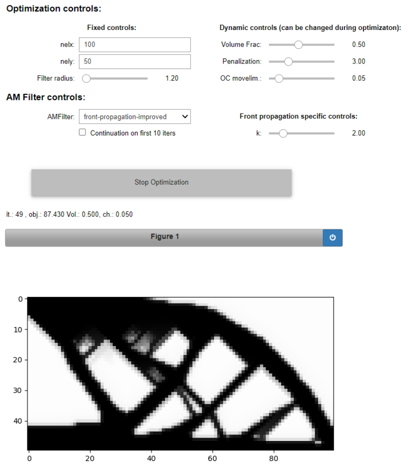
The front propagation formulation of the layer-by-layer filter allows a component-level comparison between the two methods. It is shown that the speed function used by the layer-by-layer function cannot be applied to unstructured meshed due to the large anisotropy of the speed function. However, the speed-density relation of the layer-by-layer filter has the advantageous characteristic that it allows the support of intermediate-density elements by elements of the same or higher density, improving convergence in the early stage of the optimization. Additionally, the different implementations of the minimum/maximum operators are investigated. As the continuous filter does not use a smooth minimum operator, the resulting sensitivities for a uniform density field are incorrect. It is possible to use smoothed operators, but in practice the difference is small, as the sensitivities are only inaccurate for a uniform density field: a minutely perturbed density field results in correct sensitivities.

Finally, the speed-density relation used by the layer-by-layer filter is transferred to the continuous overhang filter, resulting in an overhang filter with the advantageous characteristics of the layer-by-layer filter, but directly applicable to unstructured meshes.

A Matlab implementation of the continuous filter is provided with this work, to aide with the implementation of a front propagation based overhang filter in any existing density-based topology optimization.

5.7. Replication of results

An implementation of the front propagation-based overhang filter presented in this work is provided in the form of an interactive Jupyter notebook, shown in Fig. 5.21. It



Optimization controls:

Fixed controls:

nelx: 100

nely: 50

Filter radius: 1.20

Dynamic controls (can be changed during optimizaton):

Volume Frac: 0.50

Penalization: 3.00

OC movelim.: 0.05

AM Filter controls:

AMFilter: front-propagation-improved

□ Continuation on first 10 iters

Front propagation specific controls:

k: 2.00

it.: 49 , obj.: 87.430 Vol.: 0.500, ch.: 0.050

Figure 5.21: Snapshot of the Jupyter notebook provided in the supplementary material.

is based on the Python implementation of the 88-line topology optimization code (Andreassen *et al.*, 2011). It contains the front propagation-based overhang filter as presented in van de Ven *et al.* (2018b), the improved variant presented in this work, and a Python adaption of the layer-by-layer filter provided in Langelaar (2017). In order to run the notebook, open a Jupyter notebook (`www.jupyter.org`), inside the notebook navigate to the supplementary material and run `TopOptAMapplet.ipynb`. Then follow the instructions inside the notebook. The reader is encouraged to play with the various overhang filters and parameters to gain an understanding of their effect on the optimization.

ACKNOWLEDGEMENTS

This research was carried out under project number S12.7.14549a in the framework of the Metals for Additive Manufacturing Partnership Program of the Materials innovation institute M2i (www.m2i.nl) and the Netherlands Organization for Scientific Research (www.nwo.nl). The authors thank Krister Svanberg for use of the MMA optimizer.

REFERENCES

Aage N, Lazarov BS (2013) Parallel framework for topology optimization using the method of moving asymptotes. Structural and multidisciplinary optimization 47(4):493–505

Aage N, Andreassen E, Lazarov BS (2015) Topology optimization using PETSc: An easy-to-use, fully parallel, open source topology optimization framework. Structural and Multidisciplinary Optimization 51(3):565–572

Adam GA, Zimmer D (2014) Design for Additive Manufacturing—Element transitions and aggregated structures. CIRP Journal of Manufacturing Science and Technology 7(1):20–28

Ahrens J, Geveci B, Law C (2005) Paraview: An end-user tool for large-data visualization. In: Hansen CD, Johnson CR (eds) Visualization Handbook, Butterworth-Heinemann, Burlington, pp 717 – 731

Allaire G, Dapogny C, Estevez R, Faure A, Michailidis G (2017) Structural optimization under overhang constraints imposed by additive manufacturing technologies. Journal of Computational Physics 351:295–328

Amir O, Mass Y (2018) Topology optimization for staged construction. Structural and Multidisciplinary Optimization 57(4):1679–1694

Andreassen E, Clausen A, Schevenels M, Lazarov BS, Sigmund O (2011) Efficient topology optimization in MATLAB using 88 lines of code. Structural and Multidisciplinary Optimization 43(1):1–16

Balay S, Gropp WD, McInnes LC, Smith BF (1997) Efficient management of parallelism in object oriented numerical software libraries. In: Arge E, Bruaset AM, Langtangen HP (eds) Modern Software Tools in Scientific Computing, Birkhäuser Press, pp 163–202

Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Eijkhout V, Gropp WD, Kaushik D, Knepley MG, McInnes LC, Rupp K, Smith BF, Zampini S, Zhang H, Zhang H (2017) PETSc users manual. Tech. Rep. ANL-95/11 - Revision 3.8, Argonne National Laboratory

Bendsøe MP (1989) Optimal shape design as a material distribution problem. Structural Optimization 1(4):193–202

Bendsøe MP, Sigmund O (1999) Material interpolation schemes in topology optimization. Archive of Applied Mechanics (Ingenieur Archiv) 69(9-10):635–654

Bendsøe MP, Sigmund O (2004) Topology Optimization: Theory, Methods and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg

Bruns TE, Tortorelli DA (2001) Topology optimization of non-linear elastic structures and compliant mechanisms. Computer Methods in Applied Mechanics and Engineering 190(26-27):3443–3459

Cloots M, Spierings AB, Wegener K (2013) Thermomechanisches multilayer-modell zur simulation von eigenspannungen in SLM-proben. In: Hildebrand J (ed) Simlationsforum 2013 - Schweissen und Waermebehandlung, Weimar, Germany

Gaynor AT, Guest JK (2016) Topology optimization considering overhang constraints: Eliminating sacrificial support material in additive manufacturing through design. Structural and Multidisciplinary Optimization 54(5):1157–1172

Geuzaine C, Remacle JF (2009) Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. International journal for numerical methods in engineering 79(11):1309–1331

Guo X, Zhou J, Zhang W, Du Z, Liu C, Liu Y (2017) Self-supporting structure design in additive manufacturing through explicit topology optimization. Computer Methods in Applied Mechanics and Engineering 323:27–63

Hoffarth M, Gerzen N, Pedersen C (2017) ALM Overhang Constraint in Topology Optimization for Industrial Applications. In: Proceedings of the 12th World Congress on Structural and Multidisciplinary Optimisation, Braunschweig, Germany, p 12

Jiang J, Stringer J, Xu X, Zhong RY (2018) Investigation of printable threshold overhang angle in extrusion-based additive manufacturing for reducing support waste. International Journal of Computer Integrated Manufacturing 31(10):961–969

Karypis G, Kumar V (1998) A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. Journal of Parallel and Distributed Computing 48:71–85

Kranz J, Herzog D, Emmelmann C (2015) Design guidelines for laser additive manufacturing of lightweight structures in TiAl6V4. Journal of Laser Applications 27(S1):S14,001

Langelaar M (2016) Topology optimization of 3D self-supporting structures for additive manufacturing. Additive Manufacturing 12:60–70

Langelaar M (2017) An additive manufacturing filter for topology optimization of print-ready designs. Structural and Multidisciplinary Optimization 55(3):871–883

Langelaar M (2018) Combined optimization of part topology, support structure layout and build orientation for additive manufacturing. Structural and Multidisciplinary Optimization 57(5):1985–2004

Liu J, Gaynor AT, Chen S, Kang Z, Suresh K, Takezawa A, Li L, Kato J, Tang J, Wang CCL, Cheng L, Liang X, To AC (2018) Current and future trends in topology optimization for additive manufacturing. Structural and Multidisciplinary Optimization 57(6):2457–2483

Mercelis P, Kruth JP (2006) Residual stresses in selective laser sintering and selective laser melting. Rapid Prototyping Journal 12(5):254–265

Qian X (2017) Undercut and overhang angle control in topology optimization: A density gradient based integral approach. International Journal for Numerical Methods in Engineering 111(3):247–272

Ranjan R, Ayas C, Langelaar M, van Keulen A (2018) Towards design for precision additive manufacturing: A simplified approach for detecting heat accumulation. In: Proceedings of the ASPE and EUSPEN Summer Topical Meeting

Sethian JA, Vladimirsky A (2003) Ordered Upwind Methods for Static Hamilton–Jacobi Equations: Theory and Algorithms. SIAM Journal on Numerical Analysis 41(1):325–363

Sigmund O, Petersson J (1998) Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima. Structural Optimization 16(1):68–75

Sih SS, Barlow JW (2004) The Prediction of the Emissivity and Thermal Conductivity of Powder Beds. Particulate Science and Technology 22(4):427–440

Svanberg K (1987) The method of moving asymptotes—a new method for structural optimization. International Journal for Numerical Methods in Engineering 24(2):359–373

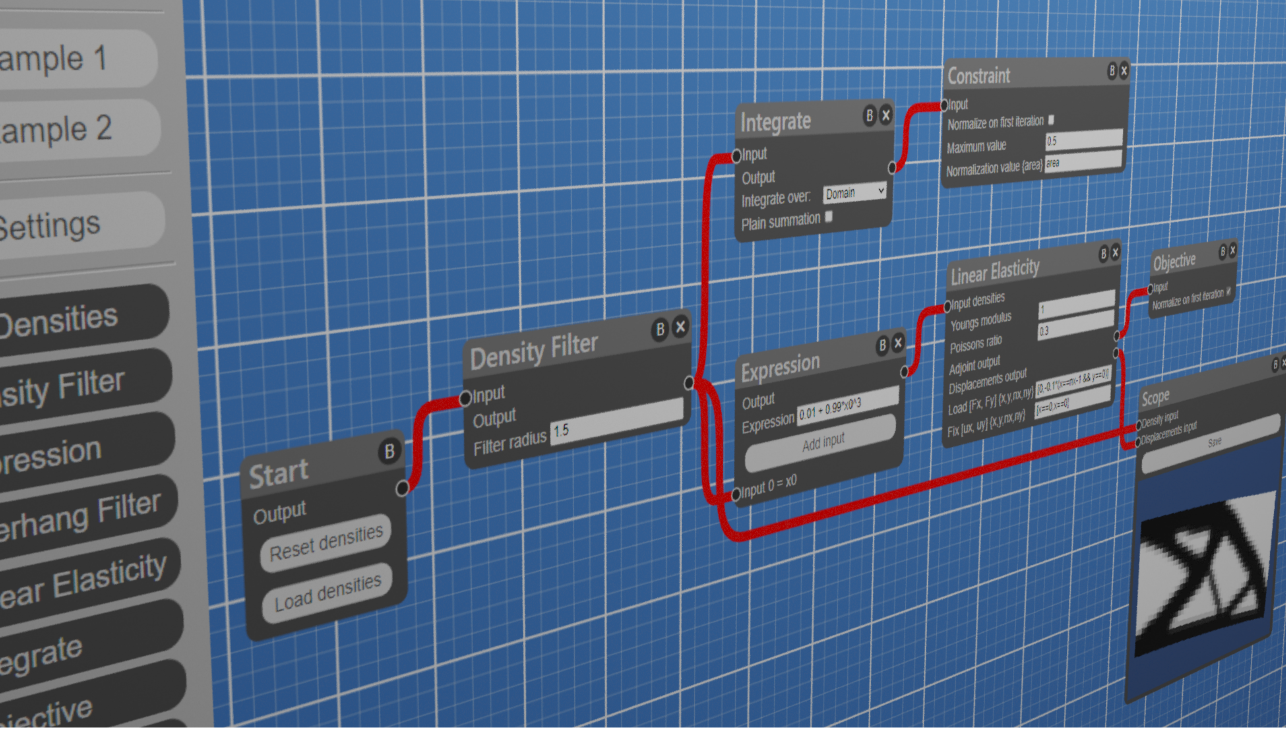
Thomas D (2009) The Development of Design Rules for Selective Laser Melting. PhD thesis, University of Wales

van de Ven E, Ayas C, Langelaar M, Maas R, van Keulen F (2018a) A PDE-Based Approach to Constrain the Minimum Overhang Angle in Topology Optimization for Additive Manufacturing. In: Schumacher A, Vietor T, Fiebig S, Bletzinger KU, Maute K (eds) Advances in Structural and Multidisciplinary Optimization, Springer International Publishing, Cham, pp 1185–1199

van de Ven E, Maas R, Ayas C, Langelaar M, van Keulen F (2018b) Continuous front propagation-based overhang control for topology optimization with additive manufacturing. Structural and Multidisciplinary Optimization 57(5):2075–2091

Wang D, Yang Y, Yi Z, Su X (2013) Research on the fabricating quality optimization of the overhanging surface in SLM process. The International Journal of Advanced Manufacturing Technology 65(9-12):1471–1484

Zhang K, Cheng G, Xu L (2019) Topology optimization considering overhang constraint in additive manufacturing. Computers & Structures 212:86–100



6

Accessibility of support structures in topology optimization

Additive manufacturing (AM) and topology optimization (TO) have a synergetic relation, as AM can produce complex TO designs, and TO provides high-performance parts that utilize the form freedom provided by AM. Recently, TO has been tailored more towards AM with the inclusion of the minimum allowable overhang angle as a design constraint: resulting designs can be built without any support structures. This work is an extension thereof, by allowing support structures only if they are accessible, such that they can be removed after manufacturing. This is achieved by applying a conventional overhang filter twice, combined with basic operations such as geometry inversion, union and intersection. The result is an accessibility-aware overhang filter that can be incorporated in TO. Compared to conventional overhang filtered designs, the accessibility filter results in increased part performance and better convergence behavior. Furthermore, a modular filter structure is presented to easily construct the accessibility filter, and its effectiveness is demonstrated on several numerical cases.

