Topology optimization for Additive Manufacturing

- ***Computing and constraining the overhang angle***

Sebastián Avelino Gómez Sánchez

sebgo787

971021 – T112

Linköping University – Sweden

Spring term 2018-2019

ISRN: LIU-IEI-TEK-G--19/01690--SE

LINKÖPINGS UNIVERSITET

Table of Contents

Table of Contents 2

Abstract 4

Introduction 5

Definition of structural optimization 5

Manufacturability: The Motivation of this Work 6

Additive Manufacturing 6

3D Printing with polymers 7

Topology Optimization with FEM 8

SIMP Penalization 10

Solving the Optimization Problem 11

Detecting Overhang 11

Boundary detection routine 12

Pre-processing the image 15

Computing the Overhanging Angle and comparing with the Printing Angle 18

Implementation of the constraint in a Topology Optimization code 20

Mathematical formulation of the problem 20

Objective Function 21

Volume Constraint 21

Overhang Constraint 22

Sensitivity Analysis 24

Sensitivity of the Objective Function 24

Sensitivity of the Volume Constraint 24

Sensitivity of the Overhang Constraint 25

Results of testing the implementation 26

Cantilever beam example 26

Concluding Remarks 27

Acknowledge 28

References 29

Abstract

The objective of this work is to implement a constraint in a topology optimization problem, in order to produce designs that are ready to be 3D printed without the need of supportive structures. There are several motivations to study this, some of the most important ones are to make the 3D printing process shorter and to avoid the need of post printing processes like removing the supportive structure. These together make the whole process more economical. The basis of the project are [1] and [2], and this work will try to replicate their findings.

In order to prevent future researchers to commit the same mistakes as we did in the beginning and save them time, we will share some of the main problems that were found during the research process. That is why we have divided the research in two phases: phase one, just trying to detect the boundary elements of fixed final designs, and phase two, implementing the overhang constraint in the topology optimization code.

Introduction

Definition of structural optimization

Structural optimization can be defined as the discipline that tries to make structures "the best" under certain conditions. A structure can be any part that resist loads. The "best" can have different meanings, depending on the situation. So, it could be the smallest structure, the lightest one or basically any criteria that we can think of [3]. In this work, we are going to study the optimization problem that tries to obtain the stiffest structure under a limited mass or volume.

There are three types of structural optimization: sizing optimization, shape optimization and topology optimization. Sizing optimization is focused on changing the dimensions of the structure [3]. An example of this is to change the cross-sectional areas of trusses, making them bigger or smaller. Shape optimization pursues the definition of the shape that solves the problem in the most satisfactory way [3]. A good example of this may be to define the shape of a beam to resist a specific group of loads (Figure 1).

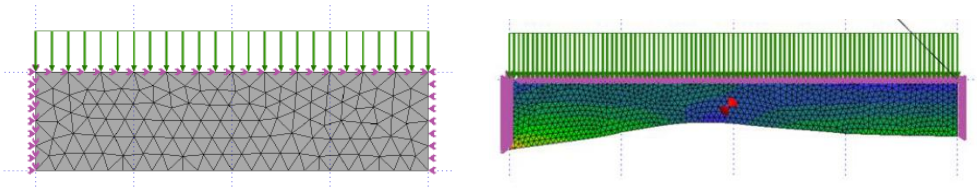

Figure 1: An example of a beam under a distributed load (left) and the optimal shape of the beam (right) if the lower boundary can be varied.

Topology optimization tries to find the best way to place material in a certain area [3] (Figure 2). This project is focused on topology optimization.

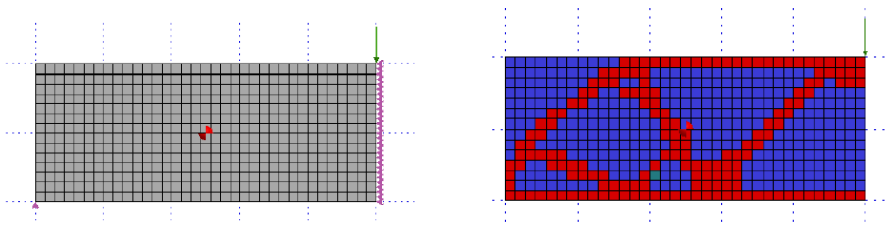

Figure 2: Example of topology optimization, a beam with a force in the upper right corner (left) and the topology optimization result of minimizing compliance with a volume constraint (right).

Manufacturability: The Motivation of this Work

Manufacturability is one of the most important aspects about any design. There is no reason to get a design that is impossible or very uneconomical to materialize. But, it is common to get designs with complex geometry from topology optimization. Despite being better, in most of the cases it is not reasonable to build them with the classical manufacturing procedures [4].

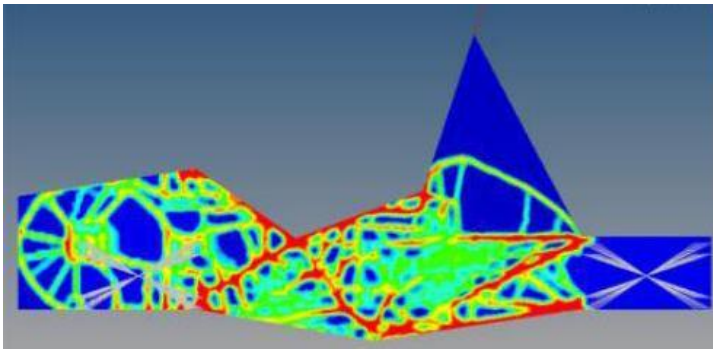

Figure 3: A topology optimization of an electric racing car. We can see that the structure has a very complicated geometry that may be very difficult to replicate [5].

Nowadays, additive manufacturing has been developed and complex geometries are not as difficult to create as thirty years ago. We can convert topology optimized designs to a CAD file and print them with different techniques.

Additive Manufacturing

Additive manufacturing is a fabrication technology that began to be developed in the 1960s at Battelle Memorial Institute. In the past 30 years it has become popular and has evolved in different techniques so far. The first one was stereolithography, based on solidifying a photo-sensible liquid. We have now more advanced techniques like 3D printing designed for polymers; Selective Laser Sintering, suitable for metals, polymers, glass and ceramics; Fused Deposition Modelling and Laminated Object Manufacturing [4]. In this work we will focus on the 3D printing method.

3D Printing with polymers

The technique is simple: it glues small amounts of material together, materializing the design layer by layer (Figure 4). It has some advantages, like being able to print a wide range of polymers, being fast and having a relatively low cost [4].

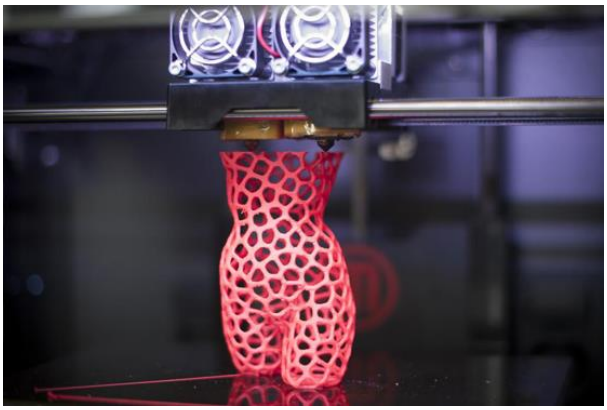

Figure 4: A 3D-printing machine working [6].

