



HAL
open science

Réseaux Antagonistes Génératifs 3d pour le design, l'optimisation et la validation numérique en fabrication additive

Waad Almasri

► **To cite this version:**

Waad Almasri. Réseaux Antagonistes Génératifs 3d pour le design, l'optimisation et la validation numérique en fabrication additive. Informatique. HESAM Université, 2023. Français. NNT : 2023HESAE009 . tel-04090062

HAL Id: tel-04090062

<https://pastel.hal.science/tel-04090062v1>

Submitted on 5 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ÉCOLE DOCTORALE SCIENCES ET MÉTIERS DE L'INGÉNIEUR

Laboratoire d'ingénierie des systèmes physiques et numériques,
Campus de Cluny

THÈSE

présentée par : **Waad ALMASRI**

soutenue le : **19 janvier 2023**

pour obtenir le grade de : **Docteur d'HESAM Université**

préparée à : **École Nationale Supérieure d'Arts et Métiers**

Spécialité : **Informatique**

3D Generative Adversarial Networks for design, optimization, and validation in additive manufacturing

THÈSE dirigée par :

Professeur **ABABSA Fakhreddine**

et co-encadrée par :

Docteur **DANGLADE Florence**

Jury

M. Frédéric SEGONDS	Professeur des universités, Laboratory LCPI, École Nationale Supérieure d'Arts et Métiers	Président
M. Djamal MERAD	Professeur des universités, Laboratory LIS CNRS, Université Aix-Marseille	Rapporteur
M. Volker SCHULZE	Professeur des universités, Institute for Production Engineering (wbk), Karlsruhe Institute of Technology	Rapporteur
M. Evgeny BURNAEV	Professeur des universités, Skoltech Applied AI center, Skolkovo Institute Of Technology	Examineur
M. Fakhreddine ABABSA	Professeur des universités, Laboratory LISPEN, École Nationale Supérieure d'Arts et Métiers	Examineur
Mme. Florence DANGLADE	Maitre de conférence, Laboratory LISPEN, École Nationale Supérieure d'Arts et Métiers	Examinatrice
M. Dimitri BETTE-BGHOR	Data Practice Lead, Expleo Group	Invité
M. Faouzi ADJED	Docteur Chercheur , IRT System-X	Invité

T
H
È
S
E

To my Dad ...

Acknowledgment

As Sir Isaac Newton once said, “If I have seen further, it is by standing on the shoulders of giants”, this thesis would not have been accomplished without the help of many people on several levels, from the administrative to technical and finally personal level.

I want to start by thanking my ex-managers, Mr. David RENAUD and Dr. Jean-Christophe MAGNIENT, the administrative managers, Mr. Olivier MINGUI et Dr. Perrine BROY, my advisors from Expleo Group, Dr. Dimitri BETTEBGHOR, and Dr. Faouzi ADJED, and finally my advisors from l’Ecole des Arts et Métiers, Prof. Fakhreddine ABABSA, and Dr. Florence DANGLADE, whom without the thesis would not have seen the light.

I want to thank Dr. Dimitri BETTEBGHOR for this fascinating research subject that he proposed. I was honored to work alongside him for these last three years. I have learned a lot from his experience and benefited enormously from his advice.

I want to thank Dr. Faouzi ADJED for his help in finding the LISPEN research lab and for his thorough supervision and support on the technical and personal levels. I have enormously appreciated every input, comment, and advice I received from you, Dr. Faouzi ADJED.

A special thank you to my advisor Prof. Fakhreddine ABABSA for his thorough supervision and continuous encouragement.

I want to thank Dr. Florence DANGLADE for her support not only on the technical side but most notably on the personal side. Dr. DANGLADE taught me perseverance and tenacity.

I am lucky to have had the chance to work alongside these brilliant people I respect and admire.

I would also like to thank my interns, Mr. Corentin MAGYAR and Mr. Steve Baggio SEDjro NOUATIN, for their brilliant work.

I want to thank the Cassiopee Platform Team, Mr. Théophile GROSS, Mr. Boris PIOTROWSKI,

ACKNOWLEDGMENT

and Mr. Camille BURTON for their support on the computational resources. Without you, no dataset would have been consolidated, and none of the work could have been completed.

I want to thank Mr. Jérôme VIALLET, Mr. Ludovick BELLUNE, Mr. Corentin MAGYAR, and Mr. Nicolas DAVID for helping implement and maintain our open-accessible application GAIMD up and running on the cloud.

I want to thank my ex-EXPLEO colleague, Dr. Mohamed BENZERGA, for his technical support when tackling the mathematical sections during the state-of-the-art phase.

Also, I want to thank the mechanical design team at EXPLEO for their help in the mechanical validation of our approach, particularly Mr. Olivier VUILLAUME, Mr. Erkan ERGIN, and Mr. Jean-Philippe GRUAU.

I would also like to thank Mr. Mohanamurali NALLANATHAN for his help on the mechanical validation part. But, most importantly, I want to thank him for making EXPLEO a great place to work for me with his friendly and funny presence.

I would also like to thank Dr. Sondes GHARIANI, without whom no design would have been printed.

I want to thank my colleagues, Mr. Hugo SECHIER, Mrs. Lamiae MKADMI, Dr. Perrine BROY, and the R&D data team for their support and continuous encouragement.

I want to thank my friends, Anthony, Chantal, Charbel, Hanady, Joanne, Mira, and Roland, for being there whenever I felt down.

In the spirit of saving the best for last, it is time to give heartfelt thanks to my mom, Ginan, dad, Amer, and siblings, Nadine, Rana, and Amer, and to the love of my life, Louay. I dedicate my work to you, for you have always inspired me. You always had my back and have been the one primary reason for who I have become and accomplished today. No matter what I do, I will never fully show my gratitude and appreciation. But I hope I made you proud.

ACKNOWLEDGMENT

ACKNOWLEDGMENT

Abstract

The growing need for fast, organic, versatile, cost- and material-efficient products in the industrial world drives the research to develop design approaches accounting for these criteria altogether. Topology optimization (TO) falls in the category of these design approaches. It allows the generation of shapes with curvatures and fine details, given parameters such as loads, boundary conditions, and a volume fraction. The resulting design can have any shape and be defined as a 2D binary image or a 3D binary voxel grid, such that the presence of a non-empty pixel/voxel means the presence of the material. TO gained tremendous success in the 20th century during the industrial revolution of the automotive and aerospace sectors, given its powerful potential to optimize a structure in terms of material used while maintaining its recommended mechanical specifications and properties. Complementary comes additive manufacturing (AM), which allows the printing of any form. However, this synergy is not as idealistic as it looks. On the one hand, TO uses an iterative, FE-based method that is computationally expensive and relatively slow. On the other hand, AM requires the design's shape to comply with some geometrical criteria, like dealing with curvatures, overhanging patterns, the need for supports, thin features, Etc., which are hardly integrated into TO's formulation [1]. Thus, designing optimal printable parts requires the intervention of experts to interpret the shapes proposed by TO to comply with the manufacturing criteria. Nonetheless, this reinterpretation phase can compromise the initial design's optimality. Moreover, it is time-consuming and depends on the engineer's expertise. With the new advanced TO software, outputting a single design subjected to boundary conditions and load configurations takes seconds to days, depending on the design's complexity and surrounding constraints. This aspect is acceptable if the process is limited to this single step. However, a mechanical engineer never relies on the first design (s)he tries, especially since TO, in its commercially available form, does not consider geometrical constraints, manufacturing criteria, or other customized industrial constraints (automotive, aeronautic, hydraulic, Etc.). (S)He always needs to explore sev-

eral set-ups to find the optimal topology, shape, and sizing of the design to be manufactured. (S)He also must ensure that his(her) final draft is creative, cost-efficient, and manufacturable. Hence, this accumulated iterative exploration process can become slow, especially for large-scale designs. Consequently, with the flourishing role of AM in the industry, it is further interesting to find a method that simultaneously considers mechanical and geometrical constraints at the conceptual level known as the discipline of Design for Additive Manufacturing (DfAM). Furthermore, a recent survey by Subedi et al. [2] has shown that half of TO practitioners regret the absence of geometric and manufacturing-related plugins in TO's software. Accelerating DfAM's cycle is a current hot research and industrial topic. Consequently, researchers try to find workarounds to bypass these constraints and accelerate the design optimization phase. We can find in the literature four general approaches to tackle the problem. Accelerating the DfAM process consists of four major state-of-the-art approaches:

1. Formalizing AM design rules for engineers to consider at the CAD drawing phase. However, this does not guarantee the initial design optimality and mechanical performance. Nevertheless, this approach helped formalize AM constraints for novice users and guided engineers into creating print-ready designs.
2. Integrating certain AM constraints at the FE-TO formulation. This approach allows the engineers to have a first draft accounting for both constraints, thus preventing getting stuck in a loop of re-drawing and re-testing the mechanical performance of the design. Nevertheless, not all AM constraints can be analytically formulated. Also, this approach inherits the density-based TO's flaws. (1) FE-TO identifies the general shape at early iterations, inhibiting the method of modifying the shape to take account of geometrical constraints. (2) The convergence problem is not guaranteed, especially when there are multiple constraints; it is hard to find a nash equilibrium where all conditions are validated.
3. Assisting FE-TO methods with Machine learning (ML) and Deep Learning (DL). This approach focuses on accelerating the TO phase of the DfAM process and does not dodge getting stuck in a loop in later stages. Like the previous approach, most of TO's problems are still inherited.
4. Replacing FE with ML and DL in TO. However, this approach does not integrate any manufacturing constraints and does not circumvent getting stuck in a loop in later phases of the DfAM process.

Our objective is to accelerate the whole DfAM process and have the best quality/cost ratio. Indeed, the design phase has the lowest cost and highest impact on the overall product cost, quality, and hence its lifecycle [3]. However, accelerating this phase alone is not enough, as we have seen with state-of-the-art approaches. Moreover, while AM-FE-TO methods aim to accelerate the whole DfAM process by preventing repetitive iterations, the analytical formulation of AM constraints, the methods' convergence, and their computational costs are still a hurdle. Conversely, the introduction of ML and DL only accelerated one phase. Thus, we propose to get the best of both worlds in this work. Furthermore, this new DL approach allows manufacturing constraints' integration within mechanical ones concurrently at the same level. Therefore, any conditions, even ones lacking a mathematical definition like experts' rules and knowledge, benefit from DL's speed and scalability advantages. It is imperative to note that this approach is not intended to replace robust FE-TO but to help compensate for its difficulties in integrating various complex constraints. Our work's main contribution is not only accelerating the TO phase but also identifying a way to tailor the design's geometry and manufacturability in order to generate a first draft design that complies with both mechanical and geometric-manufacturing constraints to avoid getting stuck in a loop of updating and testing designs and accelerate the whole DfAM process. The major contributions of our work can be summarized as follows:

- Integration of mechanical and manufacturing constraints at the same level.
- Creation of a synthetic dataset of 2D designs alongside their mechanical and geometrical manufacturing constraints.
- Benefitting from DL's ability to learn spatial correlations and its speed and scalability. This approach integrates input mechanical and geometrical conditions at the conceptual level, generates designs accordingly, and is computationally independent of the inputs' complexity.
- Convergence is no longer an issue as long as the model is trained on converged designs.
- Any constraint can be easily integrated thanks to the flexibility of concatenating several input types into DL models.

The advantages of our approach are:

- Any AM constraint or other type of constraint (min overhang, thermal distortion, buckling, experts' informal rules, Etc.) can be integrated into the design model, not only analytically formulated ones.
- Acceleration of the whole DfAM process, and not only the design phase, by generating a first design draft complying with geometrical manufacturing and mechanical constraints and avoiding repetitive iterations of design updating and evaluation.
- DL-AM-TO could be industrialized as a lighter generative design module to be implemented in industrial design software in the future.

It is essential to highlight that this work is focused on SIMP among all TO approaches for two major reasons. First, SIMP deals with the design as a distribution of material, in other terms, as an image, which is compatible with our way of approaching the design using DL-based computer vision techniques. Second, SIMP is the most implemented TO approach in industrial software, for its simplest and less computationally expensive TO approach; SIMP is found in SolidWorks, ABAQUS, Siemens NX, Altair OptiStruct, and ANSYS Mechanical, which also includes the level set approach [4]. The latter argument encourages us to work on compensating for its difficulties with DL and, in the future, propose a lighter AI-based TO module to be incorporated into industrial software.

The work was divided into five parts:

1. Generation of a synthetic 2D-designs-dataset from the top common industrial topology optimization method, SIMP (Solid Isotropic with material Penalization). This dataset consisted of 2D designs alongside the mechanical conditions, called DB1.
2. Since the designs of DB1 was truss-like structures, we have started with a simple geometrical constraint, the number of bars, to build a proof-of-concept Deep Learning model that takes as input the mechanical constraints alongside the number of bars (the geometrical one) and generates a 2D design accordingly. We have used the GAN framework with three discriminators to train our DL-TO model. The first one evaluated the generated designs' conformity to the mechanical constraints. The second one predicted the number of bars in the generated designs and penalized DL-TO if they did not respect the input geometrical constraint. Finally, the third one predicted the energy of deformation, known as the compliance over the generated designs, and penalized DL-TO if the compliance was very high.

3. Afterward, we needed to integrate more concrete geometric manufacturing-related constraints. The problem was that there was no database with designs alongside their mechanical and geometrical constraints. Thus we had to create this dataset ourselves. First, we generated more DB1 designs from given mechanical constraints (boundary conditions, loads configuration, and volume fraction). Second, a DL model that maps designs to their corresponding mechanical conditions, using the SIMP designs, is built; the model is referred to as DL-Mechanical-Conditions-Predictor. Third, synthetic designs (inspired by the shapes of DB1 designs) with various geometric constraints are generated using pygmsh. Fourth, the previously learned DL-Mechanical-Conditions-Predictor is used to predict their mechanical conditions. Finally, the target dataset "GMCAD" is consolidated; it consists of pairs of designs and their mechanical and geometric constraints. The geometrical constraints are minimum/maximum thicknesses, minimum/maximum bar lengths, angles, number of components, Etc.

4. Having GMCAD, we could start training a model that simultaneously integrates the mechanical and geometrical-manufacturing-related constraints; we call this model DL-AM-TO. The geometrical constraints chosen are the minimum overhang (the angle between the printing orientation and the normal of a component's surface in a design), the maximum length of a component, the minimum thickness, and the number of components, such that a component is defined as a rectangular bar. Building DL-AM-TO was divided into two steps. The first step consisted of isolating every geometric constraint to check its effect on the design's geometry and the other conditions; in other terms, doing step 2 per geometrical constraint. Second, putting all geometric constraints altogether to see their effects.

5. Finally, we validated DL-AM-TO by generating several geometries for the same mechanical constraints, which differ only by one geometrical constraint each time. We compared these geometries to the one proposed by SIMP, where the shape cannot be geometrically adjusted. DL-AM-TO allows us to adjust the geometry of a design to meet the geometric-manufacturing constraints in a few seconds. These geometries have a mechanical performance of the same order as SIMP. Nevertheless, they are 1.4 times faster in the printing phase and require less support.

Keywords : Topology Optimization (TO), Additive Manufacturing (AM), Design for Additive Manufacturing (DfAM), Deep Learning (DL), Generative Adversarial Networks (GAN), Convolutional

ABSTRACT

Neural Networks (CNN).

ABSTRACT

ABSTRACT

Résumé

Le besoin croissant de produits rapides, organiques, polyvalents, efficaces en termes de coûts et de matériaux dans le monde industriel, incite la recherche à développer des approches de conception tenant compte de tous ces critères. L'optimisation topologique (TO) entre dans la catégorie de ces approches. Elle permet de générer des formes avec des courbures et des détails fins, en fonction de paramètres tels que les charges, les conditions aux limites et la fraction volumique. Le design résultant peut avoir n'importe quelle forme et être défini comme une image binaire 2D ou une grille de voxels binaires 3D, de sorte que la présence d'un pixel/voxel non vide signifie la présence du matériau. La TO a connu un succès considérable au 20^e siècle lors de la révolution industrielle des secteurs de l'automobile et de l'aérospatiale, étant donné son puissant potentiel d'optimisation d'une structure en termes de matériau utilisé tout en conservant ses spécifications et propriétés mécaniques recommandées. D'autre part, la fabrication additive (FA) vient en complément ; elle permet l'impression de n'importe quelle forme. Cependant, cette synergie entre TO et FA n'est pas aussi idéaliste qu'il n'y paraît. D'une part, le TO utilise une méthode itérative, basée sur les éléments finis, qui est coûteuse en calculs et relativement lente. D'autre part, la FA exige que la forme du design soit conforme à certains critères géométriques, comme la gestion des courbes, des modèles en surplomb, la nécessité de supports, les caractéristiques minces, etc. Ainsi, la conception de pièces imprimables optimales nécessite l'intervention d'experts pour interpréter les formes proposées par la TO afin de respecter les critères de fabrication. Néanmoins, cette phase de réinterprétation peut compromettre l'optimalité du design initiale. De plus, elle prend du temps et dépend de l'expertise de l'ingénieur. Avec les nouveaux logiciels avancés de TO, la sortie d'un design unique soumis à un ensemble de conditions limites et de configurations de charge est une question de secondes à quelques jours selon la complexité de ce design et des contraintes environnantes. Cet aspect est acceptable si le processus se limite à cette seule étape. Cependant, un(e) ingénieur(e) en mécanique ne se fie jamais à la première ébauche qu'il/elle essaie, d'autant

plus que la TO, dans sa forme commerciale, ne tient pas compte des contraintes géométriques, des critères de fabrication ou d'autres contraintes industrielles personnalisées (automobile, aéronautique, hydraulique, etc.). Il/Elle doit toujours explorer plusieurs configurations pour trouver la topologie, la forme et le dimensionnement optimaux du modèle à fabriquer. Il/Elle doit également s'assurer que son projet final est créatif, rentable et manufacturable. Par conséquent, ce processus d'exploration itérative accumulée peut devenir lent, surtout pour les designs à grande échelle. Par conséquent, avec le rôle florissant de la fabrication additive dans l'industrie, il est intéressant de trouver une méthode qui tienne compte simultanément des contraintes mécaniques et géométriques au niveau conceptuel ou ce que l'on appelle la discipline de la conception pour la fabrication additive (Design for Additive Manufacturing, DfAM). De plus, une enquête récente menée par Subedi et al. [2] a montré que la moitié des praticiens de la TO regrettent l'absence de plugins géométriques liés à la fabrication dans le logiciel de la TO. L'accélération du cycle du DfAM est un sujet de recherche et un sujet industriel d'actualité. Par conséquent, les chercheurs tentent de trouver des solutions pour contourner ces contraintes et accélérer la phase d'optimisation du design. Nous pouvons trouver dans la littérature quatre approches générales pour aborder le problème. L'accélération du processus de DfAM se compose de quatre approches majeures de l'état de l'art :

1. Formaliser les règles de design de la FA pour que les ingénieurs les prennent en compte lors de la phase de dessin de la conception assistée par ordinateur (CAO). Cependant, cela ne garantit pas l'optimalité de la conception initiale et les performances mécaniques. Néanmoins, cette approche a permis de formaliser les contraintes de FA pour les utilisateurs novices et de guider les ingénieurs dans la création de designs prêts à être imprimés ;
2. intégrer de certaines contraintes FA lors de la formulation TO basée sur les éléments finis (FE); FE-TO . Cette approche permet aux ingénieurs de disposer d'une première ébauche prenant en compte les deux contraintes, évitant une boucle de dessin et de test des performances mécaniques du design. Toutefois, toutes les contraintes FA ne peuvent pas être formulées analytiquement. En outre, cette approche hérite des défauts de la TO basée sur la densité. (1) FE-TO identifie la forme générale dès les premières itérations, ce qui empêche la méthode de modifier la forme pour tenir compte des contraintes géométriques. (2) Le problème de convergence n'est pas garanti, surtout lorsqu'il y a des contraintes multiples ; il est difficile de trouver un équilibre où toutes les conditions sont validées ;

3. assister les méthodes FE-TO avec du Machine Learning (ML) et du Deep Learning (DL). Cette approche se concentre sur l'accélération de la phase TO du processus DfAM et n'évite pas de rester coincé(e) dans une boucle dans les étapes ultérieures. Comme pour l'approche précédente, la plupart des problèmes de TO sont encore hérités ;
4. remplacer complètement les FE par du ML et DL dans la TO. Cependant, cette approche n'intègre aucune contrainte de fabrication et ne permet pas d'éviter la boucle dans les phases ultérieures du processus DfAM.

Notre objectif est d'accélérer l'ensemble du processus DfAM et d'obtenir le meilleur rapport qualité/coût. En effet, la phase de conception a le coût le plus bas et l'impact le plus fort sur le coût global du produit, sa qualité et donc son cycle de vie[3]. Cependant, l'accélération de cette seule phase n'est pas suffisante, comme nous l'avons vu avec les approches de pointe. De plus, si les méthodes TO-FA basées sur les FE visent à accélérer l'ensemble du processus DfAM en évitant les itérations répétitives, la formulation analytique des contraintes FA, la convergence des méthodes et leurs coûts de calcul restent un obstacle. À l'inverse, l'introduction de ML et DL n'a accéléré qu'une seule phase. Ainsi, nous proposons d'obtenir le meilleur des deux mondes dans ce travail. En outre, cette nouvelle approche DL permet d'intégrer les contraintes de fabrication aux contraintes mécaniques simultanément au même niveau. Par conséquent, toutes les conditions, même celles qui n'ont pas de définition mathématique, comme les règles et les connaissances des experts, bénéficient des avantages de vitesse et d'évolutivité de la DL. Il est impératif de noter que cette approche n'est pas destinée à remplacer les FE-TO robustes mais à compenser leurs difficultés à intégrer diverses contraintes complexes. La principale contribution de notre travail n'est pas seulement d'accélérer la phase de TO mais aussi d'identifier un moyen d'adapter la géométrie et la fabricabilité du design afin de générer une première ébauche qui respecte les contraintes mécaniques et géométriques de fabrication pour éviter de rester coincé dans une boucle de mise à jour et de test de designs et accélérer l'ensemble du processus de DfAM. Les principales contributions de notre travail peuvent être résumées comme suit :

- l'intégration des contraintes mécaniques et de fabrication au même niveau ;
- la création d'un jeu de données synthétique de designs 2D avec leurs contraintes mécaniques et géométriques de fabrication, GMCAD[5] ;

- DL-AM-TO est une approche basée sur les réseaux convolutionnels de neurones. Elle bénéficie de la capacité de ces réseaux à apprendre les corrélations spatiales, de leur rapidité à l'inférence et de leur scalabilité. Ce qui lui permet de considérer les conditions mécaniques et géométriques au même niveau conceptuel, et générer des designs tout en étant computationnellement indépendante, sur le plan informatique, de la complexité des entrées ;
- la convergence vers une ébauche sujette à plusieurs contraintes n'est plus un problème tant que le modèle DL-AM-TO est entraîné sur des designs convergents ;
- la flexibilité d'intégration de tout type de contraintes grâce à la souplesse de concaténation de plusieurs types d'entrées dans les modèles DL ;
- finalement, DL-AM-TO permet l'accélération de l'entité du process DfAM et non seulement de la phase de design.

Les avantages de notre approche sont :

- toute contrainte FA ou autre type de contrainte (surplomb, déformation thermique, flambage, règles métiers, etc.) peut être intégrée dans le modèle de conception, et pas seulement celles formulées analytiquement ;
- l'accélération de l'ensemble du processus de DfAM en générant une première ébauche respectant les contraintes géométriques de fabrication et mécaniques et en évitant les itérations répétitives de mise à jour et d'évaluation du design ;
- un module de design génératif plus léger qui peut être implémenté dans un logiciel industriel de design à l'avenir.

Il est essentiel de souligner que ce travail se concentre sur l'approche densité de TO, SIMP (Solid Isotropic with material Penalization), parmi toutes les approches de TO pour deux raisons majeures. Premièrement, SIMP traite le design comme une distribution de matériel, en d'autres termes, comme une image, ce qui est compatible avec notre façon d'aborder le design comme une image en utilisant des techniques de computer vision basées sur le DL. Deuxièmement, SIMP est l'approche TO la plus implémentée dans les logiciels industriels, pour son approche TO la plus simple et la moins coûteuse en calcul ; on retrouve SIMP dans SolidWorks, ABAQUS, Siemens NX, Altair Optistruct, et ANSYS

Mechanical, qui inclut également l’approche level set [4]. Ce dernier argument nous incite à travailler à la compensation de ses difficultés par la DL et, à l’avenir, à proposer un module de TO plus léger basé sur l’Intelligence Artificiale (IA), à intégrer dans les logiciels industriels.

Ce travail a été divisé en cinq parties :

1. la génération d’un ensemble de données synthétiques de designs 2D à partir de la méthode TO industrielle la plus courante, SIMP. Cet ensemble de données, appelé *DB1*, est composé de designs en 2D et de conditions mécaniques ;
2. pourvu que les designs de *DB1* sont des structures en forme de treillis, nous avons commencé par une contrainte géométrique simple, le nombre de barres, pour construire un modèle DL-TO, une preuve de concept, qui prend en entrée les contraintes mécaniques ainsi que le nombre de barres (la contrainte géométrique) et génère un design 2D en sortie. Pour entraîner DL-TO, nous avons utilisé l’approche générative GAN (Generative Adversarial Networks) avec trois discriminateurs. Le premier évalue la conformité des designs générés aux contraintes mécaniques. Le second prédit le nombre de barres dans les designs générés et pénalise DL-TO si ces derniers ne respectent pas la contrainte géométrique d’entrée. Enfin, la troisième prédit l’énergie de déformation (la compliance) des designs générés et pénalise DL-TO si elle est très élevée ;
3. ensuite, afin d’intégrer des contraintes géométriques de FA, et pourvu que la base requise n’existe pas, nous avons été amenés à créer une base de designs 2D avec que leurs contraintes mécaniques et géométriques. Tout d’abord, nous avons augmenté la base *DB1*. Deuxièmement, à l’aide de *DB1*, nous avons entraîné un modèle DL qui prédit les conditions mécaniques à partir de design 2D en entrée ; le modèle est appelé DL-Mechanical-Conditions-Predictor. Troisièmement, des designs synthétiques (inspirés des formes des designs dans *DB1*) avec diverses contraintes géométriques sont générés à l’aide de pygmsh, cette base est appelée *DB2* . Quatrièmement, le prédicteur DL-Mechanical-Conditions-Predictor précédemment appris est utilisé pour prédire leurs conditions mécaniques. Enfin, le jeu de données cible “GMCAD” est consolidé ; il se compose de paires de designs 2D et de leurs contraintes mécaniques et géométriques. Les contraintes géométriques sont les épaisseurs minimales/maximales, les longueurs de barres minimales/maximales, les angles, le nombre de composants, etc. ;
4. avec GMCAD, nous avons implémenté un modèle qui intègre simultanément les contraintes

mécaniques et géométriques liées à la fabrication ; nous appelons ce modèle Deep Learning Additive Manufacturing driven Topology Optimization, DL-AM-TO. Les contraintes géométriques choisies sont le surplomb minimal (le surplomb est défini comme l'angle entre l'orientation de l'impression et la normale de la surface d'un composant dans un design), la longueur maximale d'un composant, l'épaisseur minimale et le nombre de composants, de sorte qu'un composant soit défini comme une barre rectangulaire. La construction de DL-AM-TO a été divisée en deux étapes. La première étape consistait à isoler chaque contrainte géométrique pour vérifier son effet sur la géométrie du design et les autres conditions géométriques, en d'autres termes, à effectuer l'étape 2 par contrainte géométrique. Dans un deuxième temps, nous avons rejoint toutes les contraintes géométriques au même niveau que celles mécaniques pour entraîner notre approche DL-AM-TO ;

5. enfin, nous avons validé DL-AM-TO en générant plusieurs géométries pour les mêmes contraintes mécaniques qui ne diffèrent que par une contrainte géométrique à chaque fois. Nous avons comparé ces géométries à celle proposée par SIMP où la forme ne peut être ajustée géométriquement. DL-AM-TO permet d'ajuster la géométrie d'un design afin de respecter les contraintes géométriques de fabrication en quelques secondes. Ces géométries ont une performance mécanique du même ordre que SIMP. Mais, elles sont 1.4 fois plus rapide dans la phase d'impression et demandent moins de support.

Mots clés : Optimisation Topologique (TO), Fabrication Additive (FA), Design pour la Fabrication Additive (DfAM), Apprentissage Profond (DL), Réseaux génératifs antagonistes (GAN), Réseaux Convolutionnels de neurone (CNN)s.

Contents

Acknowledgment	v
Abstract	ix
Résumé	xvii
List of Tables	xxx
List of figures	xli
Acronyms	xliii
Introduction	1
1 State of the art	7
1.1 Context	8
1.1.1 Additive Manufacturing (AM)	8
1.1.2 Topology Optimization (TO)	11
1.2 Objective	13
1.2.1 State of the art of DfAM acceleration approaches	13
1.2.2 Our approach	19
1.3 Ingredients	23
1.3.1 Machine Learning (ML)	23

CONTENTS

1.3.2	Fully Connected Neural Networks (NN)	24
1.3.3	Deep Learning (DL)	26
1.3.4	Convolutional Neural Networks (CNN)	26
1.3.5	Generative adversarial networks (GAN)	27
1.3.6	Conditional Generative adversarial networks (cGAN)	29
1.4	French Summary	31
2	General Method of Deep Learning driven topology optimization	35
2.1	Methodology: Data driven Design for Additive Manufacturing	36
2.2	Inputs: Mechanical and Manufacturing/Geometrical constraints	37
2.3	Model's architecture	40
2.4	Training Framework	41
2.5	Training Dataset	42
2.6	Discussion	43
2.7	French Summary	44
3	The first approach: Deep Learning driven topology optimization	49
3.1	Methodology	50
3.1.1	Training framework	51
3.1.1.1	Generator's architecture	51
3.1.1.2	Discriminators' architectures	52
3.1.1.2.1	Traditional discriminator's architecture	54
3.1.1.2.2	Bar counter discriminator's architecture	54
3.1.1.3	Loss function	54
3.2	Topology Optimized designs dataset consolidation	55
3.3	Experiments and Results	56
3.3.1	DL-TO's performance	59

CONTENTS

3.3.2	Counter-discriminator’s performance	60
3.3.3	Overall performance	60
3.4	Improving the performance of DL-TO	64
3.4.1	Compliance predictor discriminator’s architecture	65
3.4.2	Compliance-predictor’s performance	65
3.4.3	Training Loss Function	66
3.4.4	DL-TO’s Performance after training with three discriminators	67
3.4.5	Tailoring the design’s geometry via DL-TO	69
3.4.6	Generating designs with a new unseen constraint	71
3.5	Web application of DL-TO	73
3.6	Discussion	76
3.7	French summary	79
4	GMCAD: Geometric and Mechanical CAD dataset	83
4.1	Motivation	84
4.2	Workflow	85
4.2.1	Generation of the SIMP dataset <i>DB1</i>	86
4.2.2	Mechanical Conditions Prediction	87
4.2.2.1	Deep Learning based boundary conditions predictor	87
4.2.2.2	Deep Learning based force Predictor	88
4.2.3	Generation of the synthetic geometric dataset <i>DB2</i>	89
4.2.4	The Consolidation of the target dataset GMCAD	90
4.3	Results	90
4.3.1	Evaluation of the Mechanical Conditions Predictions	90
4.3.2	GMCAD: dataset of designs with their geometrical & predicted-mechanical constraints	92

CONTENTS

4.4	Discussion	93
4.5	French summary	96
5	The second approach: Deep Learning Additive Manufacturing driven topology optimization	99
5.1	Deep Learning Additive Manufacturing driven Topology Optimization	101
5.2	Methodology	101
5.2.1	Training framework	102
5.3	First variant of Deep Learning Additive Manufacturing-driven Topology Optimization (DL-AM-TO)'s training	102
5.3.1	Training Dataset	102
5.3.2	Generator's architecture	103
5.3.3	Discriminators' architectures	103
5.3.4	Loss function	104
5.3.5	Results	106
5.3.6	Discussion	107
5.4	Improving the geometrical discriminator's performance	110
5.4.1	Training	111
5.4.2	training dataset	112
5.4.3	Results	112
5.4.4	Uncertainty quantification of the DL geometrical discriminators	120
5.5	Training a model per geometrical variable	124
5.5.1	Training with Minimum overhang (Θ_{min})	124
5.5.2	Training with Maximum bar length (len_{max})	126
5.5.3	Training with Number of bars (Nbr_{bars})	129
5.6	Second variant of DL-AM-TO	131
5.6.1	Results	131

CONTENTS

5.6.2	DL-AM-TO's overall performance	131
5.6.3	Tailoring a design's geometry with DL-AM-TO	136
5.7	Printing designs generated by DL-AM-TO	138
5.7.1	Printing geometries of different Nbr_{bars} constraint	143
5.7.2	Printing geometries of different len_{max} constraint	144
5.7.3	Printing geometries of different Θ_{min} constraint	146
5.7.4	DL-AM-TO accelerates the DfAM process and is a lighter module in industrial design software	148
5.8	Discussion	151
5.9	French summary	155
	Conclusion	161
5.10	Conclusion	162
5.11	Perspectives	162
	Bibliography	165
	Liste of appendices	180
	A Appendices	181
A.1	Geometrical discriminators' performance	181

CONTENTS

List of Tables

1.1	Approaches to accelerate the Design for Additive Manufacturing (DfAM) process. . .	14
1.2	The state of the art approaches to accelerate the DfAM process. A blank cell means that the concerned characteristic does not apply for the method. A “+” sign defines the presence of the characteristic; the higher the number of the “+” sign, the better the method is when it comes to this characteristic. A “-” sign defines the absence of the characteristic.	20
3.1	Generator (Deep Learning-based Topology Optimization (DL-TO)) Res-U-Net Architecture. DL-TO’s architecture is composed of three major components: the encoder, the decoder, and the bridge, which links the encoder to the decoder. The encoder is formed of 4 blocks, each consisting of a down-sampling layer (a convolution of stride 2) and a residual unit([6]). The decoder comprises five blocks, each consisting of an up-sample layer (a transpose convolution of stride 2) and a residual unit followed by a convolution of kernel size 1×1 and a sigmoid activation. The bridge connection has the same architecture as an encoder-block and combines the encoder with the decoder. nf is the number of feature maps i.e. number of channels. The input’s dimension is $101 \times 101 \times 6$ and the output’s dimension is $101 \times 101 \times 1$. <i>A residual unit block is a sequence of two blocks, each consisting of a batch normalization[6] followed by a ReLU activation and a convolution of kernel size = 3×3 and stride 1. Its input is summed with its output via an identity mapping connection. An identity mapping connection consists of a convolution of kernel size = 1×1, a stride of 1, and padding of 0 followed by a batch normalization layer.</i>	53

LIST OF TABLES

3.2 Average Design Generation Time (in seconds) of Solid Isotropic with Material Penalization (SIMP) (FE-based) versus DL-TO. 63

3.3 Generation Time (in seconds s) and Computational Complexity (in Gega Floating Point Operations per Second $GFLOPS$) of SIMP (FE-based) versus DL-TO. Central Processing Unit (CPU) and Graphics Processing Unit (GPU) stands for central and graphical processing unit, respectively 64

3.4 Comparison of DL-TOs trained without/with a Compliance Predictor as a 3rd discriminator. MSE is the average mean squared error. $e_{V\%} \leq 5\%$ is the percentage of Deep Learning (DL)-designs showing a volume fraction less than 1.05 the SIMP-designs' one. $|e_{C\%}| \leq 10\%$ and $|e_{C\%}| \leq 5\%$ are the percentages of DL-designs within ± 1.1 and ± 1.05 the compliance achieved by SIMP, respectively. $|\Delta Nbr_{bars}| \leq 2$ is the percentage of DL-designs conforming with the Nbr_{bars} constraint within an error margin of ± 2 bars. Adding a Compliance Predictor improved the generated designs' compliance by **21.3%** and **45.4%** for an error margin of 10% and 5%, respectively. 67

5.1 Macro average accuracy score of Minimum bar thickness (th_{min}), len_{max} , and Θ_{min} . . 118

5.2 Our approach versus the state of the art. A blank cell means that the concerned characteristic does not apply for the method. A “+” sign defines the presence of the characteristic; the higher the number of the “+” sign, the better the method is when it comes to this characteristic. A “-” sign defines the absence of the characteristic. . 154