This chapter is published in *International Journal for Numerical Methods in Engineering* 122(8), 2038–2056 (2021).

6.1. Introduction

The most attractive feature of additive manufacturing (AM) is the unprecedented degree of permitted form freedom. No longer bounded by the design limitations of conventional manufacturing techniques, such as casting or milling, the challenge becomes to find the geometrical layout of the component that truly gives the best performance (Frazier, 2014; Hague *et al.*, 2003). This is exactly what is provided by topology optimization (TO). Compared to other structural optimization methods, such as size or shape optimization, it exploits maximum design freedom, and is widely recognized as a design approach perfectly suitable for AM (Bendsøe and Sigmund, 2004; Christensen and Klarbring, 2008).

Although AM can produce geometrically complex parts created by TO, there are design restrictions that have to be taken into account to prevent an increase in production cost. For example, a design might require a large amount of supports if the angle between a down-facing surface and the base plate is below a material/process specific value, the so called critical overhang angle α_{oh} (Cloots *et al.*, 2017; Wang *et al.*, 2013; Yasa *et al.*, 2009). Support structures not only increase the printing cost due to increased material and post-processing cost, they can even become impossible to remove once the printing is completed.

Several studies have been published that determine design rules for AM (Adam and Zimmer, 2014; Kranz *et al.*, 2015; Thomas, 2009; Thompson *et al.*, 2016), and implementation of these design rules into TO is an active research topic; see Liu *et al.* (2018) for an overview. Especially, minimizing the need for support structures received considerable attention. Three approaches have emerged: first by enforcing a minimum overhang angle on the surface of the geometry (Allaire *et al.*, 2017a; Guo *et al.*, 2017; Mirzendehdel and Suresh, 2016; Qian, 2017), second by sweeping through the structure layer-by-layer to detect areas that violate the minimum overhang angle (Gaynor and Guest, 2016; Johnson and Gaynor, 2018; Langelaar, 2016b, 2017; van de Ven *et al.*, 2018), and, finally, by modelling a physical aspect of the printing process, accounting for the heating or the self weight of each layer (Allaire *et al.*, 2017a,b; Amir and Mass, 2018; Ranjan *et al.*, 2018).

Most of the methods mentioned above focus on completely eliminating all the supports. Consequently, the design becomes printable at the expense of its performance. However, in practice support structures might be acceptable if they are significantly beneficial for the part weight or performance. For example in high performance applications, the cost of support fabrication and removal might be irrelevant. Then, no performance shall be sacrificed to prevent support structures, unless these supports are inaccessible, i.e. impossible to remove after the build. Therefore, only those regions which are inaccessible, such as internal channels and cavities, must be free of overhang. This can be enforced by the above mentioned support elimination schemes. Consequently, it is of paramount importance to be able to asses where the design is accessible.

The partial admittance of support structures in TO has been addressed in several studies. In Mirzendehdel and Suresh (2016) support structures are assumed to be columnar, and the total support volume is constrained, while in Langelaar (2016a) the support layout is optimized simultaneously with the part, and minimized according to a cost function. The approach presented in Langelaar (2016a) is extended in Langelaar (2019) to also optimize the support structure for machining forces required in post-printing operations. Although in these studies supports are reintroduced to improve performance, the ease of support removal is not taken into account. This could result in undesirable configurations where the admitted support cannot be removed after printing. Furthermore, in contrast to Langelaar (2016a) and Langelaar (2019), the aim of this study is not to generate an explicit support layout, but merely account for regions in which supports could be removed afterwards. As such, no additional design variables are introduced for the support layout. In this study, we propose a TO methodology to enforce the minimum allowable overhang angle exclusively in regions where support structures are difficult to remove. This results in an accessibility filter for TO, which eliminates regions that both violate the minimum allowable overhang angle *and* are inaccessible. An integral part of the method is the use of an overhang filter. In this study the front-propagation-based overhang filter presented in van de Ven *et al.* (2018, 2020) is used, which can identify and suppress overhanging regions in an efficient manner on unstructured meshes.

Because the proposed accessibility filter is a combination of several individual filters, a framework is presented in which filters can be easily combined and reordered. This framework is then used to define the accessibility filter set, which is demonstrated on numerical examples.

This paper is organised as follows. In Section 6.2, the concept of the accessibility filter is detailed, and the filter combination framework is explained. Then, numerical examples are given in Section 6.3, followed by the discussion (Section 6.4) and conclusions (Section 6.5).

6.2. METHOD

After a brief introduction to TO, the accessibility filter is presented in this section. As the accessibility filter consists of a combination of other filters, we present a general framework to combine filters and calculate sensitivities, which allows for quick experimentation and adaptation. In this framework, the individual filter steps of the accessibility filter are described in detail.

6.2.1. TOPOLOGY OPTIMIZATION

There are several approaches to TO (Deaton and Grandhi, 2014; Eschenauer and Olhoff, 2001; Sigmund and Maute, 2013), and in this paper we use density-based TO (Bendsøe, 1989; Bendsøe and Sigmund, 2004), where the layout of a part is specified throughout the design domain by a pseudo-density field $\boldsymbol{\rho}$, which indicates for each location if it contains material ($\rho = 1$), or is void ($\rho = 0$). In order to avoid an integer programming problem, the optimization problem is relaxed by allowing intermediate densities ($0 \leq \rho \leq 1$).

During the topology optimization, the density field is optimized to provide the best performance for an objective under certain constraints. This is an iterative procedure, where in every iteration the design is evaluated and then updated based on the sensitivities. The latter are the derivatives of the objective and constraints with respect to the design variables (the density field $\boldsymbol{\rho}$), and indicate the effect of a change in density value on the objective and constraint values. The optimization is usually initiated with a homogeneous density field, e.g. $\rho = 0.5$.

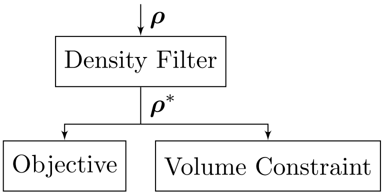
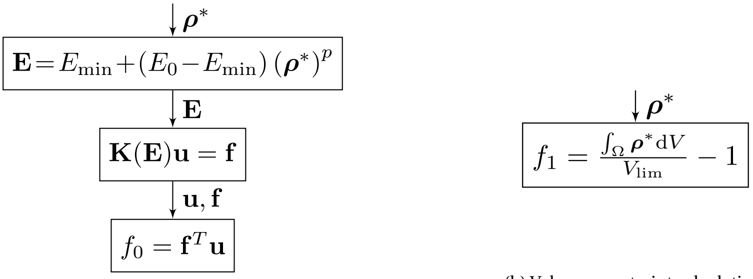


Figure 6.1: Basic TO scheme.


$\boldsymbol{\rho}^*$
$\mathbf{E} = E_{\text{min}} + (E_0 - E_{\text{min}})\,(\boldsymbol{\rho}^*)^p$
$\mathbf{E}$
$\mathbf{K}(\mathbf{E})\mathbf{u} = \mathbf{f}$
$\mathbf{u}, \mathbf{f}$
$f_0 = \mathbf{f}^T\mathbf{u}$
$\boldsymbol{\rho}^*$
$f_1 = \frac{\int_\Omega \boldsymbol{\rho}^* \mathrm{d}V}{V_{\text{lim}}} - 1$

(a) Objective calculation

(b) Volume constraint calculation.

Figure 6.2: Detailed objective and constraint evaluation for a compliance TO.

One of the most basic and well-studied optimization problems is compliance minimization subject to a volume constraint. The most common TO scheme for compliance minimization is given in Figure 6.1. The density field $\boldsymbol{\rho}$, provided by the optimization algorithm, is first processed by a density filter, which smooths the density field to prevent artefacts such as checkerboarding (Sigmund and Petersson, 1998) and introduces a length scale to prevent mesh-dependence. In this study, the filtered densities ρ^* are a weighted average of the densities $\boldsymbol{\rho}$ within a certain filter radius r around the point of interest $\mathbf{x}_j$ (Bruns and Tortorelli, 2001):

$$\rho_i^* = H_{ij}\rho_j, \tag{6.1}$$

$$H_{ij} = \omega_{ij} / \textstyle\sum_k \omega_{ik}, \tag{6.2}$$

$$\omega_{ij} = \max(r - \|\mathbf{x}_i - \mathbf{x}_j\|, 0), \tag{6.3}$$

where $\mathbf{H}$ is the filter matrix and r the length scale that is introduced. The filtered densities $\boldsymbol{\rho}^*$ are then used to evaluate the objective and the volume constraint. The process of calculating the objective f_0 is depicted in Figure 6.2a. First, the Young's modulus is evaluated in each element, which is scaled with the penalized filtered densities. The penalization is done using SIMP interpolation, in order to suppress intermediate densities in the final design (Bendsøe, 1989; Rozvany *et al.*, 1992). The effective Young's moduli for material and void regions are E_0 and E_{min}, respectively. The lower bound E_{min} is introduced to avoid singularity of the stiffness matrix $\mathbf{K}$. Then, the displacements are calculated with a finite element analysis (FEA), where $\mathbf{f}$ and $\mathbf{u}$ are the discretized force and displacement vectors resulting from the FEA, and the stiffness matrix $\mathbf{K}$ is a function of the Young's modulus field $\mathbf{E}$. Finally the compliance is calculated. For the volume

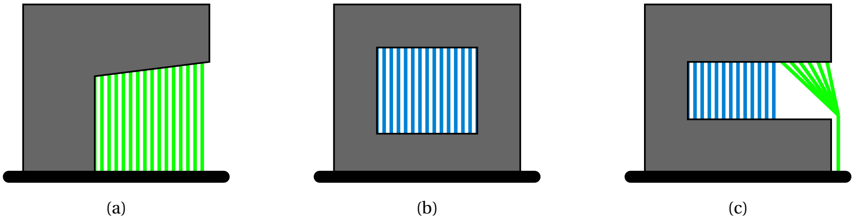


Figure 6.3: Examples of accessible support that can be easily removed (green), and inaccessible supports that are difficult to remove (blue).

constraint f_1, the volume fraction is calculated by dividing the volume integral of the filtered densities ρ^* by the volume limit V_{lim}, depicted in Figure 6.2b. Altogether, the minimization problem can be formulated as

$$\begin{aligned} \min_{\boldsymbol{\rho}} \quad & f_0 \\ \text{s.t.} \quad & \mathbf{Ku} = \mathbf{f}, \\ & f_1 \leq 0, \\ & \mathbf{0} \leq \boldsymbol{\rho} \leq \mathbf{1}. \end{aligned} \tag{6.4}$$

6.2.2. ACCESSIBILITY OF SUPPORTS

Recall that the aim of this paper is to obtain a geometry using topology optimization, of which the supports that are required to print the part are accessible for ease of removal. We define this requirement as only allowing supports that are both (1) printable themselves, and (2) connected to the base plate. Examples of support that are accessible and inaccessible are given in Figure 6.3. Any location that cannot be supported by *accessible* supports, should be free of overhang. The first requirement is straightforward: supports must be realized through printing. The effect of requiring supports to connect to the base plate is to prevent supports from forming in internal voids, from which they cannot be removed (Figure 6.3b). Furthermore, it also prevents supports in internal channels, as the required supports cannot be built from the base plate (Figure 6.3c). Finally, when the supports connect to the base plate, only one side of the supports connects to the part. Supports that are connected to the part on both ends are thus not allowed. This can save post-processing cost required to polish the surfaces affected by supports. Note that internal voids are inadmissible in certain printing technologies, but for this paper we do not take that into consideration as other works have addressed this topic (Gaynor and Johnson, 2020; Li *et al.*, 2016).

In order to achieve the accessibility of supports, a filter is required that can take any geometry as input, and removes the regions that require supports that are inaccessible. Such a filter can then be inserted into the TO scheme. Before going into the details of the formulation, the concept of the accessibility filter will be explained on the 2D geometry displayed in Figure 6.4. If a conventional overhang filter would be applied to this structure, with $\alpha_{\text{oh}} = 45°$, both overhanging areas (shaded red) are identified as not printable and thus removed by the filter. The overhanging area on the left should indeed

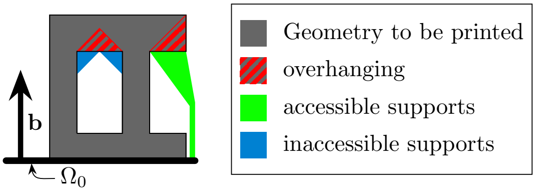


Figure 6.4: Geometry that is to be printed on the base plate Ω_0 in the build direction indicated by **b**. Potential support structures are indicated. Supports of the left overhanging section in the cavity are inaccessible.