It also has some disadvantages, like a rough surface finish and a size limitation [4], and physical restrictions. One of the most important physical restrictions is the overhang angle [1]. The 3D printing process may need to place additional material in the printing process, as supports, in order to print down-facing surfaces with a low angle relative to the base (Figure 5). Typically, there is a threshold angle of 45º, and if we want to print a surface below that angle, supportive structures must be printed too [1].

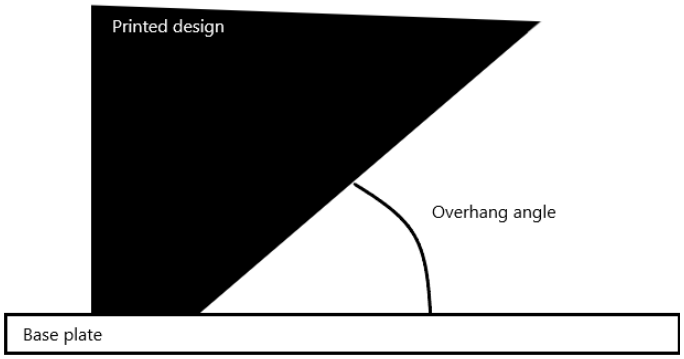

Figure 5: Overhang angle that is restricted.

So, we may be able to print any part, but also need to place the supportive material. That can be very costly, consuming time and material [1]. It also introduces some other problems, like a different surface quality between surfaces directly printed and surfaces obtained after removing

the supportive material. Thus, we would like to avoid the use of supportive structures whenever it is possible.

To solve this problem, there have been different approaches, like for example the optimization of these supports so that they consume the least material and time possible [7]. Other solutions involve a post-processing of the designs obtained from topology optimization before the printing stage. In this work, we will take advantage of the main characteristic of structural optimization, the fact that we can add conditions to the outcome designs. Therefore, we will merge design and manufacturability, establishing a condition to meet the 3D printing requirements and therefore avoiding supportive structures. We will call it the overhang constraint, and follow the approach suggested by Garaigordobil et al. [1] and [2], with an adaptation of the SUSAN technique (Smallest Univalue Segment Assimilation Nucleus). This is a reliable way to detect image boundaries in the field of digital image processing. The adaptation suggested in the paper [1] will be adopted, that uses 3x3 elements masks to scan the design, and replaces the pixel intensity with the density value.

Topology Optimization with FEM

We will now introduce the basic concepts about topology optimization with a computer. One of the most important ones is the Finite Element Method (FEM). It is a tool to solve the topology optimization problem of distributed parameter systems (systems with an infinite number of variables). Since this kind of problems are not suitable to be solved with a computer directly, we need to transform them into problems with a finite number of variables, and this is done using FEM [3]. In this work, we are going to solve 2D problems in a rectangular or squared design domain.

The material distribution in the design domain Ω is defined by the density function ρ. The density function must take values from $\underline{\rho}$ to $\overline{\rho}$ (lower and upper bounds). The set of admissible designs is

$$H = \left\{ \underline{\rho} \leq \rho \leq \overline{\rho} \quad \text{for all x} \in \Omega, \int_{\Omega} \rho \, d\,\text{v} = \text{V} \leq \gamma V_0 \right\}, \tag{1}$$

where V_0 is the maximum volume, calculated when ρ=1 for all the elements and γ is a scaling parameter between zero and one. Typically, the values of the lower and upper bounds are zero

and one, zero indicating that there is no material in that element, and one indicating that there is material. Zero drives to numerical instabilities and it is usually substituted by a low positive value that is interpreted as no material, but in this work we will redefine the Young modulus to solve this problem, as described in the implementation section below, equation (20).

The most popular topology optimization problem is minimizing compliance, a property that is inversely related to stiffness, under a volume constraint. The problem can be written as:

$$\begin{cases} \min_{u,\rho} l(\boldsymbol{u}) \\ s.t. \begin{cases} \boldsymbol{u} \in \boldsymbol{K} \; such \; that \; a(\rho, \boldsymbol{u}, \boldsymbol{v}) = l(\boldsymbol{v}) \; for \; all \; \boldsymbol{u} \in \boldsymbol{K} \\ \rho \in H \end{cases} \end{cases} \tag{2}$$

where $l(\boldsymbol{u})$ is the compliance, and

$$a(\rho, \boldsymbol{u}, \boldsymbol{v}) = \int_{\Omega} (\partial \boldsymbol{u})^T \, \rho \boldsymbol{D} \partial \boldsymbol{v} \, dA. \tag{3}$$

In our case, the 2D problem, the displacement vector $\overline{\boldsymbol{u}} = (u, v)$ and

$$\partial \boldsymbol{u} = \begin{pmatrix} \frac{\partial u}{\partial x} & 0 \\ 0 & \frac{\partial v}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial u}{\partial y} \end{pmatrix}, \tag{4}$$

and assuming an isotropic material in plane stress,

$$\boldsymbol{D} = \frac{E}{1 - \nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1 - \nu}{2} \end{bmatrix}, \tag{5}$$

where ν is Poisson's modulus and E is Young's modulus. The compliance is

$$l(\mathbf{u}) = \int_{\Gamma_t} \mathbf{u}^T \, \boldsymbol{t} \, ds. \tag{6}$$

Here, $\mathbf{t}$ represents the external boundary forces. The integration goes for the part of the boundary that is not fixed (Γ_t). The admissible displacements belong to the set

$$\boldsymbol{K} = \{\boldsymbol{v} : \Omega \rightarrow R^2 \mid \boldsymbol{v} = \mathbf{0} \; on \; \boldsymbol{\Gamma_u}\}, \tag{7}$$

which states that the displacement vector on $\boldsymbol{\Gamma_u}$, the part of the boundary of Ω that is fixed, must be zero.

We now divide the design domain into nel finite elements to formulate the discretized problem

$$(P_a)\begin{cases}\min\limits_{u,x} \boldsymbol{F^T u} \\ s.t.\begin{cases}\boldsymbol{K(x)u = F} \\ \boldsymbol{x^T a} = V \\ \underline{\rho} \le x_e \le \overline{\rho}, e = 1,..,nel\end{cases}\end{cases} \tag{8}$$

where the stiffness matrix **K** is given by

$$\boldsymbol{K(x)} = \sum_{e=1}^{nel} x_e \boldsymbol{K_e^0}. \tag{9}$$

Here x_e is the approximate value of ρ on element e, assuming a constant value of ρ in each finite element. Furthermore, $\boldsymbol{K_e^0} = \boldsymbol{C_e^T k_e^0 C_e}$ where $\boldsymbol{C_e}$is a matrix that is changing local degrees of freedom for global degrees of freedom. The $\boldsymbol{k_e^0}$ matrix is the element stiffness matrix when ρ = 1 [3].

SIMP Penalization

In the 2D case, the density function can be interpreted as the thickness of the element. That is not possible in 3D cases, and therefore we like to have values of ρ that are closer to one or zero. In order to get that, we will penalize the intermediate values of ρ. We will do it using the SIMP method. The Solid Isotropic Material with Penalization (SIMP) simply changes the value of the effective Young modulus, by changing the definition of the D matrix, introducing a new term multiplying E [3]. With SIMP,

$$\boldsymbol{D} = \frac{\rho^q E}{1-\nu^2}\begin{bmatrix}1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}, \tag{10}$$

being q a constant, generally between 1 and 3. This simple change makes uneconomical the selection of intermediate density values [3]. The effective Young's modulus will be now $\rho^q E$.