List of Figures

1	Reprinted/Adapted from “Computer Aided Design and Manufacturing, Chapter 1, figure 1.7, page 10, figure 1.8, page 11 and figure 1.9, page 11” [3]	4
1.1	Design for Additive manufacturing process as described by Martin Leary. Reprinted from “Design for Additive Manufacturing, Chapter 3, figure 3.5, page 39” [7].	8
1.2	AM constraints.	9
1.3	A design is represented as a distribution of elementary material densities x_i with $0(\text{absence of material}) < x_{min} \leq x_i \leq 1(\text{presence of material})$	11
1.4	An example of the mesh-dependency setback of TO. SIMP outputs two different geometries for two different mesh sizes for the same input mechanical conditions.	11
1.5	A simple neural network of three layers: an input layer (consisting of three units x_1 , x_2 and b), a hidden layer (consisting of four units h_1 , h_2 , h_3 and b) and an output layer (consisting of one unit \hat{o}). w_{ij}^l is the weight of the connection between unit i from layer l to the unit j from layer $l + 1$. b^l is the bias from layer l . \hat{o}_1 is the output’s predicted value and o_1 the ground truth. The forward pass can be summarized by $h_i^l = \sum_{j=1}^n (w_{ij}^{(l-1)} \times x_j) + b_i^{(l)}$ with n the number of units in the layers $l - 1$. The loss function L compares \hat{o}_1 and o_1 . In the backward pass, weights are updated using the following equation $w_{ij}^l = w_{ij}^l - \alpha \times \frac{\partial L}{\partial w_{ij}^l}$ with α being the learning rate or the step size; it defines how quickly the solution converges.	25
1.6	Schema of a convolutional neural network.	28
1.7	Diagram of a Conditional Generative Adversarial Networks (GAN).	28

2.1 Deep Learning Design Generation considering both Mechanical and Geometric/Manufacturing Constraints. The training procedure is based on the GAN framework. 38

2.2 Input to the generator $G(C, \theta)$. The input consists of (1) the mechanical constraints: Boundary conditions (BC_x, BC_y), the loads (F_x, F_y) along the x and y axis and the volume fraction Volume Fraction (V) (2) and the geometrical constraints: the total number of bars (Nbr_{bars}), the maximum bar length (len_{max}), the minimum overhang (Θ_{min}), the minimum bar thickness (th_{min}). Following option three in section 2.2, the scalar geometrical constraints are transposed to 2D matrices to be concatenated with the mechanical ones. Consequently, the input to $G(C, \theta)$ is a nine-channel 2D image. 41

3.1 Training Procedure. (From left to right)

1) The mechanical constraints (Boundary conditions, loads configuration and volume fraction) are concatenated with the geometrical condition (the Nbr_{bars}). The later are input into the Generator, which outputs the designs (Generated Designs). The generated designs are concatenated with all of the input conditions and fed to the traditional discriminator, which predicts a score (the adversarial loss). Then they are concatenated with only the mechanical conditions and fed to the counter discriminator, which predicts their corresponding Nbr_{bars} . The counting loss is the MSE between the input Nbr_{bars} and predicted one. Finally, the quality of generated designs is compared versus the real ones (the Reconstruction error). All three loss are summed (a weighted sum) and the final score is fed-back to the generator to update its weights. 2) The Traditional discriminator is trained at two levels: the first one with the real designs alongside the input conditions and the second one with the generated designs alongside the input conditions. The adversarial loss is fed-back to the network to update its weights. 3) The counter-discriminator is trained only with the real designs and their mechanical conditions. The counting loss is fed-back to the network to update its weights. 51

3.2 The architecture of the Res-U-Net Generator. The network can be divided into an encoder, a bridge, and a decoder. The encoder is formed of 4 blocks, each consisting of a down-sampling layer (a convolution of stride 2) and a residual unit([6]). The decoder comprises five blocks, each consisting of an up-sample layer (a transpose convolution of stride 2) and a residual unit followed by a convolution of kernel size 1×1 and a sigmoid activation. The bridge connection has the same architecture as an encoder-block and combines the encoder with the decoder.
A residual unit block is a sequence of two blocks, each consisting of a batch normalization[6] followed by a ReLU activation and a convolution of kernel size = 3×3 and stride 1. Its input is summed with its output via an identity mapping connection. An identity mapping connection consists of a convolution of kernel size = 1×1 , a stride of 1, and padding of 0 followed by a batch normalization layer. 52

3.3 Input to SIMP. to output a 2D design, SIMP takes as input the dimension of the design space (height n_x and width n_y), the mechanical constraints (the boundary conditions Boundary Conditions (BC), the loads Forces (F), and the volume fraction V), the material criteria (the Young modulus E and the Poisson ration ν), finally the mesh-independency filter r_{min} and the penalization power p 57

3.4 Types of bars in a design. The design shown here is clamped from the upper-left edge and loaded with two punctual external forces located in the bottom corners tilted 7 and 38 degrees. Indeed, It has five clamped (red) bars, two externally loaded (green) bars outgoing from two nodes corresponding to the forces locations and six internal-transmission (blue) bars, thus, a total of 13 bars. 57

3.5 Input of the DL-TO. The boundary conditions (BC_x, BC_y), load configurations (F_x, F_y), volume fraction V and geometrical constraint Nbr_{bars} (i.e. total number of bars) are formulated as a six-channel image. 57

LIST OF FIGURES

3.6 This figure compares the aesthetics, volume fraction ($e_{V\%}$), compliance ($e_{C\%}$) and geometry (ΔNbr_{bars}) of the original (SIMP-based) versus generated (DL-based) designs with and without Threshold. In both cases, DL-designs are barely indistinguishable, in terms of shape, from SIMP-designs, achieve lower Volume Fractions ($e_{V\%} \leq 0$) and respect the Complexity constraint ($|\Delta Nbr_{bars}| \leq 2$). However, while the Compliance of DL-designs is higher than SIMP-designs before the threshold, it is significantly reduced after the threshold. 58

3.7 Distribution of the reconstruction error in the train and test sets before and after threshold. The Mean Error Squared $MSE = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2$ where x_i and \hat{x}_i are the original (SIMP-based) and generated (DL-based) design. 61

3.8 Overall Geometrical Performance of DL-TO. This figure plots the distribution of the Nbr_{bars} difference before and after threshold in the train and test sets. 61

3.9 Relative error of the volume fraction 62

3.10 Relative error of the compliance 62

3.11 Overall Mechanical Performance of DL-TO. This figure plots the distribution of the relative error of Volume fraction and compliance before and after threshold in the train and test sets. 62

3.12 This figure shows the relative prediction error of the DL-Compliance-Predictor and compares its computational efficiency versus the traditional FE-based-compliance-calculator. 65

3.13 The distributions of the relative errors of volume fraction and compliance ($e_{V\%}$ and $e_{C\%}$) and the difference between the number of design components (ΔNbr_{bars}) computed over the test set for the DL-TO trained with two versus three discriminators. 68

3.14 Comparison between the generated designs from two DL-TO models, designs in Fig. 3.14(a) are generated by a DL-TO trained with an additional DL compliance predictor. The generated designs' mechanical performance (the compliance) is improved after integrating the DL compliance predictor as a third discriminator. 69

3.15 Tailoring the geometry of a design. a) The mechanical constraints (BC , F and V) are fixed and the geometrical condition Nbr_{bars} is variable. b) BC and F are fixed while Nbr_{bars} and V are variable. 71

3.16 Compliance versus Nbr_{bars} variation for a set of fixed mechanical constraints. The maximum admissible compliance is 450 J (the green line). The minimum admissible Nbr_{bars} here is 15 bars, and the lowest compliance is achieved at $Nbr_{bars} = 30$ 72

3.17 The effect of a local variation in the Nbr_{bars} input matrix. Six modifications are considered: a) the Nbr_{bars} input matrix is altered in the top-left by a matrix of 6×6 with the value equaling to 5, b) in the middle-left by a matrix of 6×6 with the value equaling to 5, c) in the bottom-left by a matrix of 6×6 with the value equaling to 5, d) in the top-left by a matrix of 21×21 with the value equal to 5, e) in the middle-left by a matrix of 21×21 with the value equal to 5 and f) in the bottom-left by a matrix of 21×21 with the value equaling 5. 72

3.18 Sample of DL-TO designs generated without versus with constraints on design area's boundary as input (V_{input}). All designs are threshold. a) Additional material is added on the right edge. b) A hole (passive element) of radius $r = 30$ is enforced at the bottom left of the design space. d) An active element of radius $r = 15$ is enforced at the centre of the design space. The BC (in red) and F (in green) conditions are only shown on the first row. 74

3.19 A sample of SIMP designs generated without versus with constraints on design area's boundary as input (V_{input}). All designs are threshold. a) Additional material is added on the right edge. b) A hole (passive element) of radius $r = 30$ is enforced at the bottom left of the design space. d) An active element of radius $r = 15$ is enforced at the centre of the design space. The BC (in red) and F (in green) conditions are only shown on the first row. 75

3.20 Interface of GAIMD. GAIMD is an open-accessible web application that takes the mechanical conditions: the boundary conditions (BC), the loads (F), and the volume fraction (V) and the geometrical condition (Nbr_{bars}) and generates a 2D design using a DL-TO and optionally the SIMP method. Only edge nodes (blue nodes) from the design space can be selected to be the BC and F nodes. 76

LIST OF FIGURES

3.21 Interface of GAIMD. GAIMD is an open-accessible web application that takes the mechanical conditions: the boundary conditions (BC), the loads (F), and the volume fraction (V) and the geometrical condition (Nbr_{bars}) and generates a 2D design using a DL-TO and optionally the SIMP method. Only edge nodes (blue nodes) from the design space can be selected to be the BC and F nodes. 77

4.1 The workflow is divided into two global steps. The first step consists of consolidating the target training dataset Geometric and Mechanical Computer Aided Designs dataset (GMCAD). The second step involves training the DL-AM-TO, a DL model that generates designs from input mechanical and geometrical constraints. 86

4.2 Example of the evaluation metrics computation of BC-DL-Predictor. This figure compares a predicted to a true BC matrix. The total number of predicted BC nodes is correct, while their locations are erroneous. This error is detected by the metric $e\%_{locatins-BC-nodes}$ which counts the number of BC nodes in the correct locations, here none. 86

4.3 In this figure, the design space (3×3) is subjected to 2 loads: \vec{F}_1 on the top-left edge with $\theta = 45^\circ$ and \vec{F}_2 on the bottom left edge with $\theta = 90^\circ$. The 2D load matrix (4×4) consists of null-values everywhere except for the loaded nodes set to their orientations. Next, since we only deal with edge-like loads in this work, we reduce the load matrix to the circumference nodes. Finally, the 1D-load vector is normalized (a division by 180). To facilitate the angles prediction, we augment the load matrices by a constant filter of kernel size 5×5 , but for presentation purposes, the filter size shown in the figure is 2×2 . 87

4.4 A sample of the synthetic dataset $DB2$, inspired from SIMP layouts, with various geometric constraints: minimum/maximum element-length, minimum/maximum element-thickness, minimum overhang, and the number of elements. The minimum overhang is computed with respect to the build direction \vec{n} . Here, \vec{n} is along the y-axis. NB: the overhang is the angle between the normal vector of an element (purple arrow) and the build direction \vec{n} (pink arrow). 89

4.5 Mechanical Conditions Predictors Performance over the train and test set. 91

LIST OF FIGURES

4.6 Comparison between the true and predicted boundary conditions (BC in red) and loads' locations and orientations (F in green). The BC and F locations are predicted with high precision. Most F orientations predictions differ from the true values within $[-20^\circ, +20^\circ]$. NB: The F orientations are in the anti-clockwise direction. 92

4.7 GMCAD's Descriptive Statistics. This figure shows the distributions of the geometrical (Min/Max Length/Thickness, Number of elements, Min Overhang) and mechanical variables (Number of BC nodes and F orientations) of GMCAD. 93

4.8 Synthetic Designs with mechanical and geometrical constraints. In this figure, we show a sample of synthetic designs alongside their predicted mechanical conditions. We also evaluate their mechanical performance (Compliance in Joules). 94

4.9 Mechanical Evaluation of synthetic designs. In this figure, we vary one geometric constraint and evaluate the mechanical performance (Compliance in Joules) of the resultant design. The geometric constraints are the minimum Overhang, the minimum constant bar-thickness (i.e., all elements have the same thickness), and the minimum variable bar-thickness (here, the contour bars have a thickness twice the inner bar). 95

5.1 Training Procedure 101

5.2 Distribution of the geometrical constraints values in the train and test sets 103

5.3 Performance of the geometric discriminators showing the predicted vs the true values of th_{min} , len_{max} , Θ_{min} , & Nbr_{bars} , from left to right, respectively, for the train, validation, and test sets. The worst performant discriminator is clearly the th_{min} predictor. 105

5.4 DL-AM-TO's performance when trained with and without th_{min} 107

5.5 Comparison between the real and generated designs in their full and skeleton formats on the test set. 108

5.6 The distribution of the geometrical metrics ($e_{len_{max}\%}$, $\Delta\Theta_{min}$, ΔNbr_{bars}) manually measured over 100 designs of the test set. 108

5.7 DL-AM-TO’s response versus increasing len_{max} & Θ_{min} . In (a), to comply with len_{max} , the skeleton’s layout changes drastically with the increase of len_{max} . The same behavior is noticed with the variation of Θ_{min} , at a point where additional bars start appearing (circled in red, 4th design in (b)). *The pink arrow is the build orientation, the red arrow is the normal of the beam.* 109

5.8 Ordinal regression models’ rank consistency. 113

5.9 A Visual explanation of how CORN model predicts the ordinal class y_i of an input x_i . 113

5.10 A Visual explanation of how the CORN loss is computed using the conditional training subsets. Reprinted from [8]. Assuming three training examples $x^{[1]}, x^{[2]}, x^{[3]}$ with three rank labels $y = [y^{[1]}, y^{[2]}, y^{[3]}] = [1, 3, 4]$, the overall loss $L(X, y) = \frac{1}{\sum_i |y_i|} \sum_i L_i = \frac{1}{3+2+2+1}(L_1 + L_2 + L_3 + L_4)$ 114

5.11 Distribution of the geometric intervals of len_{max} , th_{min} , & Θ_{min} , from left to right respectively, in the training sets of the len_{max} , th_{min} , & Θ_{min} CORN discriminators, from top to down respectively. 115

5.12 Performance of the geometric discriminators showing the predicted vs the true classes of th_{min} , len_{max} , & Θ_{min} , from left to right, respectively, for the train, validation, and test sets. The red line corresponds to the predictions being equal the true class. The green lines correspond to the previous and the following class; they represent the admissible misclassified predictions. 117

5.13 The confusion matrix of the th_{min} CORN discriminator computed on the test set. . . 118

5.14 The confusion matrix of the len_{max} CORN discriminator computed on the test set. . . 119

5.15 The confusion matrix of the Θ_{min} CORN discriminators computed on the test set. . . 120

5.16 Uncertainty quantification computed over the predictions of the len_{max} geometrical discriminator on the test set. 122

5.17 Uncertainty quantification computed over the predictions of the Θ_{min} geometrical discriminator on the test set. 123

5.18 Uncertainty quantification computed over the predictions of the th_{min} geometrical discriminator on the test set. 123

LIST OF FIGURES

5.19 Geometries proposed by the DL model trained to tailor Θ_{min} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different len_{max} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing's computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}). 127

5.20 Geometries proposed by the DL model trained to tailor Θ_{min} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different len_{max} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing's computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}). 128

5.21 Geometries proposed by the DL model trained to tailor len_{max} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different len_{max} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}). 130

5.22 Geometries proposed by the DL model trained to tailor Nbr_{bars} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different Nbr_{bars} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}). 132

5.23 Particular geometries generated by a DL model trained to control the Θ_{min} along the mechanical constraints. 133

LIST OF FIGURES

5.24 Distribution of the Structural Similarity (*SSIM*), Peak Signal to Noise ratio (*PSNR*), and the reconstruction error (the Mean Squared Error, *MSE*), from left to right respectively, computed on the test set, for three variants of the designs: raw, threshold, and skeletons. the raw designs are DL-AM-TO’s outputs without any transformation. 133

5.25 Distribution of the geometrical conformity metrics, $\Delta_{len_{max}}$, $\Delta_{th_{min}}$, $\Delta_{\Theta_{min}}$, and $\Delta_{Nbr_{bars}}$, from left to right respectively, with $\Delta_X = X_{generated} - X_{input}$; $X \in len_{max}, th_{min}, \Theta_{min}$, and Nbr_{bars} . The green line corresponds to $\Delta_{len_{max}} = \pm 0.05units$, $\Delta_{th_{min}} = \pm 0.01units$, $\Delta_{\Theta_{min}} = \pm 5^\circ$, and $\Delta_{Nbr_{bars}} \pm 2bars$ in the $\Delta_{len_{max}}$, $\Delta_{th_{min}}$, $\Delta_{\Theta_{min}}$, and $\Delta_{Nbr_{bars}}$ plots, respectively. 134

5.26 Distribution of the mechanical metrics, the relative error of compliance ($e_{C\%}$) and the relative error of volume fraction ($e_{V\%}$). The green line corresponds to $e_{C\%} = 20\%$ and $e_{V\%} = 10\%$ in the $e_{C\%}$ and $e_{V\%}$ plots, respectively. 134

5.27 Tailoring th_{min} 138

5.28 In this figure, we show the same mechanical constraints and changed the len_{max} 139

5.29 In this figure, we show the same mechanical constraints and changed the Θ_{min} 140

5.30 In this figure, we show the same mechanical constraints and changed the Nbr_{bars} 141

5.31 In this figure, we show the same mechanical constraints and changed three different geometrical constraints each at a time. Input $Nbr_{bars} = 5$, input $len_{max} = 1u$, & input $\Theta_{min} = 18^\circ$ 142

5.32 In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (without and with the supports) and the estimated design’s mass and material filament’s length needed, outputted by Cura[10], and finally the printed end-design. The designs’ Nbr_{bars} are 7, 17, and 29, from left to right, respectively. The $\Theta_{minprinter}$ is set to 59° for which support structures are added. 145

5.33 In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (without and with the supports) and the estimated design’s mass and material filament’s length needed, outputted by Cura[10], the displacements computed by Patran-Nastran in mm , and finally the printed end-design. The designs’ len_{max} are $0.54u$, $0.75u$, and $1.08u$, from left to right, respectively. The $\Theta_{minprinter}$ is set to 45° 147

LIST OF FIGURES

5.34 In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (without and with the supports) and the estimated design’s build time B_T , mass M , and material filament’s length needed L , outputted by Cura[10]; $B_{T_{real}}$ and L_{real} are the build time and filament length measured in the printing phase (figure 5.36), the mechanical performance illustrated with the displacements in mm outputted by Patran-Nastran, and finally the printed end-design. The designs’ Θ_{min} are 0° , 30° , 45° , and 54° , from left to right, respectively. The $\Theta_{minprinter}$ is set to 45° 149

5.35 In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (with the supports) and the estimated design’s build time B_T , mass M , and material filament’s length needed L , outputted by Cura[10] with $\Theta_{minprinter} = 60^\circ$; $B_{T_{real}}$ and L_{real} are the build time and filament length measured in the printing phase (figure 5.36), and finally the mechanical performance illustrated with the displacements in mm outputted by Patran-Nasteran. 150

5.36 Designs of figures 5.34 and 5.35 printed by the 3D printer Creality Ender 3. 151

5.37 DL-AM-TO’s architecture and computational cost detailed. This figure shows the number of DL-AM-TO’s trainable parameters, their memory usage in Mega bytes (Mb), their number of floating point operations ($GFLOPs$, giga $FLOPs$), their Multiply-Accumulate operations ($MMAC$, mega $MACs$), and Direct Memory Access operations ($GDMAs$, giga $DMAs$) on the model’s forward pass. This information is outputted by <https://pypi.org/project/torchscan/>, a python torch library to summarize a torch model’s computational cost. 152

A.1 The confusion matrices of the th_{min} geometric discriminator. 182

A.2 The confusion matrices of the len_{max} geometric discriminator. 183

A.3 The confusion matrices of the Θ_{min} geometric discriminator. 184

LIST OF FIGURES

Acronyms

Θ_{min} Minimum overhang.

BC Boundary Conditions.

C Compliance, the energy of deformation.

F Forces.

Nbr_{bars} Number of bars.

V Volume Fraction.

len_{max} Maximum bar length.

th_{min} Minimum bar thickness.

AM Additive Manufacturing.

AM-FE-TO Additive Manufacturing driven Finite Elements based Topology Optimization.

CAD Computer Aided Design.

cGAN conditional Generative Adversarial Networks.

CNN Convolutional Neural Networks.

CPU Central Processing Unit.

GPU Graphics Processing Unit.

DfAM Design for Additive Manufacturing.

DL Deep Learning.

DL-TO Deep Learning-based Topology Optimization.

DL-AM-TO Deep Learning Additive Manufacturing-driven Topology Optimization.

FE Finite Elements.

FEA Finite Elements Analysis.

FE-TO Finite Elements based Topology Optimization.

GAN Generative Adversarial Networks.

GMCAD Geometric and Mechanical Computer Aided Designs dataset.

ML Machine Learning.

NN Neural Networks.

SIMP Solid Isotropic with Material Penalization.

TO Topology Optimization.

Introduction

The growing need for fast, organic, versatile, cost- and material-efficient products in the industrial world drives the research to develop design approaches accounting for these criteria altogether. Topology optimization (TO) falls in the category of these design approaches. It allows the generation of shapes with curvatures and fine details, given parameters such as loads, boundary conditions, and a volume fraction.

Complementary comes additive manufacturing (AM), which allows the printing of any form. However, this synergy is less idealistic than it looks. Indeed, AM requires the design's shape to comply with some geometrical criteria, like dealing with curvatures, overhanging patterns, the need for supports, thin features, Etc., which are hardly integrated into TO's formulation [1]. Thus, designing optimal printable parts requires the intervention of experts to interpret the shapes proposed by TO to comply with the manufacturing criteria. Nonetheless, this reinterpretation phase can compromise the initial design's optimality, is time-consuming, and depends on the engineer's expertise.

Consequently, with the flourishing role of AM in the industry, it is further interesting to find a method that simultaneously considers mechanical and geometrical constraints at the conceptual level or the discipline of Design for Additive Manufacturing (DfAM). A recent survey by Subedi et al. ([2]) has shown that half of TO practitioners regret the absence of geometric and manufacturing-related plugins in TO's software. This survey's results are unsurprising, for "manufacturing is one of the fundamental constitutions of a nation's economy [3]." As a matter of fact, the top ten manufacturing countries in the world are China (with 28.7% of global manufacturing output), the United States (US) (16%), Japan (7.5%), Germany (5.3%), India (3.1%), South Korea (3%), Italy (2.1%), France (1.9%), United Kingdom (1.8%) and Indonesia (1.6%) (Global Upside website); these countries constitute the top most powerful economic forces in the world. Consequently, research has been focusing lately on improving the chain of manufacturing from defining guidelines with consortium and standards to design approaches, to manufacturing processes and materials, to production, and finally to certification and marketing.

Indeed, manufacturing systems started as craft systems; the end products are used for essential functions. In the 18th century, the English manufacturing systems were established; the machines replaced humans in repetitive and laborious tasks, and the objective shifted from functionality to profit. At the beginning of the 19th century, the American Manufacturing system introduced mass production: interchangeable parts, assembly, and distributional manufacturing emerged; productivity and quality

of production became central objectives. The manufacturing industry proliferated during this period until saturating the market and reducing profit margins. Thus, came lean manufacturing to overcome these problems; waste in production was eliminated, and product costs were reduced. Recently, we are in the sustainable manufacturing system, a system that is aware of environmental problems (global warming, scarcity of natural resources, carbon footprint, Etc.).

Along with their evolution, manufacturing systems replaced human involvement in the process one step at a time. Nowadays, human-machine interaction consists of information and material flow activities. Humans are still present in the product lifecycle: from the design stages to manufacturing and marketing. Currently, the goal is to find the optimal solutions to balance and harmonize this interaction to ensure a performant manufacturing system.

The manufacturing systems' performance was proven to be optimized by computer-aided technologies (CATs) in terms of lead time and costs.

While CATs are present in the design, analysis, manufacture, and assembly phases, the phase that shows the highest impact on the KPI is the design phase. The design phase costs the least, has the highest impact on the overall product cost ($>70\%$), depends on the least on hardware, allows handling of defects at a lower cost, and consequently has the highest added value in terms of products competitiveness (Figure 1). A defective design will cost the company time and money to circumvent these defects at the manufacturing and assembly phases. Defects detected in the design phase save the company time and money (about 75% of the fixing costs) spent on eliminating the impacts of these defects all along the product life cycle chain. From this, the concept of "first-time correct" has emerged. The goal is to find the most optimal and free-from-defects design at the very early phase of the product lifecycle [3].

Thus, much research has been put into exploring new design approaches to improve the design phase and adapt it to the production constraints. While the top design approaches are the density-based approach and the level-set approach, the method that is mainly implemented and found in about 70% of the design software is the density-based approach, for it is easily implemented and requires less computational power. These design approaches are far from perfect, and with the increasing pace of technology, they are having difficulty keeping up with the new manufacturing processes. AM is one of the emerging manufacturing processes and is most attractive in several industries, particularly aerospace and automotive. It allows the production of complex organic shapes and prevents the need

INTRODUCTION

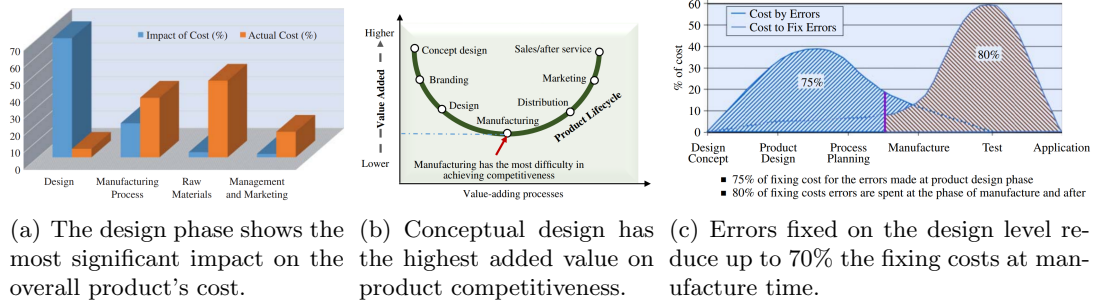


Figure 1: Reprinted/Adapted from “Computer Aided Design and Manufacturing, Chapter 1, figure 1.7, page 10, figure 1.8, page 11 and figure 1.9, page 11” [3]

for assembly. This makes parts cheaper (with many wholes, i.e., less material) and accelerates the production chain (no assembly needed). Also, AM allows for customization, particularly in the medical implant sector, where this characteristic is crucial; every person is unique and would need a unique implant shape.

Thus, the research tried to adapt the design standards and approaches to consider the new constraints that this new production process requires. In state-of-the-art, several experiments were conducted to create a worksheet with recommendations for novice AM designers to help them create compliant shapes that are less prone to fail during the printing phase. The second part worked on integrating AM constraints into the design approaches. However, their efforts were limited by the nature of most manufacturing constraints; most of them are geometrical and informal rules coming from experience rather than mathematical logic; and by the additional computational power needed for the integration of these constraints, when possible; several approaches reported convergence issues when the constraints become further complex. The third part worked on replacing the design approaches with Deep Learning models and left the adaptation of the shape to AM to the designer. The former approach accelerated the design phase without resolving the issue of getting stuck in the reinterpretation phase.

Hence, the problem is the following: How can we integrate manufacturing constraints into the design phase while ensuring convergence to alleviate getting stuck in the reinterpretation phase?

Our solution is to merge the previous two approaches and get the best of both worlds. We will explore Deep Learning generative capabilities, especially convolutional neural networks, to learn geometric manufacturing-related constraints and to allow the integration of two types of constraints: mechanical

and geometrical, concurrently at the same level of the design space. Our Deep Learning Additive Manufacturing-driven topology optimization approach is trained within a Generative Adversarial networks framework, a recent state-of-the-art generative approach for neural networks.

Consequently, this thesis is organized as follows: Chapter 1 details the state of the art of Additive Manufacturing, the design approaches known as Topology Optimization, the state-of-the-art approaches developed to accelerate the design for the Additive Manufacturing process, the Deep Learning, the convolutional neural networks, and the GAN training framework. Chapter 2 describes the general methodology of DL-AM-TO. The first approach of DL-AM-TO, DL-TO, a proof of concept, is presented in Chapter 3; it integrates the mechanical constraints and one geometrical constraint, the number of bars, at the same conceptual level and generates a 2D design. In chapter 4, the training dataset GMCAD, the Geometric Mechanical CAD dataset, is consolidated. In chapter 5, DL-AM-TO is detailed and put into action. DL-AM-TO takes the mechanical and four geometrical manufacturing-related constraints as input, the number of bars, the minimum overhang, the maximum bar length, and the minimum bar width, and generates a 2D design. This chapter validates DL-AM-TO's capability to tailor a design's geometry. The geometries generated by DL-AM-TO were generated and validated mechanically and geometrically. Finally, chapter 5.9 summarizes the methodology and its outcomes and presents future perspectives.

Chapter 1

State of the art

Content

1.1	Context	8
1.1.1	Additive Manufacturing (AM)	8
1.1.2	Topology Optimization (TO)	11
1.2	Objective	13
1.2.1	State of the art of DfAM acceleration approaches	13
1.2.2	Our approach	19
1.3	Ingredients	23
1.3.1	Machine Learning (ML)	23
1.3.2	Fully Connected Neural Networks (NN)	24
1.3.3	Deep Learning (DL)	26
1.3.4	Convolutional Neural Networks (CNN)	26
1.3.5	Generative adversarial networks (GAN)	27
1.3.6	Conditional Generative adversarial networks (cGAN)	29
1.4	French Summary	31

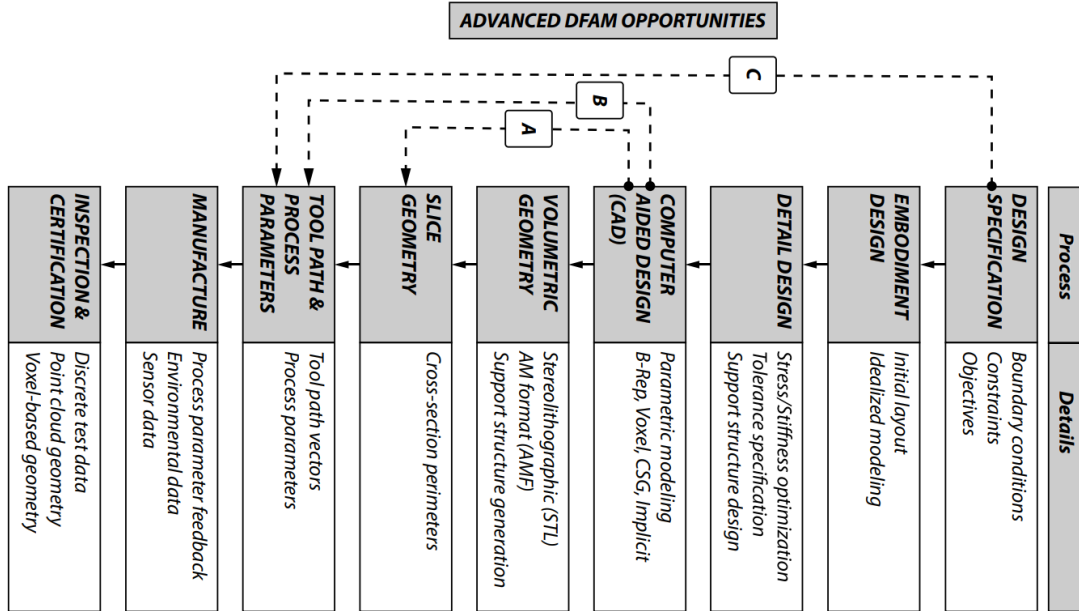


Figure 1.1: Design for Additive manufacturing process as described by Martin Leary. Reprinted from “Design for Additive Manufacturing, Chapter 3, figure 3.5, page 39” [7].

This section introduces Additive Manufacturing (Additive Manufacturing (AM)), topology optimization (Topology Optimization (TO)), and the design for AM (DfAM) approaches. It addresses their limitations and proposes a solution that leverages the usage of Deep Learning (DL) generative methods to integrate geometrical manufacturing-related constraints at the conceptual level of design.

1.1 Context

1.1.1 Additive Manufacturing (AM)

Additive manufacturing (AM) or as known as 3D printing, is an all-digital manufacturing technology that fabricates products by sequential element-wise addition of material based on a discrete digital geometry.

For several reasons, AM has gained hype in the industrial world in the last 50 years. First, it offers a higher degree of freedom in design ; complex geometries challenging to fabricate with conventional subtractive manufacturing (SM) are no longer problematic. Moreover, optimal layouts (in terms of material and functionality) are not compromised anymore to ensure manufacturability [11].

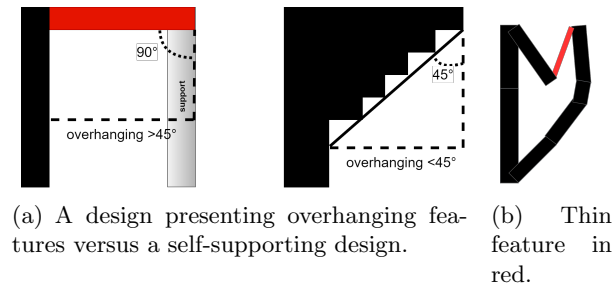


Figure 1.2: AM constraints.

For high production volume-based manufacturing, AM dominates SM in products where complexity plays a role, especially that AM prints the part at once, thus eliminating the forging and joining processes. The products falling in this category are mainly prostheses and implants.

Mass customization is one advantageous property of AM; AM allows the production of individually customized products through an agile and flexible process, which is not the case for conventional manufacturing where molds need to be fabricated every time a new custom product is required.

AM allows mechanisms that further reduce the need for material; hence, the cost of the part fabricated, such as infills[12], multi-degree-of-freedom AM systems that reduce the need for support structures, and the laborious post-processing [13].

AM reduces the fabrication's waste and hence cost, particularly in low-volume production lines, and allows customized products of complex geometries using custom materials. Most importantly, it leverages generative design approaches due to its digital nature [7, 14]. Thus, AM is synergetic with TO; TO proposes complex geometries printable via AM processes.

Design for AM (DfAM) summarizes the digital design process of a product. It consists of six major steps, as detailed in Chapter 2 of [7] (Fig. 1.1):

1. Design specification: design requirements, constraints, and objectives are defined.
2. Embodiment design: digital design methods, mainly TO approaches, are applied to obtain the first layout.
3. Detail design: computer-aided engineering (CAE) analysis, parametric optimization, tolerance optimization, Etc.
4. Computer-aided design (Computer Aided Design (CAD)) creation: the designer draws the digital design inspired by the geometry proposed by the previous steps while considering geometric

manufacturing-related constraints.

5. **Manufacture:** it includes the generation of the volumetric geometry, which is a data format compatible with the AM machine. This volumetric geometry is then sliced; for AM machines prints a design layer by layer, in other terms, slice by slice, such that each layer has a constant material thickness. Afterward, special software optimizes the tool path and process parameters. Finally, the AM machine prints the design.
6. **Inspection and certification:** this step includes functional and quality assessment of the design before delivery.

This process is iterative and time-consuming for several reasons.

Despite the success of AM, not all the designs could be manufactured. Steep curvatures, overhanging patterns, the need for supports, tiny chamfers, thin features, and other geometrical constraints are still a hurdle[1] (Fig.1.2). Moreover, AM constraints lack standardization [15] and are contradictory between processes, materials, and applications. For example, in a metal Powder Bed Fusion (PBF) process, support structures have a double role. They support overhanging features and help dissipate the laser heat to prevent thermal deformation or cracks due to residual stress. On the contrary, in polymer PBF, the unsintered powder material provides support for the overhanging features, and hence support structures are not needed [16]. Thus, an optimal design identified in steps two and three is prone to collapse during manufacture in step 5 if the manufacturing process changes. Similarly, a modified design created in step 4 might not pass the certification in step 6.

Second, the TO are finite-elements (Finite Elements (FE)) based methods, hence computationally expensive.

Third, in the CAD creation phase, the geometry is usually modified by the designer to comply with the manufacturing geometrical constraints. Thus, the starting optimality computed in the third phase might get degraded into an acceptable level, obliging the designer to re-draw a new shape and get stuck in a loop.

Furthermore, manufacturing constraints are mostly geometric ones that are, at best, approximated analytically, and integrating them into FE based TO, i.e., at the second phase, usually compromises its convergence and limits its freedom[17, 18].

Finally, a recent survey[2] has shown that half of TO practitioners regret the absence of geometric and

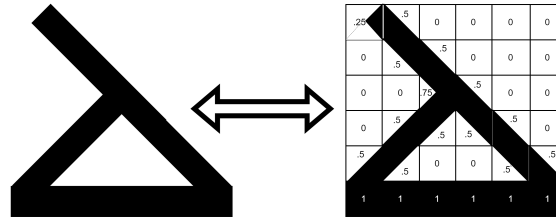


Figure 1.3: A design is represented as a distribution of elementary material densities x_i with $0(\text{absence of material}) < x_{min} \leq x_i \leq 1(\text{presence of material})$.