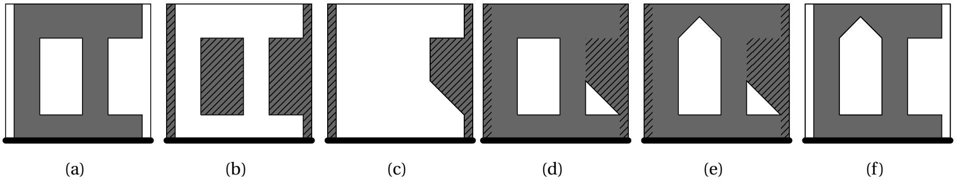


Figure 6.5: The progression of the geometry given in Figure 6.4 as the accessibility filter is applied step-by-step.

be removed, as it is in an internal void and therefore the supports will be inaccessible for removal. However, the overhanging area on the right can be supported with a support connected to the base plate, as indicated by the green region in Figure 6.4. This overhanging region is thus accessible and should not be removed. Note that in order to allow supports to connect to the base plate, as shown by the accessible supports in Figure 6.4, the design domain should be slightly wider than the original design space. This is achieved by placing a void non-design space around the geometry.

The basic principle of the accessibility filter is to apply a conventional overhang filter to the union of the original geometry and those regions where accessible supports can be printed. In order to achieve this, the following steps, visualized in Figure 6.5, are followed, starting with the input geometry shown in Figure 6.5a:

1. Obtain the region in which supports can be placed by inverting the domain, as shown in Figure 6.5b. In order to differentiate between the original structure and the support region, the support region is shaded.

2. From the region in which supports can be placed, remove the regions that are not printable and/or accessible by applying a conventional overhang filter. In Figure 6.5c it can be seen that the enclosed void is indeed removed from the domain.

3. Combine the support region attained from the previous step and the original structure. The result, displayed in Figure 6.5d, is the original structure, combined with all possible accissible supports.

4. Remove the regions from the original structure that cannot be supported by accessible supports by applying the conventional overhang filter once more. The result is given in Figure 6.5e, showing that overhanging areas in internal voids are not

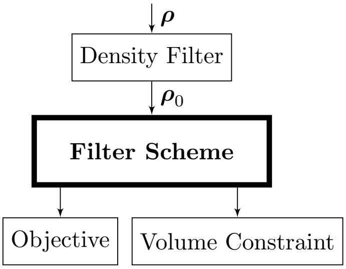


Figure 6.6: TO layout with generic filter scheme.

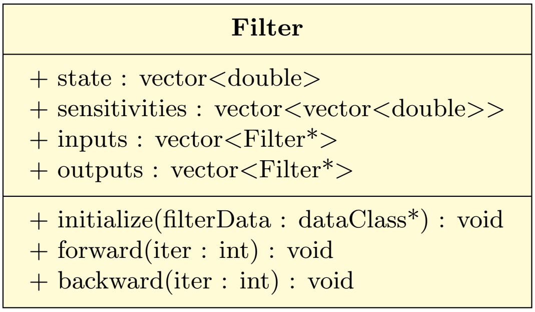

Filter
+ state : vector<double>
+ sensitivities : vector<vector<double>>
+ inputs : vector<Filter*>
+ outputs : vector<Filter*>
+ initialize(filterData : dataClass*) : void
+ forward(iter : int) : void
+ backward(iter : int) : void

Figure 6.7: Diagram of the `Filter` class.

printable since no accessible supports can be placed to print its overhanging top surface.

5. Finally, remove all the support material that was added in Step 3 by taking the intersection of the previous result with the original structure (Figure 6.5f). The inaccessible overhanging area is removed, while the accessible overhanging area is retained.

Because of the many steps required for the accessibility filter, a general framework is set up to systematically organize and implement such a chain of filter operations. This is discussed in the next section.

6.2.3. FILTER STRUCTURE

In order to construct the accessibility filter as outlined in the previous section, a scheme composed of several filters is to be inserted between the density filter and the objective and constraint evaluation, as can be seen in Figure 6.6. These filters might consist of simple expressions executed entry-wise on a density field, similar to, for example, the SIMP penalization in Figure 6.2a, or more complex operations such as a density or overhang filter.

Conventionally, the filter operations would be programmed in the order they occur, and the sensitivities have to be determined for the filter scheme. This can be a tedious task if the filter scheme consists of many operations. Furthermore, inserting an additional filter to the scheme requires reprogramming and compilation of the code. As the accessibility filter scheme contains a relatively large number of filters, a more flexible approach is taken, detailed in this section.

Although each filter performs a different operation, all the filters share some properties: they all produce a density field as output, and will have a function to process the filter and a function to calculate the sensitivities. To facilitate the systematic construction of complex filter chains, it is therefore convenient to approach the filter scheme in an object-oriented manner, where every filter is derived from the same base class: `Filter`, depicted in Figure 6.7. The `Filter` class has several attributes:

- `state`: a vector that contains the density field computed by the filter.
- `sensitivities`: a vector that for every response function contains a vector of the derivative of that response function to the output density field.
- `inputs, outputs`: vectors of pointers to the filters that are inputs/outputs to the current filter.

Furthermore, the `Filter` class has three functions: `initialize()`, which initializes the filter (e.g. set up the filtering matrix for a density filter), `forward()`, which applies the filter, and `backward()`, which processes the sensitivities. Filter-specific data, e.g. filter radius or overhang angle, and general data such as mesh information or filter inputs and outputs, are passed to the Filter object when `initialize()` is called. To the other two functions, only the iteration number is passed.

Filter types

From the base class `Filter`, the following filter classes are derived:

Expression filter The expression filter evaluates a mathematical expression containing only componentwise operations. It is the only filter that can have multiple inputs. This allows for example interpolation between several density fields, or boolean operations on the geometry. Furthermore, the expression can have a continuation variable c that takes a value based on the iteration number. An example of the expression filter is the application of SIMP penalization in Figure 6.2a.

Overhang filter The overhang filter removes regions that violate the minimum overhang angle from a design, and leaves the printable part. In this work the front propagation based implementation as presented in van de Ven *et al.* (2018) has been used.

Filter layout and initialization

The filter schemes used in this work can all be represented as a directed acyclic graph: the filter scheme has one input I_0, one output O_k per response function k, and the scheme does not contain any loops. A typical example is given in Figure 6.8, which takes the filtered densities $\boldsymbol{\rho}_0$, applies the overhang filter, and then interpolates between $\boldsymbol{\rho}_0$ and the overhang filtered densities $\boldsymbol{\rho}_1$, based on a continuation parameter c. The output is the density field $\boldsymbol{\rho}_2$, which is passed to outputs O_0 and O_1, which are connected to the objective and volume constraint (Figure 6.6).

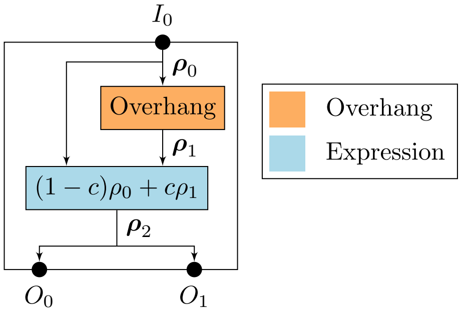


Figure 6.8: Example of a filter structure where an overhang filter is followed by an expression filter that interpolates between the original density field $\boldsymbol{\rho}_0$ and the overhang-filtered densities $\boldsymbol{\rho}_1$.

The filter scheme can be stored in a data file (e.g. XML (Bray *et al.*, 2008) or JSON (noa, 2017)), such that it can be read at runtime. The data file will contain a list of the filters, their inputs and outputs, and parameters such as the expression or continuation scheme for the expression filter, or the minimum overhang angle and build direction for the overhang filter. At runtime, the filters are read from the file and added to a list, `filterList`, and initialized. During initialization, the `state` and `sensitivities` vectors are allocated, `inputs` and `outputs` vectors set, and filter specific initializations performed.

During the evaluation of the filter scheme, every filter can only be evaluated once all its inputs have been evaluated. For example, in Figure 6.8, the expression filter can only be evaluated after the overhang filter. It is therefore convenient to order the `filterList` in topological ordering, such that for every filter in `filterList`, its inputs have lower indexes in `filterList`. The topological ordering can be achieved manually, by specifying the input file in the correct order, or by using a topological ordering algorithm such as Kahn's algorithm (Kahn, 1962).

EVALUATING THE FILTER SCHEME

Since the `filterList` is in topological ordering, one can simply loop over the `filterList` and call the `forward()` function of each filter to evaluate the filter scheme. Each filter can access the `state` vectors of its inputs through the `inputs` vector, and perform the necessary manipulations to calculate its own `state` vector.

EVALUATING SENSITIVITIES

In order to calculate the sensitivities of each response function k towards the input densities of the graph $\boldsymbol{\rho}_0$, the chain rule of differentiation is used to systematically evaluate the sensitivities filter by filter. For a filter i, whose output density field $\boldsymbol{\rho}_i$ is input to the filters N, the sensitivity of its output density field with respect to a response function f_k is defined as

$$\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_i} = \sum_{j\in N} \left[\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_j}\right]^{\mathbf{T}} \frac{\partial \boldsymbol{\rho}_j}{\partial \boldsymbol{\rho}_i}, \tag{6.5}$$

Algorithm 5 Sensitivity calculation for response k.

```
Initialize:
    $N_f = \text{LENGTH}(\texttt{filterList})$
    $\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_i} = \mathbf{0}, \quad i = 1,\dots,N_f$
    For filter $i$ directly connected to $O_k$ set: $\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_i} = \frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_{O_k}}$
for $i = N_f$ to 1 do
    $N_i \leftarrow$ filters that are input to filter $i$
    for $j \in N_i$ do
        $\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_j} \mathrel{+}= \left[\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_i}\right]^{\mathbf{T}} \frac{\partial \boldsymbol{\rho}_i}{\partial \boldsymbol{\rho}_j}$
    end for
end for
```

where $\partial f / \partial \mathbf{a}$ represents a column vector $\mathbf{g}$ with entries $g_l = \partial f / \partial a_l$, and $\partial \mathbf{a}/\partial \mathbf{b}$ represents a matrix $\mathbf{A}$ with entries $A_{lm} = \partial a_l / \partial b_m$. The sensitivities are propagated through the graph by calculating the right hand side of Equation (6.5) in the filters $j \in N$, and adding this contribution to $\mathrm{d}f_k/\mathrm{d}\boldsymbol{\rho}_i$. Since the `filterList` is in topological ordering, traversing it backwards, from end to beginning, will ensure that each filter j will only evaluate its contribution to its inputs when the term $\mathrm{d}f_k/\mathrm{d}\boldsymbol{\rho}_j$ is already calculated.

The sensitivity evaluation is initiated at the filters connected to the outputs O_k, and then propagated backwards through the filter scheme, as described by Algorithm 5. For the scheme displayed in Figure 6.8, this would result in the following sensitivities

$$\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_0} = \left[\frac{\mathrm{d}f_k}{\mathrm{d}\boldsymbol{\rho}_{O_k}}\right]^{\mathbf{T}} \left(\frac{\partial \boldsymbol{\rho}_2}{\partial \boldsymbol{\rho}_0} + \frac{\partial \boldsymbol{\rho}_2}{\partial \boldsymbol{\rho}_1}\frac{\partial \boldsymbol{\rho}_1}{\partial \boldsymbol{\rho}_0}\right), \quad k = 1,2. \tag{6.6}$$

The sensitivities $\mathrm{d}f_k/\mathrm{d}\boldsymbol{\rho}_{O_k}$ are obtained using adjoint sensitivity analysis for compliance minimization for the objective f_0, and standard differentiation of the volume constraint f_1 (Bendsøe and Sigmund, 2004). Note that in Algorithm 5, the Jacobian matrix $\partial\boldsymbol{\rho}_i/\partial\boldsymbol{\rho}_j$ can be calculated explicitly, as is usually done for e.g. the density filter. This matrix is usually sparse, e.g. for componentwise operations it is diagonal. For more complex filters, e.g. the overhang filter, an adjoint approach can be used to avoid explicit calculation of the Jacobian matrix, and instead directly calculate $\partial f_k/\partial\boldsymbol{\rho}_i$.