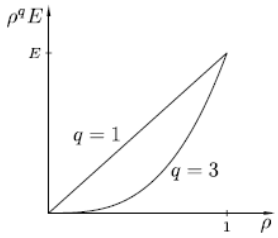

Figure 6: Graph representing the relation of E and ρ for different values of q [1].

As we can see in Figure 6, with q = 1 we get a linear relation between stiffness and density. When we change the value of q, to a bigger value, the relation becomes non-linear. Doing this, elements with intermediate density values will add volume to the design without a big increment on the stiffness. Therefore, the program will prefer elements with density one or zero rather than intermediate values [3]. This method will work when minimizing compliance under a volume constraint, but might not produce black-and-white solutions to other problems.

There are other techniques of penalizing the intermediate values, such as the RAMP method, but in this project we will be using SIMP. SIMP requires regularization to avoid checkerboards and mesh dependency [3], and the formulation for that will be directly introduced in the implementation section below (so that it will be clearer to understand later). Checkerboards are regions of alternative black and white elements, ordered like a chessboard. They are caused by a bad numerical modelling of stiffness in this material distribution. Mesh dependency means that when solving the problem with a finer mesh, instead of getting a more detailed figure of the same solution as with a bigger mesh, we get a different solution. It can happen because there does not exist a solution to converge to [8].

Solving the Optimization Problem

We are using Matlab fmincon function, which implements an interior point method for nonlinear optimization problems. In future developments, it is suggested to implement MMA, as done in [1] and [2].

Detecting Overhang

In this section, we explain how every design is processed, to know whether they are suitable or not for 3D-Printing without supportive material. We will solve this for any arbitrary 2D rectangular domain, in which the parameters Lx and Ly are the dimensions of the design domain in the x and y axis respectively, and the number of elements in the x and y axis are nelx and nely.

The investigations and tests performed to achieve a successful implementation have been divided in two phases, each one took approximately half of the working time. In the first phase, the tests were made over a fixed design, outside the topology optimization code, and was intended to understand how the boundary of a design is detected. In the second phase, we started the implementation on the topology optimization code. This is very important to understand that we will now talk about differences between the initial approach and the implemented approach, due to factors that were not able to consider, or did not affected outside the topology optimization procedure, but caused problems later, like smoothness of some functions approximations, and parameters that were initially not limited in phase one but they were too big for the second phase and had to be reduced.

Boundary detection routine

The first thing we need to do is to identify the boundaries of our design. For that purpose, the entire design is scanned with a 3x3 elements mask, as we can see in Figure 7 [1]. Notice that, in the Figure 7, there are different grey, black and white elements. Black represents ρ = 1 and white represents ρ = 0. The intermediate values are represented with lighter and darker greys.

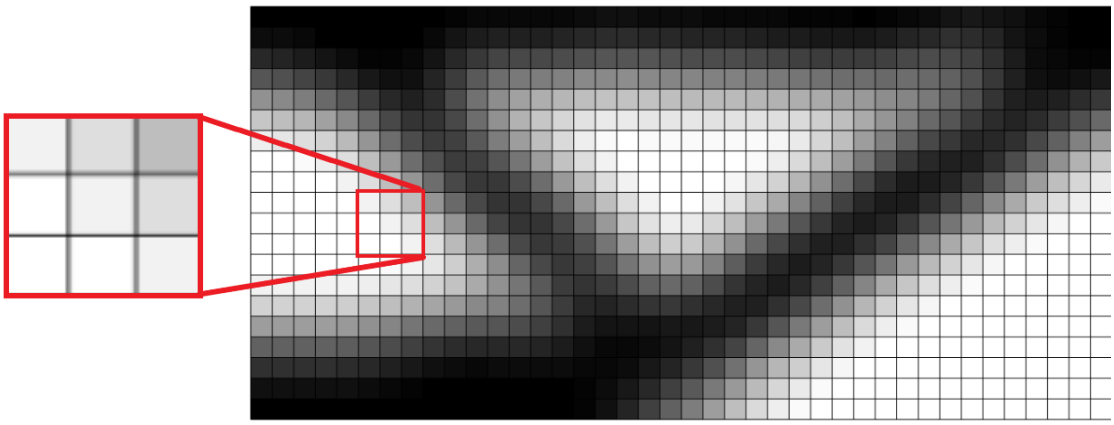

Figure 7: 3x3 Elements mask

We calculate the mass centre of the mask, using the basic equations of mechanics,

$$x_{cg} = \frac{\sum_e \widetilde{\rho_e} x_{ce}}{\sum_e \widetilde{\rho_e}}; \tag{11}$$

$$y_{cg} = \frac{\sum_e \widetilde{\rho_e} y_{ce}}{\sum_e \widetilde{\rho_e}}; \tag{12}$$

where the sum is over the elements that belongs to the mask, and the variable $\tilde{\rho}$ is the density function after pre-processing the image. The pre-processing is done mainly to solve the problem with raw designs that, with a lot of different grey elements with intermediate ρ values, we cannot identify the boundary because there is not a clear boundary to identify. The solution for that suggested in [1] is pre-processing the image, forcing all the elements to be black or white (so, have $\rho = 0$ or $\rho = 1$). That method is explained in detail in the following section.

Note that special considerations must be made for the boundary elements (first and last columns, uppest and lowest rows), as they cannot be surrounded by a 3x3 mask. In the phase one of this project, we used special masks of 3x2 or 2x3 in the boundary elements of the design domain and 4x4 in the corners, and thus modified the mask for the boundary elements. The Figure 8 was developed with this first approach. Nevertheless, in the development of the implementation step (phase two), this was changed and it was decided to follow the approach purposed in [1]. Therefore, there were assumed one extra column and row of elements surrounding the design region, with fixed $\tilde{\rho}$ values (zero all of them except the lowest extra row that will have value one, representing the printing base plate). This makes it possible to have a 3x3 mask in all the elements of the design domain, and has given better results.

The following step is to calculate the vector that goes from the geometrical centre $\boldsymbol{c_c}$ of the mask, to the mass centre of the mask $\boldsymbol{c_{cg}}$, the vector $\boldsymbol{v} = \boldsymbol{c_{cg}} - \boldsymbol{c_c}$ for each mask [1]. The **v** vectors are represented in Figure 8.

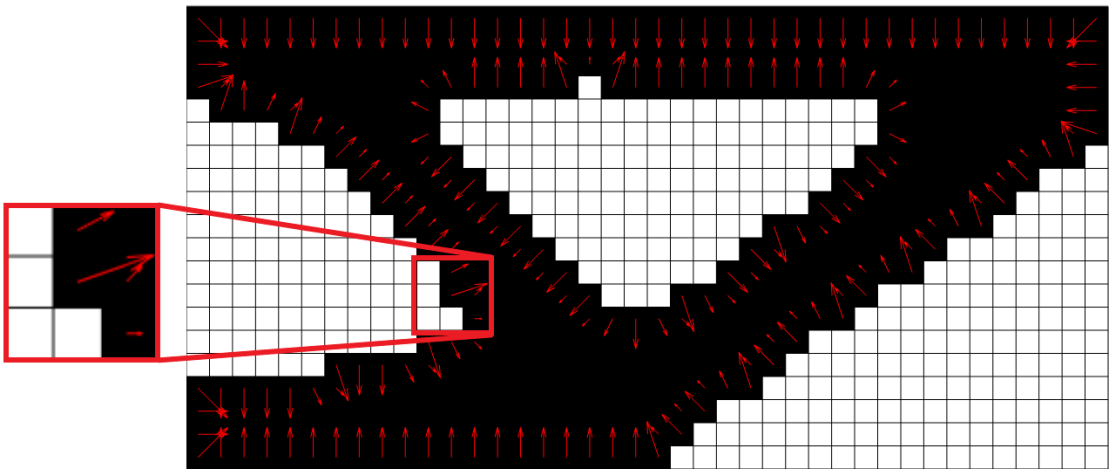

Figure 8: We can see the big red arrow, that represents the v vector assigned to the central element. It goes from the geometrical centre of the 3x3 mask, that is also the geometrical centre of the element, to the mass centre of that mask.