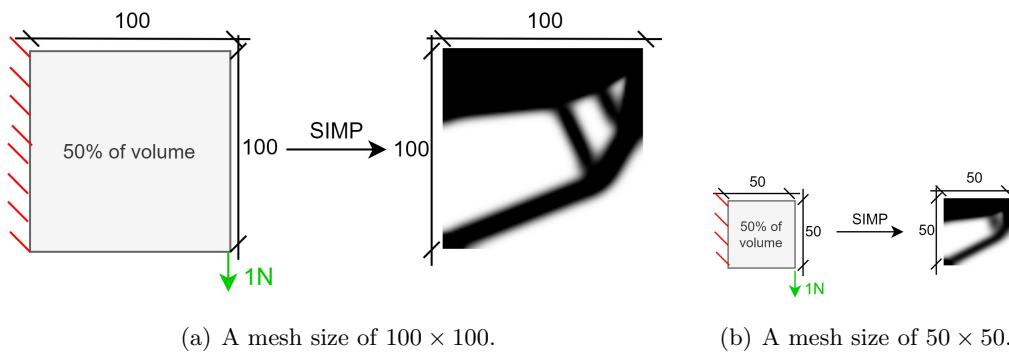


Figure 1.4: An example of the mesh-dependency setback of TO. SIMP outputs two different geometries for two different mesh sizes for the same input mechanical conditions.

manufacturing-related plug-ins in TO’s software. Thus, research is focusing on integrating geometric and manufacturing constraints into TO’s formulation.

Consequently, researchers tried finding workarounds to accelerate the DfAM process. The following section details the different approaches found in state of the art.

1.1.2 Topology Optimization (TO)

Structural optimization (SO) is defined as the process of finding the optimal material distribution within a physical volume domain to support the applied loading conditions and other constraints, e.g., increasing stiffness, reducing stress, reducing displacement, Etc. There is three major SO approaches (i) size, (ii) shape, and (iii) TO [19].

In size/shape optimization, the designers are restricted to manipulating the structural elements’ size/ shape between their limits. However, if they are unaware of the optimal structure’s shape/ size limits beforehand, then TO is their sole solution. TO is the most general form of SO [19]. It simultaneously addresses the topology, shape, and sizing problems and ensures efficient material use.

The increase in raw material cost, the environmental challenges (like energy consumption), the rise of computational power, and the development of advanced programming methods made TO a prevalent discipline in industrial design (Querin et al. [19]; Sigmund and Maute [20]). TO approaches can be grouped into two categories: (i) Optimality Criteria (OC) like Homogenization, Solid Isotropic Material with Penalization (SIMP), level-set and (ii) heuristic methods like Computer-Aided Optimization (CAO)[19, 21, 22, 23]. For further reading about the TO approaches, one can refer to the review article of Sigmund and Maute [20]. The topmost common commercial approach is Solid Isotropic Material with Penalization (SIMP); also called the power-law approach [21, 24]. SIMP is a density gradient-based iterative method that uses the penalization scheme of the intermediate non-binary values of density material x to converge to an optimal binary design. SIMP represents a design as a distribution of discretized square material elements e (material properties are assumed constant within each e). They are modeled as the relative material density raised to some power times the material properties of solid material. SIMP approach is based on the following assumptions:

- A design is represented by a specific distribution of discretized square material elements e ;
- Material properties are assumed constant within each element e used to discretize the design domain. They are modeled as the relative material density raised to some power times the material properties of solid material;
- Variables are the element relative densities x_i . x_i represents either absence (0) or presence (1) of the material at each point of the design domain (Fig.1.3); and
- A design is physically valid as long as the power $p \geq 3$ for Poisson's ratio = 1/3 [25] and is combined with a parameter constraint, a gradient constraint or filtering techniques.

In this work, the objective function to minimize is the energy of deformation or compliance $C(x)$ in Joules (J):

$$\min_x C := U^T K U = \sum_{e=1}^N x_e^p u_e^T k_e u_e \quad (1.1)$$

subject to:

$$\begin{aligned} \frac{V(x)}{V_0} &\leq f \\ K U &= F \end{aligned} \quad (1.2)$$

$$0(\text{absence of material}) < x_{\min} \leq x \leq 1(\text{presence of material})$$

1.2. OBJECTIVE

where U and u_e are the global and element-wise displacements, F the forces vector, K and k_e are the global and element-wise stiffness matrices and N the number of e . x_{min} the minimum x (non-zero to avoid singularity), p the penalization power (typically 3). V_0 and $V(x)$ are the design domain and material volumes¹ respectively and f the volume fraction.

To efficiently solve the problem stated above, [24] adopted the optimality criterion approach. A mesh-independency filter is applied to the element sensitivities to avoid checkerboard patterns. Following OC, the design variables are updated as follows:

$$x_e = \begin{cases} \max(x_{min}, x_e - m) & \text{if } x_e \beta_e^\eta \leq \max(x_{min}, x_e - m) \\ x_e \beta_e^\eta & \text{if } \max(x_{min}, x_e - m) < x_e \beta_e^\eta < \min(1, x_e + m) \\ \min(1, x_e + m) & \text{if } \min(1, x_e + m) \leq x_e \beta_e^\eta \end{cases} \quad (1.3)$$

where m is a positive move limit, $\eta = \frac{1}{2}$ is a numerical damping coefficient and β_e is found from the optimality condition as $\beta_e = \frac{-\frac{\partial c}{\partial x_e}}{\lambda \frac{\partial V}{\partial x_e}}$ such that λ is a Lagrangian multiplier that can be found by a bisectioning algorithm and the sensitivity $\frac{\partial c}{\partial x_e}$ of the objective function is found as $\frac{\partial c}{\partial x_e} = -p x_e^{p-1} u_e^T k_0 u_e$.

To ensure the existence of solutions to the problem and avoid checker-board patterns[24], some restrictions should be introduced to the resulting design. One example is to apply a mesh-independency filter on the element sensitivities to ensure that the resulting design is mesh-independent. Thus, the sensitivity of the objective function becomes $\frac{\hat{\partial} c}{\partial x_e} = \frac{1}{x_e \sum_{f=1}^N \hat{H}_f} \sum_{f=1}^N \hat{H}_f x_f \frac{\partial c}{\partial x_f}$ where the convolution operator (weight factor) \hat{H}_f

$$\hat{H}_f = \begin{cases} r_{min} - dist(e, f) & \text{with } f \in N | dist(e, f) \leq r_{min} \text{ for } e = 1, \dots, N \\ 0 & \text{outside the filter area.} \end{cases}$$

and $dist(e, f)$ is the distance between the center of element e and the center of element f .

1.2 Objective

1.2.1 State of the art of DfAM acceleration approaches

With the emergence of 3D printing processes, mainly AM, printing designs of higher complexity and creativity became doable[1], and thus made TO further attractive in the research areas. TO aims at finding the optimal material distribution inside a design domain given a set of mechanical constraints. The resulting design can have any shape and be defined as a 2D binary image or a 3D binary voxel grid, such that the presence of a non-empty pixel/voxel means the presence of the material.

¹The design domain (V_0) is the design space defined by the designer at the beginning of the optimization; it is the maximum width and height possible of the design. The design material (V_x) volume is the space taken by the geometry at the end of the optimization. Hence, $V_x \leq V_0$.

1.2. OBJECTIVE

Approaches to accelerate the DfAM process			
Formulating AM design rules	Integration of AM constraints to Finite Elements based Topology Optimization (FE-TO)	Machine and Deep Learning Usage to assist FE-TO	Deep Learning Usage to replace FE-TO
Adam et al. (2014) [1], Booth et al. (2016) [26], Gaynor et al. (2016) [27]	Mirzendehtdel et al. (2016) [28] Allaire et al. (2017) [29], Guo et al. (2017) [30], Langelaar et al. (2017) [31], Mass et al. (2017) [32], Zhang et al. (2018) [33], Yoely et al. (2018) [34], Zhang et al. (2019) [17], Han et al. (2019) [35], Bi et al. (2020) [36], Xu et al. (2020) [37], Li et al. (2020) [38], Wang et al. (2020) [39], Matos et al. (2020) [40], Fernández et al. (2020) [41].	Oh et al. (2018) [42], Sosnovik et al. (2019) [43], Wang et al. (2020) [44], Kallioras et al. (2020) [45], Bi et al. (2020) [46], Chandrasekhar and Suresh (2020) [47], Nie et al. (2020) [48].	Ulu et al. (2016) [49], Yu et al. (2019) [50], Sharpe and Seepersad (2019) [51], Rawat and M-H Herman (2019) [52], Hoyer et al. (2019) [53], Abueidda et al. (2020) [54], Malvia (2020) [55], Rade et al. (2020) [56]

Table 1.1: Approaches to accelerate the DfAM process.

1.2. OBJECTIVE

TO gained tremendous success in the 20th century during the industrial revolution of the automotive and aerospace sectors, given its powerful potential of optimizing a structure in terms of material used while maintaining its recommended mechanical specifications and properties. Afterward, it spread its applications to a broader range of disciplines: fluids, acoustics, electromagnetics, optics, Etc. However, TO uses an iterative, FE-based method that is computationally expensive and hence relatively slow. With the new advanced TO software, outputting a single design subjected to a set of boundary conditions (BC) and load (F) configurations is a matter of seconds to days depending on the design's complexity and its surrounding constraints. This aspect is acceptable if the process is limited to this single step. However, a mechanical engineer never relies on the first design (s)he tries, especially since TO, in its commercially available form [57, 58, 59, 60, 61], does not consider geometrical constraints nor manufacturing criteria, nor other customized industrial constraints (automotive, aeronautic, hydraulic, Etc.). The mechanical engineer always needs to explore several set-ups to find the optimal topology, shape, and sizing of the design to be manufactured. (S)He must also ensure its final draft is creative, cost-efficient, and manufacturable. Hence, this accumulated iterative exploration process can become slow, especially for large-scale designs.

Accelerating DfAM's cycle is a current hot research and industrial topic. Consequently, researchers try to find workarounds to bypass these constraints and accelerate the design optimization phase. We can find in the literature four general approaches to tackle the problem (Tab.1.1).

In an attempt to identify process-independent AM-based design rules, the first line of research focused on formalizing AM guidelines. Adam et al.[1] carried out several experiments over three types of processes ([1]) and formalized design rules over gap heights and widths of transitions of non-bonded elements, lengths of overhangs, positions of islands, Etc. Booth et al. [26] created a one-page visual DfAM worksheet for novice AM users. The authors of [62] opted to modify the optimal layouts to meet the overhang constraints and obtain printable designs without the need for support structures. However, changing the optimal layout suggested by TO might compromise its intended functionality. Thus, others integrated AM constraints analytically into TO's formulation. Therefore, engineers tend to manually reconstruct a shape inspired by TO's output, implicitly considering the geometric and manufacturing constraints. This re-interpretation phase is not straightforward, can compromise the initial design's optimality, and can be time-consuming; it depends on the engineer's experience and expertise.

1.2. OBJECTIVE

The second approach benefited from the synergy between TO and AM; the former proposes organic geometries that the latter allows producing, and it integrated specific AM constraints at TO's level. Zhou et al.[63] integrated minimum length scale to TO via a computationally efficient FE-free filtering-threshold scheme. The authors of [27, 29, 32, 33, 35, 36] adapted TO approaches to account for overhang limitations; as it is one of the most general AM design-rule; and deliver self-supporting and print-ready designs. Zhang et al. [17] generalized and improved their previous work to consider directional-dependent overhang constraint and minimum length control and generated high-resolution 3D structures. Yoely et al. [34] proposed an optimization approach constraining the areas of holes and curvatures of boundaries using the B-spline representation to obtain a manufacturable design. Xu et al.[37] estimated a formulation for the hanging features and thin features and penalized the evolution of densities in the domain space accordingly via a Bidirectional Evolutionary Structural Optimization (BESO) to obtain AM friendly designs. Authors in [17, 38, 39, 40] focused on including the build orientation into TO, especially since AM part's accuracy and finishing (e.g., staircase effect, support volume, support area, number of supports required, build time, surface roughness and quality, process planning, post-processing, and cost) are affected mainly by its build orientation [40]. Fernandez et al.[41] proposed to control not only minimum and maximum member sizes but also the minimum gap between structural members to avoid the presence of a large number of thin features and small cavities.

These approaches are limited for several reasons:

- (1) AM constraints lack standardization [15] and are contradictory between processes, materials, and applications, as explained previously in section 1.1.1.
- (2) AM constraints are mostly geometric constraints that are, at best, approximated analytically, and integrating them into FE based TO compromises its convergence and limits its freedom[17, 18]. As a matter of fact, despite the previously listed advantages of TO (section 1.1.2), it suffers from a principal setback. The general shape is usually identified at the very early iterations, making it difficult for TO to modify it to account for other changes imposed by other constraints; hence, the output shape ends up a local minimum [64]. Furthermore, for density-based TO, geometry cannot be clearly defined as it depends on the mesh size (Fig.1.4). Hence, the geometric constraint controlling the design's manufacturability cannot be easily incorporated in TO's formulation. Last but not least, TO suffers from a convergence problem despite the penalization scheme; the grey regions in the output designs are

hardly interpretable physically (Fig.1.4). Moreover, TO's convergence is further compromised when the constraints increase in number and complexity; authors who tried integrating manufacturing constraints into traditional FE based TO methods reported a convergence to a design one time out of 10[65].

(3) These methods are still based on FE analysis and therefore are iterative and computationally inefficient. While it converges for simple input constraints in a few seconds, its efficiency remarkably drops when input design space (e.g., mesh size) increases and input conditions complexify. Converging to a solution can quickly explode from seconds to days. Moreover, integrating geometric and manufacturing constraints into its formulation, when possible, adds computational costs and often compromises the convergence of the method[65].

Consequently, to compensate for these limitations, other research has turned to Machine Learning (Machine Learning (ML), section 1.3.1) and DL (section 1.3.3) techniques to assist the traditional FE-TO process by speeding up the evaluation of complex quantities of interest directly into the process. DL architectures have proven efficiency and robustness in learning complex spatial correlations and extracting high-level features (including geometrical or shape-related features) from real-world images [66, 67, 68].

The third part used DL to assist traditional FE-TO [42, 43, 45, 46, 44, 47, 69, 48]. Oh et al.[42] used a Boundary Equilibrium GAN (BEGAN) to propose new and creative wheel designs followed by TO to assure mechanical validity. Sosnovik et al.[43] formulated the topological problem as an image segmentation task and built an auto-encoder, which maps an intermediate structure outputted by traditional TO solver to its final structure. Similarly, Wang et al.[44] created a super-resolution neural network (Neural Networks (NN)) for structural TO to map low-resolution intermediate designs to their corresponding high-resolution designs. Kallioras et al.[45] integrated a Deep Belief Network between the first and last phases of Solid Isotropic Material with Penalization (SIMP) TO in order to accelerate the optimization process. Bi et al.[46] replaced the gradient estimation step with a DL network and used parallel computation on multiple central processing units (CPUs) and graphical computing units (GPUs) to accelerate traditional TO. Chandrasekhar and Suresh[47] used the NN's activation functions as a representation of the SIMP density field to ensure FE mesh independence. They enhanced their previous work [47] with a Fourier space projection to control the min/max length scales (a geometrical constraint) without the need to explicitly add additional constraints to SIMP

1.2. OBJECTIVE

and automated the sensitivity computation through NN’s back-propagation [69]. Finally, Nie et al.[48] generated 2D structures using a conditional Pixel2Pixel GAN architecture: the BC are transformed into physical fields like the von Mises stress, strain energy, and displacement fields. The latter are inputted to a DL-generator to output the optimal designs.

The fourth part tried to replace completely TO’s formulation by ML/DL [49, 50, 51, 52, 53, 54, 55, 56]. Ulu et al.[49] used Principal Component Analysis (PCA) to encode high-dimensional input configuration (BC , F , etc.) to lower-dimensional features which were input into a shallow NN to output the final optimal structure. Yu et al.[50] learned the mapping from BC to a high-resolution optimal structure using two steps: an auto-encoder that encodes the boundary constraints and then outputs a low-resolution design followed by a conditional GAN that reconstructs its original resolution. On the other side, Sharpe and Seepersad[51] directly used a conditional GAN to output the optimal structure given the BC as compact vectors in order to explore other designs (with different sets of constraints) in a shorter time. Simultaneously, Rawat and M-H Herman[52] explored a conditional Wasserstein GAN to generate 2D designs given the volume fraction as the input condition. However, they encountered a mode collapse: the generator succeeded in outputting only good designs for one value of the volume fraction condition (0.4). Hoyer et al.[53] used Convolutional Neural Networks (CNN)s and the Limited memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) optimizer to build a differentiable, mesh-independent TO that outperforms traditional TO, especially in large scale problems. Abueidda et al.[54] used the ResUnet architecture to generate 2D structures with nonlinearities; the encoder compresses high-dimensional BC , (F) configurations, and volume fractions into low-dimensional matrices, and the decoder reconstructs the corresponding optimal 2D structure. Malvia [55] tested the performance of four different types of CNN and GAN-based networks in generating 2D designs given the boundary, loads, and volume fraction conditions. Their objective was to explore the potential of DL and data augmentation techniques in accelerating the TO process. Rade et al.[56] built a framework of three networks to predict the final optimal density given the initial compliance and volume fraction. The first network takes the initial compliance and volume fraction and outputs an intermediate density prediction. The latter is inserted into a Compliance Prediction Network that predicts the intermediate compliance. These two intermediate matrices are inserted back into the first network. After five iterations of the previous steps, the last density prediction is inserted into a third network that outputs the final optimal density.

1.2. OBJECTIVE

The previously listed works' main contribution was accelerating partially or totally traditional TO. Some still relied on iterative computationally expensive FE methods [43, 45] or FE analyses [48, 47, 69] and none, except for Chandrasekhar and Suresh [69], integrated any geometric or manufacturing constraints.

To sum up, accelerating the DfAM process consists of four major state-of-the-art approaches:

1. Formalizing AM design rules for engineers to consider at the CAD drawing phase, but that does not guarantee the initial design optimality and mechanical performance. Nevertheless, this approach helped formalize AM constraints for novice users and guided engineers into creating print-ready designs.
2. Integrating certain AM constraints at the FE-TO formulation. This approach allows the engineers to have a first draft accounting for both constraints, thus preventing getting stuck in a loop of re-drawing and re-testing the mechanical performance of the design. Nevertheless, not all AM constraints can be analytically formulated. Also, this approach inherits the density-based TO's flaws. (1) FE-TO identifies the general shape at early iterations, inhibiting the method of modifying the shape to take account of geometrical constraints. (2) The convergence problem is not guaranteed, especially when there are multiple constraints; it is hard to find a nash equilibrium where all conditions are validated.
3. Assisting FE-TO methods with ML and DL. This approach focuses on accelerating the TO phase of the DfAM process and does not dodge getting stuck in a loop in later phases. Like the previous approach, most of TO's problems are still inherited.
4. Replacing FE with ML and DL in TO. This approach does not integrate any manufacturing constraints and does not circumvent getting stuck in a loop in later phases of the DfAM process.

1.2.2 Our approach

This chapter reviews the state-of-the-art approaches proposed to accelerate the DfAM process. They were classified into four major approaches.

The first one consisted of creating AM guidelines for design engineers to help them in the re-interpretation phase, i.e., the CAD drawing phase. While this approach was a significant step into the adoption of design for AM, the designers still got stuck in the drawing phase due to the mechanical performance

1.2. OBJECTIVE

Advantages	Approaches to accelerate the DfAM process			
	Formulating AM design rules	Integration of AM constraints to FE-TO	ML and DL-assisted FE-TO	Full DL-TO
Speed	+	+	++	+++
Scalability			+	++
Convergence		-	+	++
Mesh-Independency	-	-	+	++
Geometrical control at the conceptual level		+		
Light generative design module in industrial software		-	+	++
DfAM process acceleration	+	++	+	++

Table 1.2: The state of the art approaches to accelerate the DfAM process. A blank cell means that the concerned characteristic does not apply for the method. A “+” sign defines the presence of the characteristic; the higher the number of the “+” sign, the better the method is when it comes to this characteristic. A “-” sign defines the absence of the characteristic.

1.2. OBJECTIVE

deterioration induced by updating the shape to comply with AM constraints.

Thus, for the reason mentioned previously, the second approach proposed integrating AM constraints into FE-TO methods so that generated designs would be compliant with both mechanical and AM constraints simultaneously. Nevertheless, this approach faced several challenges. AM constraints are primarily informal and geometrical, which makes formulating them analytically challenging, especially with FE-TO the geometry is unknown beforehand. Moreover, with FE-TO, the shape is identified at early iterations, restraining the shape from changing to account for other constraints. Finally, FE-TO has a hard time converging (i.e., finding the optimal design complying with all the constraints) with the increasing number of constraints, and when it converged, the computational power needed exploded.

The third approach modeled a hybrid TO method, they replaced some FE blocks with ML techniques. Unfortunately, this approach focused on FE-TO and not the whole DfAM process and inherited the flaws of FE like convergence and computational power.

The fourth approach totally replaced FE with DL models. This last approach accelerated TO and eliminated all FE-TO setbacks but did not integrate any AM constraints and did not impede getting stuck in a loop in later phases of the DfAM process.

Our objective is to accelerate the whole DfAM process and have the best quality/cost ratio. As we have seen in the introduction, the design phase has the lowest cost and the highest impact on the overall product cost, quality, and lifecycle. However, accelerating this phase alone is not enough, as we have seen with state-of-the-art approaches. Moreover, while Additive Manufacturing driven Finite Elements based Topology Optimization (AM-FE-TO) methods aim to accelerate the whole DfAM process by preventing repetitive iterations, the analytical formulation of AM constraints, the methods' convergence, and their computational costs are still a hurdle. On the other side, the introduction of ML and DL only accelerated one phase. Thus, we propose to get the best of both worlds in this work. This new DL approach allows manufacturing constraints' integration within mechanical ones concurrently at the same level, and any constraints, even ones lacking a mathematical definition [70] like experts' rules and knowledge while benefiting from DL's speed and scalability advantages. It is imperative to note that this approach is not intended to replace robust FE-TO but to help compensate for its difficulties when it comes to integrating various complex constraints (Tab. 1.2).

Our work's main contribution is not only accelerating the TO phase but also identifying a way to tailor

1.2. OBJECTIVE

the design’s geometry and manufacturability in order to generate a first draft design that complies with both mechanical and geometric-manufacturing constraints to avoid getting stuck in a loop of updating and testing designs and accelerate the whole DfAM process.

Thus, we formulate the problem as follows. We want to find the optimal material distribution in a domain space subject to a set of mechanical and non-mechanical, or what is called in this work geometric, constraints. More precisely, to find a generative function $G(C, \theta)$ of parameters θ , which takes these constraints C as input and suggests a design x accordingly. The optimization problem can be formulated as the following:

$$\begin{aligned}
 x &= G(C, \theta) \\
 C_{m_1} &: \frac{V(x)}{V_0} \leq f \\
 C_{m_2} &: KU = F \\
 C_{g_i} &: c_{g_i} \leq v_{g_i} \quad \text{or} \quad c_{g_i} \geq v_{g_i} \quad \text{for } i = 1, \dots, |C_g|
 \end{aligned} \tag{1.4}$$

where C_m are the mechanical constraints with C_{m_1} being the volume fraction constraint and C_{m_2} the loads constraint, while C_g are the set of geometric constraints, v_{g_i} is the minimum/maximum value acceptable for the geometric constraint g_i . V_0 , V_x , f , K , U and F are defined in section 1.1.2.

The major contributions of our work:

1. Integration of mechanical and geometrical manufacturing-related constraints at the same level.
2. Creation of a synthetic dataset of 2D designs alongside their mechanical and geometrical manufacturing constraints.
3. Benefitting from DL’s ability to learn spatial correlations and its speed and scalability. This approach integrates input mechanical and geometrical conditions at the conceptual level, generates designs accordingly, and is computationally independent of the inputs’ complexity.
4. Convergence is no longer an issue as long as the model is trained on converged designs.
5. Flexibility of integrating any types of constraints thanks to the flexibility of concatenating several input types into DL models

With our approach, we would like to test and validate the following hypotheses:

Hypothesis 1 (H1): *Any AM constraint or another type of constraint (min overhang, thermal-distortion, buckling, experts’ informal rules, Etc.) can be integrated into the design model, not only analytically formulated ones.*

Hypothesis 2 (H2): *Acceleration of the whole DfAM process by generating a first design draft complying with geometrical manufacturing and mechanical constraints and avoiding repetitive iterations of design updating and evaluating.*

Hypothesis 3 (H3): *A lighter generative design module that can be implemented in industrial design software in the future.*

It is essential to highlight that this work is focused on SIMP among all TO approaches for two major reasons. First, SIMP deals with the design as a distribution of material, in other terms, as an image, which is compatible with our way of approaching the design using DL-based computer vision techniques. Second, SIMP is the most implemented TO approach in the industrial software [4], for its simplest and less computationally expensive TO approach. It is found in SolidWorks[71], ABAQUS[72], Siemens NX[73], and Altair Optistruct[74], and ANSYS Mechanical[75], which also includes the level set approach [4]. The latter argument encourages us to work on compensating for its difficulties with DL and, in the future, propose a lighter AI-based TO module to be incorporated in industrial software.

1.3 Ingredients

1.3.1 Machine Learning (ML)

Machine Learning is the discipline of algorithms and statistical models that learn to perform a task based on data instead of being explicitly programmed to do so.

Machine Learning algorithms can be divided into three major learning paradigms that differ according to the available variables and task in hand: supervised, unsupervised, and reinforcement Learning (RL).

In supervised learning, the algorithm learns to map some input to an output based on pairs of training input-output data samples. The learning is forwarded by initializing a cost function that will update the algorithm's parameters until convergence via gradient descent; convergence is achieved when the cost function is minimized. The trainable parameters are called the weights of the ML algorithm. Linear/Logistic regression, Decision trees, Naive Bayes, Support vector machine, Etc., fall into the supervised ML category.

1.3. INGREDIENTS

Conversely, Unsupervised ML deals with data where no labels are present. Clustering algorithms like K-Means, K Nearest Neighbor, and feature reduction algorithms like Principle Component Analysis fall into this category.

Similarly, in RL, no labels for data samples are present. RL defines a reward function instead of a loss function and updates the weights of the algorithm in order to maximize this reward [76].

RL shares the same principle as unsupervised learning; no-true labels are available. However, it differs from it in the way the learning works. RL involves an agent, an environment, and a reward function. The learning consists of the agent acting in the environment and getting rewarded or penalized accordingly.

ML algorithms fall into discriminative and generative categories. Discriminative algorithms model the boundary between the different classes. It learns directly the conditional probability $p(y|x)$ of the label (y) given the data (x); hence, they are used in supervised learning. Generative algorithms model the joint distribution $p(x, y)$ [77].

1.3.2 Fully Connected Neural Networks (NN)

A fully connected neural network (NN) is an ML technique that consists mainly of three layers: an input, a hidden, and an output layer. Its architecture was inspired by the human neural brain. Each layer consists of a number of units or nodes connected to all nodes in the adjacent layer. The connections linking nodes of different layers are defined as the weights. The inputs are multiplied by the hidden weights and then summed. The latter undergo a non-linear function called an activation function like the rectified linear unit (ReLU = $f(x) = \max(x, 0)$), the sigmoid ($f(x) = \frac{1}{1+\exp -x}$), the softmax ($f(x) = \frac{\exp x_i}{\sum_j \exp x_j}$) or the hyperbolic tangent ($\tanh = f(x) = \frac{\exp x - \exp -x}{\exp x + \exp -x}$). The resulting values are multiplied by the weights from the connections between the hidden and output layers, then summed and passed through an activation function to get the output values [77]. The training consists of updating the weights to map the outputs to the ground truth labels. The weight updating procedure consists of computing a loss function between the predicted output and the ground truth, like the mean absolute error (L_1 norm), the mean squared error (L_2 norm), etc. This error is back-propagated through the network using the backpropagation algorithm [78]. "Backpropagation uses gradient descent for error reduction, by adjusting the weights based on the partial derivative of the error with respect to each weight [77]." Figure 1.5 shows the diagram of a simple NN of three layers.

1.3. INGREDIENTS

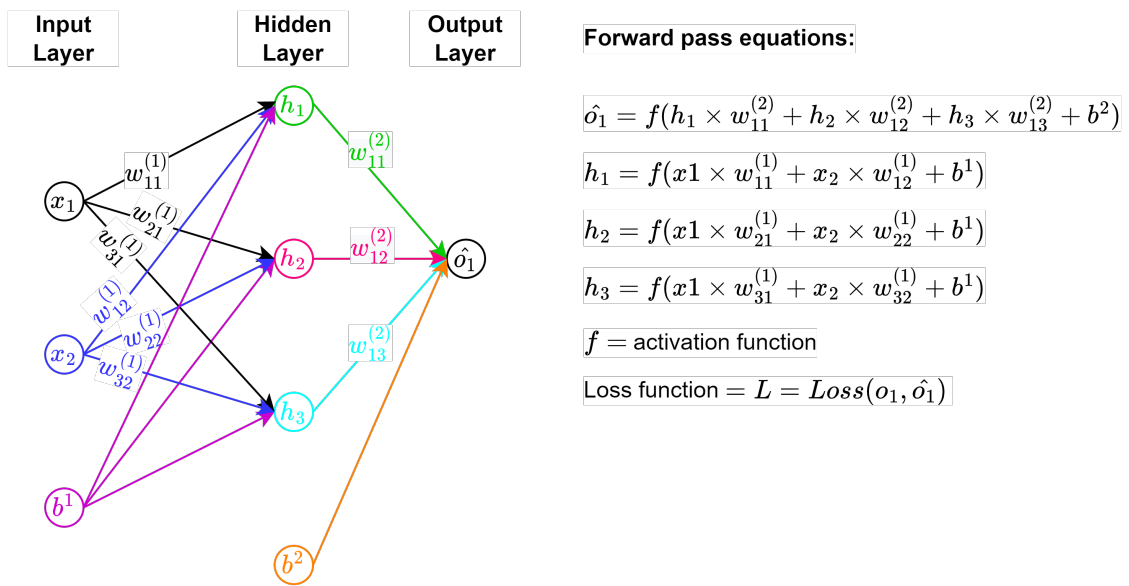


Figure 1.5: A simple neural network of three layers: an input layer (consisting of three units x_1 , x_2 and b), a hidden layer (consisting of four units h_1 , h_2 , h_3 and b) and an output layer (consisting of one unit \hat{o}). w_{ij}^l is the weight of the connection between unit i from layer l to the unit j from layer $l + 1$. b^l is the bias from layer l . \hat{o}_1 is the output's predicted value and o_1 the ground truth. The forward pass can be summarized by $h_i^l = \sum_{j=1}^n (w_{ij}^{(l-1)} \times x_j) + b_i^{(l)}$ with n the number of units in the layers $l - 1$. The loss function L compares \hat{o}_1 and o_1 . In the backward pass, weights are updated using the following equation $w_{ij}^l = w_{ij}^l - \alpha \times \frac{\partial L}{\partial w_{ij}^l}$ with α being the learning rate or the step size; it defines how quickly the solution converges.

1.3.3 Deep Learning (DL)

Most of ML applications are applied on processed inputs via hand-crafted features engineering. However, this task requires domain expertise and is time-consuming. Hence, identifying these features automatically could be beneficial technically and time-wise by finding new hidden features faster. The latter characteristic is the Deep Learning (DL) specialty [79]. "Deep learning is a specific kind of machine learning" [80]. Deep NN is a NN of more layers, thus, the term deep. The deeper the network, the more complex the learned features [77].

Recent state-of-the-art DL architectures have revolutionized the way we solve several problems in computer vision like object detection and recognition, video compression and generation via convolutional neural networks (CNN) and their variants (Alexnet [81], VGGnet [82], GoogLeNet [83], Inception-v3 [84], Inception-v4 [85]), to text generation and text-to-image generation via recurrent neural network (RNN) [86] and transformers [87], and the restricted Boltzmann machines [88].

In this work, we are dealing with 2D sketches that we approach as images. Thus, we are more interested in CNN-based architectures, which is explained in the following section.

1.3.4 Convolutional Neural Networks (CNN)

CNNs differ from traditional NNs in the way neurons between successive hidden layers are connected [89]. In a hidden layer, a neuron is only connected to a subset of neurons in the previous layer called the local receptive field. This sparse connectivity implicitly allows CNNs to learn feature maps and reduce the network's complexity and overfitting phenomenon. Additionally, neurons of the same feature map share the same weights reducing the parameters' number and hence the network's complexity.

More formally, the k^{th} feature map F_k can be defined as:

$$F_k = f(x * W_k) \tag{1.5}$$

with x the input, $f(\cdot)$ the activation function, W_k the convolutional filter of the k^{th} feature map, and $*$ the 2D convolutional operator.

A convolution layer is modeled as a moving filter, which passes through the image. Its values are multiplied by a neighborhood of pixels. Afterward, a pooling operation is applied to reduce the number of parameters, known as down-sampling; the pooling operations are the sum, the average, or

the maximum operations. This pooling layer is advantageous for the feature detection in the input to be scale and orientation invariant. Finally, a non-linear activation function is applied. A simple convolution operation is shown in figure 1.6(a). In reality, every convolutional layer applies several filters that map an input into several outputs called feature maps, such that a filter k_i maps the input to a particular feature map F_{k_i} (Fig.1.6(b)).

A CNN is a stacking of several convolution and pooling layers (Fig.1.6(c)).

Deeper networks result in hierarchical feature extraction. CNNs have proven robustness in learning spatial correlations in images for different use cases from object classification to object detection [90].

1.3.5 Generative adversarial networks (GAN)

Among the various ML techniques, the generative algorithms have gained great success recently for their ability to learn the real distribution of the data and perhaps accelerate and improve the generation of different types of data. The top most common methods are the variational auto-encoder (VAE) [91] and the generative adversarial networks (GAN) [92]. VAE learns to generate dispersed samples by minimizing a pixel-wise reconstruction error between the generated and input samples, making the VAE stable during training. However, this stable training, which consists of optimizing the average reconstruction loss, suffers from a major drawback: blurry generated images. In 2014, Goodfellow introduced the GAN architecture [92]. It outperformed VAE in generating sharper, aesthetically more plausible, and more creative images. Nonetheless, GAN suffers from many issues during training, with the most common being: difficulty in balancing the learning speed between the generator and the discriminator resulting in a mode collapse (i.e., when the generator learns only to produce one mode of samples) and instability of the generator and discriminator's losses (they oscillate continuously and never converge into a fixed point). Currently, research is focusing on solving these issues: Arjovsky et al.[93] proposed to replace the Kullback-Leibler loss function with the earth mover distance. Metz et al. [94] used an unrolled optimization of the discriminator's objective during training. Others explored different training hacks (batch normalization, transposed convolutions, the choice of activation functions and optimizers, Etc.)[95].

GAN is a method that learns to mimic any input data distribution. Its advantage is that it can generate new samples following the same input data distribution.

1.3. INGREDIENTS

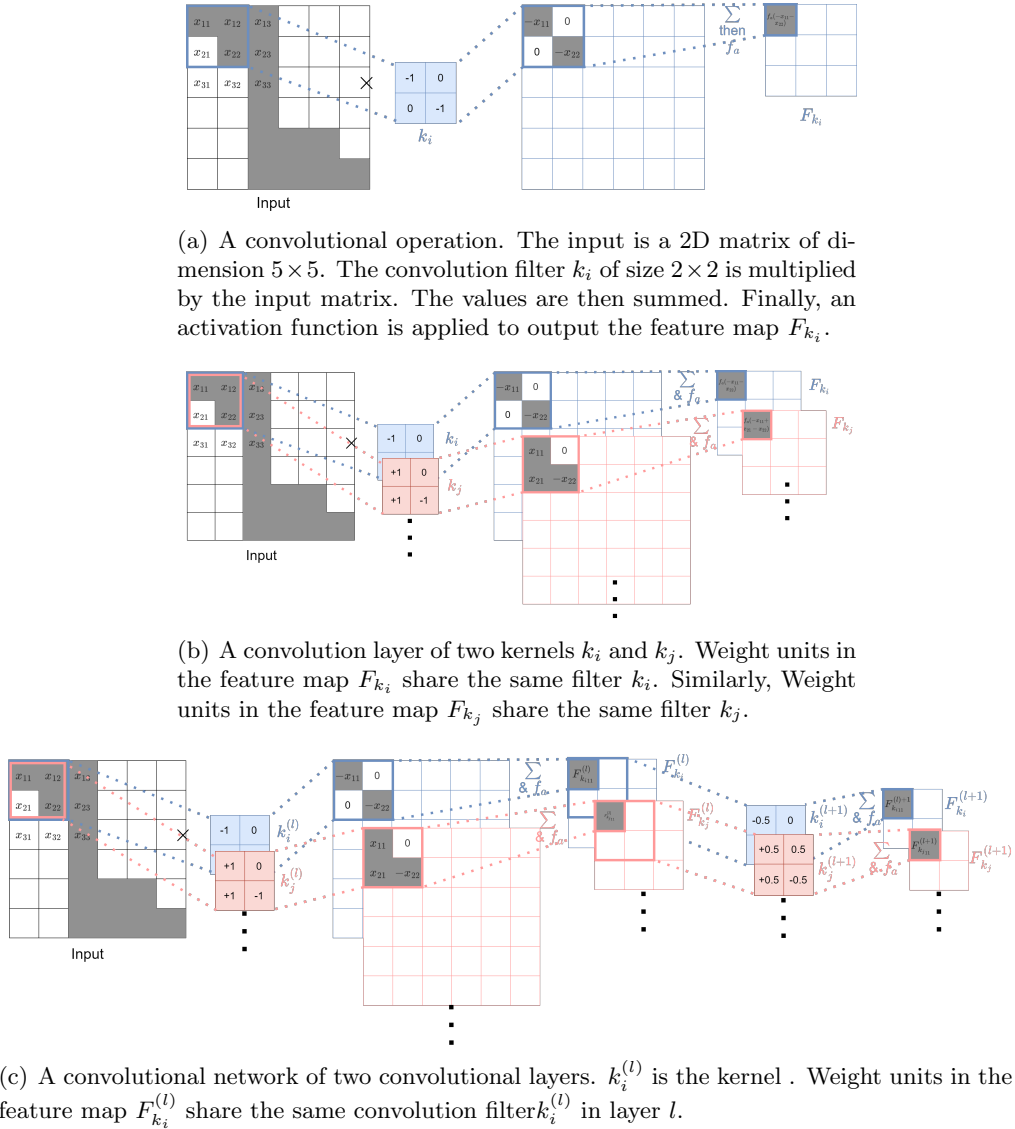


Figure 1.6: Schema of a convolutional neural network.