6.2.4. Accessibility filter scheme

With the filter framework as defined in the previous section, the concept of the accessibility filter as presented in Section 6.2.2 can now be formulated in a similar manner. The full scheme is given in Figure 6.9. The first five filters exactly represent the five steps presented in Section 6.2.2, and the density fields $\boldsymbol{\rho}_1$-$\boldsymbol{\rho}_6$ resemble Figures 6.5a to 6.5f when applied to the geometry given in Figure 6.5a. Numerical experience shows that convergence is improved when the accessibility filter is inactive for the first 10 iterations, and is then gradually activated over the next 10 iterations, until it is fully active. This is achieved

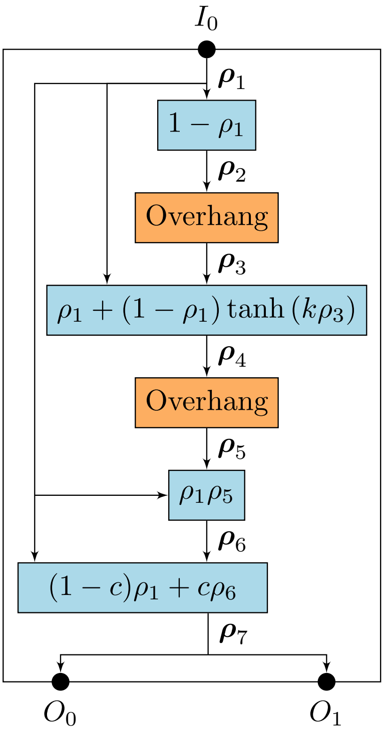


Figure 6.9: The accessibility filter scheme. The first six density fields correspond to the density fields shown in Figure 6.5.

in the last step, which is an interpolation between the original ($c=0$) and the accessibility filtered geometry ($c=1$), similar to Figure 6.8.

The inversion filter ($\boldsymbol{\rho}_1$ to $\boldsymbol{\rho}_2$) and the intersection filter ($\boldsymbol{\rho}_5$ to $\boldsymbol{\rho}_6$) are straightforward operations on the density fields. However, the superimposing filter ($\boldsymbol{\rho}_3$ to $\boldsymbol{\rho}_4$), is less trivial and will be discussed in more detail. The simplest union operation would be $\boldsymbol{\rho}_4 = \boldsymbol{\rho}_3 + \boldsymbol{\rho}_1$. In theory, since $\boldsymbol{\rho}_2 = 1-\boldsymbol{\rho}_1$, and $\boldsymbol{\rho}_3 \leq \boldsymbol{\rho}_2$ because the overhang filter only removes material, $\boldsymbol{\rho}_3+\boldsymbol{\rho}_1 \leq 1$. However, in practice, the overhang filter can slightly smooth the density fields van de Ven *et al.* (2018), and $\boldsymbol{\rho}_3+\boldsymbol{\rho}_1$ can be larger than 1. There are multiple possibilities to avoid this behaviour, and after testing several options the union operation is chosen as

$$\boldsymbol{\rho}_4 = \boldsymbol{\rho}_1 + (1-\boldsymbol{\rho}_1)\tanh(k\boldsymbol{\rho}_3). \tag{6.7}$$

This operation is an addition of the original density field $\boldsymbol{\rho}_1$, and a term smaller or equal to $1-\boldsymbol{\rho}_1$, such that the combination cannot exceed a density of 1. The hyperbolic tangent function was used as it is bounded by 1. The parameter k scales how fast the second term of the right hand side in Equation (6.7) approaches 1, and is chosen as $k=20$.

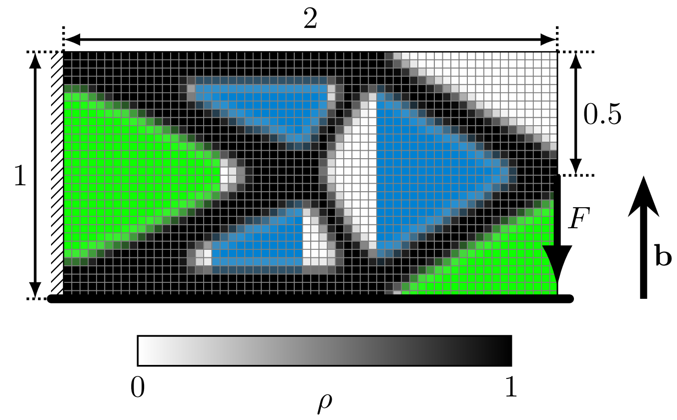


Figure 6.10: Domain and boundary conditions for the cantilever case, with optimal geometry for a conventional stiffness maximization without accessibility filter. The green and blue regions indicate the accessible and inaccessible support, respectively, required for manufacturing with $\alpha_{\mathrm{oh}} = 45°$. Objective $f = 74.2\,\mathrm{Nm}$.

6.3. RESULTS

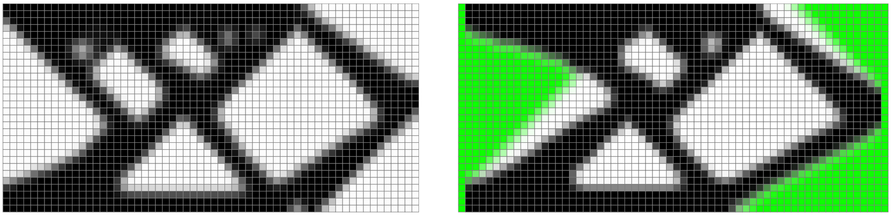
In this section, the accessibility filter is demonstrated both in 2D and 3D, for stiffness maximisation as formulated in Equation (6.4). The Young's modulus is chosen as $E_0 = 1\,\mathrm{Pa}$ for material and $E_{\min} = 1 \times 10^{-6}\,\mathrm{Pa}$ for void, with a Poisson's ratio $\nu = 0.3$, and SIMP penalization factor $p = 3$. For the optimization, the Method of Moving Asymptotes (MMA) (Svanberg, 1987) is used as optimization algorithm. The optimization is terminated when the maximum change of any of the design variables in subsequent iterations is smaller than 0.01, or when a maximum of 300 iterations has been reached.

6.3.1. ACCESSIBILITY FILTER IN 2D

2D CASE: CANTILEVER

In 2D, the accessibility filter is demonstrated on a 2:1 aspect ratio cantilever beam case. For this purpose, a rectangular design domain is mechanically clamped on the left side, with a vertical point force acting on the right, as displayed in Figure 6.10. A 50% volume constraint is applied. The domain is discretized with 60×30 square Q4 elements, and a density filter with a radius of $2h$ is applied, where h is the element edge length. For demonstration purposes, it is assumed that the out-of-plane direction in the third dimension cannot be used for printing, or support removal. The geometry is assumed to be printed in the vertical direction as indicated with the vector **b** in Figure 6.10. Unless specified otherwise, the displayed density field is the filtered density field that forms the input to the objective and volume constraint (see Figure 6.6).

The resulting density field generated by a conventional optimization without overhang or accessibility filter is displayed in Figure 6.10. The required support to print the part for a minimum allowable overhang angle $\alpha_{\mathrm{oh}} = 45°$, is indicated in blue and green. The support material required inside the three interior holes of the structure, indicated in blue, cannot be removed in a 2D setting, thus the design is not suitable for AM. The



(a) Overhang filter. Objective $f = 87.6\,\text{Nm}$.

(b) Accessibility filter, allowing all supports that can be printed with $\alpha_{\text{oh}} = 45°$. Objective $f = 78.3\,\text{Nm}$.

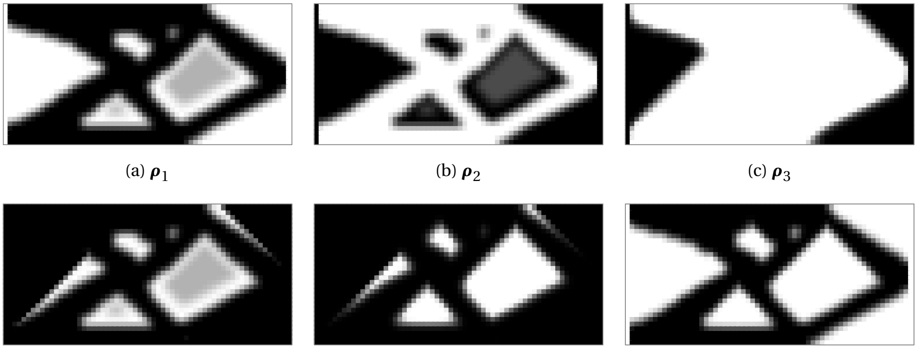
Figure 6.11: Cantilever case result with overhang filter (a), and accessibility filter (b), with $\alpha_{\text{oh}} = 45°$. The result in (a) can be printed without supports, while the green field in (b) indicates $\boldsymbol{\rho}_3$ from the accessibility filter (Figure 6.9), which is the printable support region. The design in (b) has a better performance because less material is sacrificed for manufacturability.

simplest method to enforce printability is to completely avoid the requirement of support material, with the addition of an overhang filter (see Figure 6.8 and van de Ven *et al.* (2018)). The resulting design is displayed in Figure 6.11a, which can be printed without supports.

Depending on the type of component, one might allow supports, as long as they are removable, if, e.g., performance can be gained. In this case, the accessibility filter is applied, as is displayed in Figure 6.9, which only suppresses overhanging regions that are inaccessible. The resulting design is displayed in Figure 6.11b. Notice that the design domain is extended by one element in width on both sides to allow support material to connect to the base plate on both sides of the structure. The extra columns of elements are non-design regions, and the boundary conditions are now one element from the actual boundary, such that the original aspect ratio is not influenced. As can be seen, the enclosed voids that are inaccessible can be printed without the need of support material, while support is required for the outside boundaries that are partially not printable without the use of support material. The field corresponding to $\boldsymbol{\rho}_3$ in Figure 6.9, is displayed in green in Figure 6.11b. This field indicates the accessible region, thus everywhere in the domain where supports can be printed that are directly connected to the base plate. The actual support layout will be only partially filling the green domain.

The consecutive filtering operations of the accessibility filter as illustrated in Figure 6.9, leading to density fields labelled as $\boldsymbol{\rho}_1$-$\boldsymbol{\rho}_6$, are plotted in Figure 6.12. This figure is the numerical equivalent of the schematic picture in Figure 6.5. The filter is initiated with the filtered density field, and the regions of extension on both sides are non-design regions where $\boldsymbol{\rho}_1 = 0$ (Figure 6.12a). This design is inverted (Figure 6.12b), and overhanging regions are removed (Figure 6.12c). The resulting field $\boldsymbol{\rho}_3$ (cf. the green regions in Figure 6.11b) indicates the printable support region. It is then combined with the original design $\boldsymbol{\rho}_1$ (Figure 6.12d), and the overhanging regions of $\boldsymbol{\rho}_4$ are removed (Figure 6.12e). Finally, by removing the printable support region $\boldsymbol{\rho}_3$, the accessibility filtered design $\boldsymbol{\rho}_6$ is obtained (Figure 6.12f).

From the printable support field $\boldsymbol{\rho}_3$ in Figure 6.11b, it can be seen that supports are required that are at the limit of the overhang angle. One can, for example, enforce columnar support by changing the minimum overhang angle used in the overhang fil-



(a) $\boldsymbol{\rho}_1$ (b) $\boldsymbol{\rho}_2$ (c) $\boldsymbol{\rho}_3$

(d) $\boldsymbol{\rho}_4$ (e) $\boldsymbol{\rho}_5$ (f) $\boldsymbol{\rho}_6$

Figure 6.12: Numerical results of the progression of the geometry shown in Figure 6.11b through the accessibility filter.

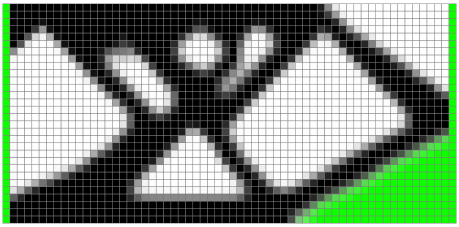


Figure 6.13: Cantilever case result with accessibility filter enforcing vertical supports. The green field indicates $\boldsymbol{\rho}_3$ from the accessibility filter (Figure 6.9), which is the printable support region. Objective $f = 81.8\,\mathrm{Nm}$.

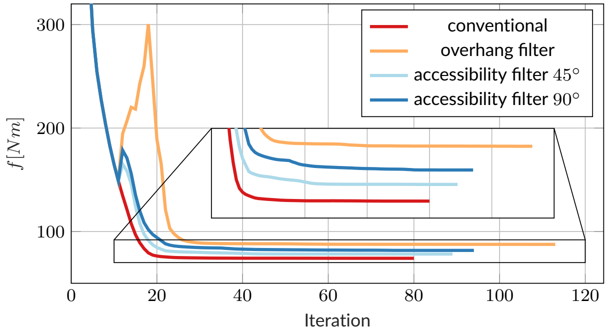


Figure 6.14: Convergence plots for the cantilever case. The accessibility and overhang filters are gradually introduced by continuation from iteration 10 to 20, which causes the increase in objective in that interval.

ter in Step 2 of the accessibility filter (Section 6.2.2), referred to as α_{sup}, to $\alpha_{\text{sup}} = 90°$. The resulting design is displayed in Figure 6.13. Now, also part of the outer boundary is constrained by the overhang filter, and the design is a hybrid between Figure 6.11a and Figure 6.11b, which is also reflected in the performance, discussed next.