This vector's length was used to measure the quality of each element as a boundary. We started in the phase one by considering as boundary elements those that fulfil two basic conditions:

- The elements must have a $\widetilde{\rho_e} \geq \underline{\rho}$ to be considered boundary. This criterion is motivated because an element that has no material cannot be a boundary.
- The elements must have a value of $\|\boldsymbol{v}\| = \sqrt{v_x^2 + v_y^2}$ bigger than a certain value ε. In the boundaries, the geometrical centre and the mass centre are not the same point, but in areas full of material or without material they do. Thus, if $\|\boldsymbol{v}\|$ is zero, we know that it is not a boundary element.

The value of ε was set to the minimum value that v can have in a boundary element. This minimum value was found to be when the 3x3 mask has only one empty element, as shown in Figure 9. So, ε is calculated for the rectangular domain as:

$$\varepsilon = \begin{cases} \frac{\mathrm{Lx}}{8*\mathrm{nelx}}, & \text{if} \quad \frac{\mathrm{Lx}}{8*\mathrm{nelx}} < \frac{\mathrm{Ly}}{8*\mathrm{nely}} \\ \frac{\mathrm{Ly}}{8*\mathrm{nely}}, & \text{if} \quad \frac{\mathrm{Lx}}{8*\mathrm{nelx}} \geq \frac{\mathrm{Ly}}{8*\mathrm{nely}} \end{cases}; \tag{13}$$

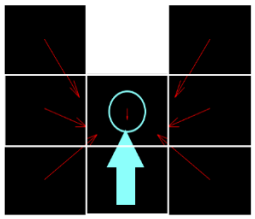

Figure 9: Boundary element 3x3 mask; case where $\|v\|$ is minimum

With these two simple criteria we can detect the boundaries of different designs. As an example, we present the following picture (Figure 10), where the boundaries are identified in white. Note that the left picture was taken before pre-processing the image, and the right picture represents the boundaries of the already-processed figure. In Figure 11 we can see a picture of Figure 10 left, after the pre-processing.

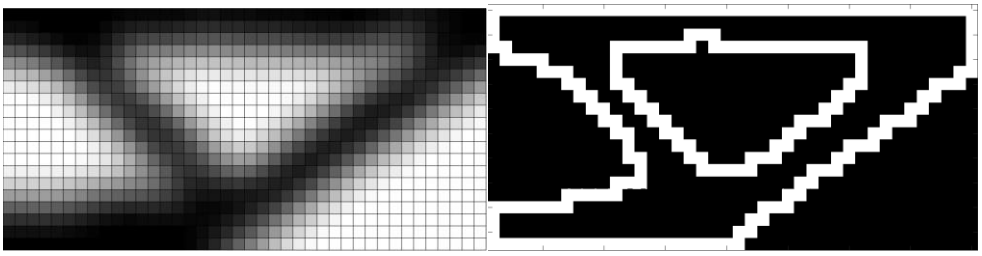

Figure 10: Boundary detection following the described method. Left, the raw design that comes from the topology optimization. Right, the boundary detected after pre-processing the image.

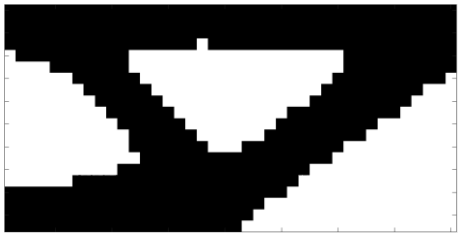

Figure 11: Design of Figure 10 after the processing, before the boundary detection.

This two-criteria approach looked promising in phase one, but later in phase two there were difficulties to detect boundaries of intermediate designs and a different approach was adopted. The simplest way to motivate this change is that, when projecting the density function value to one or zero, the heaviside approximation function becomes too "peaky" to be continuous and differentiable, and therefore a good projection cannot be achieved for all the iterative designs until reaching the final one. With a non-black and white design, these two criteria explained above loose coherency with the process. We will therefore see another way to detect overhang angles and constraint them, by classifying well supported and not well supported elements instead of looking just for the boundary element angles. It will be explained while defining the overhang constraint in the implementation chapter of this work.

Pre-processing the image

In phase one, we thought that one could get a black and white design for every iteration of the optimization process. Therefore, we needed to implement a pre-processing method. In phase two, we do not assume black and white designs in every iteration, but we will still use this pre-processing method, as recommended in [1], so we will explain the method.

As we said in the previous section, in phase one we required an almost black and white design in order to identify the boundary. This was intended to solve one problem: that a grey scale transition between smaller and bigger densities in the design makes it impossible, in a general case, to get a design boundary. This problem is shown in the following figures (Figure 12), where the boundary detection routine tries to detect the design boundary without pre-processing the image, and fails because there are more density values than zero and one:

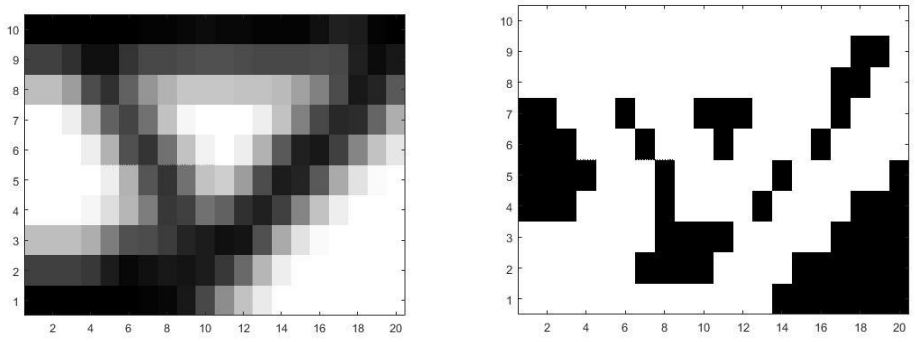

Figure 12: Image without pre-processing (left) and boundary detected in white colour (right). We can see how the method fails to detect the boundaries without pre-processing the design.

So, we can clearly see how some not boundary elements are detected as if they were. This happens because the relative mass distribution in the 3x3 mask with the centre in light grey elements that are true boundary elements in Figure 12, is similar to the mass distribution in the mask with the centre in medium dark grey.

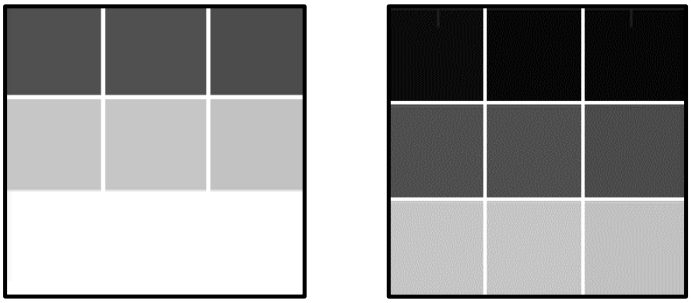

Figure 13: Left, 3x3 mask centred in a light grey element. Right, 3x3 mask centred in a darker grey element. In both cases, the mass distribution is approximately the same, and their v vector will be similar. This fact confuses the boundary detection procedure.