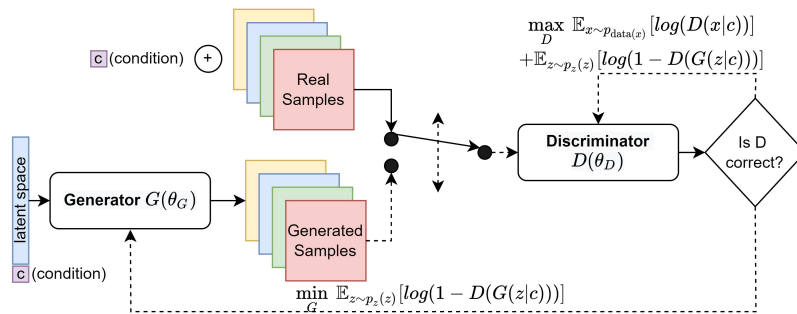


Figure 1.7: Diagram of a Conditional GAN.

As explained in their paper [92], Goodfellow et al. compared the GAN to the money counterfeiters versus the police situation. The counterfeiters try to produce fake money to fool the police, and the police's job is to detect this fake money. Every time the police catch fake money, the counterfeiters need to enhance their skills to induce realism into fake money so that it goes indiscernible.

GAN consists of two differentiable functions, the generator $G(z, \theta_g)$; with θ_g its parameters and z a latent random vector following a noise prior distribution p_z ; and the discriminator $D(x, \theta_d)$; with θ_d its parameters and x a real sample; each working against the other. $G(z, \theta_g)$ tends to output samples with a distribution p_g similar to the real data distribution p_{data} . However, $D(x, \theta_d)$ tries to discriminate real samples (i.e., from p_{data}) from synthesized ones (i.e., from $G(z, \theta_g)$). GAN's primary purpose is to optimize p_g to produce creative and diverse samples without memorizing the original ones. Both functions are learned in a minimax framework to improve the same loss function: the cross-entropy loss $L(G, D)$. This loss, as defined in the original paper (Goodfellow et al. [92]), is:

$$\begin{aligned} L(G, D) &= \min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} [\log(D(x, \theta_d))] \\ &+ \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z, \theta_g)))] \end{aligned} \quad (1.6)$$

The solution of this function is $p_g = p_{data}$, i.e. when the generator starts to output data samples following the same distribution as the real ones.

1.3.6 Conditional Generative adversarial networks (cGAN)

On the other hand, a Conditional Generative Adversarial Networks (cGAN) [96] is a GAN's extension enabling the generation to be oriented by a specific input condition c . In this framework (Fig.1.7), the basics of CGAN become: the conditional generator as $G(z|c, \theta_g)$, the conditional discriminator as $D(x|c, \theta_d)$ and the loss function as:

$$L(G, D) = \min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} [\log(D(x|c, \theta_d))] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z|c, \theta_g)))] \quad (1.7)$$

It is worth mentioning that conditional GANs have recently grabbed the interest of research. Its applications are spread amongst all fields, in computer vision (image synthesis, image-to-image, text-to-image) [97], video generation [98], engineering [99], medical [100], agriculture [101], and cybersecurity [102].

1.3. INGREDIENTS

To sum up, a convolutional CGAN is a CGAN with $G(z|c, \theta_g)$ and $D(x|c, \theta_d)$'s architectures being based on CNNs.

1.4 French Summary

Avec l'émergence des procédés d'impression 3D, principalement la fabrication additive (FA), imprimer des designs plus complexes et plus créatifs est devenue réalisable, ce qui a rendu l'optimisation topologique (TO) encore plus attirante dans les domaines de recherche. La TO vise à trouver la distribution optimale des matériaux dans un espace de conception, compte tenu d'un ensemble de contraintes mécaniques. La topologie résultante peut avoir n'importe quelle forme et être définie comme une image binaire 2D ou une grille de voxels binaires 3D, de sorte que la présence d'un pixel/voxel non vide signifie la présence du matériau. La TO a connu un succès considérable au 20th siècle lors de la révolution industrielle des secteurs de l'automobile et de l'aérospatiale, étant donné son puissant potentiel d'optimisation d'une structure en termes de matériau utilisé tout en maintenant ses spécifications et propriétés mécaniques recommandées. Par la suite, elle a étendu ses applications à un plus grand nombre de disciplines : fluide, acoustique, électromagnétique, optique, etc. Cependant, la TO utilise une méthode itérative, basée sur les éléments finis (FE), qui est coûteuse en calcul et donc relativement lente.

Avec le nouveau logiciel de TO avancé, la production d'un design unique soumis à un ensemble de conditions aux limites et de configurations de charge est une question de secondes à quelques jours selon la complexité du design et des contraintes environnantes. Cet aspect est acceptable si le processus est limité à cette seule étape. Cependant, un(e) ingénieur(e) en mécanique ne se fie jamais au premier design qu'il essaie, d'autant plus que la TO, sous sa forme commerciale [57, 58, 59, 60, 61], ne tient pas compte des contraintes géométriques ni des critères de fabrication, ni des autres contraintes industrielles spécifiques au secteur dans lequel la pièce est fabriquée (automobile, aéronautique, hydraulique, etc.). L'ingénieur(e) en mécanique doit toujours explorer plusieurs configurations pour trouver la topologie, la forme et le dimensionnement optimaux de la conception à fabriquer. Il/elle doit également s'assurer que son projet final est créatif, rentable et manufacturable. Par conséquent, ce processus d'exploration itératif peut devenir lent, en particulier pour les designs à grande échelle. L'accélération du cycle du design pour la FA est un sujet de recherche et un sujet industriel d'actualité. Par conséquent, les chercheurs tentent de trouver des solutions pour contourner ces contraintes et accélérer la phase de la TO. Nous pouvons classer les approches de la littérature qui ont abordé le sujet en quatre approches de pointe (Tab.1.1) :

1. la formalisation des règles de FA pour que les ingénieurs les prennent en compte lors de la phase de dessin des CAO, mais cela ne garantit pas l'optimalité initiale du design en termes de performance mécanique. Cette approche a permis de formaliser les contraintes de FA pour les utilisateurs novices et de guider les ingénieurs dans la création de designs imprimables ;
2. l'intégration de certaines contraintes de FA lors de la formulation FE-TO. Cette approche permet aux ingénieurs de disposer d'une première ébauche prenant en compte les deux contraintes, évitant ainsi de rester bloqués dans une boucle de redessiner et retester la performance mécanique de l'ébauche. Néanmoins, toutes les contraintes FA ne peuvent pas être formulées analytiquement. En outre, cette approche hérite des défauts de la TO basée sur la densité. (1) FE-TO identifie la forme générale dès les premières itérations, ce qui empêche la méthode de modifier la forme, dans les itérations restantes, pour tenir compte des contraintes géométriques. (2) La convergence n'est pas garantie, surtout lorsqu'il y a des contraintes multiples ; il est difficile de trouver le design optimal où toutes les conditions sont validées ;
3. l'utilisation du Machine Learning (ML) et du Deep Learning (DL) pour assister les méthodes FE-TO. Cette approche se concentre sur l'accélération de la phase TO du processus de design pour la FA et n'évite pas de rester coincé dans une boucle dans les phases ultérieures. Comme pour l'approche précédente, la plupart des problèmes de TO sont encore hérités ;
4. le remplacement total des éléments finis par du ML et du DL dans TO. Cette approche n'intègre aucune contrainte de fabrication et ne permet pas d'éviter d'être coincé dans une boucle dans les phases ultérieures du processus de design pour la FA.

Notre objectif est d'accélérer l'ensemble du processus de design pour la FA et d'obtenir le meilleur rapport qualité/coût. Comme nous l'avons vu dans l'introduction, la phase de conception a le coût le plus bas et l'impact le plus élevé sur le coût global du produit, sa qualité et donc son cycle de vie. Cependant, l'accélération de cette seule phase n'est pas suffisante, comme nous l'avons vu avec les approches de l'état de l'art. De plus, alors que les méthodes FE qui intègrent quelques contraintes de FA visent à accélérer l'ensemble du processus de design pour la FA en évitant les itérations répétitives, la formulation analytique des contraintes FA, la convergence de ces dernières et leurs coûts de calcul restent un obstacle. D'autre part, l'introduction de ML et du DL n'a accéléré qu'une seule phase. Ainsi, dans ce travail, nous proposons d'obtenir le meilleur des deux mondes. Cette nouvelle approche

généralive du DL permet d'intégrer les contraintes de fabrication aux contraintes mécaniques simultanément au même niveau conceptuel du design, et toutes les contraintes, même celles qui n'ont pas de définition mathématique, comme les règles métiers, tout en bénéficiant des avantages de vitesse et d'évolutivité du DL. Il est impératif de noter que cette approche n'a pas pour but de remplacer le robuste FE-TO mais de compenser ses difficultés lorsqu'il s'agit d'intégrer diverses contraintes complexes (Tab. 1.2).

La principale contribution de notre travail n'est pas seulement d'accélérer la phase de TO, mais aussi d'identifier un moyen d'adapter la géométrie et la fabricabilité de la conception afin de générer une première ébauche de design qui respecte à la fois les contraintes mécaniques et géométriques de fabrication, afin d'éviter de rester coincé dans une boucle de mise à jour et de test des conceptions et d'accélérer l'ensemble du processus de design pour la FA.

Ainsi, nous formulons le problème comme suit. Nous souhaitons trouver la distribution optimale des matériaux dans un espace de domaine soumis à un ensemble de contraintes mécaniques et non mécaniques, ou ce que nous appelons dans ce travail des contraintes géométriques. Plus précisément, trouver une fonction généralive $G(C, \theta)$ de paramètres θ , qui prend en entrée ces contraintes C et propose une conception x en conséquence. Le problème d'optimisation peut être formulé analytiquement comme suit :

$$\begin{aligned}
 x &= G(C, \theta) \\
 C_{m_1} &: \frac{V(x)}{V_0} \leq f \\
 C_{m_2} &: KU = F \\
 C_{g_i} &: c_{g_i} \leq v_{g_i} \quad \text{or} \quad c_{g_i} \geq v_{g_i} \quad \text{for } i = 1, \dots, |C_g|
 \end{aligned} \tag{1.8}$$

où C_m sont les contraintes mécaniques, C_{m_1} étant la contrainte de fraction de volume et C_{m_2} la contrainte de charges, tandis que C_g sont l'ensemble des contraintes géométriques, v_{g_i} est la valeur minimale/maximale acceptable pour la contrainte géométrique g_i . V_0 , V_x , f , K , U et F sont définis dans la section 1.1.2.

Les principales contributions de notre travail sont les suivantes :

1. l'intégration des contraintes mécaniques et géométriques de fabrication au même niveau conceptuel du design ;
2. la création d'un jeu de données synthétiques de designs 2D avec leurs contraintes mécaniques et géométriques de fabrication ;
3. bénéficier de la capacité du DL à apprendre les corrélations spatiales ainsi que de sa vitesse et

de son évolutivité. Cette approche intègre les conditions mécaniques et géométriques d'entrée au niveau conceptuel, génère des conceptions en conséquence et est indépendante sur le plan informatique de la complexité des entrées ;

4. la convergence n'est plus un problème tant que le modèle est entraîné sur des designs convergents ;
5. la flexibilité d'intégration de tout type de contraintes grâce à la possibilité de concaténer plusieurs types d'entrées dans les modèles DL.

Avec notre approche, nous souhaitons tester et valider les hypothèses suivantes :

1. toute contrainte FA ou autre type de contrainte (contrainte de surplomb, distorsion thermique, flambage, règles métiers, etc.) peut être intégrée dans le modèle de design, et pas seulement celles formulées analytiquement ;
2. l'accélération de l'ensemble du processus de design pour la FA en générant une première ébauche conforme aux contraintes géométriques de fabrication et mécaniques et en évitant les itérations répétitives de mise à jour et d'évaluation du design;
3. un module de design génératif plus léger computationnellement qui peut être implémenté dans un logiciel industriel de design à l'avenir.

Il est essentiel de souligner que ce travail se concentre sur l'approche densité, Solid Isotropic Material with penalization (SIMP), parmi toutes les approches de TO pour deux raisons majeures. Premièrement, SIMP traite le design comme une distribution de matériaux, en d'autres termes comme une image, ce qui est compatible avec notre façon d'aborder le sujet en utilisant des techniques de vision par ordinateur basées sur le DL. Deuxièmement, SIMP est l'approche TO la plus implémentée dans les logiciels industriels [4], grâce à ces avantages informatiques ; elle est la plus simple et la moins coûteuse en calcul. On la retrouve dans SolidWorks[71], ABAQUS[72], Siemens NX[73], et Altair Optistruct[74], et ANSYS Mechanical[75], qui comprend également l'approche level set [4]. Ce dernier argument nous incite à explorer des solutions afin de compenser les difficultés de TO par la DL et à proposer à l'avenir un module de TO plus léger basé sur le DL à intégrer dans les logiciels industriels.

Chapter 2

General Method of Deep Learning driven topology optimization

Content

2.1	Methodology: Data driven Design for Additive Manufacturing	36
2.2	Inputs: Mechanical and Manufacturing/Geometrical constraints	37
2.3	Model's architecture	40
2.4	Training Framework	41
2.5	Training Dataset	42
2.6	Discussion	43
2.7	French Summary	44

2.1 Methodology: Data driven Design for Additive Manufacturing

As we mentioned in section 1.2.2, our work’s main objective is to identify a way to accelerate the whole DfAM process. Moreover, as we identified previously, it comes to avoiding getting stuck in a loop of updating the design’s geometry to comply with manufacturing constraints and testing the mechanical performance. This is achieved by integrating the manufacturing constraints in earlier stages of the DfAM process, i.e., the design phase.

Previous work adopted this approach via FE-TO methods. Nevertheless, they reported several limitations. Some geometrical-related constraints between structural members (e.g., angles between structural members) cannot be easily formulated analytically, and some (e.g., hanging features, AM experts rules) lack an exact mathematical description [70]. In [41, 17], several parameters must be chosen carefully to avoid the introduction of non-linearities and ensure convergence of the optimization problem. Moreover, although some work tried to impose geometric constraints via density-filters [63] to alleviate the computational cost of FE-TO or used NNs’ back-propagation [69] to compute expensive sensitivity analysis, they still relied on iterative and computationally expensive FE-solvers and FE analysis (Finite Elements Analysis (FEA)), and sometimes ended up with local optima.

Finally, whereas it is less intricate to impose a few geometric constraints, it is further complicated to handle numerous ones simultaneously, especially inter-member constraints of a single structure, with the analytical AM-oriented-TO formulations.

To compensate for these limitations, We propose a data-driven approach to explore the DL capability, particularly CNNs, to integrate the geometrical constraints concurrently at the same level as the mechanical ones. CNNs have demonstrated their potential in learning spatial correlations and extracting high-level features (including geometrical or shape-related features) from real-world images [66, 67, 68], and DL outputs converged crisp black-and-white designs if trained on converged training designs. Besides, their computational cost, in terms of operations and prediction time, is independent of the inputs’ complexity, unlike FE methods, for which the number of operations and computational time grow exponentially with the input’s complexity.

Since the problem is to generate designs, a data-driven generative method, particularly the Generative Adversarial Networks (GAN), is chosen. GANs are distinguished for their flexible framework incorporating additional knowledge into the generator.

We recall that our goal is to find the optimal material distribution in a domain space subject to a set of

2.2. INPUTS: MECHANICAL AND MANUFACTURING/GEOMETRICAL CONSTRAINTS

mechanical and non-mechanical, or what is called in this work geometric, constraints. More precisely, to find a generative function $G(C, \theta)$ of parameters θ , which takes these constraints C as input and suggests a design x accordingly. The optimization problem can be formulated as the following:

$$\begin{aligned}
 x &= G(C, \theta) \\
 C_{m_1} &: \frac{V(x)}{V_0} \leq f \\
 C_{m_2} &: KU = F \\
 C_{g_i} &: c_{g_i} \leq v_{g_i} \quad \text{or} \quad c_{g_i} \geq v_{g_i} \quad \text{for } i = 1, \dots, |C_g|
 \end{aligned} \tag{2.1}$$

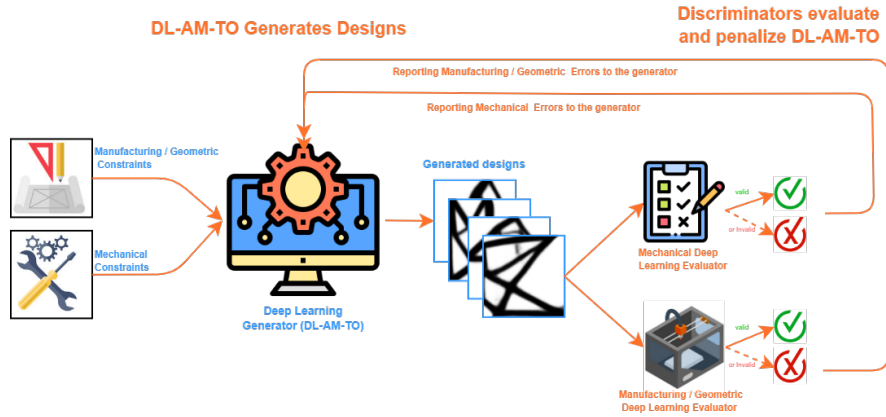
where C_m are the mechanical constraints with C_{m_1} being the volume fraction constraint and C_{m_2} the loads constraint, while C_g are the set of geometric constraints, v_{g_i} is the minimum/maximum value acceptable for the geometric constraint g_i . V_0 , V_x , f , K , U and F are defined in chapter 1.1.2.

Inspired by the GAN framework, this problem can be solved by setting $G(C, \theta)$ as the conditional generator, D_m as a discriminator validating the conformity of the generated designs with the mechanical constraints, and D_{g_i} as the i^{th} discriminator validating the conformity of the generated designs with the i^{th} geometrical constraint, with $i = 1, \dots, |C_g|$. $G(C, \theta)$ is penalized not only for the aesthetics of generated designs but also for their mechanical and geometrical conformity with the input conditions. This is illustrated in figure 2.1. Figure 2.1(a) shows the training phase of the generator $G(C, \theta)$ (Deep Learning Additive Manufacturing driven Topology Optimization, DL-AM-TO). Indeed, DL-AM-TO takes the mechanical and manufacturing/geometrical constraints as input and generates a 2D design. The latter is input to the discriminators; the mechanical discriminator penalizes the generator if the design does not respect the mechanical input constraints, and the geometrical discriminator penalizes the generator if the design does not comply with the geometrical/geometrical constraints. The errors computed by the discriminators are fed back to the generator to update its weights and improve its generation quality. Figure 2.1(b) shows DL-AM-TO during usage time, i.e., inference time. At inference time, DL-AM-TO generates designs for any given input constraints within a fraction of a second. These designs can be evaluated by the discriminators that helped with DL-AM-TO's training, as shown in figure 2.1(c).

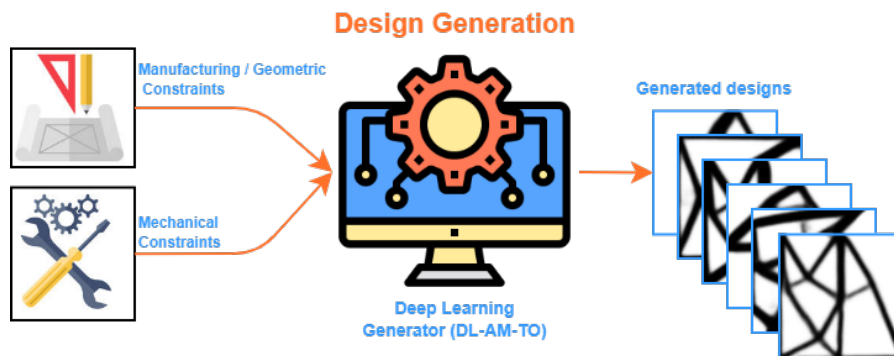
2.2 Inputs: Mechanical and Manufacturing/Geometrical constraints

As mentioned previously, $G(C, \theta)$ takes several types of constraints, which do not share the same format (i.e., shape) most of the time. This format problem can be solved by transforming all inputs into the same shape. However, identifying this consolidated format is not straightforward and should

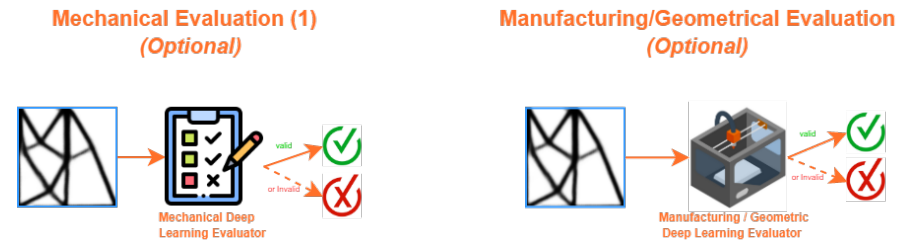
2.2. INPUTS: MECHANICAL AND MANUFACTURING/GEOMETRICAL CONSTRAINTS



(a) During Training, the mechanical and geometric/manufacturing constraints are input into the Deep Learning Additive Manufacturing-driven Topology Optimization (DL-AM-TO), the generator, which generates 2D designs. The latter is evaluated by the mechanical discriminator penalizing the generator when the generated designs do not comply with the mechanical conditions. The geometric/manufacturing discriminator penalizes the generator when the generated designs do not comply with the geometric/manufacturing conditions. These evaluators are also Deep Learning-based networks that are trained simultaneously along with the generator.



(b) During Inference, the generator is used to output designs conforming to input mechanical and inference and geometric/manufacturing constraints.



(c) Evaluation of Generated designs (Optional). To evaluate the generated designs, the user could use the Deep Learning evaluators.

Figure 2.1: Deep Learning Design Generation considering both Mechanical and Geometric/Manufacturing Constraints. The training procedure is based on the GAN framework.

2.2. INPUTS: MECHANICAL AND MANUFACTURING/GEOMETRICAL CONSTRAINTS

be assessed for every case.

For our case, the mechanical constraints considered are the boundary conditions (BC : BC_x, BC_y), the loads (F : F_x, F_y) along the x and y axis and the volume fraction V .

The BC define the clamped part of the domain space; what nodes are restrained from translations and/or rotations; they are represented as 2D matrices of size $((n_x + 1), (n_y + 1))$ for a domain space of dimension (n_x, n_y) .

F define the loads along the x and y axis, such that $F_x = |F| \times \cos(F_\theta)$ and $F_y = |F| \times \sin(F_\theta)$ with $|F|$ the load's magnitude (in this work, it is set to $1N^1$) and F_θ the load's orientation. F_x and F_y are represented as 2D matrices of size $((n_x + 1), (n_y + 1))$ for a domain space of dimension (n_x, n_y) .

V defines the fraction of the total material in the domain space. In other terms, the final design must consist of, at most, $V\%$ material. In SIMP-TO, this constraint is initialized by setting the starting x to a matrix of V value everywhere; the average value of an array of the same value val equals val . Hence, V is represented as a 2D matrix of size $((n_x + 1), (n_y + 1))$ for a domain space of dimension (n_x, n_y) with the same value V .

From section 1.2.1, we can conclude that the most influential geometrical manufacturing-related constraints are the minimum overhang and the minimum and maximum member size.

1. The minimum overhang (Θ_{min}). It is the most general AM constraint. It is the angle between the normal of a bar and the build orientation. More importantly, it is the most impactful manufacturing constraint that gained the highest attraction in research. Indeed, a design violating this constraint has hanging features and hence needs support structures. The latter yields additional material, slows the build time, damages the part's accuracy and finishing, and induces the need for post-processing, further delaying the fabrication [40]
2. The thin feature represented in this work as the minimum bar thickness (th_{min}). Thin features usually present several defects: unmelted powder inclusions, internal voids, cracks, and shape irregularities [103]. Moreover, controlling the bar thickness alleviates large thermal gradients and improves resistance to buckling or localized damage [41].
3. The maximum bar length (len_{max}). Long bars, called bridges, are unsupported features in a design that collapse during manufacturing [26].

¹ N = Newton; an SI unit measure for forces.

2.3. MODEL’S ARCHITECTURE

4. The total number of bars (Nbr_{bars}). Engineers have a preference for simple geometries. Thus, we defined the simplicity of a design by the total number of bars, primarily since our dataset consists of truss-like structures. A simpler structure is a design having fewer bars.

These constraints are scalar values, thus, of a different shape than the mechanical ones.

In this case, there are three solutions to allow the concatenation of both constraints:

1. Compress the mechanical constraints i.e. convert the 2D matrices of size $((n_x + 1), (n_y + 1))$ into scalar values and represent the inputs as a 1D vector of dimension 1×9 .
2. Expand the geometrical constraints into 2D matrices (by repeating the same value in the matrix) and end up with an input of size $9 \times (n_x + 1) \times (n_y + 1)$; a nine-channel $((n_x + 1) \times (n_y + 1))$ image,
3. Keep all inputs’ shapes untouched and configure the generator’s architecture to handle the constraints at different levels.

In our case, the second option was adopted for performance reasons; the generated designs were of better quality, and the response to the geometrical constraints was more remarkable by representing them as matrices than by scalars. This result is not conclusive. For other situations, the remaining approaches might work as well.

For a better understanding, figure.2.2 illustrates the inputs to $G(C, \theta)$.

2.3 Model’s architecture

The model’s architecture depends on the inputs and outputs formats, whether they are scalars, images, texts, Etc.

In our case, inputs are a mix of scalar and 2D matrix constraints converted into image-like inputs. Thus, $G(C, \theta)$ is supposed to take these image-like constraints, encode them, and extract the features that map them to an image-like geometry or a density-pixel distribution.

Consequently, the adequate architecture would consist of a sequence of convolutional neural networks (CNN) 1.3.4 to encode the constraints, followed by a sequence of de-convolutions to decode the design. Several state-of-the-art architecture based on CNNs can be found in the literature, we cite: ResNet

2.4. TRAINING FRAMEWORK

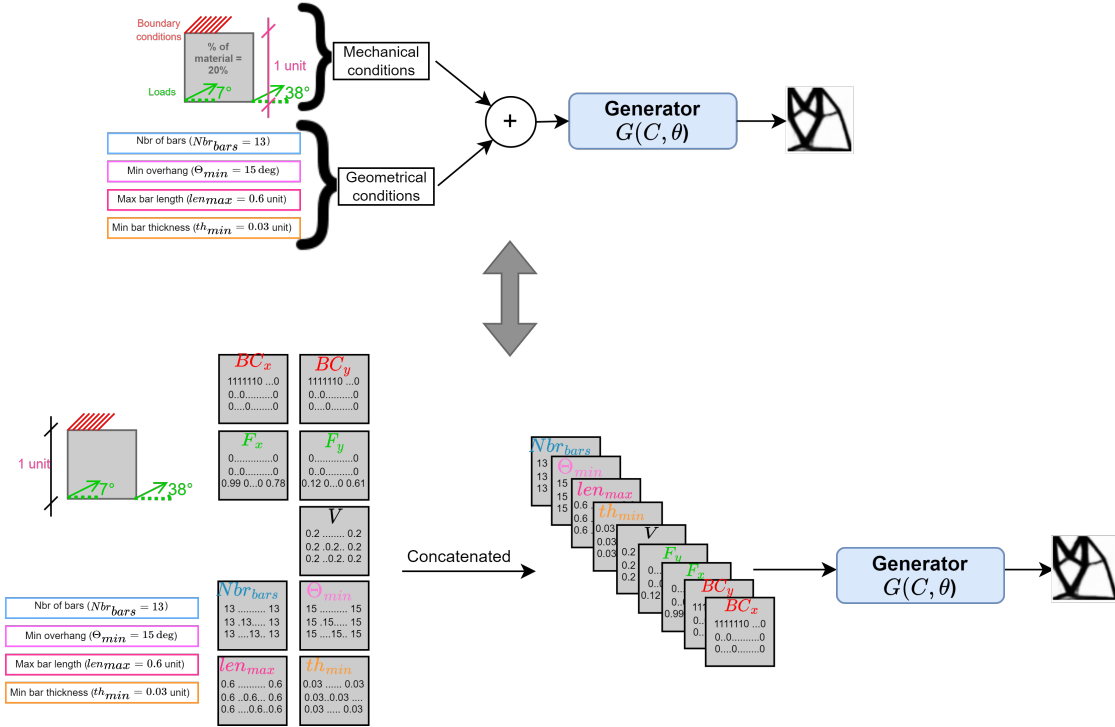


Figure 2.2: Input to the generator $G(C, \theta)$. The input consists of (1) the mechanical constraints: Boundary conditions (BC_x , BC_y), the loads (F_x , F_y) along the x and y axis and the volume fraction V (2) and the geometrical constraints: the total number of bars (Nbr_{bars}), the maximum bar length (len_{max}), the minimum overhang (Θ_{min}), the minimum bar thickness (th_{min}). Following option three in section 2.2, the scalar geometrical constraints are transposed to 2D matrices to be concatenated with the mechanical ones. Consequently, the input to $G(C, \theta)$ is a nine-channel 2D image.

[104], U-net[105], Res-U-net [6], Inception architectures [83, 84, 85], etc. $G(C, \theta)$'s architecture chosen in this work is detailed in chapters 3 and 5.

2.4 Training Framework

Now that $G(C, \theta)$'s architecture is chosen, It is time to set up the training workflow.

A simple training workflow consists of formulating a loss function that compares the generated designs versus the ground truth, i.e., computing a reconstruction loss, which is propagated to penalize $G(C, \theta)$. However, this training procedure does not fulfill our requirements. The objective is not simply to generate aesthetically plausible image-like designs but to generate mechanically valid designs conforming to the input geometrical constraints; in other terms, designs respecting the same mechanical specs but are not necessarily identical when it comes to geometry. Hence, every input constraint must be

checked separately at every output design.

This mindset of training recalls the generative adversarial networks (GAN)[92] (section 1.3.5) where the generator outputs data samples, and a discriminator tries to identify whether they came from the real dataset or whether the DL generator outputted them.

The only difference between the original GAN and our case is that one discriminator is not sufficient here because the input constraints are several and differ in nature (mechanical versus geometrical) and magnitudes (for example, F_x and F_y are continuous bounded values $\in [0, 1]$ while the Nbr_{bars} are integer scalars ≥ 1).

Therefore, the training workflow consists of two types of discriminators: mechanical and geometrical ones, such that a geometrical discriminator per a geometrical condition and one mechanical discriminator representing the mechanical constraints.

The reason for all mechanical constraints to share the same discriminator while every geometrical condition has its own is that the mechanical constraints, altogether, make a design, and thus evaluating each mechanical constraint separately does not make any sense; it is not physically interpretable. However, two designs with two different numbers of bars can be interpreted as mechanically similar as long as the mechanical key performance indicator (KPI) computed on both is similar. Moreover, we recall that our initial objective is to be able to propose several geometries for the same well-defined mechanical conditions in order to comply with the production requirements. Also, having an evaluator per geometrical condition helps us in the evaluation phase; they will alleviate the need to manually measure the geometrical constraints of all the generated designs.

2.5 Training Dataset

As mentioned, a dataset of designs alongside their mechanical and geometrical constraints is needed to train $G(C, \theta)$.

Available datasets include mesh geometries (Boundary representations, B-rep)[106, 107], sketch graphs [108] and parametric Computer Aided Designs (CADs) [109]. Unfortunately, these datasets are not convenient for the task at hand; they contain geometrical information about designs, but none provide the mechanical specs. Thus, we need to consolidate a synthetic dataset containing the designs alongside the mechanical and geometrical constraints.

2.6 Discussion

The objective is to create a generative model that maps a set of mechanical and geometrical manufacturing-related constraints to a shape. Thus, we need a dataset of designs consisting of several shapes per set of mechanical constraints, such that these shapes differ in only one geometrical constraint at a time. Consequently, we have identified a workflow to consolidate this dataset.

First, We will need to generate designs from a traditional FE-TO method (SIMP) by sampling the mechanical constraints. Second, we will train Deep Learning models to predict the mechanical constraints from designs using the SIMP dataset. Third, we will need to draw CADs inspired by the shapes proposed by TO; these shapes differ in the geometrical constraints. Finally, we will predict the mechanical constraints of these synthetic CADs using the trained DL models. Hence, we will end up with a dataset of designs and their geometrical and mechanical constraints.

Nonetheless, this data consolidation workflow is very complex and time-consuming. Another challenge is the training dataset dimension; a tremendous amount of designs is needed to train a generative model based on DL. successfully Consequently, before diving into this step, we will start with a proof of concept to validate our proposed method. The first approach consists of choosing a simple geometry-related constraint that is hardly analytically formulated; thus is challenging to be integrated into FE-TO approaches. The first approach consists of creating a dataset by sampling the mechanical constraints and generating designs using the SIMP method. The SIMP designs are truss-like structures; hence, as a first geometry-related constraint, the number of bars (Nbr_{bars}) is chosen as the first geometrical constraint. This approach trains a DL model that takes the mechanical constraints and the (Nbr_{bars}) as inputs and outputs a 2D design accordingly. The goal is to demonstrate that for a set of fixed mechanical constraints, changing the geometrical condition will lead to several geometries while always respecting these mechanical constraints in a fraction of a second. Thus, the design proposed is not only mechanically valid but also geometrically valid, hence print-ready. Thus, the DfAM process is accelerated not only the TO phase; the designer will no longer be blocked in a loop of designing and testing the mechanical performance, and the geometry proposed in the design phase is already compliant with the manufacturing constraint.

Thus, chapter 3 details the proof of concept approach and shows the results obtained. Next, the consolidation of the mechanical and geometrical CAD dataset is described in chapter 4. Then, chapter 5 improves on the proof of concept using the dataset created in the previous chapter and validates

our objective, which is that a data-driven TO approach integrating both mechanical and geometrical manufacturing-related constraints accelerates the whole DfAM process and not only the design phase. Finally, Chapter 5.9 summarizes the methodology and its outcomes and presents future perspectives.

2.7 French Summary

Comme nous l'avons mentionné dans la section 1.2.2, l'objectif principal de notre travail est d'identifier un moyen d'accélérer l'ensemble du processus de design pour la fabrication additive (FA). En effet, il s'agit d'éviter la boucle des tâches de mise à jour de la géométrie pour respecter les contraintes de fabrication et de tests mécaniques. Cet objectif est atteint en intégrant les contraintes de fabrication à des étapes plus précoces du processus de design pour la FA, c'est-à-dire la phase de conception.

Des travaux antérieurs ont adopté cette approche via des méthodes d'optimisation topologique (TO) basées sur les éléments finis. Néanmoins, ils ont signalé plusieurs limitations.

Certaines contraintes géométriques entre les éléments structurels (par exemple, les angles) ne peuvent pas être facilement formulées de manière analytique, et d'autres (par exemple, les contraintes de surplomb, les règles métiers) manquent d'une description mathématique exacte [70]. Enfin, alors qu'il est moins compliqué d'imposer quelques contraintes géométriques, il est encore plus compliqué d'en traiter de multiples contraintes simultanément, en particulier celles entre les membres d'une même structure, avec les méthodes de TO orientées FA.

Pour compenser ces limitations, nous proposons une approche basée sur les données pour explorer la capacité des DL, en particulier les réseaux convolutionnels de neurones (CNN), à intégrer les contraintes géométriques simultanément au même niveau que les contraintes mécaniques. Les CNN ont démontré leur potentiel dans l'apprentissage des corrélations spatiales et l'extraction de caractéristiques de haut niveau (y compris des caractéristiques géométriques ou liées à la forme) à partir d'images du monde réel. En outre, leur coût de calcul, en termes d'opérations et de temps de prédiction, est indépendant de la complexité des entrées, contrairement aux méthodes basées sur les éléments finis, pour lesquelles le nombre d'opérations et le temps de calcul croissent de manière exponentielle avec la complexité de l'entrée.