CONVERGENCE

The convergence behaviour of the presented 2D cantilever beam optimizations are shown in Figure 6.14. The overhang and accessibility filters are gradually introduced by a continuation scheme, which linearly ramps up the continuation parameter c (Figure 6.9) from 0 to 1 between iteration 10 to 20. Therefore, the convergence in the first 10 iterations is identical for all cases. As expected, the conventional optimization without overhang or accessibility filter has the lowest objective (maximum stiffness), at $f = 74.2\,\text{Nm}$. The overhang filtered optimization, which is the most constrained, has the highest objective at $f = 87.6\,\text{Nm}$. Since the accessibility filter allows for more overhang, the objective is between the conventional and overhang filtered designs at $f = 78.3\,\text{Nm}$ for $\alpha_{\text{sup}} = 45°$, and $f = 81.8\,\text{Nm}$ for $\alpha_{\text{sup}} = 90°$. Furthermore, it can be seen that the less constrained optimizations, next to obtaining a lower objective, also converge slightly faster for this 2D example studied.

6.3.2. ACCESSIBLITY FILTER IN 3D

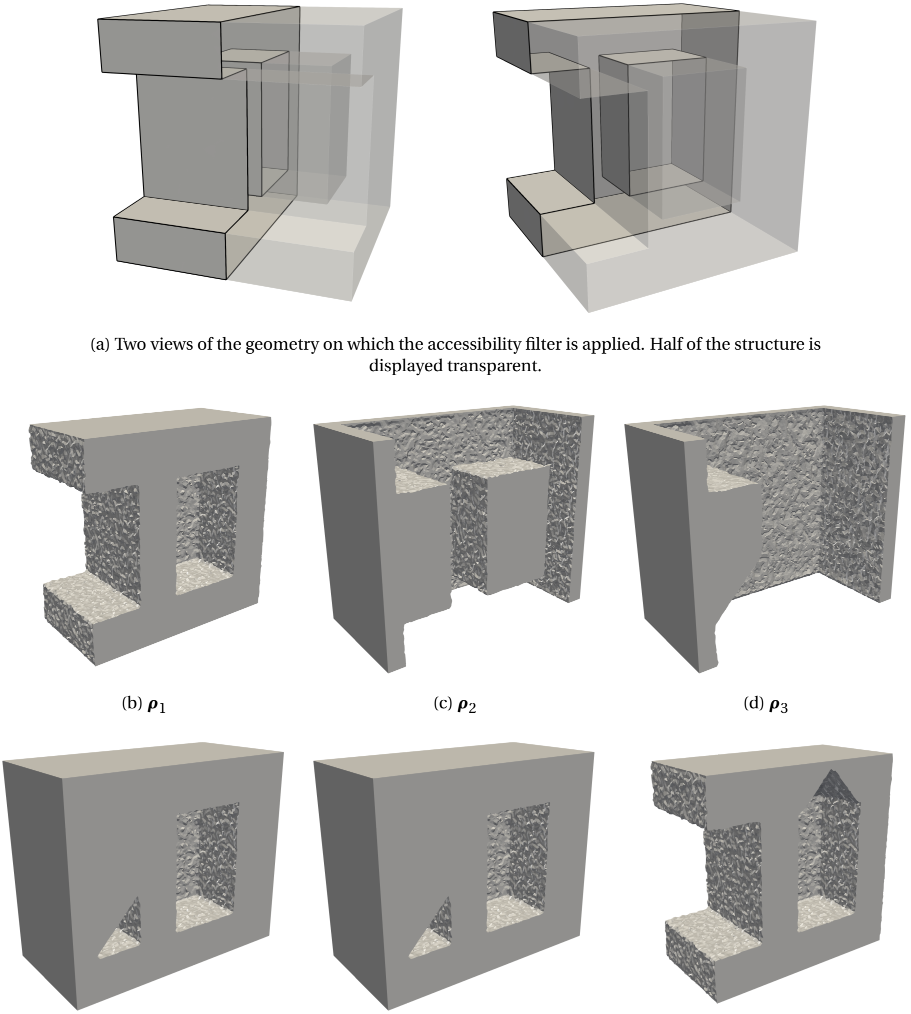
As there is no direct dimension dependency in the accessibility filter, the same filter scheme used in 2D can be directly applied to 3D, provided that a 3D overhang filter is used. Fortunately, many of the published overhang filters have provided a 3D implementation (Johnson and Gaynor, 2018; Langelaar, 2016b; van de Ven *et al.*, 2020). For the 3D results, the front propagation based filter presented in van de Ven *et al.* (2020) is utilized.

The 3D results presented in this section are obtained with an optimization code based on the Portable, Extensible Toolkit for Scientific Computing (PETSc) (Balay *et al.*, 1997, 2017; Karypis and Kumar, 1998). The domains are discretized using Gmsh (Geuzaine and Remacle, 2009), and visualised with ParaView (Ahrens *et al.*, 2005). The displayed geometries are isosurfaces of the density field at $\rho = 0.5$. Finally, the PETSc-based MMA implementation presented in Aage and Lazarov (2013); Aage *et al.* (2015) has been used as optimization algorithm.

Before topology optimized results are presented, the filter operations of the accessibility filter are verified in 3D. The 3D layout illustrated in Figure 6.15a is reminiscent of the layout in Figure 6.4. The geometry contains two overhanging regions: one in the internal cavity, and one on the outside. After the accessibility filter is applied, the overhang filter removes the overhang in the internal cavity, as can be seen Figure 6.15g. Therefore, the remaining structure is manufacturable, as supports can be removed on the outside, but not in the internal cavity.

3D CASE: TORSION BOX

In order to demonstrate the full potential of the accessibility filter, it is applied to a case where the optimal design contains overhanging regions on its outer surface, and in an internal cavity. Closed shell structures, resulting in internal cavities, are often optimal



(a) Two views of the geometry on which the accessibility filter is applied. Half of the structure is displayed transparent.

(b) $\boldsymbol{\rho}_1$

(c) $\boldsymbol{\rho}_2$

(d) $\boldsymbol{\rho}_3$

(e) $\boldsymbol{\rho}_4$

(f) $\boldsymbol{\rho}_5$

(g) $\boldsymbol{\rho}_6$

Figure 6.15: Numerical results of the progression of a geometry (a) through the accessibility filter (b-g). Only half of the geometry is displayed in (b-g).

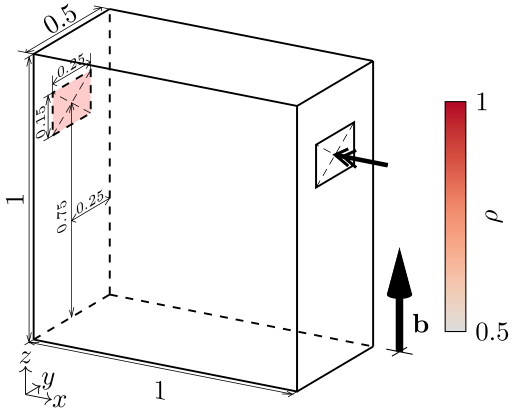
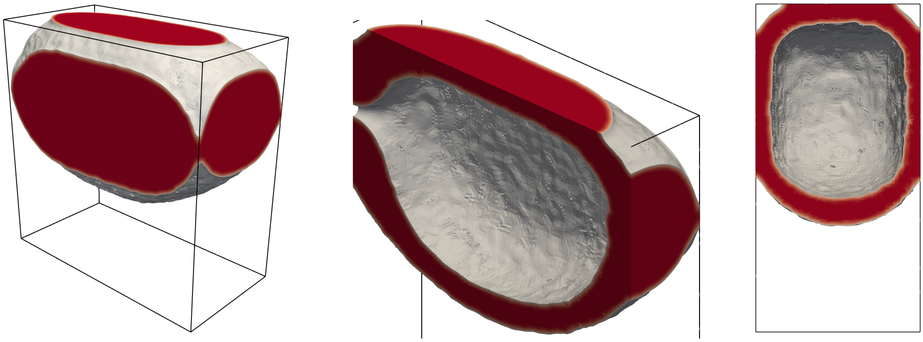


Figure 6.16: Domain and boundary conditions for the torsion box case. A moment is applied on the rectangular patch in the front, while a similar patch on the back, shaded red, is fixed. The color scheme as shown is used in all consecutive figures to represent the density field.

under torsion Sigmund *et al.* (2016). Consequently, the test case is defined as shown in Figure 6.16. A torsional load is applied on a rectangular patch as displayed, and the domain is mechanically fixed on a similar patch at the opposite side. The torsional load is applied by applying a distributed load of 1 N/m on the outer perimeter of the rectangular patch, oriented such that a moment normal to the patch is generated. In order to force overhanging regions on the outer surface, the loading and clamping are placed well above the base plate (i.e. the bottom xy-plane). Furthermore, the volume is constrained at 30% of the design domain. The domain is discretized with an unstructured mesh, containing roughly $7 \cdot 10^6$ tetrahedral elements and $1 \cdot 10^6$ nodes. Finally, the density filter that is applied before the accessibility filter has a filter radius of $2h$, where $h = 0.01\,\mathrm{m}$ is the average element edge length, and a minimum allowable overhang angle for structure and support is chosen as $\alpha_{\mathrm{oh}} = 45°$.

The resulting design for a conventional stiffness optimization, i.e. without accessibility or overhang filtering, is displayed in Figure 6.17. The interior is displayed in Figure 6.17b, where the domain is cut along the $y = 0.25$ plane, and the $x = 0.5$ plane (Figure 6.17c). As expected, as there is enough material available, a closed shell structured is formed to carry the torsional load. Material is placed as far from the centreline of the load as possible, creating an internal cavity. Supports inside the cavity are thus required in order to additively manufacture the component. However, these supports will be impossible to remove after the build.

Instead, when an overhang filer is included to suppress overhang inside the cavity, the result displayed in Figures 6.18a and 6.18b is obtained. As can be seen in the cross-section in Figure 6.18b, there is an internal structure in the cavity with permitted degree of overhang, therefore the cavity can be printed without the need for supports. However, the overhang filter is also active on the outside of the structure, requiring the bottom of the component to be printable without supports. This is not realistic, as the bounds of the optimization domain do not need to correspond with the boundaries of the com-

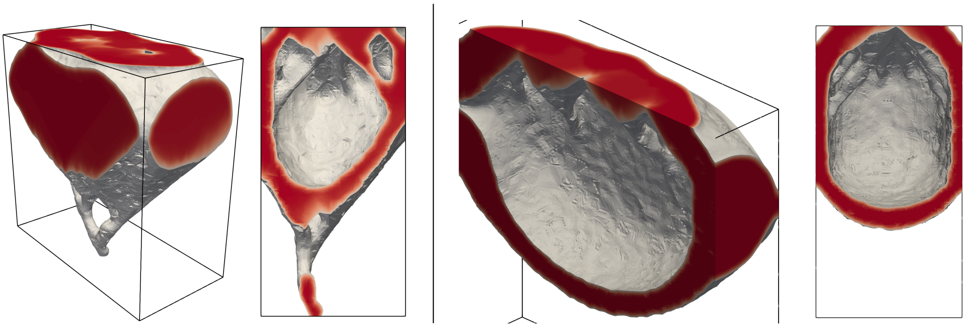


(a) Side view.

(b) Side view, cut at $y = 0.25$ plane.

(c) Front view, cut at $x = 0.5$ plane.

Figure 6.17: Optimal design for a conventional stiffness maximization on the torsion box case.



(a) Side view.

(b) Front view, cut at $x = 0.5$ plane.

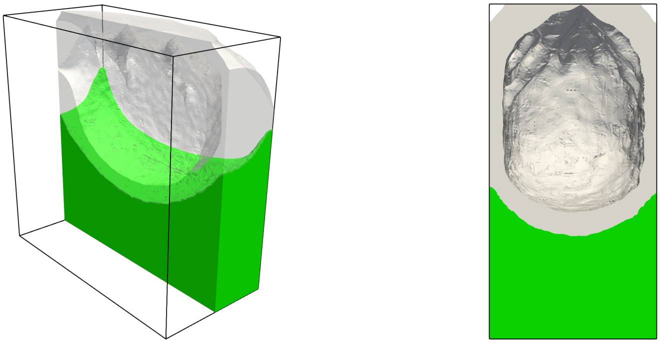
(c) Side view, cut at $y = 0.25$ plane.

(d) Front view, cut at $x = 0.5$ plane.

Figure 6.18: Optimized design for the torsion box case with overhang filter (a-b), and accessibility filter (c-d), with $\alpha_{\mathrm{oh}} = 45°$. With overhang filter, the structure has to connect to the base plate at the bottom to satisfy the overhang constraint, sacrificing performance (a-b). For the accessibility filter (c-d), the structure can be printed on supports, shown in Figure 6.19, increasing the performance.

ponent. Also, certainly with metal AM, the part would not be printed on the base plate directly, but on a layer of supports. Therefore, it makes no sense to apply the overhang filter on the bottom surface. Furthermore, it can be seen that there are some semi-dense areas left in Figure 6.18b, and holes are present in the top of the structure (Figure 6.18a). Because of the overhang filter, the complete geometry pivots on the bottom supports. If the density in one of the bottom nodes is slightly altered, the complete structure can become overhanging and removed by the overhang filter. This large non-linearity makes it difficult for the optimization to converge, and causes the semi-dense regions.