Therefore, we pre-process the image. Thus, we establish a threshold value of ρ, and use a continuous approximation of the heaviside function to make the values of ρ over that value one, and the values under the threshold zero. So, we define a new density function, $\tilde{\rho}$, using the density function that we get from the topology optimization code, ρ. Our continuous approximation is

$$\widetilde{\rho_e} = \frac{\tanh(\beta * T) + \tanh(\beta * (\rho_e - T))}{\tanh(\beta * T) + \tanh(\beta * (1 - T))} \quad [1]. \tag{14}$$

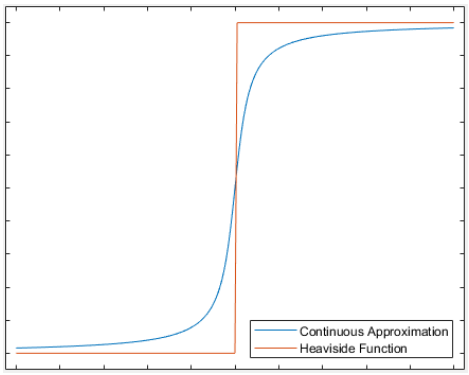

Figure 14: Graphical description of the heaviside function and the continuous approximation.

Where β is a scaling parameter controlling the steepness of the continuous approximation and T is the threshold parameter of the Heaviside function. When we make the design more black and white, the mass distribution is tremendously different between design boundary element's centred 3x3 masks and elsewhere centred masks. This property is used to find the boundary elements. We can see an example of a boundary detection when the design is pre-processed in Figure 15.

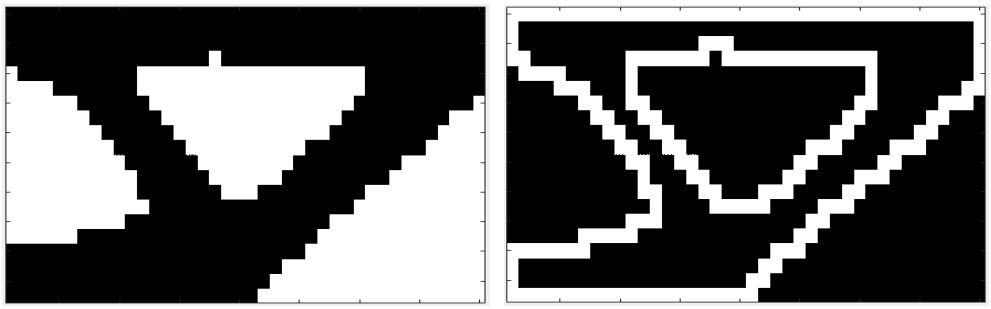

Figure 15: Image of the design after a processing it with the Heaviside function (left). New boundary detected (right).

But, to get this pure black and white designs, high values of β were required. This, as we explained before, helps to identify boundaries of the final design, but brings numerical instabilities in the iterative process in phase two, because the heaviside approximation function is too peaky. Therefore, we will still use this method, since it leads to an improvement in the process, but with low values of β to avoid numerical instabilities and calculating the overhang angle of all the elements instead of pretending to do it only in some fake design boundary that does not exist in intermediate iterative designs.

Computing the Overhanging Angle and comparing with the Printing Angle

The overhang angle can be obtained with the v vector. It is the angle between the down-facing surfaces and the base plate. Since the ground is always perpendicular to the printing direction (in the 3D printing machines considered in this thesis), and the v vector is perpendicular to the surfaces, the overhang angle is the same as the angle between the printing direction and the v vector (Figure 16) [1].

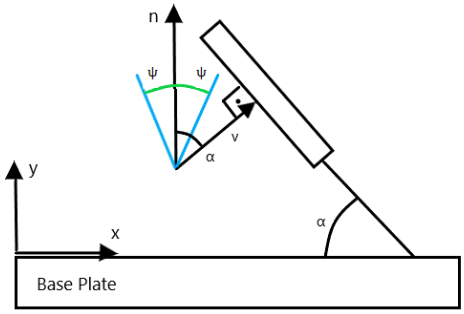

Figure 16: Describing the alpha angle between a down-facing surface and the base, with the v vector and the printing direction. This image has been redrawn after [1]

From the definition of the scalar product of two vectors in R^2, we get the following formula for the overhang angle α.

$$\alpha = \operatorname{acos}\left(\frac{n_x * v_x + n_y * v_y}{\|v\|}\right). \tag{15}$$

Here $n_x = \cos(\theta)$ and $n_y = \sin(\theta)$ are the components of the unit vector in the θ angle, the printing direction angle between the printing direction and the positive x direction. The problematic elements are identified as the ones with an alpha angle between $-\psi$ and ψ, being ψ the overhang angle limit. In phase one, we compared directly the angles α and ψ to decide which elements had a problematic overhang angle. An example of the problematic elements detected is showed in Figure 17. This was effective with a fixed and final design, but similar problems as above appeared in phase two. It was not efficient and presented mathematical indeterminations when calculating alpha and v was (0,0) with equation (15). In phase two, this angle comparison will be implemented in an efficient way, comparing the vertical projection of each v vector and the maximum value that it is allowed to have, as a function of ψ.

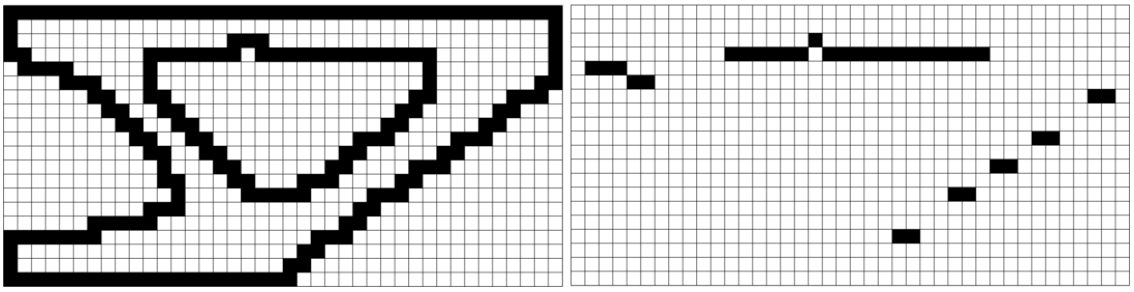

Figure 17: Left, all the boundary elements, right, boundary elements with an overhang angle problematic. In this case, the printing angle is $\theta = \pi/2$ (vertical) and the overhang angle limit $\psi = \pi/4$.