Le problème étant de générer des designs, une méthode générative pilotée par les données, notamment les réseaux antagonistes génératifs (GAN), est choisie. Les GAN se distinguent par leur flexibilité

quant à incorporer des connaissances supplémentaires dans le générateur.

Nous rappelons que notre objectif est de trouver la distribution optimale de matériaux dans un espace de design soumis à un ensemble de contraintes mécaniques et non mécaniques (i.e. géométriques).

Plus précisément, il s'agit de trouver une fonction générative $G(C, \theta)$ de paramètres θ , qui prend en entrée ces contraintes C et propose un design x en conséquence (section 1.4, chapitre 1).

Inspiré des GAN, ce problème peut être résolu en définissant $G(C, \theta)$ comme le générateur conditionnel, D_m comme un discriminateur validant la conformité des designs générés avec les contraintes mécaniques, et D_{g_i} comme le $i^{\text{ème}}$ discriminateur validant la conformité des designs générés avec la $i^{\text{ème}}$ contrainte géométrique, avec $i = 1, \dots, |C_g|$. Ceci est illustré dans la figure 2.1. La figure 2.1(a) montre la phase d'apprentissage du générateur $G(C, \theta)$ (appelé Deep Learning Additive Manufacturing driven Topology Optimization, DL-AM-TO). En effet, DL-AM-TO prend en entrée les contraintes mécaniques et de fabrication/géométriques et génère un design 2D. Ce dernier est soumis aux discriminateurs ; le discriminateur mécanique pénalise le générateur si le dessin ne respecte pas les contraintes mécaniques d'entrée et le discriminateur géométrique pénalise le générateur si le dessin ne respecte pas les contraintes géométriques/géométriques. Les erreurs calculées par les discriminateurs sont renvoyées au générateur pour mettre à jour ses poids et améliorer la qualité de sa génération.

La figure 2.1(b) montre DL-AM-TO pendant le temps. Au moment de l'inférence, DL-AM-TO génère des designs pour toute contrainte d'entrée donnée en une fraction de seconde. Ces designs peuvent être évalués par les discriminateurs qui ont participé à la formation de DL-AM-TO, comme le montre la figure 2.1(c). Les entrées de DL-AM-TO sont les contraintes mécaniques ; les conditions aux bords au long des axes x et y (BC_x, BC_y), les forces (F_x, F_y) et la fraction volumique (V) ; et les contraintes géométriques ; le minimum surplomb (Θ_{min}), la minimum épaisseur d'une barre (th_{min}), la longueur maximale d'une barre (len_{max}) et le nombre de barres (Nbr_{bars}). En TO, un domaine d'espace de dimension (n_x, n_y) est discrétisé en $((n_x + 1), (n_y + 1))$ éléments. Par suite, les BC , F , et V sont formulées comme des matrices 2D de dimension $((n_x + 1), (n_y + 1))$. Pour les conditions de bords, nous les formulons comme des matrices avec des valeurs nulles partout à l'exception des nœuds où les BC sont appliquées, ils prennent la valeur 1.0. Pareil, les F sont des matrices avec des valeurs nulles partout à l'exception des nœuds où les F sont appliquées, ils prennent la valeur $|F|.cos(F_\Theta)$ pour F_x et $|F|.sin(F_\Theta)$ pour F_y avec $|F|$ la magnitude de la force en *Newton* et F_Θ l'orientation de la force. La fraction volumique V est la contrainte de pourcentage de matériau, il est formulé comme une matrice

2D de même dimension $((n_x + 1), (n_y + 1))$ avec la même valeur V partout.

D'autre part, les contraintes géométriques sont plutôt des valeurs scalaires de 1D, ce qui n'est pas compatible en termes de format avec les contraintes mécaniques de 2D. Par conséquent, pour des mesures de compatibilité et de performance, nous avons choisis de transposer les contraintes géométriques en 2D ; nous avons transformé la valeur scalaire en une matrice 2D avec cette même valeur répétée $((n_x + 1), (n_y + 1))$ fois (2.2). Pour entraîner DL-AM-TO, une base de données encapsulant des designs 2D avec leurs contraintes mécaniques et géométriques. Malheureusement, cette base n'existe pas. Par conséquent, nous avons identifié un flux de travail pour consolider cet ensemble de données.

Premièrement, nous devons générer des designs à partir d'une méthode FE-TO traditionnelle (Solid Isotropic with Material Penalization, SIMP) en échantillonnant les contraintes mécaniques. Deuxièmement, nous formerons des modèles de DL pour prédire les contraintes mécaniques à partir de designs en utilisant le jeu de données SIMP. Troisièmement, nous devons dessiner des conceptions assistées par ordinateur (CAO) inspirées des géométries proposées par TO ; ces formes diffèrent par les contraintes géométriques. Enfin, nous prédirons les contraintes mécaniques de ces CAO synthétiques à l'aide des modèles DL entraînés. Nous obtiendrons ainsi un ensemble de données sur les dessins et leurs contraintes géométriques et mécaniques. Néanmoins, ce flux de consolidation des données est très complexe et prend beaucoup de temps. Un autre défi est la dimension de l'ensemble de données d'entraînement ; une quantité énorme de designs est nécessaire pour entraîner un modèle génératif basé sur la DL.

Par conséquent, avant de plonger dans cette étape, nous allons commencer par une preuve de concept pour valider la méthode que nous proposons. La première approche consiste à choisir une contrainte simple liée à la géométrie qui est difficilement formulable analytiquement et donc difficile à intégrer dans les approches de TO basées sur les éléments finis. La première approche consiste à créer un jeu de données en échantillonnant les contraintes mécaniques et en générant des designs à l'aide de la méthode SIMP. Les designs SIMP sont des structures en forme de treillis ; par conséquent, le nombre de barres (Nbr_{bars}) est choisi comme première contrainte géométrique. Cette approche forme un modèle DL qui prend les contraintes mécaniques et le nombre de barres (Nbr_{bars}) en entrée et produit un design 2D en conséquence. L'objectif est de démontrer que pour un ensemble de contraintes mécaniques fixes, le changement de la condition géométrique conduira à plusieurs géométries en respectant toujours ces contraintes mécaniques en une fraction de seconde. Ainsi, le design proposé est non seulement valide

2.7. FRENCH SUMMARY

mécaniquement mais aussi géométriquement, donc prêt à être imprimé. Ainsi, le processus de design pour la FA n'est pas seulement accéléré dans la phase TO ; le concepteur ne sera plus bloqué dans une boucle de design et de test des performances mécaniques et la géométrie proposée dans la phase de design est déjà conforme à la contrainte de fabrication.

Ainsi, le chapitre 3 détaille la démarche de preuve de concept et montre les résultats obtenus. Ensuite, la consolidation du jeu de données CAO mécanique et géométrique est décrite dans le chapitre 4. Ensuite, le chapitre 5 améliore la preuve de concept en utilisant le jeu de données créé dans le chapitre précédent et valide notre objectif, qui est qu'une approche TO pilotée par les données et intégrant les contraintes mécaniques et géométriques liées à la fabrication accélère l'ensemble du processus de design pour la FA et pas seulement la phase de design. Enfin, le chapitre 5.9 résume la méthodologie et ses résultats et présente les perspectives.

2.7. FRENCH SUMMARY

Chapter 3

The first approach: Deep Learning driven topology optimization

Content

3.1	Methodology	50
3.1.1	Training framework	51
3.2	Topology Optimized designs dataset consolidation	55
3.3	Experiments and Results	56
3.3.1	DL-TO's performance	59
3.3.2	Counter-discriminator's performance	60
3.3.3	Overall performance	60
3.4	Improving the performance of DL-TO	64
3.4.1	Compliance predictor discriminator's architecture	65
3.4.2	Compliance-predictor's performance	65
3.4.3	Training Loss Function	66
3.4.4	DL-TO's Performance after training with three discriminators	67
3.4.5	Tailoring the design's geometry via DL-TO	69
3.4.6	Generating designs with a new unseen constraint	71
3.5	Web application of DL-TO	73
3.6	Discussion	76
3.7	French summary	79

3.1 Methodology

This chapter is adapted from two of our articles:[110], and the other is currently in production.

The first approach aims to create a DL-TO method that simultaneously integrates mechanical and geometrical constraints at the conceptual level. In other words, the training must ensure that DL-TO's network is penalized on the mechanical and geometrical constraints (input conditions). Therefore, a conditional dual-discriminator GAN [96] is chosen to train DL-TO (the generator). The input constraints are formulated as images (Fig. 3.5). Hence, a CNN -based architecture was adopted.

As mentioned previously, DL-TO is trained along with two discriminators. Their role is to ensure that it generates designs respecting input conditions. The first discriminator, the GAN's traditional one, differentiates between the real and the generated designs. The second one predicts the generated designs' Nbr_{bars} .

The training procedure can be summarized as follows.

At every iteration of the training, the DL-generator (Res-U-Net Generator) takes as input the mechanical (Boundary conditions, loads configuration, and volume fraction) and geometrical (Nbr_{bars}) constraints and outputs 2D designs (the Generated Designs). Then, the traditional discriminator is trained in two stages. It is trained with the real designs alongside the constraint in the first stage and with the generated designs alongside the constraints in the second. The counter discriminator is trained with only the real designs and the mechanical constraints. At this level, the discriminators are considered optimal and are used to evaluate the generated designs: the scores output by both discriminators (the adversarial and counting losses) alongside the reconstruction loss are injected back into the generator to train its weights.

The training workflow is illustrated in Fig.3.1.

This proof-of-concept approach targets the validation of the following hypotheses:

Hypothesis 4 (H4): *DL-TO generates designs of good quality, mechanically and geometrically valid.*

Hypothesis 5 (H5): *DL-TO tailors the design's geometry by simply modifying the input geometrical variable Nbr_{bars} .*

Hypothesis 6 (H6): *DL-TO generates designs with passive and active elements without being trained to do so.*

3.1. METHODOLOGY

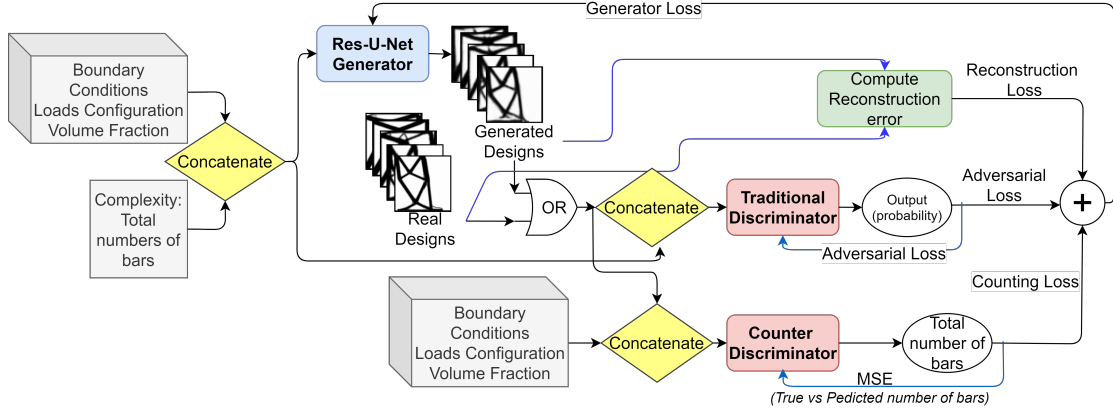


Figure 3.1: Training Procedure. (From left to right)

1) The mechanical constraints (Boundary conditions, loads configuration and volume fraction) are concatenated with the geometrical condition (the Nbr_{bars}). The later are input into the Generator, which outputs the designs (Generated Designs). The generated designs are concatenated with all of the input conditions and fed to the traditional discriminator, which predicts a score (the adversarial loss). Then they are concatenated with only the mechanical conditions and fed to the counter discriminator, which predicts their corresponding Nbr_{bars} . The counting loss is the MSE between the input Nbr_{bars} and predicted one. Finally, the quality of generated designs is compared versus the real ones (the Reconstruction error). All three loss are summed (a weighted sum) and the final score is fed-back to the generator to update its weights. 2) The Traditional discriminator is trained at two levels: the first one with the real designs alongside the input conditions and the second one with the generated designs alongside the input conditions. The adversarial loss is fed-back to the network to update its weights. 3) The counter-discriminator is trained only with the real designs and their mechanical conditions. The counting loss is fed-back to the network to update its weights.

3.1.1 Training framework

3.1.1.1 Generator's architecture

The generator (DL-TO) is a deep Res-U-net network [6]. It is an encoder-decoder convolutional architecture with residual (Res) and skip-connections between the outputs of encoder layers and the inputs of decoder layers or what is called U-Net. The generator encodes input conditions formulated as a six-channel-image (Boundary conditions and loads along the x and y-axis, volume fraction and Nbr_{bars}) and decodes the 2D design (Fig. 3.5). This architecture benefits from the U-Net and residual advantages. The U-Net improves the information propagation from the encoder to the decoder and compensates for the loss of finer details in the decoding process by combining low-level features with their corresponding high levels [105]. The residual connection compensates for the degradation in performance (usually due to vanishing gradients) in deeper networks (He et al. [104]). As per the

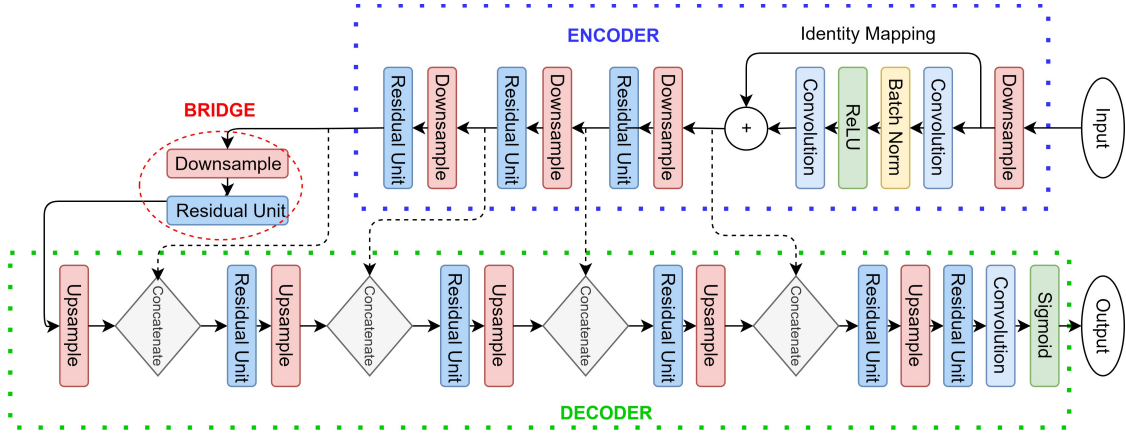


Figure 3.2: The architecture of the Res-U-Net Generator. The network can be divided into an encoder, a bridge, and a decoder. The encoder is formed of 4 blocks, each consisting of a down-sampling layer (a convolution of stride 2) and a residual unit([6]). The decoder comprises five blocks, each consisting of an up-sample layer (a transpose convolution of stride 2) and a residual unit followed by a convolution of kernel size 1×1 and a sigmoid activation. The bridge connection has the same architecture as an encoder-block and combines the encoder with the decoder.

A residual unit block is a sequence of two blocks, each consisting of a batch normalization[6] followed by a ReLU activation and a convolution of kernel size = 3×3 and stride 1. Its input is summed with its output via an identity mapping connection. An identity mapping connection consists of a convolution of kernel size = 1×1 , a stride of 1, and padding of 0 followed by a batch normalization layer.

type of neural network used per encoding layer, it is the convolutional neural network (CNN) that is chosen for CNNs capture local spatial correlations in images, which is synergetic with our goal to build a model that learns geometrical constraints. Transpose CNN s [111] are used in the decoding phase; they are fractionally-strided convolutions that allow a top-down hierarchical construction of the image, in our case, the design, from the encoded constraints.

The diagram of the complete architecture and the information about each layer’s kernel size, output size, Etc., are detailed in Fig. 3.2 and Tab. 3.1.

3.1.1.2 Discriminators’ architectures

The first one, the traditional discriminator, takes the design along with all its conditions and outputs the probability that it comes from the real data distribution to ensure the generator learns it. The second one, the DL-counter, takes the design and only its mechanical conditions and outputs its Nbr_{bars} to ensure the generator respects the input geometrical condition.

3.1. METHODOLOGY

Table 3.1: Generator (DL-TO) Res-U-Net Architecture. DL-TO’s architecture is composed of three major components: the encoder, the decoder, and the bridge, which links the encoder to the decoder. The encoder is formed of 4 blocks, each consisting of a down-sampling layer (a convolution of stride 2) and a residual unit ([6]). The decoder comprises five blocks, each consisting of an up-sample layer (a transpose convolution of stride 2) and a residual unit followed by a convolution of kernel size 1×1 and a sigmoid activation. The bridge connection has the same architecture as an encoder-block and combines the encoder with the decoder. nf is the number of feature maps i.e. number of channels. The input’s dimension is $101 \times 101 \times 6$ and the output’s dimension is $101 \times 101 \times 1$.

A residual unit block is a sequence of two blocks, each consisting of a batch normalization [6] followed by a ReLU activation and a convolution of kernel size = 3×3 and stride 1. Its input is summed with its output via an identity mapping connection. An identity mapping connection consists of a convolution of kernel size = 1×1 , a stride of 1, and padding of 0 followed by a batch normalization layer.

	Block Level	Layer	Filter	Stride	Output Size
Encoder	Block1	Downsample	$3 \times 3/nf$	2	$50 \times 50 \times nf$
		Residual unit	$3 \times 3/nf$	1	$50 \times 50 \times nf$
	Block2	Downsample	$4 \times 4/nf \times 2$	2	$25 \times 25 \times nf \times 2$
		Residual unit	$3 \times 3/nf \times 2$	1	$25 \times 25 \times nf \times 2$
	Block3	Downsample	$3 \times 3/nf \times 4$	2	$13 \times 13 \times nf \times 4$
		Residual unit	$3 \times 3/nf \times 4$	1	$13 \times 13 \times nf \times 4$
	Block4	Downsample	$3 \times 3/nf \times 8$	2	$7 \times 7 \times nf \times 8$
		Residual unit	$3 \times 3/nf \times 8$	1	$7 \times 7 \times nf \times 8$
Bridge	Block5	Downsample	$3 \times 3/nf \times 16$	2	$4 \times 4 \times nf \times 16$
		Residual unit	$3 \times 3/nf \times 16$	1	$4 \times 4 \times nf \times 16$
Decoder	Block6	Upsample	$3 \times 3/nf \times 8$	2	$7 \times 7 \times nf \times 8$
		Residual unit	$3 \times 3/nf \times 8$	1	$7 \times 7 \times nf \times 8$
	Block7	Upsample	$3 \times 3/nf \times 4$	2	$13 \times 13 \times nf \times 4$
		Residual unit	$3 \times 3/nf \times 4$	1	$13 \times 13 \times nf \times 4$
	Block8	Upsample	$3 \times 3/nf \times 2$	2	$25 \times 25 \times nf \times 2$
		Residual unit	$3 \times 3/nf \times 2$	1	$25 \times 25 \times nf \times 2$
	Block9	Upsample	$4 \times 4/nf$	2	$50 \times 50 \times nf$
		Residual unit	$3 \times 3/nf$	1	$50 \times 50 \times nf$
	Block10	Upsample	$4 \times 4/\frac{nf}{2}$	2	$101 \times 101 \times \frac{nf}{2}$
		Residual unit	$3 \times 3/\frac{nf}{2}$	1	$101 \times 101 \times \frac{nf}{2}$
	Block 11	Convolution	$3 \times 3/1$	1	$101 \times 101 \times 1$

3.1.1.2.1 Traditional discriminator’s architecture The traditional discriminator’s network consists of DL-TO’s encoder followed by a dropout, then a fully connected layer. It outputs a probability p regarding the design being real ($p \approx 1$) or fake ($p \approx 0$). It helps DL-TO learn the mapping from constraints to designs (quality) and capture various constraints (diversity).

3.1.1.2.2 Bar counter discriminator’s architecture The counter discriminator consists of a stem, an Inception/Reduction Resnet-v1-block-A, an Inception/Reduction Resnet-v1-block-B, an Inception Resnet-v1-block-C followed by an average pooling layer, a dropout layer, and a fully connected layer. The stem and inception/reduction blocks used defers from the original paper[85] only by the number of input/output feature maps. They allow for increasing the network’s complexity without generating any extra computational costs.

In this work, the counter is used at three levels: (1) to augment the training dataset, (2) to train, and (3) to evaluate DL-TO.

It is pre-trained on manually labelled SIMP-designs (4347 samples) before the GAN’s full training; the Nbr_{bars} present in each design were manually counted (Fig.3.4).

This pre-trained counter is used to predict the Nbr_{bars} on unlabeled train designs (to augment the labeled training dataset). This pre-training hack improved DL-TO’s convergence.

At each training iteration, this discriminator is used to predict the generated designs’ Nbr_{bars} and penalize DL-TO.

At inference time, it is used to predict the Nbr_{bars} on generated designs to evaluate DL-TO’s geometrical performance (section 3.3).

The counter’s performance is detailed in section 3.3.2.

3.1.1.3 Loss function

This dual-discriminator GAN strives to train a DL-TO method that generates 2D designs of a good quality conforming to input mechanical and geometrical conditions. Thus, the original adversarial loss function (L_{adv}) used to train the generator (Eq. 1.7) was adjusted by the addition of a reconstruction loss (L_r) and a counting loss (L_{count}). The modified generator loss function L_G adapted in the training process is the following:

$$L_G = \lambda_1 L_r + \lambda_2 L_{adv} + \lambda_3 Acc_{count} L_{count} \quad (3.1)$$

3.2. TOPOLOGY OPTIMIZED DESIGNS DATASET CONSOLIDATION

Where $L_r = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2$, $L_{count} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$, with x_i , \hat{x}_i the true and predicted 2D design, y_i the input total bar-count, \hat{y}_i the predicted total bar count in the generated designs and n the batch size. The accuracy of the counter discriminator $Acc_{count} = \frac{1}{N} \sum_{i=1}^N (\hat{t}_i == y_i)$ with y_i , \hat{t}_i the true and predicted total bar count in the real designs and N the total number of training samples; this accuracy is updated at the end of each epoch. λ_i with $i \in 1, 2, 3$ are regularizers to ensure the generator is equally penalized over all three losses.

The adversarial loss encourages creativity and diversity in the generated 2D designs. The reconstruction loss boosts their aesthetic quality and conformity with the volume fraction constraint; the design’s volume fraction is the average of the pixel values (element-wise densities). The counting loss assures the generator respects the input geometrical constraint .

GAN’s training framework is known for its instability; losses oscillating continuously and never converging, thus making a GAN converge requires careful updating and crafting training hacks [95]. In this work, an additional challenge was encountered. The generator’s loss consists of different types of losses having different orders of magnitude: $0 \leq L_r \leq 1$, $0 \leq L_{adv} \leq 100$; due to Pytorch Implementation of the Binary Cross Entropy Loss, $0 \leq Acc_{count} \leq 1$ and $0 \leq L_{count} \leq 961$; the maximum total bar count in the training dataset is 31, hence the maximum L_{count} is $(31 - 0)^2 = 961$. Practically, during training, L_r , L_{adv} , and L_{count} seemed to decrease sharply after only few iterations: $0 \leq L_r \leq 0.1$, L_{adv} ($0 \leq L_{adv} \leq 1$) and $0 \leq L_{count} \times Acc_{count} \leq 30$; the counter discriminator is pre-trained before the training ($Acc_{count} \approx 0.8$). Thus, to ensure that the generator is equally penalized over the three losses, λ_1 , λ_2 and λ_3 were set to 10, 1 and 0.1 respectively.

The traditional discriminator’s loss remains intact: the binary cross-entropy (Eq. 1.7). The counter discriminator’s loss is the Mean Error Squared (MSE) between the true and predicted total bar count on the real designs (i.e., SIMP-based designs).

3.2 Topology Optimized designs dataset consolidation

To train the model, 21538 2D designs were generated following the SIMP via an in-house object-oriented Python version of the academic open-source TO code written by Sigmund([112]). This code is available on the GitHub repository: https://github.com/dbetteb/TOP_OPT.git. The geometrical constraint (Nbr_{bars}) is added by manual labeling over 4347 samples (which are used to pre-train the counter discriminator), and then the Nbr_{bars} of the remaining samples is predicted using the counter

discriminator.

A 2D design of size $n_x \times n_y$ is discretized into a mesh of $(n_x + 1) \times (n_y + 1)$ nodes and is subject to two major constraints: the boundary conditions (BC) i.e. the clamped nodes and the load configurations (F) i.e. the loaded nodes (figures 3.4 and 3.3).

Only truss-like structures were kept to train the GAN-based approach.

To generate a wide variety of designs, the mechanical constraints are sampled using the following strategy [43]:

- The volume fraction follows a normal distribution of the mean of 0.3 and standard deviation of 0.05
- The number of loads follows a Poisson distribution of $\lambda = 2$
- The loads' orientation follows a uniform distribution between 0 and 360 degrees
- The number of clamped nodes follows a Poisson distribution of $\lambda = 50$
- The locations of clamped and load nodes are limited to the edge ones and $n_x = n_y = 100$.

Following [112], the Young modulus E and the Poisson ration ν were set to 1.0 and 0.3.

BC along the x and y axis (BC_x , BC_y) are represented as $(n_x + 1) \times (n_y + 1)$ matrices with null values everywhere except for the clamped nodes set to 1.0; for simplicity, encastrated designs are only considered i.e. BC_x and BC_y are similar. Loads F_x and F_y are represented as $(n_x + 1) \times (n_y + 1)$ matrices with null values everywhere except for the loaded nodes; a loaded node n_e located at line i and column j tilted θ degrees has $F_x(i, j) = F \cdot \cos(\theta)$ and $F_y(i, j) = F \cdot \sin(\theta)$; the magnitude F of the loads were set to 1.0 N . To concatenate input conditions altogether, the 2D design, the volume fraction (V), and the Nbr_{bars} are reshaped into a $(n_x + 1) \times (n_y + 1)$ matrix each (Fig. 3.5).

The dataset was separated into a train (80%) and a test (20%). The test set is used to evaluate the generator's performance.

3.3 Experiments and Results

In this section, we evaluate the aesthetics of the generated designs, as well as their conformity with the input volume fraction and geometrical (Nbr_{bars}) constraints. We also compute their compliance

3.3. EXPERIMENTS AND RESULTS

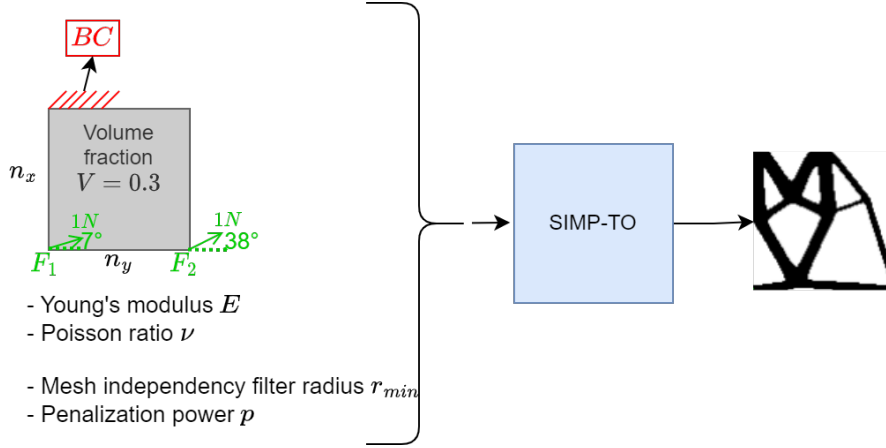


Figure 3.3: Input to SIMP. to output a 2D design, SIMP takes as input the dimension of the design space (height n_x and width n_y), the mechanical constraints (the boundary conditions BC , the loads F , and the volume fraction V), the material criteria (the Young modulus E and the Poisson ration ν), finally the mesh-independency filter r_{min} and the penalization power p .

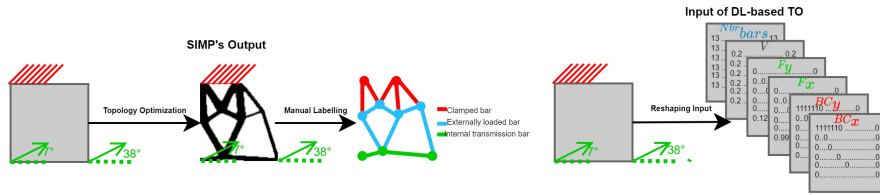


Figure 3.4: Types of bars in a design. The design shown here is clamped from the upper-left edge and loaded with two punctual external forces located in the bottom corners tilted 7 and 38 degrees. Indeed, It has five clamped (red) bars, two externally loaded (green) bars outgoing from two nodes corresponding to the forces locations and six internal-transmission (blue) bars, thus, a total of 13 bars.

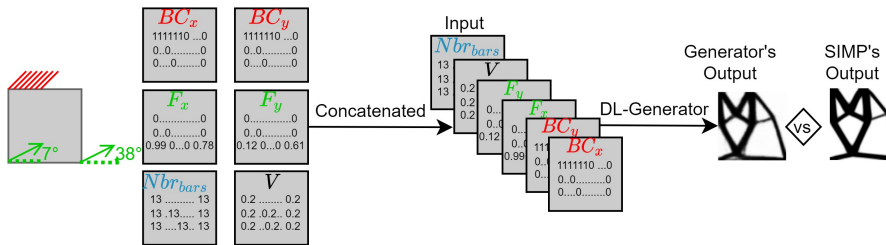


































Figure 3.5: Input of the DL-TO. The boundary conditions (BC_x , BC_y), load configurations (F_x , F_y), volume fraction V and geometrical constraint Nbr_{bars} (i.e. total number of bars) are formulated as a six-channel image.

3.3. EXPERIMENTS AND RESULTS

Design	Original Design								
	Generated Design								
Metrics	$eV_{\%}$	-3.9	-2.07	-3.64	-7.94	-1.58	2.37	-3.73	-1.11
	$eC_{\%}$	20.12	12	2.41	39.27	9.39	6.94	44.24	37.7
	ΔC_x	0	0	0	+2	+1	-1	0	+1

(a) Without Threshold.

Design with Threshold	Original Design (Threshold = 0.4)								
	Generated Design (Threshold = 0.4)								
Metrics With Threshold	$eV_{\%}$	-1.36	-0.16	-3.72	-7.16	-1.34	3.45	0.17	-3
	$eC_{\%}$	3.75	-2	2.83	31.75	-2.53	0.48	4.31	21.1
	ΔC_x	0	0	0	+1	0	0	0	-1

(b) With Threshold

Figure 3.6: This figure compares the aesthetics, volume fraction ($eV_{\%}$), compliance ($eC_{\%}$) and geometry (ΔNbr_{bars}) of the original (SIMP-based) versus generated (DL-based) designs with and without Threshold. In both cases, DL-designs are barely indistinguishable, in terms of shape, from SIMP-designs, achieve lower Volume Fractions ($eV_{\%} \leq 0$) and respect the Complexity constraint ($|\Delta Nbr_{bars}| \leq 2$). However, while the Compliance of DL-designs is higher than SIMP-designs before the threshold, it is significantly reduced after the threshold.

3.3. EXPERIMENTS AND RESULTS

values to examine their mechanical performance.

Most state-of-the-art GANs evaluations are subjective, based on the creativity and aesthetics of the generated samples. In this study, generated designs are assessed not only qualitatively but also quantitatively.

The metrics used for the volume fraction V and compliance C , the energy of deformation (C) are the relative errors $e_{V\%} = \frac{V_g - V_o}{V_o} \times 100$ and $e_{C\%} = \frac{C_g - C_o}{C_o} \times 100$ respectively. The metric of Nbr_{bars} is the bar-count difference $\Delta Nbr_{bars} = Nbr_{bars_g} - Nbr_{bars_o}$. Where X_g, X_o refer to generated and original respectively and $\{X_g, X_o : X \in \{V, C, Nbr_{bars}\}\}$.

V_{design} is the mean of density material, i.e., the mean of the pixel values of the design. C_{design} is computed via a compliance-FE-calculator coded in Pytorch. $C_{x_{design}}$ is predicted via the DL-counter-discriminator.

Section 3.3.1 presents the performance of DL-TO on a sample of the test set. Section 3.3.3 summarizes the overall performance of DL-TO (considering all 4308 test samples). Section 3.4.5 demonstrates DL-TO's capability to tailor design's geometry (via the total bar count). Section 3.4.6 shows DL-TO's capability to generate creative designs with previously unseen input constraints: boundaries of design area (passive and active elements). Finally, section 3.4 shows the results of integrating a third discriminator into DL-TO's training framework and demonstrates DL-TO's capability to account for this additional constraint.

3.3.1 DL-TO's performance

The figure 3.6(a) displays a sample of original versus reconstructed designs from the test set. The generated designs are very similar to the SIMP-ones. The clamped and loaded bars are well-reconstructed, showing that DL-TO respects the input boundary conditions and load configurations. In the majority of cases, $e_{V\%}$ is negative; in its worst case, it never exceeds 5% of extra material. In other terms, DL-TO outperforms SIMP at finding the minimal material distribution (volume fraction). However, $e_{C\%}$ is relatively high (red values in Fig.3.6(a)); showing that the generated designs exhibit high external stresses.

The underlying reason is that compliance is very sensitive to intermediate-density-pixel values. And, generated designs (by SIMP or DL-TO) are continuous and hence can embed such intermediate values. Therefore, to account for this drawback, a threshold is applied to all designs, then the compliance

is recomputed. Results are reported in Fig. 3.6(b). As expected, in most cases, the compliance of the generated designs dropped and sometimes to an even lower value than that achieved by SIMP (negative $e_{C\%}$).

It should be noted that the application of a threshold causes the volume fraction to increase. Yet, DL-designs still achieve lower volume fraction than the SIMP-designs.

The geometrical constraint is well respected (after and before the threshold). Complex generated designs (with $Nbr_{bars} \geq 20$) sometimes display additional or fewer internal bars. Figure 3.6 shows that the range of bar-difference is at most of more or less two bars.

In the next section, additional statistics are computed over all the generated designs of the test set to check for the DL-TO’s overall performance.

3.3.2 Counter-discriminator’s performance

As previously mentioned, the counter discriminator is pre-trained on manually labeled SIMP designs (4347 samples) before the GAN’s full training.

The 2D designs in the dataset consist of structures with 3 to 31 components, which is a wide range for the Nbr_{bars} variable. An admissible prediction falls within an interval $|\Delta Nbr_{bars}| \leq 2$. The counter’s performance on the train (3885 designs) and test (462 designs) sets:

1. on the train set: 99.8% of the predictions fall within $|\Delta Nbr_{bars}| \leq 1$ bar.
2. On the test set: 85.4% of the predictions fall within $|\Delta Nbr_{bars}| \leq 1$ bar while 94.9% of them within $|\Delta Nbr_{bars}| \leq 2$ bars.

In conclusion, the counter-discriminator is validated.

3.3.3 Overall performance

This section summarizes the overall aesthetic, mechanical, geometrical, and computational performance of the DL-TO on the test set.

The generated designs are aesthetically plausible; the average reconstruction error (MSE) of the generated test designs is 0.025 (Fig 3.7). Additionally, they respect the geometrical constraint within an error margin of ± 2 bars. 86% of the DL-designs show, at most, 2 additional/fewer bars ($|\Delta Nbr_{bars}| \leq 2$).

3.3. EXPERIMENTS AND RESULTS

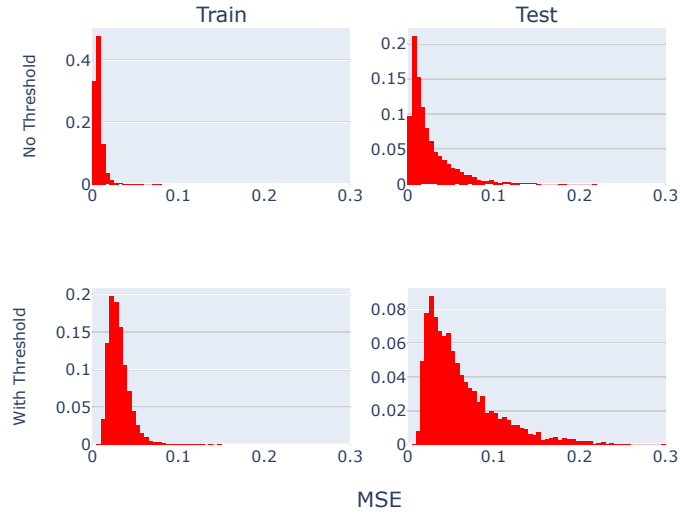


Figure 3.7: Distribution of the reconstruction error in the train and test sets before and after threshold. The Mean Error Squared $MSE = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2$ where x_i and \hat{x}_i are the original (SIMP-based) and generated (DL-based) design.