With the accessibility filter applied, a result closer to that of the conventional stiffness optimization is obtained, as can be seen in Figures 6.18c and 6.18d. The supports that are required to manufacture the bottom of the structure are accessible, and therefore the overhang filter is suppressed. Inside the cavity, the overhang filter is active and a self-supporting internal structure is generated to support the top. Similar to the 2D case,



(a) Side view, cut at $y = 0.25$ plane, with transparant geometry.

(b) Front view, cut at $x = 0.5$ plane.

Figure 6.19: The printable support field ($\boldsymbol{\rho}_3$ from the accessibility filter in Figure 6.9) indicated in green, with the final geometry in gray.

the printable support region is displayed in green in Figure 6.19. It represents possible locations for support material, in order to support the bottom of the structure.

TORSION BOX WITH FORCED HOLE

It remains to demonstrate the flexibility of the filter structure presented in Section 6.2.3 to adjust the accessibility filter for more challenging cases. For this purpose, the torsion case discussed above is slightly altered. The same domain and boundary conditions are used, but for demonstration purposes a cylindrical non-design region parallel to the z-axis with a diameter of 0.05 m is introduced, as displayed in Figure 6.20. Consequently, a small hole in the bottom of the design emerges, through which supports stemming from the base-plate can reach the cavity. Therefore, the design obtained with the accessibility filter as displayed in Figure 6.20 is similar to a conventional optimization as shown in Figure 6.17. The support region required to manufacture the design is displayed in green in Figure 6.21a. Although the supports are accessible in the sense that they are connected to the base plate, they still are difficult to remove.

To prevent this, the amount of supports inside the cavity can be reduced by increasing the overhang angle for the supports, α_{sup}, as was done in the 2D case in Figure 6.13. However, this would still leave some supports inside the cylindrical non-design region that are difficult to remove. Instead, we alter the filter to prevent supports from going through small holes. This is achieved by introducing an additional density filter before Step 2 in the accessibility filter, as shown in Figure 6.22. With the filter structure as presented in Section 6.2.3 in place, only the input file has to be changed, and the sensitivity calculation is automatically adjusted to the new scheme. The additional density filter smooths the original structure, and closes small holes, up to the filter radius. Note that also a dilate or close filter (Sigmund, 2007) could have been used, but as the linear density filter already proves effective it is preferred here. A filter radius of $r = 0.07\,\text{m}$ was used, and the result is displayed in Figure 6.21b. This results resembles the structure of the original accessibility filter in Figures 6.18c and 6.18d, except with a hole in the bottom. A structure inside the cavity supports the top side, and no supports are required that are difficult to remove.

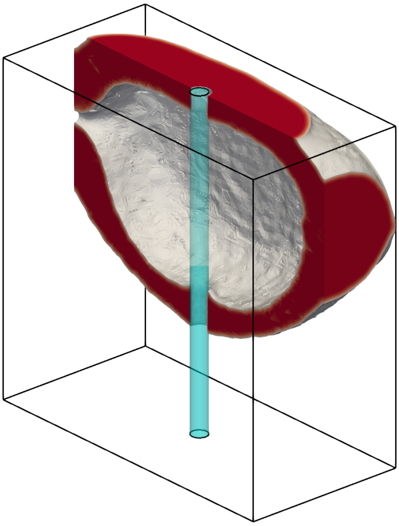
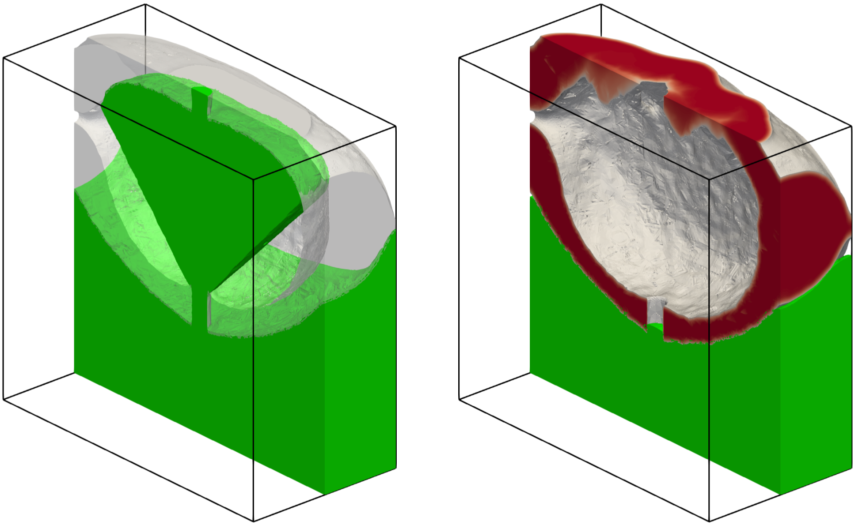


Figure 6.20: Optimized design for the torsion box case with accessibility filter, with a non-design region indicated in cyan.



(a) Support structure for normal accessibility filter.

(b) Layout and supports for modified accessibility filter.

Figure 6.21: With the conventional accessibility filter, supports can access cavities through small holes (a). By including an additional density filter, this can be prevented (b).

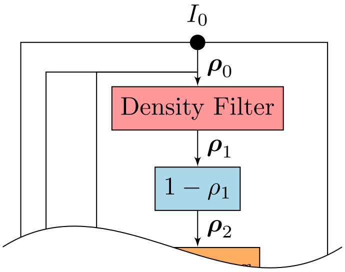


Figure 6.22: The modified accessibility filter with additional density filter as indicated by the red block. The remainder of the filter is similar to Figure 6.9.

CONVERGENCE

The convergence behaviour for the different optimizations of the torsion box case is plotted in Figure 6.23. Similar to the 2D case, the optimizations that include an overhang filter or accessibility filter show an increase in objective from iteration 10-20, as the filters are gradually activated with continuation. The exception is the accessibility filter with forced hole (Figure 6.20), which does not show an increase in objective. Because of the forced hole, the geometry is completely supported, and therefore activating the accessibility filter has no noticeable effect. Furthermore, it can be seen that the optimization with overhang filter has difficulties to converge due to the high degree of non-linearity as discussed. This might be prevented by choosing a different optimization algorithm such as globally convergent MMA (Svanberg, 2002), which adjusts the step size according to the degree of non-linearity.

The final objective values of the conventional, overhang filtered, and accessibility filtered optimization are 22.9 Nm, 30.1 Nm, and 23.2 Nm, respectively. This is in accordance to the amount of material in the final design that has to be used to make the structure printable, instead of improving the performance. That is none for the conventional case, the most for the overhang filtered case, and a small amount for the accessibility filtered case. The same holds for the optimizations with forced hole, which reach a final objective of 22.4 Nm and 22.9 Nm for the accessibility filter and modified accessibility filter, respectively.

Surprisingly, the accessibility filter with hole has a somewhat lower objective than the conventional optimization. This is caused by the fact that the two overhang filters present in the accessibility filter slightly modify the length scale, allowing sharper edges van de Ven *et al.* (2018). Consequently, there is less semi-dense material for the accessibility filtered case. This leads to a lower objective as semi-dense material has an unfavourable stiffness-mass ratio due to the penalization.

6.4. DISCUSSION

Both the 2D and the 3D results demonstrate the effectiveness of the accessibility filter in relaxing a strict minimum overhang angle constraint to a more realistic accessibility of support requirement. An overhang filter requires the structure to connect to some

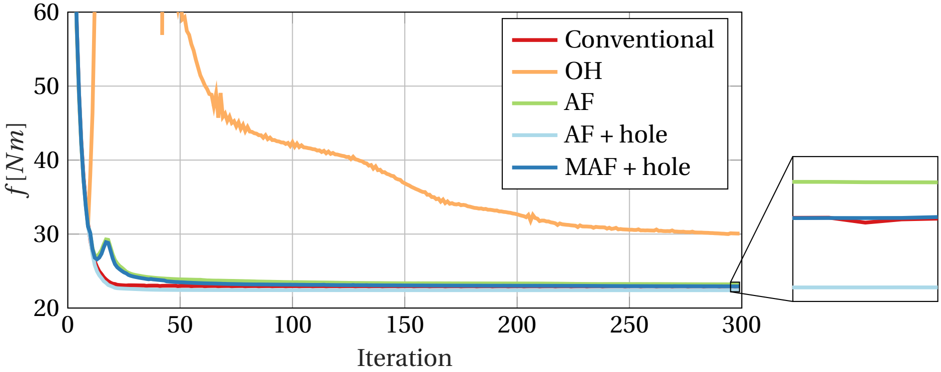


Figure 6.23: Convergence plots for the torsion box case for the conventional, overhang filter (OH), accessibility filter (AF), accessibility filter with force hole (AF + hole), and the modified accessibility filter with forced hole (MAF + hole).

predefined base-plate, which can have a large impact on performance. Although slightly exaggerated in the 3D case by placing the boundary conditions well above the base plate, a significant increase of performance is achieved when switching from overhang to accessibility filter.

Because the accessibility filter is composed of many individual filters, a filter structure was proposed that automates the filtering and sensitivity evaluation of any filter layout. This structure is much more widely applicable than just considering the accessibility filter case, and has already proved valuable for rapid prototyping of different filter configurations.

The numerical examples show that the relative impact on part performance of the accessibility filter is small, 6% for the 2D case and 1% for the 3D case, compared to 18% and 31% for the strict overhang filter. This is of course case specific, but the performance decrease is expected to be certainly lower than strict overhang constraints, as only inaccessible areas are affected. Furthermore, by inspecting different fields of the accessibility filter, one can get a good indication where supports are required. An optimization of the support structures inside the printable support domains identified by the accessibility filter could be complementary to the accessibility filter to generate an accurate support layout.

The accessibility of supports is currently assessed based on whether a support is connected to the base plate or not. With this definition of accessibility, it is possible to have a result with supports that are in practice difficult to remove, as shown by the last example in Figure 6.21a. However, this case has specifically been fabricated to showcase this possibility. In practice, it is unlikely that a hole is created in the wall of an enclosed region, as having a closed wall is a local optimum. The optimization algorithm can only exploit this configuration after a hole is already present. Furthermore, it was shown that this behavior can be avoided with a slight modification of the accessibility filter. However, it would be interesting to improve on this by incorporating a simulation of an actual tool removing the support, e.g. a type of milling constraint. This will result in a more accurate measure of accessibility.

Finally, similar to the strict overhang filter, when cavities are present, supporting structures are created inside the cavities to make them printable, as was shown in the 3D case. If the printing process can not manufacture cavities, because, e.g., powder has to be removed, a constraint to prevent the cavities can be included (Li *et al.*, 2016), or possibly the filter scheme can be adopted to detect internal cavities with front propagation.

6.5. Conclusion

In this work an accessibility filter for topology optimization is presented, and successfully demonstrated. The filter provides an optimized structure which is printable without support structures in regions where they are difficult to remove. In contrast to conventional overhang filters/constraints, which result in a structure that is printable without any support, supports are allowed if they are connected to the base plate. This resembles powder bed AM methods, where a part is never directly built onto the base plate. The accessibility filter is a composition of a number of regular filters, which is implemented in a novel modular filter structure. This arrangement makes it relatively easy to implement if an overhang filter is available. Future directions include to build on this idea and further refine the definition of support structures and their removability as required.

Acknowledgements

This research was carried out under project number S12.7.14549a in the framework of the Metals for Additive Manufacturing Partnership Program of the Materials innovation institute M2i (www.m2i.nl) and the Netherlands Organization for Scientific Research (www.nwo.nl).