From this set of problematic angles, one should notice that the ones that are directly in contact with the base should be excluded. We have investigated some way to do that depending on the selected printing direction in phase one and implemented a method that is based on getting the projection of the $\boldsymbol{c_c}$ vector of each problematic element over the printing direction ($\boldsymbol{n} \cdot \boldsymbol{c_c}$). Then, the problematic boundary elements that have the lowest projection (with a certain tolerance) are removed from the problematic set. We can see how it works in Figure 18.

$$\boldsymbol{n} \cdot \boldsymbol{c_c} = x_{ce} n_x + y_{ce} n_y, \text{ for all elements in the boundary.}$$

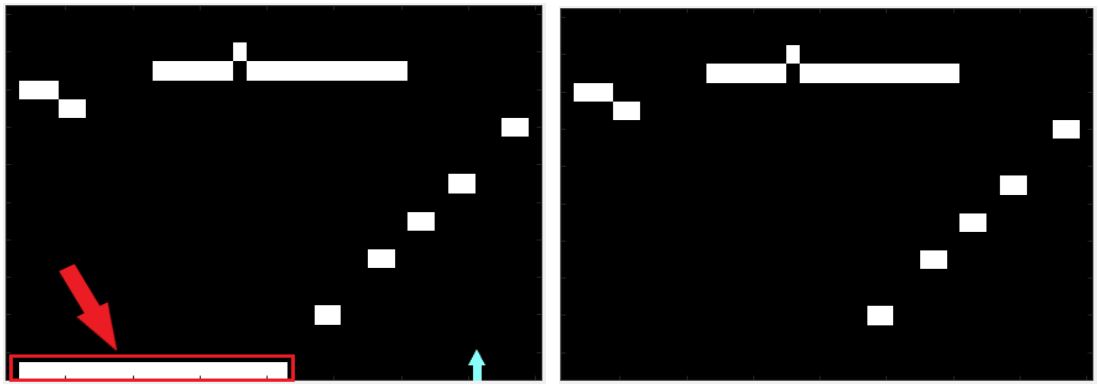

Figure 18: Left, the problematic elements set before removing the elements in contact with the base. Right, the problematic set after removing the elements in contact with the base.

This was possible with one fixed design, but not in the optimization procedure of phase two, as we set the base (and therefore, the printing direction) when giving fixed values to the dummy extra elements to calculate the v vector in the design domain boundary. Nevertheless, for future investigation with a different approach to calculate the v vector in the boundaries avoiding the set of a fixed dummy extra rows and columns, this can be a good way to exclude the base elements of the set of problematic elements. Then, a loop to investigate for every design the printing direction with a smaller number of problematic elements could be created. Since that method identifies the elements that are in contact with the base for every printing direction, it

could also be used to avoid designs that have a small number of elements in the base, and therefore are unstable to print (so they may need supportive structures for that reason).

Implementation of the constraint in a Topology Optimization code

We will now introduce the final mathematical formulation that has been implemented in Matlab. It is the result of merging formulation from [1] and [2], and then changing some expressions, in some cases for simplicity, and in others because some equations (mostly in the sensitivity analysis) from the references was not corresponding the actual derivatives that we got differentiating ourselves. Thus, we cannot expect the exact same results, but they are similar as we can see later in the corresponding section.

Mathematical formulation of the problem

The topology optimization problem that we are going to solve, has the following mathematical form:

$$\text{Optimization Problem} \begin{cases} \min\limits_{\rho} c(\tilde{\boldsymbol{\rho}}) & (Objective\ Function) \\ s.t. \begin{cases} \boldsymbol{K}\boldsymbol{u} = \boldsymbol{F} & (Equilibrium\ Constraint) \\ V(\tilde{\boldsymbol{\rho}}) \le V_0 & (Volume\ Constraint) \\ \phi(\tilde{\boldsymbol{\rho}}) \ge \phi_0 & (Overhang\ Constraint) \\ \underline{\rho} \le \rho_e \le \overline{\rho} & (Density\ function\ bounds) \end{cases} \end{cases} \tag{16}$$

Here, $\tilde{\rho}_e$ is defined in equation (14), but instead of using ρ_e, we replace it by $\hat{\rho}_e$ defined as

$$\hat{\rho}_e = \frac{\sum_{i \in S_e} w_i \rho_i}{\sum_{i \in S_e} w_i}, \tag{17}$$

in all the equations. The $\hat{\rho}$ is the result of applying a linear filter, known as regularization, and it is needed since we are using SIMP. The $\tilde{\rho}$ is the result of projecting the values of $\hat{\rho}$ with a continuous approximation of the heaviside function. The parameter β indicates how steep the heaviside approximation is, T is the threshold value for the heaviside approximation. Finally, Se is the set of elements that are not further away than the filter radius R (that we can set in the

parameters of the program). The parameters w_i are weight factors that depend on the Euclidean distance between the element e and element i, and are defined as

$$w_i = \max(0, R - |dist(e,i)|), \tag{18}$$

where R is the filter radius, that determines the minimum size of solid members of the design, and it is a percentage of the design's domain width.

Objective Function

Now, we will transform the equations that appear in the problem (17), so that it is possible to use them in the Matlab implementation of the project. We will start with the objective function, that is compliance, which is a property inversely proportional to stiffness. We will rewrite it, in terms of the finite element formulation, using the equilibrium constraint:

$$c(\tilde{\boldsymbol{\rho}}) = \boldsymbol{u}^T \boldsymbol{K} \boldsymbol{u} = \sum_{e=1}^{N} E_e(\tilde{\rho}_e) \boldsymbol{u_e^T k_0 u_e}. \tag{19}$$

In this formulation, we will apply SIMP to the Young modulus to penalize the intermediate density values. Thus, the definition of $E_e(\tilde{\rho}_e)$ will be:

$$E_e(\tilde{\rho}_e) = E_{min} + (E_0 - E_{min}) \cdot \tilde{\rho}_e{}^q, \tag{20}$$

where $\boldsymbol{u_e}$ is the displacement vector of element e, $\boldsymbol{k_0}$ is the general stiffness matrix for squared shaped elements and q is the SIMP penalty parameter, that usually takes values between 1 and 3. The value E_0 is the Young modulus and E_{min} is a small value, typically 10^{-9}, introduced to avoid numerical problems that appear because the lower bound of ρ_e is zero.

Volume Constraint

Having defined the objective function, we will now define the volume constraint $V(\tilde{\boldsymbol{\rho}}) \leq V_0$. The volume is written as

$$V(\tilde{\boldsymbol{\rho}}) = \boldsymbol{A}\tilde{\boldsymbol{\rho}}\,, \tag{21}$$

where $\boldsymbol{A}$ is a row vector whose components are all $v_e = \frac{1}{nel}$. If we normalize this volume constraint, we get the following expression (that is ready to implement in the code):

$$\boldsymbol{A}\tilde{\boldsymbol{\rho}} - V_0 \leq 0. \tag{22}$$

Overhang Constraint

Finally, we will describe the overhang constraint $\phi(\tilde{\boldsymbol{\rho}}) \geq \phi_0$. For that purpose, we will define ϕ, as the ratio between well supported and not well supported elements, ϕ^- and ϕ^+ respectively. Thus, we calculate the vector that goes from the geometrical center of each mask to the gravity center of it. For that, we use equations (11) and (12), and calculate ${x_{cg}}^m$ and ${y_{cg}}^m$ with $\tilde{\boldsymbol{\rho}}$, and the sumatories are defined for the elements e that belong to the mask m.

For this calculation in the boundary of the design domain, we will introduce two extra dummy rows and columns of elements, surrounding the design domain. Their density value will be zero, except the row below the design domain, that will have density one since there it is the base plate. Thus, we will end up with two column vectors x_{cg} and y_{cg}, with the x and y cords of each mask's gravity center. Since in our formulation we have the centroid cords in an absolute coordinate system, the vectors x_{cg} and y_{cg} will not go from the mask's geometrical center but form the origin of the coordinate system. Thus, we will make new vectors with relative coordinates changing the coordinate system from absolute to local with the following simple formula:

$${V_x}^m = {x_{cg}}^m - {x_c}^m, \tag{23}$$

$${V_y}^m = {y_{cg}}^m - {y_c}^m. \tag{24}$$

This will be calculated exactly like this in the Matlab code. Now we will use these vectors to calculate the angle between down-facing contours and the plate. This can be done because the value of $V^m = ({V_x}^m, {V_y}^m)$ is high in boundaries of the design and low in all black or all white regions (where mass and geometrical center coincide). This vector's angle with the printing direction is also the same one as the surface angle with the base plate (because it points from the geometric center to the gravity center, that is perpendicular to the surface direction). Therefore, we will constraint that angle, that we call α.