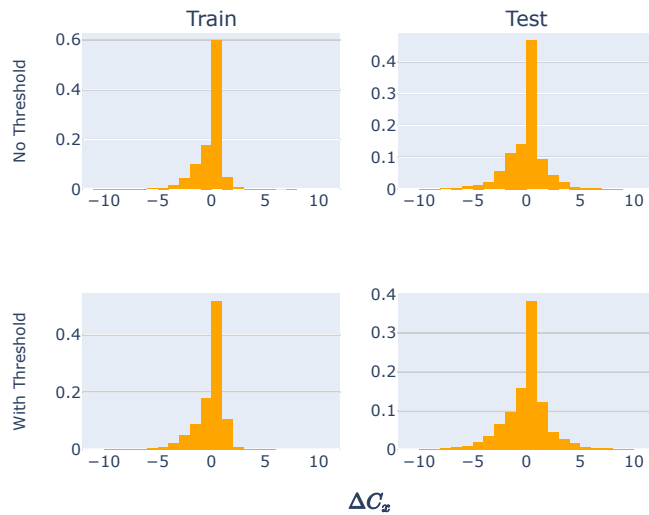


Figure 3.8: Overall Geometrical Performance of DL-TO. This figure plots the distribution of the Nbr_{bars} difference before and after threshold in the train and test sets.

3.3. EXPERIMENTS AND RESULTS

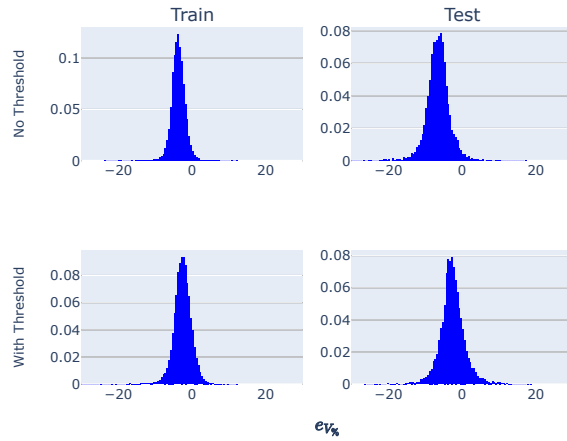


Figure 3.9: Relative error of the volume fraction

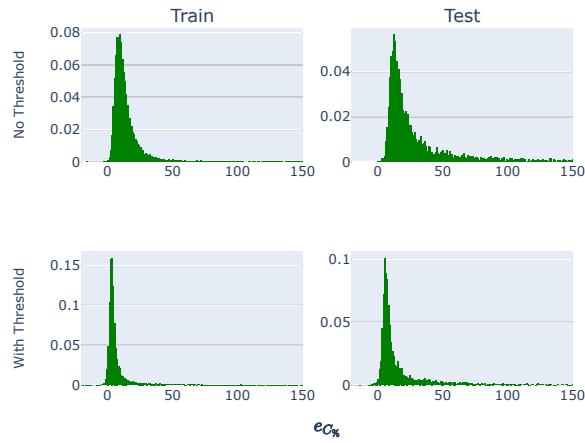


Figure 3.10: Relative error of the compliance

Figure 3.11: Overall Mechanical Performance of DL-TO. This figure plots the distribution of the relative error of Volume fraction and compliance before and after threshold in the train and test sets.

3.3. EXPERIMENTS AND RESULTS

Table 3.2: Average Design Generation Time (in seconds) of SIMP (FE-based) versus DL-TO.

Dataset	Train	Test
SIMP (on CPU)	642.81	140.85
DL-TO (on CPU)	0.043	0.0389
DL-TO (on CPU)	0.0029	0.00288

Also, 94% of DL-designs achieve a lower volume fraction than that achieved by SIMP (Fig.3.9). Nevertheless, DL-designs tend to exhibit higher external stresses. 50% of them score a compliance value more than 1.2 times that obtained by SIMP ($C_g \geq 1.2 \times C_o$, Fig.3.10).

One of the reasons is that the generator was not penalized explicitly on the compliance during the training. The integration of a compliance predictor as a third discriminator into our GAN could improve the generated designs' compliance (Section 3.4). It is interesting to point out that the DL-designs comply better with the Nbr_{bars} and volume fraction constraints, for the generator was penalized on the reconstruction error (it embeds the volume fraction constraint implicitly) and the Nbr_{bars} error during training.

The designs are also compared after the application of the threshold. As expected, after the elimination of intermediate-density values, the relative error of the compliance dropped; ($e_{C\%} \leq 20\%$) in 70% of the cases.

Second rows of Figures 3.9 and 3.10 plot $e_{V\%}$ and $e_{C\%}$ distributions after the threshold. 83% of the generated designs comply with the volume constraint with a relative error $e_{V\%} \leq 0\%$ and 14.6% with $0 < e_{V\%} \leq 5\%$. Hence, 97.6% of the test set respect the volume fraction within a relative error margin of 5% of extra material; 5% of extra material is still admissible.

Also, 80% of the generated designs comply with the geometrical constraint within an error margin of ± 2 bars.

It is important to underline that applying a threshold is critical to the design's mechanical and geometrical performance: the lower the threshold, the higher the volume fraction ($e_{V\%}$ increased slightly after the application of the threshold), the higher the threshold, the lower the bar-count (ΔNbr_{bars} also increased after the application of threshold), and possibly the appearance of discontinuities in the design. Consequently, a better approach is to apply a local threshold. Finally, the computational performances of DL-TO FE-TO SIMP are compared. DL-TO generates a design 3500 times (141/0.04)

3.4. IMPROVING THE PERFORMANCE OF DL-TO

Table 3.3: Generation Time (in seconds s) and Computational Complexity (in Gega Floating Point Operations per Second $GFLOPS$) of SIMP (FE-based) versus DL-TO. CPU and GPU stands for central and graphical processing unit, respectively

Input Constraints	Generation Time (s)		Computational Complexity ($GFLOPS$)	
	SIMP	DL-TO	SIMP	DL-TO
1 Load	68	0.02	62.81	2.27
2 Loads	132	0.02	125.89	2.27
10 Loads	656	0.02	620.28	2.27

faster on CPU and 47000 times (141/0.003) faster on GPU (Tab. 3.2).

On top of that, a DL-TO’s generation time and computational complexity are independent of the input constraints, unlike traditional TO approaches as shown in Tab.3.3. For SIMP-TO, the computational time and complexity increase with the complexity of the input constraint (here, the number of loads), while they remain unchanged for DL-TO.

Furthermore, DL-TO is advantageous over SIMP in terms of geometrical control at the conceptual level; it takes into consideration not only mechanical constraints but also a geometrical one, SIMP-TO needs post-processing to integrate this additional constraint. This aspect is demonstrated in section 3.4.5.

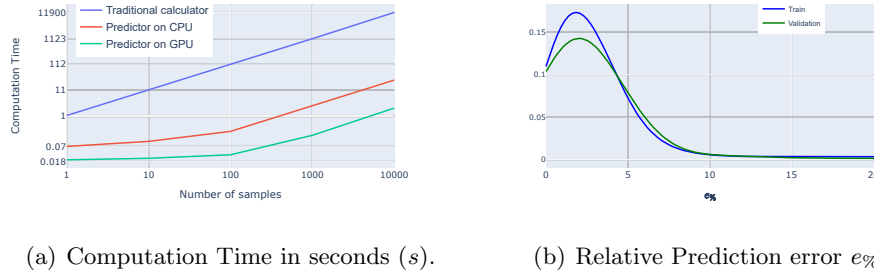
To sum up, the overall performance of DL-TO is promising. It generates mechanically and geometrically valid designs, indistinguishable to the naked eye from those generated by SIMP while being thousands of times faster.

Thus, it offers the designer an alternative way to explore designs faster and to easily adjust their geometry (here defined by the number of bars composing a design). Tuning the mechanical and geometrical conditions can be done effortlessly, and the result is obtained in a fraction of a second.

3.4 Improving the performance of DL-TO

As proposed previously in section 3.3.3, one solution to improve the compliance of the generated designs via DL-TO is to retrain the DL-TO with a compliance predictor as a 3rd discriminator. Hence, in this section, a DL-TO is trained via a triple-discriminator-GAN framework: a traditional discriminator, a DL-counter, and a DL-based-compliance-predictor. The major difference is in the loss function, which will need to consider an additional error the compliance.

3.4. IMPROVING THE PERFORMANCE OF DL-TO



(a) Computation Time in seconds (s).

(b) Relative Prediction error $e\%$.

Figure 3.12: This figure shows the relative prediction error of the DL-Compliance-Predictor and compares its computational efficiency versus the traditional FE-based-compliance-calculator.

3.4.1 Compliance predictor discriminator’s architecture

The compliance predictor’s architecture is a sequence of seven residual convolutional layers with squeeze-excitation layers in-between [113] followed by a dropout and a fully connected layer. It takes an image-like 2D design as input and outputs an estimation of the compliance value. It is pre-trained on real 2D designs (output by SIMP). Its performance is detailed in section 3.4.2. It is interesting to highlight that an FE-based compliance evaluator could have conducted the training instead of a DL-based one. However, this is disadvantageous in terms of training time. A FE-compliance computation is thousands times slower than its DL-counterpart (section3.4.2).

3.4.2 Compliance-predictor’s performance

The DL compliance predictor’s performance is evaluated by two metrics: the computation time (Fig. 3.12(a)) and prediction error $e\%$, which is the relative error between the true and predicted compliance values on the real designs (Fig. 3.12(b)). 87% of the test predictions fall into a 5% error margin, and 93% of them are made within a 10% error margin. Moreover, it computes the compliance of one design 16 times (on CPU) and 56 times (on GPU) faster than the FE-based calculator and 400 (on CPU) to 5000 (on GPU) times faster for a batch of designs (Fig. 3.12(a)). Thus, the advantage of the DL compliance predictor over its FE counterpart is its prediction speed, especially in batch prediction, allowing faster GAN training within an acceptable level of precision.

3.4.3 Training Loss Function

This triple-discriminator GAN strives to train DL-TO to generate 2D designs of good quality and compliant with the input mechanical and geometrical conditions. Thus, the original adversarial loss function (L_{adv}) used to train the generator is adjusted by adding the reconstruction (L_r), counting (L_{count}) and compliance (L_C) losses. The modified generator loss function L_G adapted in training is the following:

$$L_G = \lambda_1 L_r + \lambda_2 L_{adv} + \lambda_3 Acc_{count} L_{count} + \lambda_4 L_C \quad (3.2)$$

Where $L_r = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2$, $L_{count} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$, $L_C = \frac{1}{n} \sum_{i=1}^n |C_i - \hat{C}_i|$ with x_i , \hat{x}_i the true and predicted 2D design, y_i the input Nbr_{bars} , \hat{y}_i the predicted Nbr_{bars} of the generated design, C_i the compliance of the real design, \hat{C}_i the compliance predicted over the generated one, and n the batch size. The accuracy of the counter discriminator $Acc_{count} = \frac{1}{N} \sum_{i=1}^N (\hat{t}_i == y_i)$ with y_i , \hat{t}_i the true and predicted Nbr_{bars} in the real designs and N the total number of training samples.

The adversarial loss encourages creativity and diversity in the generated 2D designs. The reconstruction loss boosts their aesthetics and conformity with the volume fraction constraint; the design's volume fraction is the average of the pixel values. The counting loss assures the generator respects the input geometrical constraint. Finally, the compliance loss compensates for the weak mechanical performance exhibited in section 3.3.3.

GAN's training framework is known for its instability; losses oscillating continuously and hardly reaching a nash equilibrium. Thus, making a GAN converge requires careful updating and crafting training hacks (Gui et al. [95]). In this work, an additional challenge was encountered. The generator's loss consists of different types of losses having different orders of magnitude: $0 \leq L_r/L_C \leq 1$, $0 \leq L_{adv} \leq 100$; its Pytorch implementation, $0 \leq Acc_{count} \leq 1$ and $0 \leq L_{count} \leq 961$; the maximum Nbr_{bars} in the training dataset is 31, hence the maximum L_{count} is $(31 - 0)^2 = 961$.

Practically, during training, L_r , L_{adv} , and L_{count} seemed to decrease sharply after only few iterations: $0 \leq L_r \leq 0.1$, $0 \leq L_{adv} \leq 1$, and $0 \leq L_{count} \times Acc_{count} \leq 30$; the counter discriminator is pre-trained before the training ($Acc_{count} \approx 0.8$). Thus, to ensure that the generator is equally penalized over the three losses, λ_1 , λ_2 and λ_3 were set to 10, 1 and 0.1 respectively. λ_4 was set to 0.01 after several trial-and-error tests.

The traditional discriminator's loss remains intact: the binary cross-entropy (Eq.3 in the manuscript). The rest of the discriminators were frozen during the GAN training.

3.4. IMPROVING THE PERFORMANCE OF DL-TO

Table 3.4: Comparison of DL-TOs trained without/with a Compliance Predictor as a 3rd discriminator. MSE is the average mean squared error. $e_{V\%} \leq 5\%$ is the percentage of DL-designs showing a volume fraction less than 1.05 the SIMP-designs’ one. $|e_{C\%}| \leq 10\%$ and $|e_{C\%}| \leq 5\%$ are the percentages of DL-designs within ± 1.1 and ± 1.05 the compliance achieved by SIMP, respectively. $|\Delta Nbr_{bars}| \leq 2$ is the percentage of DL-designs conforming with the Nbr_{bars} constraint within an error margin of ± 2 bars. Adding a Compliance Predictor improved the generated designs’ compliance by **21.3%** and **45.4%** for an error margin of 10% and 5%, respectively.

Training Method	Aesthetics	Mechanical Performance			Geometrical Performance
	MSE	$e_{V\%} \leq 5\%$	$ e_{C\%} \leq 10\%$	$ e_{C\%} \leq 5\%$	$ \Delta Nbr_{bars} \leq 2$
Without Compliance	0.063	97.7%	52.4%	15.4%	82.4%
With Compliance	0.065	93.6%	73.7%	60%	73.7%

It is worth noting that the relative error is used to compare the generated designs’ compliance versus the SIMPs, which is the $L1 - norm$ between the compliances of the DL and SIMP designs divided by the SIMP design’s compliance. Thus, using the $L1 - norm$ in the loss function penalizing DL-TO’s training is the best choice, especially since the objective is to generate designs with lower compliance. However, $L2 - norm$ was also tested. With $L2$, the DL designs always showed higher compliances than their SIMP counterparts. Consequently, the $L1 - norm$ was chosen for the L_C .

3.4.4 DL-TO’s Performance after training with three discriminators

This section summarizes the overall performance of DL-TO and compares the results obtained in section 3.3.3. The generated designs are thresholded to dampen any intermediate density values.

Table 3.4 clearly shows that C is improved by 21.3% with the integration of the compliance predictor as a third discriminator without any loss of generality. Moreover, this improvement increases remarkably, by 45.4%, if we restrain the error to 5% (15.4% versus 60%).

For a better visualization, figure 3.13 plots $e_{V\%}$, $e_{C\%}$ and ΔNbr_{bars} distributions of the test set using DL-TO’s previous and improved version. The improvement over C is clear, the median $e_{C\%}$ is 2.52% versus 9.3%. The distributions of $e_{V\%}$ and ΔNbr_{bars} remain similar. In other terms, the addition of the third discriminator enhanced the compliance without deteriorating the initial performance.

A sample of the generated designs by DL-TO trained via the triple-discriminator-GAN framework compared to the ones generated via a model trained by the double-discriminator-GAN is illustrated by the figures 3.14(a) and 3.14(b), respectively; a global threshold is applied over the generated designs.

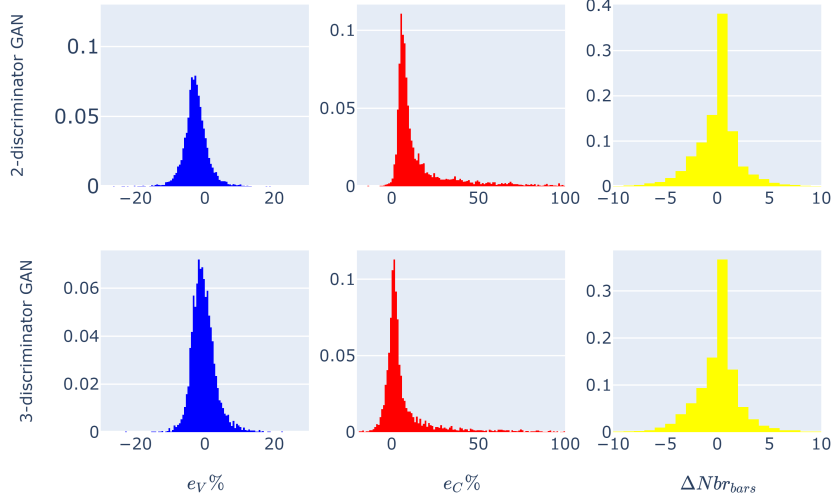










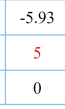
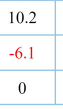
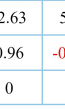

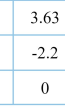
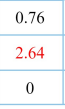


Figure 3.13: The distributions of the relative errors of volume fraction and compliance ($e_{V\%}$ and $e_{C\%}$) and the difference between the number of design components (ΔNbr_{bars}) computed over the test set for the DL-TO trained with two versus three discriminators.

As expected, the ones generated from the new DL-TO version show lower deformation energy (C). It is essential to underline that applying threshold is critical to the design’s mechanical and geometrical performance: the lower the threshold, the higher the V ; the higher the threshold, the lower the Nbr_{bars} with possibly the appearance of discontinuities in the design. Hence, the optimal approach is to privilege a local threshold.










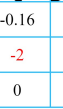

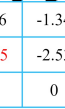
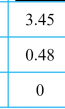



To sum up, the overall performance of DL-TO is promising. It generates mechanically and geometrically valid designs, indistinguishable to the naked eye from those generated by SIMP while being thousands of times faster. Hypothesis 4 is validated.

Thus, it offers the designer an alternative way to explore designs faster and easily adjust their geometry (here defined by Nbr_{bars}). Tuning the mechanical and geometrical conditions is accomplished fast and effortlessly. Furthermore, the choice of GANs is justified. The addition of discriminators is advantageous, and when the right balance between their losses is found, transferring new knowledge to the generator is possible and is demonstrated here.

3.4. IMPROVING THE PERFORMANCE OF DL-TO

Design with Threshold	Original Design (Threshold = 0.35)								
	Generated Design (Threshold = 0.35)								
Metrics With Threshold	$e_{V\%}$	-5.93	10.2	-2.63	5.31	4.2	3.63	0.76	7.7
	$e_{C\%}$	5	-6.1	0.96	-0.066	-3.3	-2.2	2.64	-1.73
	ΔC_x	0	0	0	+2	0	0	0	0

(a) Generation by DL-TO trained with an additional compliance predictor.

Design with Threshold	Original Design (Threshold = 0.4)								
	Generated Design (Threshold = 0.4)								
Metrics With Threshold	$e_{V\%}$	-1.36	-0.16	-3.72	-7.16	-1.34	3.45	0.17	-3
	$e_{C\%}$	3.75	-2	2.83	31.75	-2.53	0.48	4.31	21.1
	ΔC_x	0	0	0	+1	0	0	0	-1

(b) Generation by DL-TO trained via a double-discriminator-GAN framework

Figure 3.14: Comparison between the generated designs from two DL-TO models, designs in Fig. 3.14(a) are generated by a DL-TO trained with an additional DL compliance predictor. The generated designs' mechanical performance (the compliance) is improved after integrating the DL compliance predictor as a third discriminator.

3.4.5 Tailoring the design's geometry via DL-TO

DL-based generative models sometimes suffer from what is called a memorization problem; the model memorizes the training dataset, and a slight variation in the input leads to noisy and meaningless outputs; or a mode collapse; when the model reproduces only one mode, in our case, one geometry [114]. Let us recall that the primary objective of DL-TO is to tailor the design's geometry while always respecting mechanical constraints; hence, to examine that DL-TO does not memorize the input constraints and understands the mechanism of this geometric condition, this experiment is realized. To validate DL-TO's understanding of geometry, the mechanical conditions (BC , F and V) were prepended, only Nbr_{bars} is varied (Fig 3.15(a)). Indeed, the Nbr_{bars} in the design increases with the input variable.

However, the supplementary bars are blurry, and DL-TO struggles to conform with the input Nbr_{bars} , especially for the lower and upper extreme values (7 and 30).

In addition, V and Nbr_{bars} of a design are correlated. Thus, a better approach would be to vary them together (Fig. 3.15(b)).

3.4. IMPROVING THE PERFORMANCE OF DL-TO

Modifying V and Nbr_{bars} simultaneously improves DL-TO's conformity with the input geometrical and V constraints (ΔNbr_{bars} and $e_{V\%}$ decreased in Fig. 3.15(b)), and most importantly demonstrates DL-TO's creativity in adding/removing internal bars to/from the design. Nevertheless, while C improved for $Nbr_{bars} = 20$ and 30 , it exploded for $Nbr_{bars} = 7$. This phenomenon implies a minimum Nbr_{bars} for every set of mechanical conditions to guarantee the design's integrity and resilience.

To identify this minimum value, a set of mechanical constraints are fixed, and Nbr_{bars} is varied between 3 to 30 bars. Then, the designs are generated and evaluated (Fig.3.16). C and Nbr_{bars} are inversely correlated, and the minimum admissible Nbr_{bars} of the considered example is 15, after which C does not show any remarkable improvement. This information implies that the designer can choose any design with 15 to 30 components and guarantee that its mechanical performance will not deteriorate, knowing that this process takes a fraction of a second, thanks to DL.

To further explore this constraint, a new experiment is added to examine the effect of modifying locally the Nbr_{bars} input matrix on the generated designs. A sample of results are shown in Fig. 3.17.

There are six modifications in three different locations (top-left, middle-left, and bottom-left) and of different sizes (alterations of 6×6 and 21×21).

When the local change is via the smaller matrix (of dimension 6×6 , in Fig. 3.17 a), b), and c)), the shape and, remarkably, the bars' distribution are changed locally; its effect is local; it does not trigger the modification of the whole shape. However, while the chosen value is 5, the added bars were not exactly five.

On the other hand, when the local change is via the bigger matrix (of dimension 21×21 , in Fig. 3.17 d), e), and f)), its effect is similar to adding an active element (a boundary of design areas) as demonstrated in section 4.5 in the main article. Moreover, the shapes are not always interpretable, when they are, the chosen value (here, 5) is not respected.

To sum up, controlling locally the Nbr_{bars} is not precise via DL-TO. Nevertheless, it is important to note that the input Nbr_{bars} was shaped as a matrix only for convenience with the input conditions' shape (all other inputs are 2D matrices) and not to control the Nbr_{bars} locally. This objective might be handled in future work.

Hypothesis 5 is validated; DL-TO tailors the geometry of a design globally, not locally.

Finally, it is essential to highlight that the geometrical constraint used in this work (Nbr_{bars}) might not be the most demonstrative example because Nbr_{bars} is more precisely controllable by the primi-

3.4. IMPROVING THE PERFORMANCE OF DL-TO

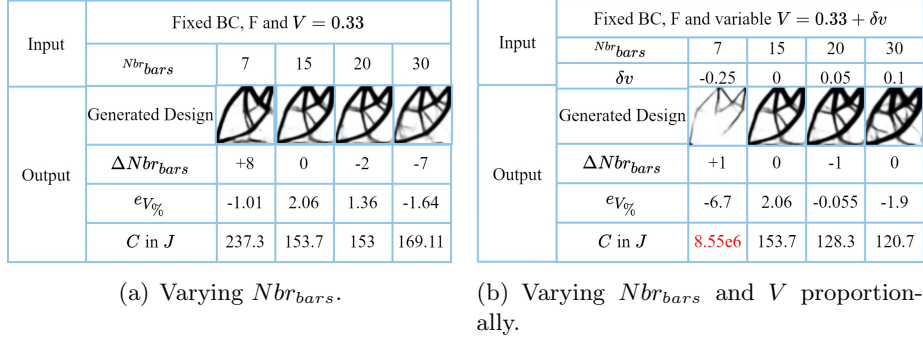


Figure 3.15: Tailoring the geometry of a design. a) The mechanical constraints (BC , F and V) are fixed and the geometrical condition Nbr_{bars} is variable. b) BC and F are fixed while Nbr_{bars} and V are variable.

tives/supershapes methods [115, 116, 117]. However, it is proof that a geometrical constraint requiring seeing the design as an image, typically an aesthetic constraint (filling the design space) or a complicated constraint to formulate and control analytically (informal experts' AM rule), can be integrated via DL.

3.4.6 Generating designs with a new unseen constraint

Some spatial constraints can be enforced on the design area's boundaries. These boundaries can be defined by the V matrix.

This section examines DL-TO's potential to propose new designs accounting for this constraint. It should be noted that it was trained using only input V without any design area's boundaries.

Since the input V is formulated as a matrix of $(n_x + 1) \times (n_y + 1)$ elements, the existence/absence of material in particular locations of the design space is obtained by increasing/decreasing the values in these locations.

This formulation allows many geometrical constraints to be integrated into the design, like passive elements (e.g., a hole for a pipe) or active elements (e.g., a filled shape for an external pillar). It also allows the addition of boundary and load conditions on these elements by modifying the BC and F matrices. This aspect is not explored in this research.

Figure 3.18 illustrates a sample of constraints without/with different design area's boundaries and the corresponding generated threshold designs. In all three cases, DL-TO filled/emptied the locations

3.4. IMPROVING THE PERFORMANCE OF DL-TO

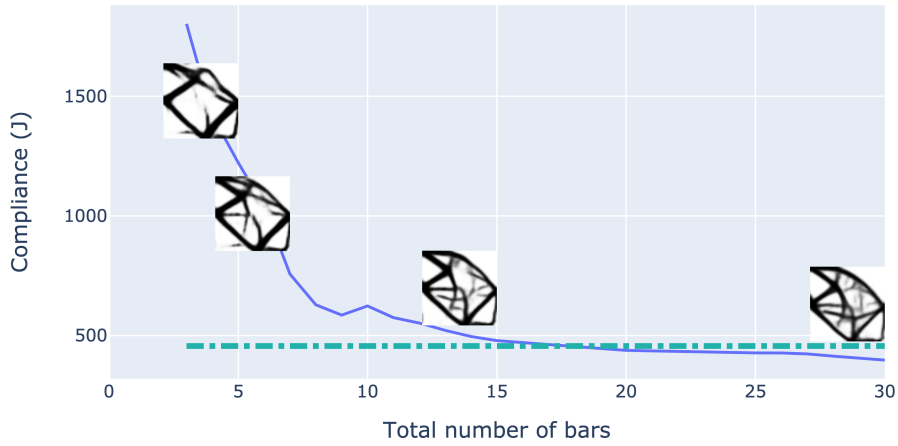


Figure 3.16: Compliance versus Nbr_{bars} variation for a set of fixed mechanical constraints. The maximum admissible compliance is 450 J (the green line). The minimum admissible Nbr_{bars} here is 15 bars, and the lowest compliance is achieved at $Nbr_{bars} = 30$.

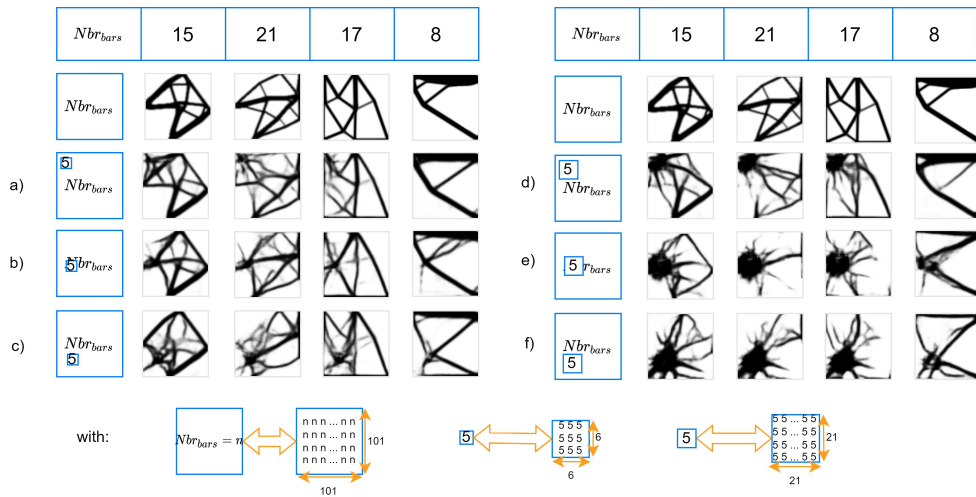


Figure 3.17: The effect of a local variation in the Nbr_{bars} input matrix. Six modifications are considered: a) the Nbr_{bars} input matrix is altered in the top-left by a matrix of 6×6 with the value equaling to 5, b) in the middle-left by a matrix of 6×6 with the value equaling to 5, c) in the bottom-left by a matrix of 6×6 with the value equaling to 5, d) in the top-left by a matrix of 21×21 with the value equal to 5, e) in the middle-left by a matrix of 21×21 with the value equal to 5 and f) in the bottom-left by a matrix of 21×21 with the value equaling to 5.

with/from material where extra/no material is forced; it reshaped the internal truss-like shape creatively to maintain the design's integrity while preserving the original design's outer shape. In other terms, it respects the input BC and F configurations. As for the compliance, it decreased in 4 out of 15 cases, its increase was modest ($e_{C\%} \leq 10\%$) in 5 cases, and its increase was more than 10% in the 6 left cases (Figures 3.18 and 3.19).

Finally, this experiment was replicated using SIMP (Fig. 3.19). As clearly seen, SIMP does not always comply with the constraint. In Fig. 3.19.a., the additional material is on the left side of the domain space; SIMP responded with thicker-bars designs instead of adding extra material only to the left side; in other terms, the volume fraction constraint is considered globally and not locally. While not conforming with the constraint, it is worth mentioning that these designs benefited from a lower compliance.

In Fig. 3.19.b. where a hole was enforced, SIMP failed to converge to a solution.

To sum up, DL-TO shows an encouraging result in creatively conforming to geometrical constraints, and its convergence is not easily compromised, especially when trained on converged designs.

To the best of the authors' knowledge, DL-TO is the first strategy that allows for the natural handling of active and passive elements while not explicitly being trained on such samples, demonstrating its generalization capability. [55] has also generated designs with passive and active elements. However, unlike our model, his was trained on such examples.

Hypothesis 6 is validated; DL-TO generates designs with active and passive elements without being trained to do so.

3.5 Web application of DL-TO

An open-accessible web application called GAIMD (Generate Artificial Intelligent-based Mechanical designs) was built by our intern Corentin MAGYAR, and presented at the 34th international conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems IEA/AIE 2021 [110] to facilitate the comparison between DL-based designs and SIMP designs.

Figure 3.20 shows the interface of the application.

The user defines the geometrical variable (Nbr_{bars}), the volume fraction, the boundary conditions (the user needs to turn on the BC button to enable their selection), the loads (the user needs to turn on the F button to enable their selection, which triggers a textbox to fill the angle value of the load).

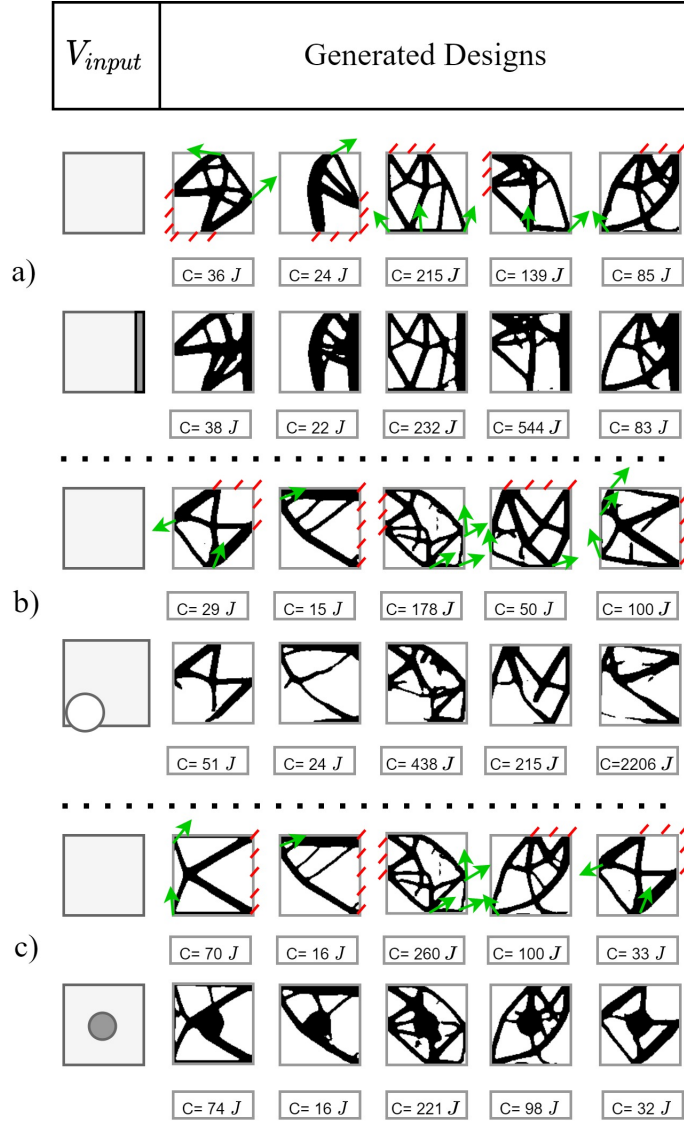


Figure 3.18: Sample of DL-TO designs generated without versus with constraints on design area's boundary as input (V_{input}). All designs are threshold. a) Additional material is added on the right edge. b) A hole (passive element) of radius $r = 30$ is enforced at the bottom left of the design space. d) An active element of radius $r = 15$ is enforced at the centre of the design space. The BC (in red) and F (in green) conditions are only shown on the first row.

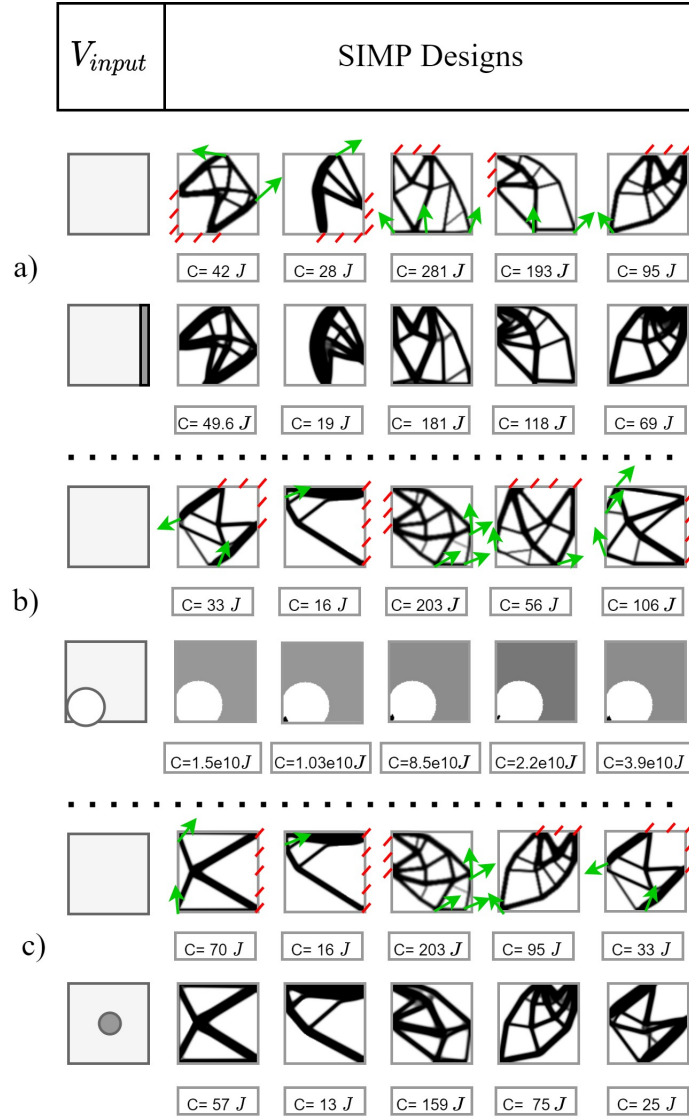


Figure 3.19: A sample of SIMP designs generated without versus with constraints on design area's boundary as input (V_{input}). All designs are threshold. a) Additional material is added on the right edge. b) A hole (passive element) of radius $r = 30$ is enforced at the bottom left of the design space. d) An active element of radius $r = 15$ is enforced at the centre of the design space. The BC (in red) and F (in green) conditions are only shown on the first row.