References

(2017) The JSON Data Interchange Syntax. Standard ECMA-404, Ecma International, Geneva, CH

Aage N, Lazarov BS (2013) Parallel framework for topology optimization using the method of moving asymptotes. Structural and multidisciplinary optimization 47(4):493–505

Aage N, Andreassen E, Lazarov BS (2015) Topology optimization using PETSc: An easy-to-use, fully parallel, open source topology optimization framework. Structural and Multidisciplinary Optimization 51(3):565–572

Adam GA, Zimmer D (2014) Design for Additive Manufacturing—Element transitions and aggregated structures. CIRP Journal of Manufacturing Science and Technology 7(1):20–28

Ahrens J, Geveci B, Law C (2005) Paraview: An end-user tool for large-data visualization. In: Hansen CD, Johnson CR (eds) Visualization Handbook, Butterworth-Heinemann, Burlington, pp 717 – 731

Allaire G, Dapogny C, Estevez R, Faure A, Michailidis G (2017a) Structural optimization under overhang constraints imposed by additive manufacturing technologies. Journal of Computational Physics 351:295–328

Allaire G, Dapogny C, Faure A, Michailidis G (2017b) Shape optimization of a layer by layer mechanical constraint for additive manufacturing. Comptes Rendus Mathematique 355(6):699–717

Amir O, Mass Y (2018) Topology optimization for staged construction. Structural and Multidisciplinary Optimization 57(4):1679–1694

Balay S, Gropp WD, McInnes LC, Smith BF (1997) Efficient management of parallelism in object oriented numerical software libraries. In: Arge E, Bruaset AM, Langtangen HP (eds) Modern Software Tools in Scientific Computing, Birkhäuser Press, pp 163–202

Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Eijkhout V, Gropp WD, Kaushik D, Knepley MG, McInnes LC, Rupp K, Smith BF, Zampini S, Zhang H, Zhang H (2017) PETSc users manual. Tech. Rep. ANL-95/11 - Revision 3.8, Argonne National Laboratory

Bendsøe MP (1989) Optimal shape design as a material distribution problem. Structural Optimization 1(4):193–202

Bendsøe MP, Sigmund O (2004) Topology Optimization: Theory, Methods and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg

Bray T, Paoli J, Sperberg-McQueen C, Mailer E, Yergeau F (2008) Extensible Markup Language (XML) 1.0 5th Edition. W3C recommendation

Bruns TE, Tortorelli DA (2001) Topology optimization of non-linear elastic structures and compliant mechanisms. Computer Methods in Applied Mechanics and Engineering 190(26-27):3443–3459

Christensen P, Klarbring A (2008) An Introduction to Structural Optimization, Solid Mechanics and Its Applications, vol 153. Springer Netherlands, Dordrecht

Cloots M, Zumofen L, Spierings AB, Kirchheim A, Wegener K (2017) Approaches to minimize overhang angles of SLM parts. Rapid Prototyping Journal 23(2):362–369

Deaton JD, Grandhi RV (2014) A survey of structural and multidisciplinary continuum topology optimization: Post 2000. Structural and Multidisciplinary Optimization 49(1):1–38

Eschenauer HA, Olhoff N (2001) Topology optimization of continuum structures: A review. Applied Mechanics Reviews 54(4):331

Frazier WE (2014) Metal Additive Manufacturing: A Review. Journal of Materials Engineering and Performance 23(6):1917–1928

Gaynor AT, Guest JK (2016) Topology optimization considering overhang constraints: Eliminating sacrificial support material in additive manufacturing through design. Structural and Multidisciplinary Optimization 54(5):1157–1172

Gaynor AT, Johnson TE (2020) Eliminating occluded voids in additive manufacturing design via a projection-based topology optimization scheme. Additive Manufacturing 33:101,149

Geuzaine C, Remacle JF (2009) Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. International journal for numerical methods in engineering 79(11):1309–1331

Guo X, Zhou J, Zhang W, Du Z, Liu C, Liu Y (2017) Self-supporting structure design in additive manufacturing through explicit topology optimization. Computer Methods in Applied Mechanics and Engineering 323:27–63

Hague R, Mansour S, Saleh N (2003) Design opportunities with rapid manufacturing. Assembly Automation 23(4):346–356

Johnson TE, Gaynor AT (2018) Three-dimensional projection-based topology optimization for prescribed-angle self-supporting additively manufactured structures. Additive Manufacturing 24:667–686

Kahn AB (1962) Topological sorting of large networks. Communications of the ACM 5(11):558–562

Karypis G, Kumar V (1998) A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. Journal of Parallel and Distributed Computing 48:71–85

Kranz J, Herzog D, Emmelmann C (2015) Design guidelines for laser additive manufacturing of lightweight structures in TiAl6V4. Journal of Laser Applications 27(S1):S14,001

Langelaar M (2016a) Topology optimization for additive manufacturing with controllable support structure costs. In: Proceedings of the 7th European Congress on Computational Methods in Applied Sciences and Engineering, National Technical University of Athens (NTUA), Greece, pp 3689–3699

Langelaar M (2016b) Topology optimization of 3D self-supporting structures for additive manufacturing. Additive Manufacturing 12:60–70

Langelaar M (2017) An additive manufacturing filter for topology optimization of print-ready designs. Structural and Multidisciplinary Optimization 55(3):871–883

Langelaar M (2019) Integrated component-support topology optimization for additive manufacturing with post-machining. Rapid Prototyping Journal 25(2):255–265

Li Q, Chen W, Liu S, Tong L (2016) Structural topology optimization considering connectivity constraint. Structural and Multidisciplinary Optimization 54(4):971–984

Liu J, Gaynor AT, Chen S, Kang Z, Suresh K, Takezawa A, Li L, Kato J, Tang J, Wang CCL, Cheng L, Liang X, To AC (2018) Current and future trends in topology optimization for additive manufacturing. Structural and Multidisciplinary Optimization 57(6):2457–2483

Mirzendehdel AM, Suresh K (2016) Support structure constrained topology optimization for additive manufacturing. Computer-Aided Design 81:1–13

Qian X (2017) Undercut and overhang angle control in topology optimization: A density gradient based integral approach. International Journal for Numerical Methods in Engineering 111(3):247–272

Ranjan R, Ayas C, Langelaar M, van Keulen A (2018) Towards design for precision additive manufacturing: A simplified approach for detecting heat accumulation. In: Proceedings of the ASPE and EUSPEN Summer Topical Meeting, ASPE

Rozvany GIN, Zhou M, Birker T (1992) Generalized shape optimization without homogenization. Structural Optimization 4(3-4):250–252

Sigmund O (2007) Morphology-based black and white filters for topology optimization. Structural and Multidisciplinary Optimization 33(4-5):401–424

Sigmund O, Maute K (2013) Topology optimization approaches: A comparative review. Structural and Multidisciplinary Optimization 48(6):1031–1055

Sigmund O, Petersson J (1998) Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima. Structural Optimization 16(1):68–75

Sigmund O, Aage N, Andreassen E (2016) On the (non-)optimality of Michell structures. Structural and Multidisciplinary Optimization 54(2):361–373

Svanberg K (1987) The method of moving asymptotes—a new method for structural optimization. International Journal for Numerical Methods in Engineering 24(2):359–373

Svanberg K (2002) A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations. SIAM Journal on Optimization 12(2):555–573

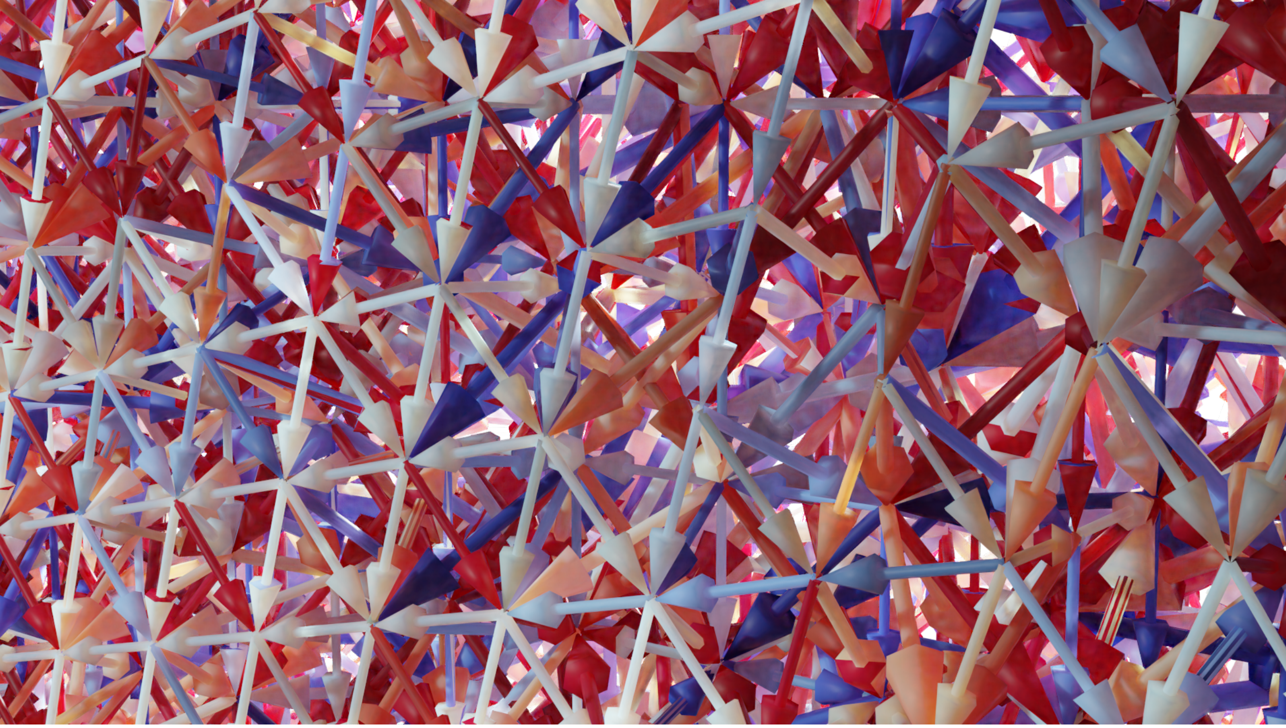
Thomas D (2009) The Development of Design Rules for Selective Laser Melting. PhD thesis, University of Wales

Thompson MK, Moroni G, Vaneker T, Fadel G, Campbell RI, Gibson I, Bernard A, Schulz J, Graf P, Ahuja B, Martina F (2016) Design for Additive Manufacturing: Trends, opportunities, considerations, and constraints. CIRP Annals 65(2):737–760

van de Ven E, Maas R, Ayas C, Langelaar M, van Keulen F (2018) Continuous front propagation-based overhang control for topology optimization with additive manufacturing. Structural and Multidisciplinary Optimization 57(5):2075–2091 van de Ven E, Maas R, Ayas C, Langelaar M, van Keulen F (2020) Overhang control based on front propagation in 3D topology optimization for additive manufacturing. Computer Methods in Applied Mechanics and Engineering 369:113,169

Wang D, Yang Y, Yi Z, Su X (2013) Research on the fabricating quality optimization of the overhanging surface in SLM process. The International Journal of Advanced Manufacturing Technology 65(9-12):1471–1484

Yasa E, Craeghs T, Badrossamay M (2009) Rapid Manufacturing Research at the Catholic University of Leuven. In: RapidTech 2009: US–Turkey Workshop on Rapid Technologies, pp 63–73



7
Conclusions

The goal of this thesis was to develop an overhang constraint for topology optimization to improve the integration of topology optimization and additive manufacturing. This is achieved by utilizing front propagation to mimic the printing process. This novel approach resulted in an overhang constraint applicable to many optimization problems, large unstructured 3D meshes, and capable of optimization for different printing processes, each with its unique limiting overhang angle. A large part of the work is dedicated to formulation and computational implementation of the overhang constraint in 2D and 3D and understanding its distinguishing features from overhang constraints in the literature. Additionally, an extension to the overhang constraint is presented that can take the accessibility of support material for later removal into account.

7.1. Front propagation in topology optimization

Front propagation is shown to be a suitable algorithm to incorporate into the topology optimization loop. Due to existing algorithms that solve front propagation problems with low computational complexity ($O(N\log N)$), it can be incorporated as a preprocessing step before the main FEA with a small additional computational cost. Despite the low computational cost involved, complex phenomena can be captured using front propagation, such as overhang control as shown in this thesis. For isotropic front propagation, implementations for structured and unstructured meshes in 2D and 3D are readily available. For anistropic front propagation, as is used for the overhang constraint, it can be difficult to construct an algorithm that progresses the front within a single element efficiently. This is an essential part of the front propagation algorithm which is the main driver of computational cost for complex propagation functions. This was one of the main challenges in the expansion of the overhang constraint algorithm from 2D to 3D, and was solved by using an analytical solution to update the front within each element.

The other challenge faced in the extension from 2D to 3D was parallelization of the front propagation algorithm. Front propagation algorithms are sequential by nature as the arrival time of a given node is evaluated from known values of neigbouring nodes. First of all, this makes it difficult to provide a well-performing algorithm in interpreted programming languages, which is why the front propagation algorithm provided as supplementary material with Chapter 5 is precompiled. Moreover, it makes it difficult to parallelize the algorithm, which is critical for large-scale problems. Parallelization algorithms for isotropic front propagation are available. In Chapter 3 we extend this to anisotropic front propagation which is shown to have a parallel scalability comparable to the FEA, and the computational cost remains an order of magnitude lower than for the parallel FEA solve, although for very large problems the scaling is to be investigated. Finally, a large advantage of the sequential nature of the algorithm is that it drastically simplifies the adjoint sensitivity calculation into a single loop over all the nodes, in opposite order as the forward solve of the front propagation.

Concluding, front propagation algorithms are very well suited for integration in topolgoy optimization, and while depending on the degree of anisotropy and scale of the problem the implementation can be challenging. It is expected that front propagation can be applied to address various other criteria besides overhang angles.

7.2. Front propagation-based overhang constraint

The overhang constraint developed in this thesis is based on solving a front propagation problem. This is shown to have several advantages. First of all, front propagation problems can be described by a partial differential equation. Consequently, the overhang constraint is formulated in the continuous domain, as opposed to some alternatives in the literature that are discretized by nature. The main advantage of having a continuous problem is that it makes the formulation discretization independent, and the same formulation can be used in 2D, 3D, on arbitrary mesh types and for arbitrary overhang angles. Furthermore, the continuous nature leads to insight into the method and can explain why some formulations only work on structured grids while others can be applied to unstructured grids, as shown in Chapter 5. Secondly, front propagation problems are well studied and efficient solution methods are available due to the sequential nature of front propagation problems. The sequential nature also drastically simplifies the adjoint sensitivity calculation but makes it difficult to parallelize the algorithm.