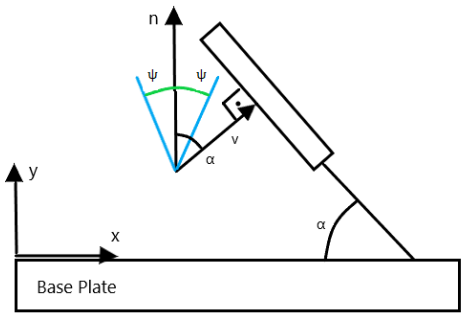

Figure 19: Illustration of the angels considered.

The constraint is that the angle α cannot be bigger than the critical overhang angle ψ. Instead of writing this constraint as $\alpha \leq \psi$, we will use a more practical definition from the paper [2] that is already normalized:

$$\varphi^m(\tilde{\boldsymbol{\rho}}) = {V_y}^m \sin(\psi) - |{V_x}^m| \cos(\psi) \leq 0. \tag{25}$$

So, we will calculate the value of $\varphi^m(\tilde{\rho})$ for each mask m of a design $\tilde{\rho}$, and then we will use it to classify all the elements between self-supported and not self-supported, as we said before, with the following equations:

$$\phi^- = -\sum_{m=1}^{M} \min(0, \varphi^m), \tag{26}$$

$$\phi^+ = \sum_{m=1}^{M} \max(0, \varphi^m). \tag{27}$$

And we will use this values to set a relative value of well supported elements over the entire design as:

$$\phi = \frac{\phi^-}{\phi^- + \phi^+}. \tag{28}$$

So, the overhang constraint will be the minimum relative amount of elements that we want to force to fullfill the overhanging constraint ϕ_0.

$$\phi_0 \cdot \frac{\phi^- + \phi^+}{\phi^-} - 1 \leq 0\,. \tag{29}$$

Sensitivity Analysis

We need to take derivatives of the objective function and the constraint equations, in order to reach the solution of the problem faster. Since all the equations are funcions of $\tilde{\rho}$, the chain rule will be applied when differentiating all the equations in the following way:

$$\frac{\partial F}{\partial \boldsymbol{\rho}} = \frac{\partial F}{\partial \widetilde{\boldsymbol{\rho}}} \cdot \frac{\partial \widetilde{\boldsymbol{\rho}}}{\partial \widehat{\boldsymbol{\rho}}} \cdot \frac{\partial \widehat{\boldsymbol{\rho}}}{\partial \boldsymbol{\rho}}. \tag{30}$$

So, there will be a common term that we will call $P = \frac{\partial \widetilde{\boldsymbol{\rho}}}{\partial \widehat{\boldsymbol{\rho}}} \cdot \frac{\partial \widehat{\boldsymbol{\rho}}}{\partial \boldsymbol{\rho}}$, and will be calculated once for all the equations:

$$\frac{\partial \widetilde{\rho_e}}{\partial \widehat{\rho_e}} = \frac{\beta[1 - tanh^2(\beta \cdot (\widehat{\rho_e} - T))]}{\tanh(\beta \cdot T) + \tanh(\beta \cdot (1 - T))}, \tag{31}$$

$$\frac{\partial \widehat{\rho_e}}{\partial \rho_e} = \frac{w_i}{\sum_{i \in S_e} w_i}. \tag{32}$$

And now, we will compute $\frac{\partial F}{\partial \widetilde{\boldsymbol{\rho}}}$ for the objective function and the constraint equations.

Sensitivity of the Objective Function

The derivative of the compliance is

$$\frac{\partial c(\widetilde{\boldsymbol{\rho}})}{\partial \widetilde{\rho_e}} = -q \cdot (E_0 - E_{min}) \cdot \widetilde{\rho_e}^{q-1} \cdot \boldsymbol{u_e^T k_0 u_e}\,. \tag{33}$$

Sensitivity of the Volume Constraint

The derivative of the volume constraint is

$$\frac{\partial V(\widetilde{\boldsymbol{\rho}})}{\partial \widetilde{\rho_e}} = v_e. \tag{34}$$

Sensitivity of the Overhang Constraint

The derivative of the overhang constraint will be:

$$\frac{\partial}{\partial \widetilde{\rho_e}}\left(\phi_0 \cdot \frac{\phi^- + \phi^+}{\phi^-} - 1\right) = \phi_0 \frac{\phi^- \cdot \frac{\partial \phi^+}{\partial \widetilde{\rho_e}} - \phi^+ \cdot \frac{\partial \phi^-}{\partial \widetilde{\rho_e}}}{(\phi^-)^2}. \tag{35}$$

Thus, we need $\frac{\partial \phi^+}{\partial \widetilde{\rho_e}}$ and $\frac{\partial \phi^-}{\partial \widetilde{\rho_e}}$, but ϕ^- and ϕ^+ are sumatories, so the derivative of both, will be the sum of the derivative of the elements that are in the sumatories. And since we were summing 0 or $\varphi^m(\widetilde{\boldsymbol{\rho}})$ depending on the case, we need to differentiate $\varphi^m(\widetilde{\boldsymbol{\rho}})$ for all masks. This translated to equations, can be written (neglecting the fact that min and max are not smooth) as:

$$\frac{\partial \phi^-}{\partial \widetilde{\rho_e}} = \sum_{m=1}^{M} \begin{cases} 0 & if\ \varphi^m > 0 \\ -\frac{\partial \varphi^m}{\partial \widetilde{\rho_e}} & if \varphi^m \leq 0 \end{cases} \tag{36}$$

$$\frac{\partial \phi^+}{\partial \widetilde{\rho_e}} = \sum_{m=1}^{M} \begin{cases} \frac{\partial \varphi^m}{\partial \widetilde{\rho_e}} & if\ \varphi^m > 0 \\ 0 & if \varphi^m \leq 0 \end{cases} \tag{37}$$

Therefore, we will need to differentiate $\frac{\partial \varphi^m}{\partial \widetilde{\rho_e}}$ for all m, in order to later get the derivative of the overhang constraint with respect to $\widetilde{\rho_e}$. We get this derivative:

$$\frac{\partial \varphi^m}{\partial \widetilde{\rho_e}} = \frac{\partial {V_y}^m}{\partial \widetilde{\rho_e}} \sin(\psi) - \frac{{V_x}^m}{|{V_x}^m|} \cdot \frac{\partial {V_x}^m}{\partial \widetilde{\rho_e}} \cdot \cos(\psi). \tag{38}$$

So, we need to calculate the derivatives of vector V^m component,

$$\begin{aligned} \frac{\partial {V_x}^m}{\partial \widetilde{\rho_e}} &= \frac{\partial}{\partial \widetilde{\rho_e}}\left(\frac{\sum_{e\in m} \tilde{\rho}_e\, x_{ce}}{\sum_{e\in m} \tilde{\rho}_e} - {x_c}^m\right) = \frac{\partial}{\partial \widetilde{\rho_e}}\left(\frac{\sum_{e\in m} \tilde{\rho}_e\, x_{ce}}{\sum_{e\in m} \tilde{\rho}_e}\right) \\ &= \begin{cases} \frac{x_{ce} - {x_{cg}}^m}{\sum_{e\in m} \tilde{\rho}_e} & if\ e\ \in m \\ 0 & otherwise \end{cases} \end{aligned} \tag{39}$$

And similarly,

$$\frac{\partial {V_y}^m}{\partial \widetilde{\rho_e}} = \begin{cases} \frac{{y_{ce} - y_{cg}}^m}{\sum_{e \in m} \tilde{\rho}_e} & if\ e\ \in m \\ 0 & otherwise \end{cases} \quad (40)$$

Results of testing the implementation

In this section, some results are shown and commented. Not all of them are perfect, but we instead will focus here on finding the strengths and weaknesses of the method and making suggestions on how to improve it in further developments, and how the parameters affect the result.