3.6. DISCUSSION

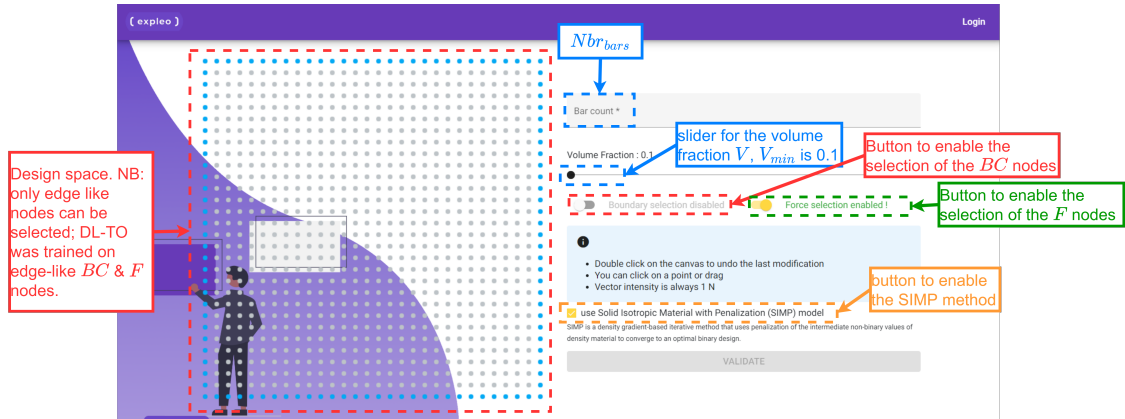


Figure 3.20: Interface of GAIME. GAIME is an open-accessible web application that takes the mechanical conditions: the boundary conditions (BC), the loads (F), and the volume fraction (V) and the geometrical condition (Nbr_{bars}) and generates a 2D design using a DL-TO and optionally the SIMP method. Only edge nodes (blue nodes) from the design space can be selected to be the BC and F nodes.

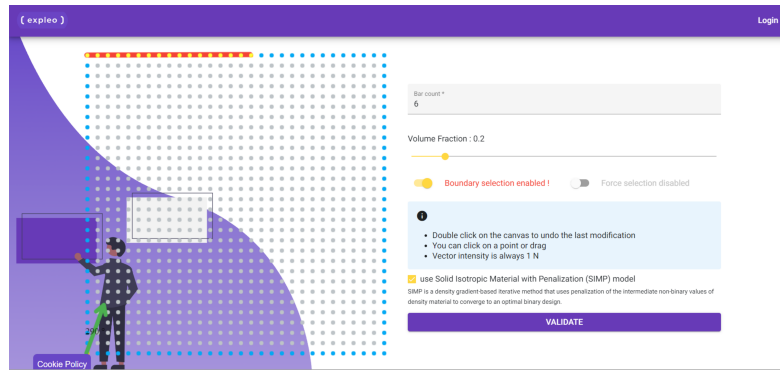
Figure 3.21 shows an example of how to use GAIME. Inputs are selected in figure 3.21(a); the box “Using Solid Isotropic Material with Penalization” is selected; hence, two designs will be outputted, one by SIMP and the other by DL-TO. Figure 3.21(b) shows the results interface. The app returns the design generated by DL-TO and the one outputted by SIMP from the top down, respectively. It also computes the energy of deformation, C , and the V constraints for both designs and compares them by calculating the relative errors. Finally, an additional threshold feature is added; the user can threshold the DL-based design and download it along the input constraints and computed mechanical performance.

3.6 Discussion

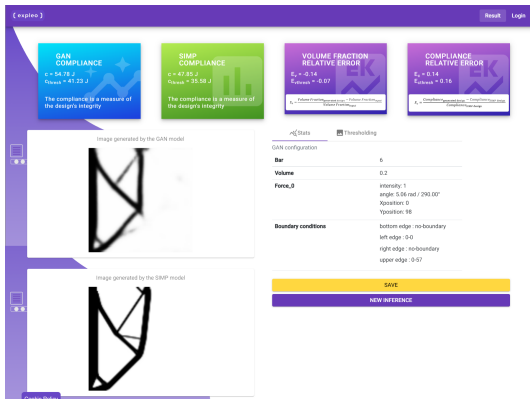
To recapitulate, the first approach, DL-TO, validates the following hypotheses. DL-TO is capable of generating designs of good mechanical and geometrical quality 4; section 3.4.4. DL-TO demonstrated its capacity into tailoring the global geometry of a design while always respecting the mechanical constraints 5; section 3.4.5. Finally, its generation performance goes beyond the training dataset; it generates designs with passive and active elements without being trained to do so 6; section 3.4.6.

In conclusion, DL can compensate for the difficulties faced by TO when dealing with the mechanical

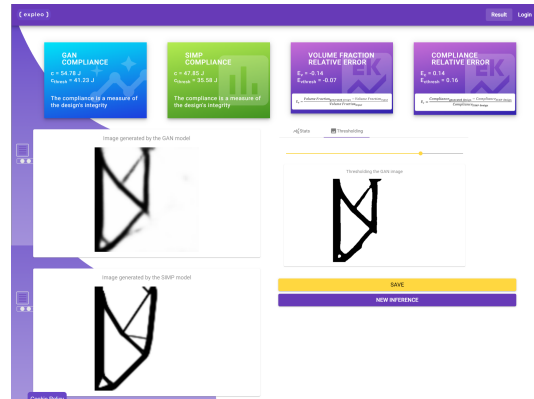
3.6. DISCUSSION



(a) Selection of the mechanical (BC are the red nodes in the design space, F is the loaded node with an angle of $^\circ$, the V is set to 0.2 i.e. 20%) and geometrical conditions (the Nbr_{bars} is set to 6).



(b) Results interface of GAIMD. It shows the generated design by DL-TO (top design) versus the generated design by SIMP. It computes the compliance values (in Joules) of both designs, the relative error of the volume fraction and of the compliance before and after the application of a pre-defined threshold value. Section stats summarizes the input conditions.



(c) Section threshold in the results interface of GAIMD. The user can apply a threshold on the generated design to obtain a crisper black-and-white design and eliminate intermediate density values.

Figure 3.21: Interface of GAIMD. GAIMD is an open-accessible web application that takes the mechanical conditions: the boundary conditions (BC), the loads (F), and the volume fraction (V) and the geometrical condition (Nbr_{bars}) and generates a 2D design using a DL-TO and optionally the SIMP method. Only edge nodes (blue nodes) from the design space can be selected to be the BC and F nodes.

and geometrical constraints concurrently without the need for explicit formulas for the geometrical constraint. Furthermore, DL-TO allows tailoring the design's geometry by modifying the input geometrical variable. It offers the designer an alternative way to explore designs faster and quickly adjust their geometry (here defined by the number of bars composing a design). Tuning the mechanical and geometrical conditions can be done effortlessly, and the result is obtained in a fraction of a second. On the other hand, DL-TO does not replace TO but compensates for its difficulties in integrating geometric constraints. Accordingly, it is advisable to proceed with a hybrid approach where both methods are used; the first draft is outputted by DL-TO and then optimized by some iterations of SIMP.

Finally, the geometrical condition considered (the Nbr_{bars}) is not a major manufacturing constraint, only a simple geometry-related constraint.

Thus, we will generalize over more concrete manufacturing constraints like overhangs, bar lengths, and bar thicknesses, which brings us to the creation of a dataset gathering the mechanical and geometrical constraints of designs (chapter 4).

3.7 French summary

Ce chapitre résume la première approche que nous proposons, DL-TO, une méthode TO basée sur le DL qui prend en entrée les contraintes mécaniques (conditions limites BC , configuration des charges F , et la fraction volumique V) et une contrainte géométrique (Nbr_{bars}) et génère un design 2D conforme à ces contraintes ; cette méthode est appelée DL-TO.

Par conséquent une approche GAN à deux discriminateurs est implémentée pour entraîner notre générateur DL-TO. La procédure de formation peut être résumée comme suit.

La procédure d'entraînement peut être résumée comme suit (Fig.3.1).

À chaque itération de l'apprentissage, le générateur (de type Res-U-Net [6]) prend en entrée les contraintes mécaniques et géométriques et produit des designs 2D. Ensuite, le discriminateur traditionnel est entraîné en deux étapes. Dans un premier temps, il est entraîné avec les designs réels et toutes les contraintes, et dans un deuxième temps, avec les designs générés et les contraintes. Le discriminateur de compteur de barres est entraîné uniquement avec les designs réels et les contraintes mécaniques. À ce niveau, les discriminateurs sont considérés comme optimaux et sont utilisés pour évaluer les designs générés : les scores produits par les deux discriminateurs (les pertes adverses et de comptage) ainsi que la perte de reconstruction (c'est l'erreur quadratique entre les designs réels et ceux générés par DL-TO) sont réinjectés dans le générateur pour mettre à jour ses poids. La base de données d'entraînement est issue d'un code SIMP (une méthode d'éléments finis de TO) qui prend en entrée les contraintes mécaniques (BC, F, V) et fournit en sortie la géométrie 2D ainsi que l'énergie de déformation ou compliance C en *Joules* et d'une labélisation manuelle de ces designs pour ajouter la contrainte de Nbr_{bars} . Cette base a été ensuite augmentée en raison de performance de DL-TO. En effet, nous avons utilisé la base de 4000 designs labélisés manuellement pour apprendre un modèle de compteur de barres basé sur le DL (qu'on a utilisé après comme discriminateur géométrique pour l'entraînement de DL-TO) et on a prédit sur les designs sortis par SIMP pour augmenter notre base de designs labélisés.

L'entraînement de DL-TO terminé, nous avons testé sa performance sur une base de test qui n'a pas été vu par le modèle pendant son entraînement. Les designs générés par DL-TO sont presque indiscernables à l'œil nu de ceux générés par SIMP, ils respectent les contraintes mécaniques et géométriques avec une marge de ± 2 barres et la génération était 3500 fois plus rapide que la méthode d'éléments finis SIMP. Mais, ils souffrent d'une compliance plus élevée que celle enregistrée avec SIMP (Figures

3.7, 3.8, 3.9, 3.11). Par conséquent, pour remédier à cet inconvénient, nous avons introduit un 3^{ème} discriminateur à la chaîne des discriminateurs de la structure GAN pour pénaliser DL-TO quand les designs générés montrent une compliance supérieure aux designs réels. En effet, la compliance est issue d'un calcul d'éléments finis itératif et coûteux pour des raisons de parallélisation de calculs, par suite, l'introduire dans notre pipeline d'entraînement ralentira le process. Donc, nous avons opté à entraîner un modèle DL qui va prendre en entrée un design 2D et prédire sa compliance. Après avoir entraîné DL-TO avec trois discriminateurs, nous avons testé DL-TO avec le test set. Nous avons amélioré 21.3% la compliance des designs générés en conservant la performance du modèle sur les autres aspects mécaniques et géométriques (Tab.3.4). Afin de valider l'apprentissage de la notion de géométrie par DL-TO tout en respectant les contraintes mécaniques, nous avons réalisé une expérience où nous figeons les contraintes mécaniques (BC, F, V) et nous changeons la contrainte géométrique (Nbr_{bars}). De cette expérience, nous avons conclu une corrélation entre la contrainte géométrique (Nbr_{bars}) et la contrainte de fraction volumique (V) et nous avons validés la capacité de DL-TO d'ajuster la géométrie d'un design afin de respecter la nouvelle valeur de la contrainte géométrique imposée tout en être toujours conforme avec les contraintes de bords et les forces imposés (Figure 3.15).

De plus, nous avons examiné la capacité de généralisation de DL-TO pour prendre en considération des contraintes jamais vues lors de l'entraînement ; des contraintes spatiales imposées aux limites du domaine d'espace. La limite du domaine d'espace est définie par la matrice de fraction de volume. Un exemple de limite du domaine d'espace : le design optimal final doit tenir compte du passage d'un tuyau, le tuyau est appelé un élément passif. Un autre exemple : le design doit tenir compte d'un pilier ; le pilier est appelé un élément actif. Dans ce cas, les valeurs correspondant à l'emplacement du tuyau dans la matrice de fraction de volume sont mises à zéro tandis que les autres restent intactes. La fraction volumique d'entrée étant formulée sous la forme d'une matrice de $((n_x+1) \times (n_y+1))$ éléments pour un design de dimension n_x, n_y pouvons toujours modifier cette matrice et forcer l'existence/absence de matière à des endroits particuliers de l'espace de design en augmentant/diminuant les valeurs de fraction volumique à ces endroits. Dans tous les cas testés, DL-TO a rempli de/vidé la matière dans les endroits où un surplus/absence de matériau était imposé et a remodelé la forme interne en forme de tronc de manière créative afin de maintenir l'intégrité du design. De plus, le design généré préserve la forme extérieure du design original (c'est-à-dire qu'il respecte les conditions aux limites et les configurations de charge d'entrée) (Figure 3.18). Il est important de noter que Malvia et Manoj [55] ont

aussi généré des designs avec des éléments actifs et passifs mais leur modèle était entraîné à le faire, le nôtre ne l'est pas.

Enfin, DL-TO a été encapsulée dans une API web appelé GAIMD (Generate AI based Mechanical designs). Il s'agit d'une application interactive où DL-TO et SIMP sont tous deux implémentées. L'utilisateur entre les contraintes mécaniques et le Nbr_{bars} et l'application génère deux designs, l'un à partir de DL-TO (où Nbr_{bars} est pris en compte) et l'autre à partir de SIMP. Elle présente les deux designs ainsi que leur évaluation, la compliance et la fraction volumique (Fig. 3.20).

Pour récapituler, la première approche, DL-TO, valide les hypothèses suivantes. DL-TO est capable de générer des designs de bonne qualité mécanique et géométrique 4 ; section 3.4.4. DL-TO a démontré sa capacité à adapter la géométrie globale d'un design tout en respectant les contraintes mécaniques 5 ; section 3.4.5. Enfin, ses performances de génération vont au-delà de l'ensemble de données d'entraînement ; elle génère des designs avec des éléments passifs et actifs sans être entraînée à le faire 6 ; section 3.4.6. En conclusion, DL peut compenser les difficultés rencontrées par TO en traitant simultanément les contraintes mécaniques et géométriques sans avoir besoin de formules explicites pour la contrainte géométrique. De plus, DL-TO permet de personnaliser la géométrie du design en modifiant la variable géométrique d'entrée. Il offre au concepteur un autre moyen d'explorer plus rapidement les designs et d'ajuster rapidement leur géométrie (définie ici par le nombre de barres composant un design). Le réglage des conditions mécaniques et géométriques peut se faire sans effort, et le résultat est obtenu en une fraction de seconde. En revanche, DL-TO ne remplace pas TO mais elle compense ses difficultés quant à l'intégration des contraintes géométriques. La meilleure méthode sera une forme hybride où les deux méthodes sont utilisées ; la première ébauche sortit par DL-TO et après optimiser par quelques itérations de SIMP. Néanmoins, la condition géométrique considérée (le Nbr_{bars}) n'est pas une contrainte de fabrication majeure, mais une simple contrainte liée à la géométrie.

Ainsi, nous allons essayer de généraliser DL-TO afin de l'adapter pour des contraintes de fabrication plus concrètes comme la contrainte de surplomb, les longueurs et épaisseurs de barres, ce qui nous amène à la création d'un jeu de données rassemblant les contraintes mécaniques et géométriques des designs (chapitre 4).

3.7. FRENCH SUMMARY

Chapter 4

GMCAD: Geometric and Mechanical CAD dataset

Content

4.1	Motivation	84
4.2	Workflow	85
4.2.1	Generation of the SIMP dataset <i>DB1</i>	86
4.2.2	Mechanical Conditions Prediction	87
4.2.3	Generation of the synthetic geometric dataset <i>DB2</i>	89
4.2.4	The Consolidation of the target dataset GMCAD	90
4.3	Results	90
4.3.1	Evaluation of the Mechanical Conditions Predictions	90
4.3.2	GMCAD: dataset of designs with their geometrical & predicted-mechanical constraints	92
4.4	Discussion	93
4.5	French summary	96

4.1 Motivation

In chapter 3, DL-TO proved the capacity of DL to tailor geometrically and mechanically a design at the same level. While the Nbr_{bars} is a geometrical condition that can be hardly formulated and integrated into FE-TO methods, it still is not the most adequate manufacturing constraint. Therefore, in this chapter, we will augment our training dataset to include the geometrical manufacturing-related constraints listed in section 2.2 in chapter 2 in order to accomplish our goal and train DL-AM-TO, a generative TO method that integrates at the same conceptual level mechanical and manufacturing constraints.

This chapter is adapted from our article GMCAD: an original Synthetic Dataset of 2D Designs along their Geometrical and Mechanical Conditions [5].

Nevertheless, creating a DL-AM-TO needs a rich training dataset with a wide variety of pairs of these designs alongside their constraints. Creating this dataset is challenging, especially since no FE-based TO method in the literature handles several geometric conditions at once. Thus, to achieve our goal, we propose to resolve the problem inversely (detailed in section 4.2, Fig. 4.1).

First of all, we generate designs from given mechanical constraints (boundary conditions, loads configuration, and volume fraction) using a modified version of the open-source code of Solide Isotropic Material with Penalization (SIMP) written by Sigmond [112]. Second, a DL model that maps designs to their corresponding mechanical conditions, using the SIMP designs, is built; the model is referred to as DL-Mechanical-Conditions-Predictor. Third, synthetic designs (inspired by the shapes of SIMP designs) with various geometric constraints are generated using pygmsh ¹. Fourth, the previously learned DL-Mechanical-Conditions-Predictor is used to predict their mechanical conditions. Finally, the target dataset “GMCAD” is consolidated; it consists of pairs of designs and their mechanical and geometric constraints. Consequently, we can train DL-AM-TO, which integrates mechanical and AM-geometric constraints simultaneously at the conceptual level and allows the user to tailor its input constraints easily and generate designs instantly. It is worth mentioning that a controllable design generation in the AM field, combining the mechanical and geometrical natures exhaustively, is recently been further explored [118]. In fact, with an adapted synthetic dataset as GMCAD and DL techniques, DL-driven generative design approaches offering to manage several parameters, and constraints jointly will motivate the implementation of lighter and faster modules in CAD software.

¹Pygmsh is a python library to draw shapes in FreeCAD, <https://pygmsh.readthedocs.io/en/latest/>

The rest of the chapter is organized as follows. Section 4.2 details GMCAD’s conception workflow. Section 3.3 evaluates the DL-Mechanical-Conditions-Predictors, presents and evaluates a sample of the resulting target dataset “GMCAD”. Section 4.4 summarizes the work and discusses the future development of the method.

4.2 Workflow

As previously mentioned, GMCAD aspires to bridge the mechanics to the geometry at the design level. TO generates layouts given mechanical constraints as inputs. However, it is rather complex for it to handle geometric constraints. Therefore, we choose to resolve the problem inversely. We can always learn to inverse TO’s job via DL, i.e., create a DL-Mechanical-Conditions-Predictor that takes a design as input and predicts its mechanical conditions. Then, we create designs (with layouts inspired by SIMP’s suggestions) accounting for the desired geometrical constraints and deduce their mechanical constraints using the former DL-Mechanical-Conditions-Predictor. Now that we dispose of a complete dataset with mechanical and geometrical constraints, we can go to the next step and train a DL model that generates designs accounting for the mechanical and geometrical aspects jointly.

The global work is divided into two steps. The first step consists of consolidating the target training dataset GMCAD, which makes the subject of the present article, in an inverse problem resolution way, from the geometry to the mechanics. The second step involves training the DL-AM-driven-TO model and is reported to future work.

The first step is partitioned into four stages: (1) The generation of *DB1*, a dataset of 2D designs from mechanical conditions using SIMP-TO. (2) The training of a DL-Mechanical-Conditions-Predictor to learn to map designs to their corresponding mechanical conditions. (3) The inception of *DB2*, a synthetic dataset of CADs (inspired from SIMP-designs’ shapes) from geometric conditions. (4) The completion of the synthetic dataset by predicting the mechanical conditions of the CADs. Consequently, the target dataset, GMCAD, is built. It is a synthetic dataset joining designs along with their geometric and mechanical conditions. GMCAD will be used in the future to train the DL-AM-driven-TO model.

4.2. WORKFLOW

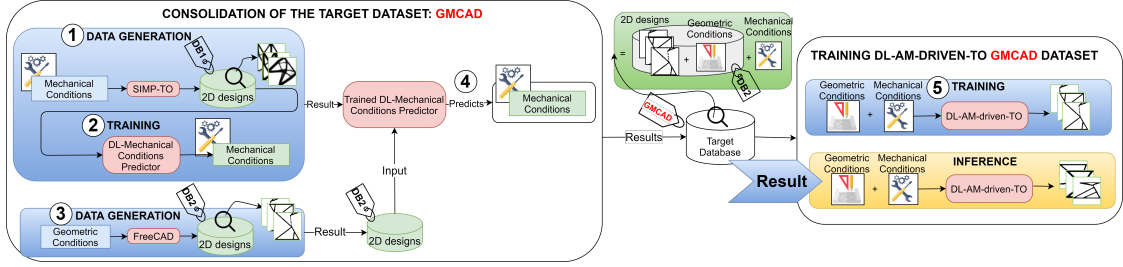


Figure 4.1: The workflow is divided into two global steps. The first step consists of consolidating the target training dataset GMCAD. The second step involves training the DL-AM-TO, a DL model that generates designs from input mechanical and geometrical constraints.

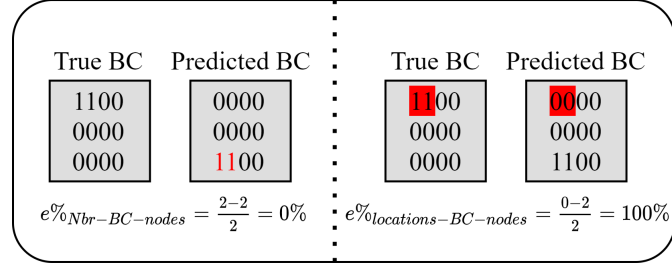


Figure 4.2: Example of the evaluation metrics computation of BC-DL-Predictor. This figure compares a predicted to a true BC matrix. The total number of predicted BC nodes is correct, while their locations are erroneous. This error is detected by the metric $e\%_{locations-BC-nodes}$ which counts the number of BC nodes in the correct locations, here none.

4.2.1 Generation of the SIMP dataset *DB1*

DB1 comes from the mechanical SIMP-TO code and consists of generating 2D designs (as images) from various mechanical conditions (boundary conditions, loads configuration, and volume fraction). For the present work, only edge-like boundary conditions and edge-like punctual loads were considered; i.e., boundary conditions and loads were chosen along the circumference of the design space. Additionally, the maximum number of loads is one. A sample of SIMP designs alongside their mechanical constraints are presented in Fig. 4.6.

It is important to emphasize that using SIMP-TO, the output layout is independent of the load intensity, and the load orientations are of modulo 180° , i.e., a design subject to a load of 90° is similar to another subject to a load of 270° .

4.2. WORKFLOW

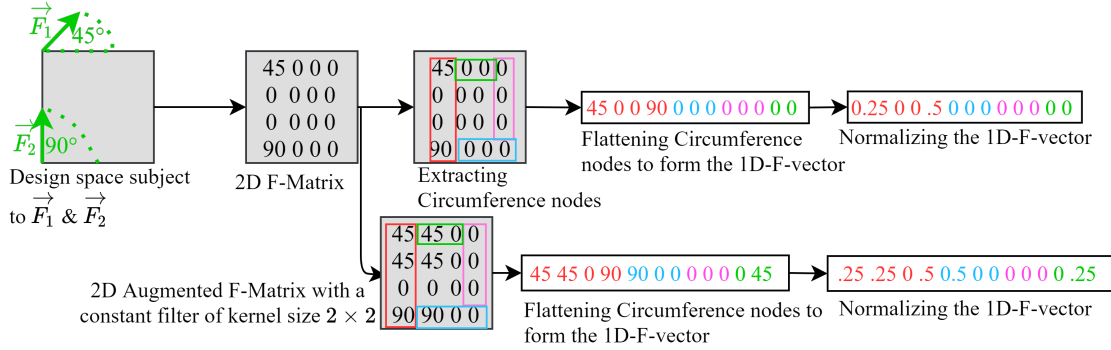


Figure 4.3: In this figure, the design space (3×3) is subjected to 2 loads: \vec{F}_1 on the top-left edge with $\theta = 45^\circ$ and \vec{F}_2 on the bottom left edge with $\theta = 90^\circ$. The 2D load matrix (4×4) consists of null-values everywhere except for the loaded nodes set to their orientations. Next, since we only deal with edge-like loads in this work, we reduce the load matrix to the circumference nodes. Finally, the 1D-load vector is normalized (a division by 180). To facilitate the angles prediction, we augment the load matrices by a constant filter of kernel size 5×5 , but for presentation purposes, the filter size shown in the figure is 2×2 .

4.2.2 Mechanical Conditions Prediction

The mechanical conditions that were considered in this work are the volume fraction V , the boundary conditions BC , and the loads' configurations F . The volume fraction constraint is the average density values of a design, i.e., the average pixel values in the image-like design. The boundary conditions (BC) and loads (F) are 2D matrices of size $(n_x + 1, n_y + 1)$ for a 2D design of size (n_x, n_y) . BC matrices are 2D matrices with null values everywhere except for the fixed nodes, which are set to 1.0 (Fig. 4.2 shows an example of a BC matrix of size 4×4). F matrices are 2D matrices with null values everywhere except for the loaded nodes, which are set to θ , where θ is the orientation of the load. Thus, we develop a convolutional DL model that takes 2D designs and reconstructs the BC and F matrices. Instead of predicting BC and F in a single shot, we create a model per constraint; the BC -DL-Predictor and the F -DL-Predictor. The SIMP-based dataset $DB1$ was split into a train (4538 samples) and a test (1140 samples) set to train the DL models. In this work, n_x and n_y are set to 100, $\theta \in [0^\circ, 180^\circ]$ and $|F| = 1N$.

4.2.2.1 Deep Learning based boundary conditions predictor

The BC -DL-Predictor's architecture was inspired by the convolutional Res-U-Net architecture [6]. The network is constituted of an encoder, a bridge, and a decoder. The encoder is formed of 4 blocks,

4.2. WORKFLOW

each consisting of a down-sampling layer (a convolution of stride 2) and a residual unit ². The decoder comprises five blocks, each consisting of an up-sample layer (a transpose convolution of stride 2) and a residual unit followed by a convolution of kernel size 1×1 and a sigmoid activation. The bridge connection has the same architecture as an encoder block and combines the encoder with the decoder. BC-DL-Predictor takes as input the 2D design and outputs a two-channel matrix corresponding to the BC along the x- and y-axis, BC_x , and BC_y , respectively. The model was trained with a learning rate of 0.0002, an Adam optimizer, a batch size of 64, and the mean squared error as a loss function.

4.2.2.2 Deep Learning based force Predictor

Predicting the loads' matrices is a very intricate task. In the training set, load matrices are sparse, where non-zero values can range from 5 to 180° (loads orientations). Thus, to alleviate the sparsity and wide range of values in the loads' matrices and knowing that we are only dealing with edge loads, we reduce the 101×101 sparse loads matrices to a 1D-normalized-vector consisting of the circumference nodes, i.e., of dimension 1×400 with values range from 0 to 1 (Fig. 4.3). We note that the minimum load orientation is 5° to avoid confusion with the null values representing the absence of load. Predicting the load location and orientation precisely consisted of two complementary models. The first is the F-DL-Locator model; it predicts the loads' locations with high precision. The second, the F-DL-Angle-Estimator model, predicts an augmented version of the load vector. The final load vector is the product of the outputs of the former models (*Load vector = Location vector \times Augmented Orientation vector*). F-DL-Locator and F-DL-Angle-Estimator share the same architecture, which consists of seven down-sampling layers, each followed by a residual unit [104]. It takes a three-channel input (the 2D design, BC_x , and BC_y) and outputs a 1D vector of dimension 1×400 . Adding the BC as input to the load predictors helps enhance the models' precision for a loaded node that is unlikely a fixed one (i.e., a BC node). F-DL-Angle-Estimator was trained with a learning rate of 0.002, an Adam optimizer, a batch size of 64, and the mean squared error as a loss function, and F-DL-Locator with a learning rate of 0.0002, an Adam optimizer, a batch size of 64, and the $L1$ as a loss function.

²A residual unit block is a sequence of two blocks, each consisting of a batch normalization followed by a ReLU activation and a convolution of kernel size = 3×3 and stride 1. Its input is summed with its output via an identity mapping connection. An identity mapping connection consists of a convolution of kernel size = 1×1 , a stride of 1, and padding of 0 followed by a batch normalization layer [6].

4.2. WORKFLOW

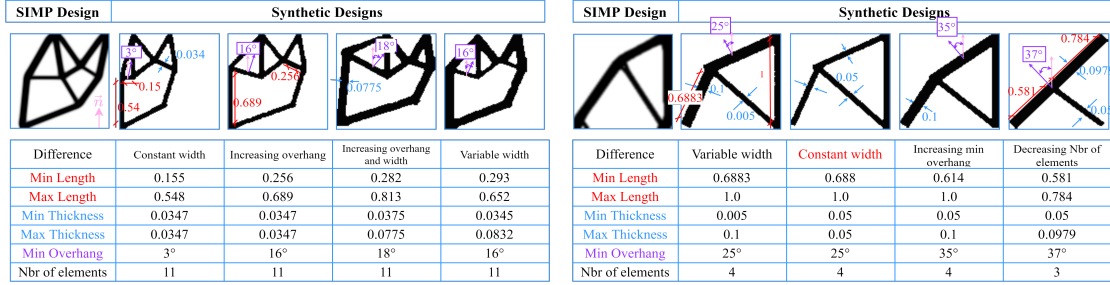


Figure 4.4: A sample of the synthetic dataset $DB2$, inspired from SIMP layouts, with various geometric constraints: minimum/maximum element-length, minimum/maximum element-thickness, minimum overhang, and the number of elements. The minimum overhang is computed with respect to the build direction \vec{n} . Here, \vec{n} is along the y-axis. NB: the overhang is the angle between the normal vector of an element (purple arrow) and the build direction \vec{n} (pink arrow).

4.2.3 Generation of the synthetic geometric dataset $DB2$

The synthetic 2D CAD dataset ($DB2$) is built using pygmsh (a python library, FreeCAD); the CAD shapes are inspired by SIMP-TO’s designs. A synthetic design is defined as a connection of beams where we know the four coordinates of every design’s beam. This geometry definition allows us to extract all the geometric information we need from the length, width, overhang (corresponding to a chosen build direction), Etc., and consequently obtain a dataset of a wide variety of geometric constraints. Next, the CADs are converted to 2D images. The conversion from CADs to images consists of reading the CAD as a cloud of points, then applying a convolutional filter followed by a bilateral filter to smooth the image without compromising the design’s beam thicknesses. $DB2$ was generated by varying geometric element-wise (lengths and thicknesses) and inter-element (angles between elements, number of elements) constraints. $DB2$ contains all the beams’ coordinates, lengths, thicknesses, and angles between connected consecutive beams. However, we present here only the geometric-AM-related measures that we will explore (the maximum and minimum element length, the maximum and minimum element thickness, the minimum overhang, and the number of elements) with the corresponding designs (Fig. 4.4). We would like to note that the length and thickness measures are computed with respect to the design space’s dimensions (n_x, n_y) (here, $n_x == n_y$, i.e. a length of 1.0 is equivalent to $1.0 \times n_x$).

4.2.4 The Consolidation of the target dataset GMCAD

At this stage, we dispose of *DB2*, a geometric dataset, a BC-DL-Predictor, and an F-DL-Predictor. Consequently, we predict the mechanical constraints of *DB2*'s designs and end up with GMCAD, a dataset with designs along with their mechanical and geometrical constraints. This step consists of first predicting the BC of the designs and then their loads' locations and orientations; since the loads-predictor also needs the BC input. GMCAD's mechanical and geometric-AM-related variables distributions are shown in Fig. 4.7.

4.3 Results

4.3.1 Evaluation of the Mechanical Conditions Predictions

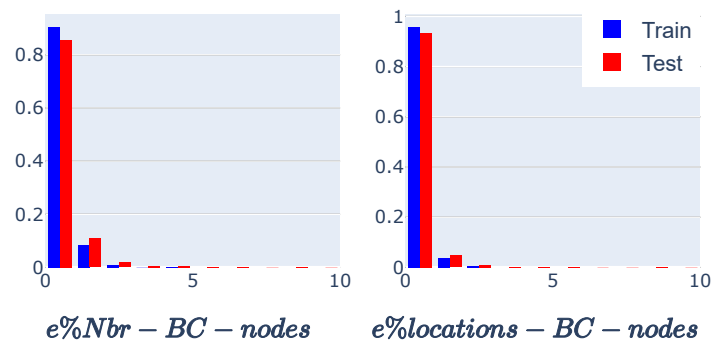
To evaluate the BC-DL-Predictor, we choose two metrics: $e\%_{Nbr-BC-nodes}$ and $e\%_{locations-BC-nodes}$ that calculates the error over the number and location of predicted BC nodes, respectively; the errors vary from 100% (erroneous model) to 0% (accurate model). However, we want to emphasize that $e\%_{Nbr-BC-nodes}$ is not sufficient to judge the BC-DL-Predictor's performance, for, in our case, this metric could be deceiving; the model predicts BC nodes in the wrong locations (an example of such case is presented in Fig. 4.2). Thus, we compute $e\%_{locations-BC-nodes}$, which is the difference in the number of BC nodes between the true and predicted BC matrices in the right locations. This metric ensures that the model well recognizes the BC nodes' locations.

The results of the former metrics are presented in Figure 4.5(a). As we can see, the precision (100%-error) of the BC-DL-Predictor is very high; 83.5% of the test predictions are detected in the exact locations, and 98% of them are within an error margin of 2%.

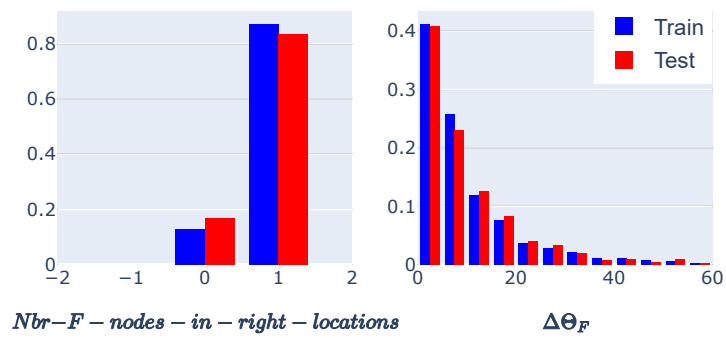
To evaluate the loads predictions, two metrics were considered: $Nbr_{F-nodes-in-right-locations}$ and $\Delta\theta_F$. $Nbr_{F-nodes-in-right-locations}$ computes the number of loads predicted in the exact location; it measures the precision of the F-DL-locator model. $\Delta\theta_F$ computes the difference between the true and predicted loads orientations in the right loads' locations; the results are illustrated in Fig. 4.5(b). In the test set, the precision of the F-DL-locator is 83.3%, and only 25% of the angle predictions differ from the true angle by $\pm 20^\circ$.

Although predicting the loads' angles is intricate, the first results are encouraging. We want to point out that improving the precision of the F-DL-Angle-Estimator is still a work in progress. According

4.3. RESULTS



(a) BC-DL-Predictor's performance.



(b) F-DL-Predictor's performance.

Figure 4.5: Mechanical Conditions Predictors Performance over the train and test set.

4.3. RESULTS

True BC and F										
Predicted BC and F										
$\Delta_{Nbr-BC-nodes}$	-6	-6	-5	0	0	-5	10	0	0	0
$\Delta_{Nbr-F-nodes}$	0	0	0	0	0	0	0	0	0	0
ΔF_{θ}	-91°	4°	-72°	5°	-31°	14°	-2°	-9°	4°	16°

Figure 4.6: Comparison between the true and predicted boundary conditions (BC in red) and loads’ locations and orientations (F in green). The BC and F locations are predicted with high precision. Most F orientations predictions differ from the true values within $[-20^{\circ}, +20^{\circ}]$. NB: The F orientations are in the anti-clockwise direction.

to our knowledge, we are the first to predict the mechanical constraints from designs via DL models to consolidate a complete dataset of designs with geometric and mechanical conditions altogether.

A sample of designs with their predicted versus true loads and boundary conditions is shown in Fig. 4.6. As we can see, the BC predictions and F locations are very accurate. Also, most load orientation predictions deviate from the true values within a range of $[-20^{\circ}, +20^{\circ}]$.

4.3.2 GMCAD: dataset of designs with their geometrical & predicted-mechanical constraints

GMCAD’s variables distributions and a sample of the synthetic designs along their predicted mechanical conditions and performance are shown in figures 4.7 and 4.8, respectively. GMCAD contains 36181 distinct combinations of the geometric (Min/Max length/thickness, Min Overhang, Nbr of elements) and mechanical (F orientations and BC nodes) variables. From Fig. 4.7, we can see that GMCAD comprises a wide variety of designs with 3 to 12 elements, Min bar length $\in [0.07, 1.0]$, Max bar thickness $\in [0.003, 0.4]$, Min overhang $\in [0^{\circ}, 65^{\circ}]$, etc. In the future, we could complement GMCAD with designs featuring more than 12 bars, a Max bar thickness ≥ 0.4 , and a Min bar length ≥ 1.0 . The data can be found in GMCAD’s Git repository.