In Chapter 2 and Chapter 3 the overhang constraint is shown to perform well on general compliance minimization problems and problems that are especially difficult for overhang-constrained topology optimization. In Chapter 4 it is demonstrated that the front propagation based overhang constraint also performs well on fluid flow and compliant mechanism optimization problems. For all these problems, there is only one parameter that was changed which is directly related to mesh size. These problems did use a gradual activation of the overhang constraint over the first 20 iterations, except for the fluid flow case which uses a different optimization algorithm. However, in Chapter 5 an improvement was proposed that eliminates the requirement of continuation of the overhang constraint.

We thus conclude that the front propagation based overhang constraint satisfies the original research question: it can handle arbitrary meshes and overhang angles and large scale topology optimization without parameter tuning.

7.3. Accessibility filter

When the overhang constraint is applied, the resulting part can be printed without any support material. Often some support is acceptable as long as it can be removed after printing. This relaxation of the overhang constraint is achieved by the accessibility filter presented in Chapter 6. By a suitable arrangement of four Boolean operations on the geometry and two overhang filters, the overhang constraint is only activated in areas where support material is difficult to remove. Numerical examples show that the accessibility filter successfully relaxes the overhang constraint in 2D and 3D, increasing the performance compared to fully overhang constrained optimizations as less material has to be dedicated to support structures. Furthermore, in order to manage the filters, a framework is presented in which filters can be easily stacked in different configurations. This allowed for rapid prototyping of different filter configurations and is highly recommended for research codes as well as commercial software to provide flexibility to the end user.

7.4. Recommendations

7.4.1. Topology optimization and additive manufacturing

At the start of this study, there were only a few studies into overhang constraints for topology optimization. In the following years, the implementation of additive manufacturing constraints has become one of the most active research areas within the topology optimization community and currently there are numerous overhang constraint variants available. Therefore, follow up research concerning additive manufacturability should focus on more advanced overhang constraints. One example is the accessibility filter presented in this thesis. It would be interesting to have a more detailed formulation of support accessibility and removability, e.g. by testing removability with an actual tool shape of a milling machine. Furthermore, optimization with a physics-based printing simulation in the loop is still a challenging research domain with the appealing prospect of predicting deformations and stresses during and after printing.

7.4.2. Front propagation applications

Although this thesis set out to develop an overhang constraint for topology optimization, a significant part of the thesis is dedicated to efficiently and effectively incorporating front propagation into the topology optimization loop. Besides the overhang constraint, there are many more applications for front propagation in topology optimization. Especially with anisotropic front propagation, one can be creative in redefining problems to fit a front propagation formulation. Some possible applications include: maximum member thickness control, toolpath generation for 5-axis milling constraints, prevention of internal voids for additive manufacturing, separation of multiple fluid flows in a hydraulic manifold for flow optimization, optics design, and any other process that is naturally governed by front propagation.

TITLE IMAGES DESCRIPTION

Every once in a while the result from a numerical simulation is surprisingly beautiful. Whether it is caused by a bug, an underestimation of the exploitative nature of optimization algorithms, or simply by the complexity of the calculation, I had the habit of collecting these outliers in the “art” folder, as I imagine many people who run simulations do. In this thesis you will find these images on the title pages of the chapters, each related to the contents of the corresponding chapter. In the following I will briefly explain the nature of each picture.

- *Chapter 1*. Render of the result of a topology optimization in metal with supports that are required for printing in red. The supports are generated with a preprocessing tool for SLA printing, where branching supports are more common then for FDM printing. It was originally intended for a presentation to show the excessive amount of material required for the supports. The picture shows the starting point of this research: how to get rid of these supports?

- *Chapter 2*. The fourth iteration of a 2D cantilever beam optimization with overhang constraint, from the same simulation as shown in Fig. 2.15. The supports that enable material at the top of the domain are already quite solid (i.e. black), while the rest of the domain has intermediate densities. In combination with the colorscheme it almost looks like a bushfire or a colorful sunset.

- *Chapter 3*. Result of a 3D version of the tensile test case presented in Section 2.5.6. The domain is clamped on the left side while is horizontal force acts on a surface on the right side. The resulting structure appeared quite Gothic (Gothic architecture is simply a 3D printable version of Romanesque architecture). It was not used in the paper because a better 3D representation of the tensile case was found (see Section 3.4.2). The high gloss red plastic material seemed fitting for this artifact.

- *Chapter 4*. Three states of a gripper mechanism similar to the one presented in Section 4.2. This version had a particular organic vibe, perhaps because of the holes in the top reminiscent of the eyes of some creature. Therefore, the mechanisms are rendered with a fleshy material, in three states of closing of the “beak”.

- *Chapter 5*. Front propagation through an optimized cantilever beam in 2D. In the background the topology is visualized (white = material, gray = void), while the arrows point in the direction from which the front was incoming. Each material node that is overhanging, adds some value to the stream. While following the arrows, the redder the arrow, the more “overhang” is accumulated along that path. This is a visualization of the adjoint used for debugging, but it turned out to be also aesthetically pleasing.

- *Chapter 6*. Snapshot of a topology optimization web implementation (www.emielvandeven.nl/topopt). One subtopic of Chapter 6 is on how to implement topology optimization in a structured, object-oriented manner. This is a bit overshadowed by the main accessibility of supports topic. I made a online demonstration of the principle such that everyone can experiment with it, and this title image is a subtle attempt to highlight the topic.

- *Chapter 7*. Front propagation visualization of the 3D crane hook case (Section 3.4.3). This is a 3D version of the title image of Chapter 5. Debugging parallelized code is a nightmare, as the bugs often only occurred on extremely large cases under very rare circumstances. For the parallel implementation of the front propagation algorithm, I faced many problems where nodes were updated by multiple processors simultaneously (i.e. race conditions), sometimes leading to situations where two nodes are dependent on each other. These arrow visualizations helped in the understanding of that problem, and in this very image one can find such a bug where two arrows are pointing towards each other. Coincidentally the combined arrows look like Where's Wally's shirt. This picture seemed fitting for the conclusions, as one needs to follow all the arrows in the right direction to reach towards the end (or find a bug).

Curriculum Vitae

Emiel van de Ven was born on April 24th, 1991 in Grubbenvorst. After graduation from secondary school in 2009 (*Dendron College*, Horst), he moved to Delft to study Mechanical Engineering at the *Delft University of Technology*. He completed his BSc degree in 2012 (cum laude), and his MSc degree in 2014 (cum laude) with the thesis titled "*Topology Optimization for Transient-Thermal-Mechanical Systems*". In 2011, together with Tan Lai Zhou, Emiel appeared on the Dutch television game show 'Lingo', but their earnings were not sufficient to cover their pension. So, in order to make a living, they both pursued their doctorate[1]. In 2015, Emiel started his PhD research on topology optimization and additive manufacturing in the department of Structural Optimization and Mechanics at the *Delft University of Technology*, and in the Collaborative Engineering Systems department at the *Royal Netherlands Aerospace Centre* (NLR). When his PhD was close to but not quite finished in 2019, Emiel started working as a thermal-mechanical analyst at the opto-mechatronics department for the *Netherlands Organization for Applied Scientific Research* (TNO). As of 2021, Emiel is working in the Flow-Thermal-Tribology research department at ASML.

[1] Zhou, TL. Blood Pressure Variability: causes and consequences [dissertation]. Maastricht; 2020.

List of Publications

Journal papers

van de Ven E., Maas R., Ayas C., Langelaar M., van Keulen F. (2021) *Overhang control in topology optimization: a comparison of continuous front propagation-based and discrete layer-by-layer overhang control.* Structural and Multidisciplinary Optimization 64, 761-778.

van de Ven E., Ayas C., Langelaar M., Maas R., van Keulen F. (2020) *Accessibility of support structures in topology optimization for additive manufacturing.* International Journal for Numerical Methods in Engineering.

van de Ven E., Maas R., Ayas C., Langelaar M., van Keulen F. (2020) *Overhang control based on front propagation in 3D topology optimization for additive manufacturing.* Computer Methods in Applied Mechanics and Engineering. 369, 113169.

van de Ven E., Maas R., Ayas C., Langelaar M., van Keulen F. (2018) *Continuous front propagation-based overhang control for topology optimization with additive manufacturing.* Structural and Multidisciplinary Optimization 57, 2075–2091.

Misiun G., **van de Ven E.**, Langelaar M., van Keulen F., van den Boogaard T., Ayas C. (2020) *Topology Optimization for Additive Manufacturing with Distortion Constraint.* Manuscript submitted for publication.

Conference proceedings

van de Ven E., Ayas C., Langelaar M., Maas R., van Keulen F. (2018) *Topology optimization for additive manufacturing: fully printable compliant mechanisms.* In: 2018 ASPE and euspen Summer Topical Meeting: Advancing Precision in Additive Manufacturing (pp. 40-44). American Society for Precision Engineering.

van de Ven E., Ayas C., Langelaar M., Maas R., van Keulen F. (2018) *A PDE-Based Approach to Constrain the Minimum Overhang Angle in Topology Optimization for Additive Manufacturing.* In: Schumacher A., Vietor T., Fiebig S., Bletzinger KU., Maute K. (Eds.) Advances in Structural and Multidisciplinary Optimization. WCSMO 2017. Springer, Cham.

van de Ven E., Hooijkamp E., Langelaar M., van Keulen F. (2015) *Topology Optimization of a Transient Thermo-Mechanical Problem using Material Penalization.* In: Qing Li, Grant P. Steven and Zhongpu (Leo) Zhang (Eds.) Advances in Structural and Multidisciplinary Optimization - Proceedings of the 11th World Congress on Structural and Multidisciplinary Optimization, page 966-971, International Society for Structural and Multidisciplinary Optimization (ISSMO), Sydney, Australia.

Book chapter

van de Ven E., Ayas C., Langelaar M. (in press). *Computational design of complex 3D printed objects.* In: Rubio A., Esposito V. (Eds.) 3D Printing for Energy Systems.

Acknowledgements

As for everyone, my PhD had its ups and its downs. However, I do recommend anyone who considers pursuing a PhD to do so. It is a rewarding experience which, overall, I thoroughly enjoyed. That is for a large part thanks to the people I met during my PhD, whom I like to thank here.

First of all the supervisory team from the TU Delft: Matthijs Langelaar, Can Ayas, and Fred van Keulen. You gave me all the freedom to explore tangent ideas and chew on problems extensively. Thank you for creating such an open environment, for your valuable input during our weekly discussions, and for gently pushing me towards the finish line. I am sure our paths will cross many times in the future. I am also grateful to the supporting staff of PME for helping out with all sorts of requests.

Next to the TU Delft I spent a mayor amount of my time at the NLR in the collaborative engineering (AVCE) group, where I was supervised by Robert Maas. Robert, your mathematical background was a welcome addition to the project, and I was always impressed how little explanation was required for you to understand the problems I faced, thank you for your support. I am also grateful to Tony ten Dam for enabling this project and for making sure I became a member of the NLR community. I thank my NLR colleagues for making me feel welcome and connected to the NLR, and for the nice lunch activities: walks to the Nieuwe Meersluis, running in the Vondelpark, and especially rehearsing with the NLR-jazz-band.

As one might imagine, this thesis is written on a strong foundation of coffee. A few months into my PhD, the TU Delft switched from okayish to terrible coffee, and in that very same week the NLR switched from kinda bad to very acceptable coffee. The latter was much appreciated by me, and also a nice demonstration of the law of conservation of misery. But most of all, coffee facilitated interaction. Because the coffee at the university was so bad (really, it made national press three times), I had to resort to the coffee in the Aula. During my PhD I've had several coffee buddies with whom I went to the Aula regularly, and those trips turned out to be quite cathartic. So thank you fellow coffee refugees for the discussions, advice, and listening to my complaints.

From coffee it is only a small step to the beers. With my friends from back in Limburg and from Delft, and the PhD colleagues we had many social events, parties, terraces, conference dinners, Papendal symposia (the breakfast is a myth), intense arguments, bouldering sessions, weekend trips, kerstdiners. Thank you to all my friends and PhD colleagues for your warmth and support.

Besides food for the body, there is also music, food for the mind. For me each chapter has its own soundtrack from a specific period of my PhD. For example, Kamasi Washington's Harmony of Difference puts me right back in my hotel room in Bologna, running cases late at night for my presentation the next day (resulting in Chapter 5). But even more than listening, I enjoy the playing. I want to thank everyone with whom I have played for the gezelligheid during the rehearsals, and memorable performances at parties. Muchacho's de bakbici, Lacassette, the NLR-jazz-band, Capital-S, and all the impromptu bands, thank you for keeping me behind the piano and for being my creative outlet.

Finally, I want to thank my family and Sarah's family for all their love and support. Sarah, I was probably not the most pleasant company when I had a bug in my code or a deadline to make. Thank you for putting up with me, for your comfort and support, and for keeping my life balanced. Joke, Frans, Marijn, and Rinus, together we are quite a colorful bunch and I would not want it any other way. As each year I am already looking forward to our week at Terschelling, now for the first time in years without a thesis to work on!