Cantilever beam example

We tried to get similar results as in Fig.9 of [1]. It is a comparison of different designs that they got for different ψ angles in a beam that is clamped at the left edge and has a unit point load at the center of the right edge. We started by replicating their example with a $\psi = \frac{\pi}{2}$. We show a comparison between our result and theirs in the Figure 20.

Figure 20: Comparison between results from [1] (left) and our implementation (right)

This result was obtained with the following parameter values: $\phi_0 = 0.97$, $\beta = 7$, $q = 18$, $T = 0.5$, $E_0 = 1$, $E_{min} = 0.001$, $R = 11\%$, γ = 0.5. The most important difference between designs being compared here is that our mesh is 60x30, because a finer mesh takes significantly more time, and the paper mesh is 160x80, so designs may look different. The run time was 53.26 s in a personal computer with an intel i7-3630QM, and the part of the code that takes the most time is the constraint. For the following examples, we will mention only the parameters that differ from these starting ones.

For the previous design, we should comment that the filter radius was bigger than for the following case, where we follow the advice of [1] setting the radius as four times the element's width, resulting is $R = 6{,}6667\%$. Here, we change the overhang angle to $\psi = \frac{\pi}{3}$. We also needed to use continuation on q (from 5 to 7) and on β (from 7 to 8), in three steps, and the run time was 554.36 s. The result is compared with the original in Figure 21.

Figure 21: Comparison between results from [1] (left) and our implementation (right)

The similarities between both figures can be seen.

Finally, the last of these three figures group to compare is Figure 22.

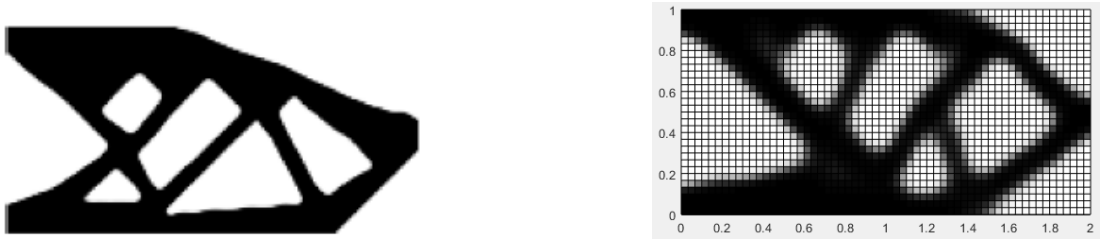

Figure 22: Comparison between results from [1] (left) and our implementation (right)

This design was obtained with continuation on q (from 4 to 10) and on β (from 7 to 8), in seven steps, and $\psi = \frac{\pi}{4}$. The run time in this case was 1285.92 s. We could assume that the main difference between the amount of holes and shape of them is caused by the mesh difference.

Concluding Remarks

The valuation of the results is that the constraint successfully works, or at least strongly affects the solution, of course with limitations. It is strongly dependent on the parameters, to mention some of the most important facts, we could say that with $\beta > 7$ it is unlikely to have convergence (at least with this problem and parameters). Another fact is that, in order to get

more black and white designs, we should trust more in getting a higher q than a higher β, but we need to use continuation. If the continuation starts too low, the final result will be a black and white bad design, because it will get more black and white something that is not close to the optimal solution. If we set q too high from the beginning, the first convergence will also give some strange shape and it will keep it for the rest of the optimization, meaning that we will not reach a good design.

We have observed that continuation is not beneficial for ϕ_0, because it drives the solution to a determined topology and later tries to change it. But, it increases when q and β are bigger and the problem is less capable of withstanding a big topological change. Therefore, the overhang constraint is not fulfilled. So, it is preferable to start with a high ϕ_0 from the beginning.

Acknowledge

I want to thank Linköping University for accepting me as an exchange student, all my teachers here for teaching me so much, and specially my thesis supervisor Carl-Johan for giving me the opportunity to do this thesis with him. I have learned so much from you.

I also want to thank my parents, for educating me to work hard and be strong in difficult times, my girlfriend for making my life happier and all my family for being there for me. No matter how far away we are, I love you all. Finally, I want to thank my little sister, who inspires me and makes me a better person. I am so proud of you. You can always count on me for everything.

References

[1] A. Garaigordobil, R. Ansola, J. Santamaría and I. F. d. Bustos, "A new overhang constraint for topology optimization of self-supporting structures in additive manufacturing," *Structural and Multidisciplinary Optimization,* 2018.

[2] A. Garaigordobil, R. Ansola, E. Veguería and I. Fernández, "Overhang constraint for topology optimization of self-supported compliant mechanisms considering additive manufacturing," in *Computer-Aided Design*, Elsevier, Ed., 2018, pp. 33-48.

[3] P. W. Christensen and A. Klarbring, An Introduction to Structural Optimization, Linköping: Springer, 2009.

[4] T. W. a. T. Gornet, "History of Additive Manufacturing," WOHLERS ASSOCIATES, INC. , 2016.

[5] M. S. b. A. Razak, M. H. b. Hasim and N. A. b. Ngatiman, "Design of Electric Vehicle Racing Car Chassis using Topology Optimization Method," *MATEC Web of Conferences,* vol. 97, p. 5, 2017.

[6] 3dnatives.com, "https://www.3dnatives.com/es/tecnologias-3d/," [Online]. Available: https://www.3dnatives.com/es/tecnologias-3d/.

[7] Y. C. C. L. Y. e. a. Kuo, "Support structure design in additive manufacturing based on topology optimization," in *Structural and Multidisciplinary Optimization*, Springer Berlin Heidelberg, 2017, p. 183–195.

[8] O. Sigmund and J. Petersson, "Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima," in *Structural Optimization*, Springer-Verlag, 1998, pp. 16-68.

[9] M. Langelaar, "Combined optimization of part topology, support structure layout," in *Structural and Multidisciplinary Optimization*, 2018, p. 57:1985–2004.

[10] I. B. a. F. Kantojärvi, "Topology optimization for additive manufacturing - A method to constrain maximum overhang," Linköping University, Linköping, 2016.

[11] G. H. a. I. Y. Omer Faruk Beyca, "Additive Manufacturing Technologies and Applications," in *Industry 4.0: Managing The Digital Transformation*, Springer, 2018, pp. 217 - 221.

[12] F. L. B. Wang and O. Sigmund, "On projection methods, convergence and robust formulations in topology optimization," in *Structural and Multidisciplinary Optimization*, Springer-Verlag, 2010, p. 767–784.

[13] D. Liang-Jian, G. Weihong and H. Ting-Zhu, "Single image super-resolution by approximated Heaviside functions," *Information Sciences,* vol. 348, pp. 107-123, 2016.