From figure 4.8, we can notice that changing one geometric condition can trigger a difference in the load orientations predictions and its performance (Compliance in Joules J). In figure 4.8(a), the first design’s performance twice improved by simply adding an element (2^{nd} design, 12 elements instead of 11). It was further improved when we increased an angle between two bars (the minimum overhang increased) and the thickness of the edge bars; the compliance dropped 16 times (from 412 to $25J$, 5^{th} design). However, increasing the thickness of all bars seemed to deteriorate the functionality of the design (the 3^{rd} design’s compliance is $1.5e9J$). The same behavior is detected in Fig 4.8(b). Con-

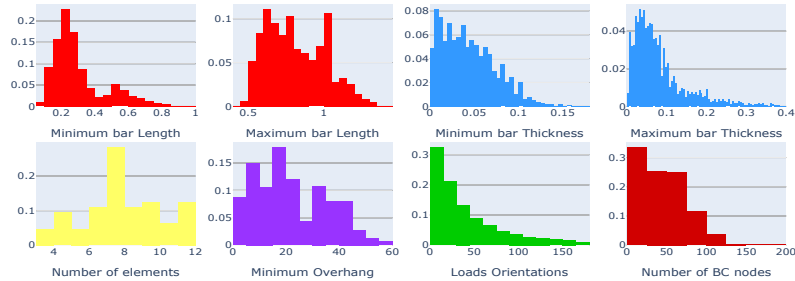


Figure 4.7: GMCAD’s Descriptive Statistics. This figure shows the distributions of the geometrical (Min/Max Length/Thickness, Number of elements, Min Overhang) and mechanical variables (Number of BC nodes and F orientations) of GMCAD.

sequently, the mechanical performance of a design is sensitive to the very minor geometrical change, and achieving the optimal design complying with all constraints takes several geometric modifications. To further understand the influence of geometric constraints on mechanical performance, we generate 16 four-element designs by varying a constraint at a time and computing their compliance (Fig.4.9). We can observe that the compliance decreases gradually with the non-constant thickness. However, it fluctuates with the minimum overhang and minimum constant thickness. Thus, there is no apparent relation between constant thickness/minimum overhang and the mechanical performance that can be deduced or generalized.

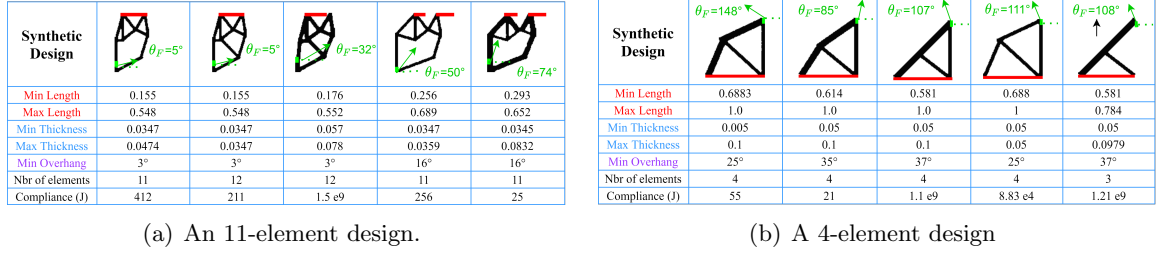
To sum up, several changes in geometric constraints can improve mechanical performance. However, the combinations of constraints are uncountable, and the inter-correlations or dependencies between them are unknown. Moreover, the influence of geometric constraints over mechanical performance cannot be generalized, which encourages the establishment of this dataset and its usage to train a DL model that can jointly capture these correlations and generate designs complying with geometric and mechanical conditions.

4.4 Discussion

GMCAD, a synthetic dataset of 2D designs alongside their mechanical and geometrical conditions, is consolidated.

First, we generate mechanical designs using the FE-density-based-TO method (SIMP) and train DL

4.4. DISCUSSION



(a) An 11-element design.

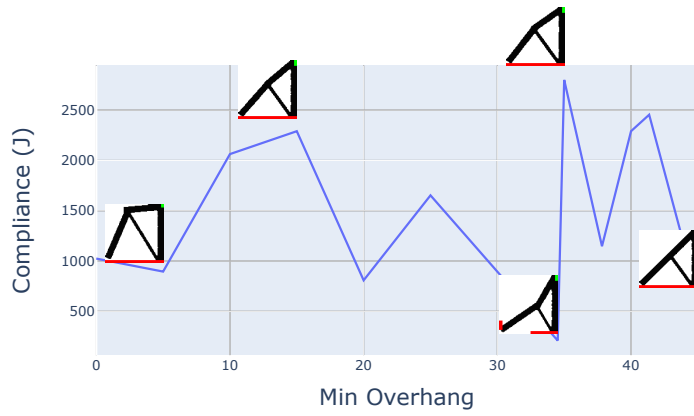
(b) A 4-element design

Figure 4.8: Synthetic Designs with mechanical and geometrical constraints. In this figure, we show a sample of synthetic designs alongside their predicted mechanical conditions. We also evaluate their mechanical performance (Compliance in Joules).

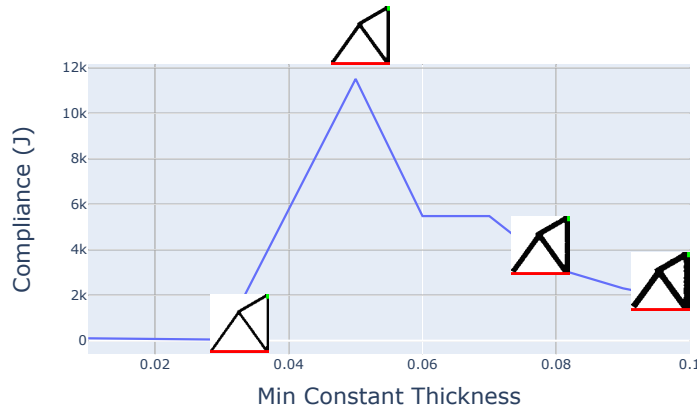
models to predict the mechanical conditions. Then, we build synthetic CADs inspired by the SIMP designs’ layouts accounting for various complex geometric constraints, which we complete by predicting the mechanical conditions using the learned DL-Mechanical-Conditions predictors. Finally, we evaluate the synthetic designs’ mechanical performance. We showed that even slight geometric changes could deteriorate the design’s functionality, and finding the best compromise between mechanical performance and geometry can be very challenging because of the unlimited combinations of geometric modifications and their unknown inter-correlations. Thus, it would be interesting to explore DL in this area for its robustness in capturing spatial correlations and hence create a model that generates designs accounting for mechanical and geometrical constraints concurrently at the conceptual level, DL-AM-TO, which is the next step (Step 5 in the workflow detailed in Fig.4.1).

Here, we proposed an innovative, unconventional approach to facilitate the integration of complex unformulated geometric AM constraints at the design level without needing the FE expensive iterative computations. Instead of formulating analytically complex geometric AM constraints, it suggests an inverse problem resolution. Designs conforming with these spatial-related constraints are first created, then DL models, particularly CNNs, are trained to learn them. The advantage of this approach is that it can be generalized to incorporate any constraints, even descriptive ones because we do not need to find a formula for the constraint but simply a sufficient number of examples describing it. Additionally, the usage of this dataset can be extended beyond AM, such as the reverse engineering field, CAD automatic reconstruction, and CAD to CAE. This dataset can also drive DL-multi-objective controllable design generation techniques that can be implemented into CAD software as a lighter and faster generative design module.

Now, with GMCAD ready, we could train DL-AM-TO, a DL based AM driven TO method that



(a) Compliance versus Overhang.



(b) Compliance versus minimum constant thickness.



(c) Compliance versus minimum variable thickness.

Figure 4.9: Mechanical Evaluation of synthetic designs. In this figure, we vary one geometric constraint and evaluate the mechanical performance (Compliance in Joules) of the resultant design. The geometric constraints are the minimum Overhang, the minimum constant bar-thickness (i.e., all elements have the same thickness), and the minimum variable bar-thickness (here, the contour bars have a thickness twice the inner bar).

integrates geometrical and mechanical constraints concurrently (Chapter 5).

4.5 French summary

Ce chapitre se concentre sur la création de la base de données qui va entraîner DL-AM-TO. Nous nommons cette base GMCAD pour Geometric and Mechanical CAD dataset. Elle contient des CAO 2D en format *.vtk* et en format d'images *.png* ainsi que les contraintes mécaniques et géométriques enregistrées dans des fichiers Excel.

Comme nous avons mentionné en chapitre 2, la création d'une base avec les deux natures de contraintes pose un grand défi. D'autant plus qu'aucune méthode TO basée sur les éléments finis dans la littérature ne traite plusieurs conditions géométriques à la fois. Ainsi, pour atteindre notre objectif, nous proposons de résoudre le problème de manière inverse (Fig. 4.1).

Tout d'abord, nous générons des designs à partir de contraintes mécaniques données (conditions aux limites, configuration des forces et fraction volumique) en utilisant la librairie https://github.com/dbetteb/TOP_OPT orientée objet et open-source, de Solid Isotropic Material with Penalization (SIMP) que nous avons développé en se basant sur [112]. Cette base comprenant les designs en format images et les contraintes mécaniques est appelée DB1.

Deuxièmement, deux modèles Deep Learning (DL) qui font correspondre les designs à leurs conditions aux bords et forces correspondantes, en utilisant la base *DB1*, sont entraînés ; les modèles sont appelés DL-BC-Predictor et DL-F-Predictor, respectivement.

L'architecture du BC-DL-Predictor s'inspire de l'architecture convolutive Res-U-Net [6]. Le BC-DL-Predictor prend en entrée la conception 2D et sort une matrice à deux canaux correspondant aux BC le long des axes x et y , BC_x et BC_y respectivement ; les BC sont représentés comme des matrices 2D avec des zéros partout sauf pour les nœuds bridés, ils sont fixés à 1.0.

La prédiction des matrices de forces est une tâche très complexe. Dans l'ensemble d'apprentissage, les matrices de forces sont des matrices creuses où les valeurs non nulles, correspondant aux nœuds chargés, peuvent varier de 5° à 180° (orientations des charges). Ainsi, pour remédier à ce problème et sachant que nous ne traitons que les forces sont appliquées aux bords de l'espace de design, nous réduisons les matrices de forces creuses à un vecteur normalisé 1D constitué des nœuds de circonférence, c'est-à-dire une dimension réduite avec des valeurs comprises entre 0 et 1 (Fig. 4.3). Nous notons que l'orientation minimale de la force est de 5° pour éviter toute confusion avec les valeurs nulles représentant l'absence de

4.5. FRENCH SUMMARY

force. La prédiction précise de l'emplacement et de l'orientation de la charge a nécessité la séparation de la tâche en deux modèles complémentaires. Le premier est le modèle F-DL-Locator. Il prédit avec une grande précision les emplacements des charges représentées par des matrices binaires 2D avec des zéros correspondant à l'absence de charges et 1.0 à leur présence. Le second modèle F-DL-Angle-Estimator prédit une version augmentée du vecteur de charge. Le vecteur de charge final est le produit des sorties des deux modèles ($Load\ vector = Location\ vector \times Augmented\ vector$). Orientation vecteur). F-DL-Locator et F-DL-Angle-Estimator partagent la même architecture. Ils prennent tous deux en entrée trois matrices concaténées (le design 2D, BC_x , et BC_y). L'ajout du BC en entrée des prédicteurs de forces permet d'améliorer la précision des modèles parce qu'un nœud chargé est normalement non encastré (c'est-à-dire un nœud BC). Troisièmement, des designs synthétiques (inspirés des formes des designs SIMP de la base DB1) avec diverses contraintes géométriques (des épaisseurs, longueurs et angles différents) sont générés à l'aide de pygmsh ³ ; cette base comprenant des designs en format images et CAOs ainsi que leurs contraintes géométriques est appelée *DB2*.

Finalement, afin de joindre les deux informations, mécanique et géométrie dans une seule base, nous prédisons à l'aide de DL-BC-Predictor et DL-F-Predictor appris précédemment les conditions mécaniques sur les designs de la base *DB2*.

Par conséquent, le jeu de données cible "GMCAD" est consolidé ; il est composé de paires de designs et de leurs contraintes mécaniques et géométriques. Les données sont déposées dans le dépôt Git de GMCAD.

³Pygmsh est une bibliothèque python permettant de dessiner des formes dans FreeCAD, <https://pygmsh.readthedocs.io/en/latest/>

4.5. FRENCH SUMMARY

Chapter 5

The second approach: Deep Learning Additive Manufacturing driven topology optimization

Content

5.1	Deep Learning Additive Manufacturing driven Topology Optimization	101
5.2	Methodology	101
5.2.1	Training framework	102
5.3	First variant of DL-AM-TO's training	102
5.3.1	Training Dataset	102
5.3.2	Generator's architecture	103
5.3.3	Discriminators' architectures	103
5.3.4	Loss function	104
5.3.5	Results	106
5.3.6	Discussion	107
5.4	Improving the geometrical discriminator's performance	110
5.4.1	Training	111
5.4.2	training dataset	112
5.4.3	Results	112
5.4.4	Uncertainty quantification of the DL geometrical discriminators	120
5.5	Training a model per geometrical variable	124
5.5.1	Training with Θ_{min}	124
5.5.2	Training with len_{max}	126
5.5.3	Training with Nbr_{bars}	129
5.6	Second variant of DL-AM-TO	131
5.6.1	Results	131
5.6.2	DL-AM-TO's overall performance	131
5.6.3	Tailoring a design's geometry with DL-AM-TO	136

5.7	Printing designs generated by DL-AM-TO	138
5.7.1	Printing geometries of different Nbr_{bars} constraint	143
5.7.2	Printing geometries of different len_{max} constraint	144
5.7.3	Printing geometries of different Θ_{min} constraint	146
5.7.4	DL-AM-TO accelerates the DfAM process and is a lighter module in industrial design software	148
5.8	Discussion	151
5.9	French summary	155

5.1. DEEP LEARNING ADDITIVE MANUFACTURING DRIVEN TOPOLOGY OPTIMIZATION

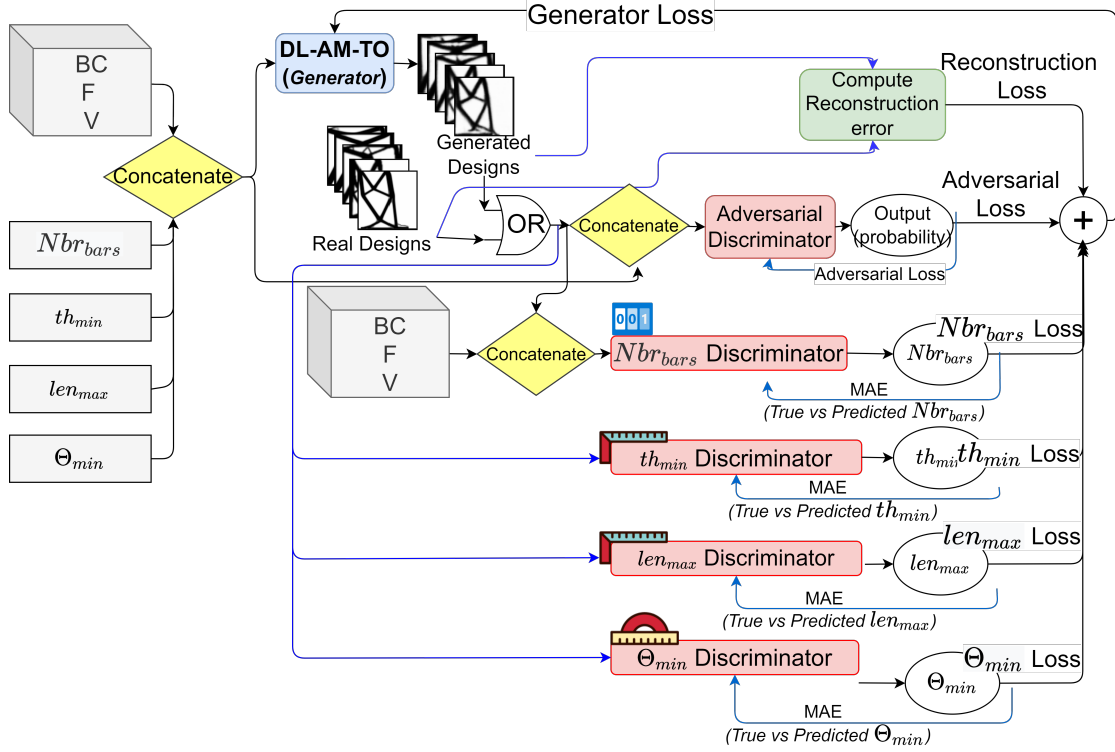


Figure 5.1: Training Procedure

5.1 Deep Learning Additive Manufacturing driven Topology Optimization

With the creation of GMCAD, DL-AM-TO detailed in section 2.1 can now be tested. Its goal is to compensate for the difficulties faced in FE-TO methods when it comes to integrating geometrical manufacturing-related and mechanical constraints at the same conceptual level.

5.2 Methodology

DL-AM-TO is a novel approach where the mechanical and geometrical constraints are no longer in competition. DL-AM-TO is a generative model that takes mechanical (Boundary conditions (BC), loads (F), and the volume fraction (V)) and geometrical conditions (the minimum thickness (th_{min}), the maximum length (len_{max}), the minimum overhang (Θ_{min}), the number of bars (Nbr_{bars})) as inputs and generates a 2D structure following these constraints.

5.2.1 Training framework

DL-AM-TO needs to learn to map the mechanical constraints to a general shape (abstract) that can be adjusted via the geometrical correlations learnt from the geometrical constraints. Moreover, geometrical constraints are themselves correlated between each other; changing one geometrical constraint triggers the changing of the other. We will also need to check the influence of a geometrical constraint with respect to the other (their inter correlation) and check the one that dominates the others. Henceforth, DL-AM-TO's generated designs need to be penalized on every geometrical constraint independently. DL-AM-TO's training could be thought of in two ways. The first variant consists of directly generating the design from all the constraints in a one shot learning. The second variant follows the hierarchical learning, i.e. we start with DL-TO then follow with another network to control the geometry. The preference for a particular variant is decided according to DL-AM-TO's performance (i.e., the generation quality).

Following the first approach detailed in chapter 3, we will first explore the first variant of DL-AM-TO.

5.3 First variant of DL-AM-TO's training

The first variant consists of implementing one model that maps directly the mechanical (Boundary conditions (BC), loads (F), and the volume fraction (V)) and geometrical (the minimum thickness (th_{min}), the maximum length (len_{max}), the minimum overhang (Θ_{min}), the number of bars (Nbr_{bars})) constraints into a design.

DL-AM-TO is trained within a five-discriminator-GAN [92] framework consisting of a generator (DL-AM-TO) and five discriminators: the traditional adversarial discriminator and four geometric discriminators, a bar counter, a th_{min} , len_{max} and Θ_{min} predictors (Fig.5.1).

5.3.1 Training Dataset

11719 samples of GMCAD are used for training and 4405 samples for test. It consists of 2D designs (in a .png format) alongside their mechanical and geometrical constraints. The distributions of the geometrical constraints values are shown in Fig. 5.2.

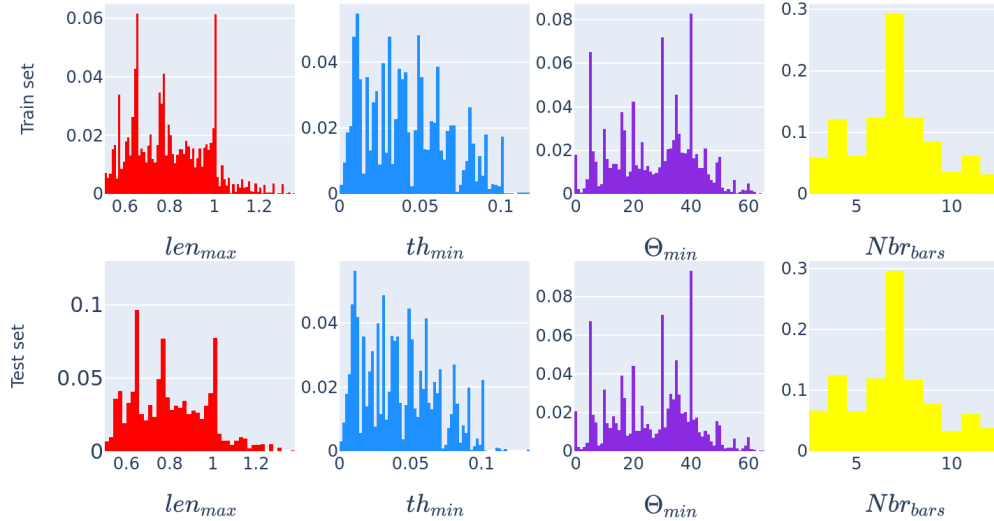


Figure 5.2: Distribution of the geometrical constraints values in the train and test sets

5.3.2 Generator’s architecture

The generator, DL-AM-TO, inherits the residual[104] convolutional encoder-decoder architecture[6] presented in section 3.1.1.1 with one difference; the skip-connections between the outputs of encoder layers and the inputs of decoder layers were eliminated here for reasons of performance. It seems that the skip-connections were making the generator memorize the constraints and not learn them; i.e. when we fix all constraints but one geometrical condition, the generator was outputting a noisy image, nothing that looked like a structure.

5.3.3 Discriminators’ architectures

The traditional discriminator consists of seven down-sample convolutional layers followed by a dropout and a final fully connected layer (like the one used in chapter 3). The Nbr_{bars} discriminator’s inception-v4 [85] regression network described and trained in chapter 3, section 3.1.1.2.2 was tested with GMCAD designs. Its input is the 2D design, the BC , the F , and the V , and its output is a scalar value depicting the predicted Nbr_{bars} present in the design. The median ΔNbr_{bars} is 2; in other words, the counter predicts on 50% of the designs a Nbr_{bars} higher

5.3. FIRST VARIANT OF DL-AM-TO'S TRAINING

than the ground truth by 2. The percentage of inadmissible predictions¹ was 35%. Thus, we retrained it for a few epochs with the training data samples, where it was poorly performing. As a result, the percentage of inadmissible predictions dropped to 0% and the median ΔNbr_{bars} is 0.

With the success of the Nbr_{bars} discriminator's architecture, we have opted for the same architecture for the new geometric (th_{min} , len_{max} , and Θ_{min}) discriminators with one difference. The stem block of the len_{max} and Θ_{min} predictors consisted of additional five residual layers. The input of the three geometric discriminators consists of the 2D design only, as there was no additional improvement in their prediction ability when adding the mechanical conditions, as was the case with the Nbr_{bars} discriminator.

To train the geometric discriminators, we augmented the training dataset with three rotations of 90°, 180°, and 270°. However, training DL-AM-TO only consisted of the training dataset with no rotation. The predictive performance of the geometrical discriminators is presented in Fig. 5.3. In order to evaluate a predictor, an admissible error interval is set (predictions within the green lines in Fig.5.3 are considered correct). As we can see, the th_{min} predictor shows the highest number of inadmissible predictions (predictions outside the green intervals). To quantify this observation, the percentage of erroneous predictions for every geometrical discriminator is computed. We choose for the th_{min} and len_{max} the relative prediction error defined as $e_{\%} = \frac{|True - Predicted|}{True} \times 100$, and for the Θ_{min} and Nbr_{bars} the difference $\Delta = |True - Predicted|$. In the test set, the percentage of predictions that fall within $e_{th_{min}\%} > 5\%$ is 46%, $e_{len_{max}\%} > 5\%$ is 1%, $\Delta\Theta_{min} > 5^\circ$ is 3.15% and $\Delta Nbr_{bars} > 1bar$ is 0.15% (Fig.5.3(c)). Consequently, we can conclude that all geometric discriminators are sufficiently precise except for the th_{min} one, which needs further improvement. In fact, if we tolerate a higher error interval of 10% for th_{min} , we would end up with 29.1% of inadmissible predictions.

5.3.4 Loss function

As we have detailed in chapter 3, section 3.1.1.3, the most challenging aspect of GANs is to find an equilibrium between the generator and the discriminator and avoid the dominance of one over the other. The loss function and other training parameters play an important role in stabilizing the training and condemning the phenomenon of oscillating losses. In this work, the loss function is further challenging; it has to also account for three additional geometrical (th_{min} , len_{max} , Θ_{min}) constraints.

¹An admissible prediction falls within an interval $|\Delta Nbr_{bars}| \leq 2$ as referred in chapter 3, section 3.3.2.

5.3. FIRST VARIANT OF DL-AM-TO'S TRAINING

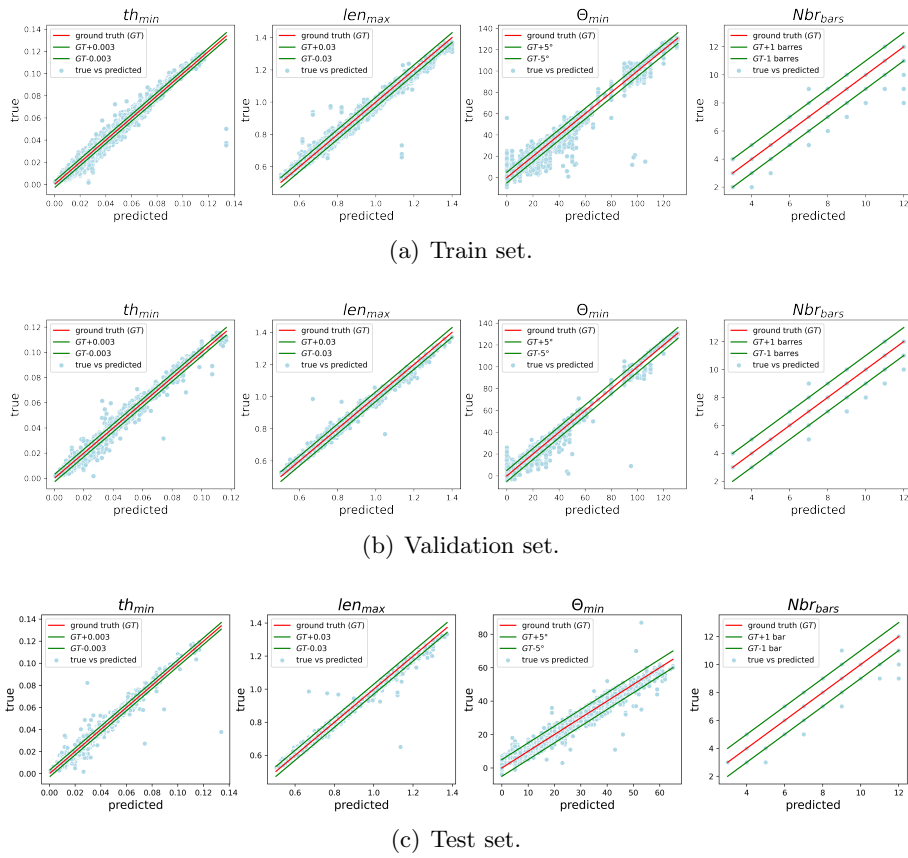


Figure 5.3: Performance of the geometric discriminators showing the predicted vs the true values of th_{min} , len_{max} , Θ_{min} , & Nbr_{bars} , from left to right, respectively, for the train, validation, and test sets. The worst performant discriminator is clearly the th_{min} predictor.

Thus, we adapted the loss function described in chapter 3, section 3.1.1.3 to account for these new constraints.

$L_G = \frac{1}{6}(L_r + \lambda_{adversarial}L_{adv} + L_{Nbr_{bars}} + L_{th_{min}} + L_{len_{max}} + L_{\Theta_{min}})$, with (1) the reconstruction loss $L_r = \frac{1}{N} \sum_{i=1}^N (x_i - \hat{x}_i)^2$ s.t. x_i and \hat{x}_i are the real and generated 2D design and N is the batch size, (2) $\{L_c = \sum_{i=1}^N |c_i - \hat{c}_i|, c \in \{Nbr_{bars}, th_{min}, len_{max}, \Theta_{min}\}\}$ s.t. c and \hat{c} are the input and predicted geometrical values respectively, and (3) the adversarial loss L_{adv} is the Binary Cross Entropy ($0 \leq L_{adv} \leq 100$ in PyTorch). Hence, $\lambda_{adversarial}$ was set to 0.01, so L_{adv} becomes of the same order of magnitude of all other losses varying between 0 and 1.

5.3.5 Results

Figure 5.4 illustrates the Structural Similarity Index Measure (SSIM)[119] of the generated designs. The blue distribution corresponds to the generator (DL-AM-TO) trained with all geometric discriminators. The average SSIM is 0.33, which demonstrates, aesthetically speaking, a weak generation. As we have mentioned previously, the th_{min} discriminator is not predicting th_{min} with high precision, and hence the generator is penalized with a less informative loss, which explains DL-AM-TO’s behavior. Furthermore, th_{min} is a continuous complex quantity to be treated by DL, particularly convolutional networks. We can find several designs with different minimum thicknesses that look indistinguishable in the dataset. Indeed, th_{min} is a texture feature, unlike Θ_{min} , an edge feature, where a slight variation can drastically modify the geometry and hence the pixels’ distribution. Additionally, the thickness information can be compensated with post-processing over the skeletons of the designs; we can erode/dilate skeletons to generate designs with arbitrary thicknesses.

In order to alleviate this setback, we re-train our model without the th_{min} variable. The generator’s loss becomes $L_G = \frac{1}{5}(L_r + \lambda_{adversarial}L_{adv} + L_{Nbr_{bars}} + L_{len_{max}} + L_{\Theta_{min}})$.

As expected, SSIM was improved by 44% as illustrated by the orange distribution in Fig.5.4.

Figure 5.5 shows a sample of real versus generated designs alongside their skeletons and the geometrical metrics: ΔNbr_{bars} , $\Delta \Theta_{min}$ and $e_{len_{max}}\%$. As a matter of fact, designers are more interested in the design’s geometry, which is best defined by the skeletons, which explains their use here for comparison. As we can clearly see, DL-AM-TO captures the geometrical information; $\Delta Nbr_{bars} = 0$, $\Delta \Theta_{min}$ rarely exceeds 5° , similarly, $e_{len_{max}}\%$ does not exceed 10%. Aesthetically, the generated designs’ skeletons are similar to the real ones; SSIM is ≈ 0.7 .

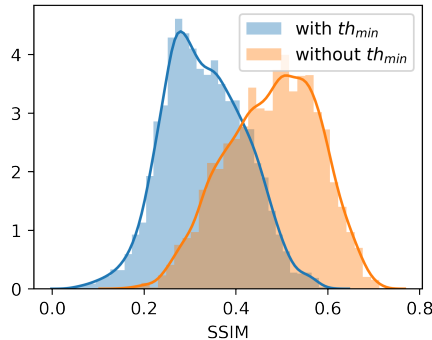


Figure 5.4: DL-AM-TO’s performance when trained with and without th_{min} .

The overall geometrical performance was evaluated manually over a sample of 100 designs of the test set; in other terms, we counted the Nbr_{bars} and measured the len_{max} , and Θ_{min} manually. We define a design complying with (i) the Nbr_{bars} constraint if $\Delta Nbr_{bars} \leq 1$, (ii) the len_{max} constraint if $e_{len_{max}\%} \leq 10\%$, and (iii) the Θ_{min} constraint if $\Delta\Theta_{min} \leq 5^\circ$. We find that 83% of the designs respect the Nbr_{bars} constraint, 76% comply with the len_{max} constraint, and 90% with the Θ_{min} constraint (Fig.5.6).

In order to further investigate the geometrical understanding of DL-AM-TO, we realized an experiment as shown in Fig.5.7. First, we fixed the mechanical constraints and altered one geometrical variable at a time (len_{max} and Θ_{min}). Then, to calibrate the design’s geometry, we modify the input value of the desired geometrical condition. As we can see, every time we increase len_{max}/Θ_{min} , the design’s shape is modified in order to comply with this variation while always conforming with mechanical constraints (the F and BC). However, we can notice that some geometrical constraints are correlated; increasing the Θ_{min} alters the len_{max} , and at a certain value, an additional bar appears (the 4th design in Fig.5.7b).

To sum up, DL-AM-TO captures the geometrical and mechanical constraints concurrently and responds to geometrical changes creatively; the obtained results encourage further model improvement.

5.3.6 Discussion

DL-AM-TO’s performance is tied to several criteria.

The first impact comes from the input data samples. In this work, the traditional SIMP was chosen to

























Real Design						
Generated Design						
Real Skeleton						
Generated Skeleton						
ΔNbr_{bars}	0	0	0	0	0	0
$\Delta\Theta_{min}$	-9°	0°	-1°	+4°	+0.5°	0°
$e_{len_{max}\%}$	3.5%	2.9%	0%	7.7%	3.7%	0%
$SSIM$	0.44	0.57	0.55	0.59	0.56	0.58
$SSIM_{skeleton}$	0.7	0.73	0.63	0.76	0.75	0.66

Figure 5.5: Comparison between the real and generated designs in their full and skeleton formats on the test set.

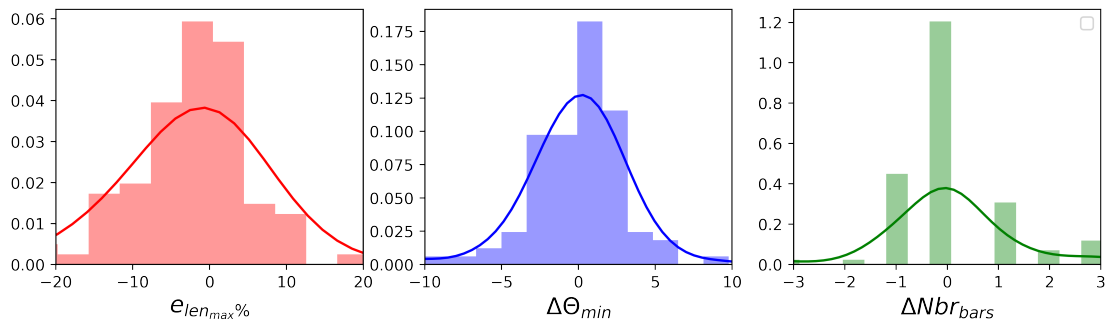


Figure 5.6: The distribution of the geometrical metrics ($e_{len_{max}\%}$, $\Delta\Theta_{min}$, ΔNbr_{bars}) manually measured over 100 designs of the test set.

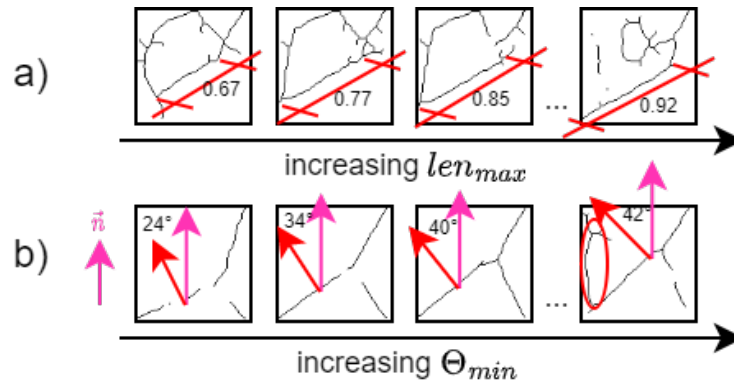


Figure 5.7: DL-AM-TO's response versus increasing len_{max} & Θ_{min} . In (a), to comply with len_{max} , the skeleton's layout changes drastically with the increase of len_{max} . The same behavior is noticed with the variation of Θ_{min} , at a point where additional bars start appearing (circled in red, 4th design in (b)). The pink arrow is the build orientation, the red arrow is the normal of the beam.

forward the GMCAD dataset creation as detailed in chapter 4. SIMP may not be the most performant, but it is the simplest and easily implemented TO algorithm (found in $\approx 70\%$ of the industrial and commercial software). Thus, the designs driving the training might not be the most optimal ones but are sufficient to validate the methodology proposed. Indeed, any new data from other, more optimal, TO algorithms can be used to train our model and improve its performance. Additionally, the mechanical conditions of GMCAD's designs are predictions of DL models, which adds a layer of uncertainty over the input training data samples.

Moreover, It is important to highlight that the main objective here is not to create a new TO algorithm but to compensate for the difficulties faced in TO when it comes to integrating the mechanical and the geometrical conditions simultaneously at the same level via DL architectures, particularly generative networks.

In the second position comes the geometrical discriminators' performance. The better the discriminator predicts the geometrical condition, the more informative the generator's loss function is; hence, DL-AM-TO is more reliable. We note that it is trained within GAN frameworks known for their unstable oscillating losses, which explains its sensitivity to the losses delivered by its discriminators. This phenomenon is observed with the th_{min} variable; integrating the latter into the model deteriorated its performance. The th_{min} discriminator should have been more precise. As a matter of fact, the image-like designs in GMCAD are CADs converted to images with computer vision filtering techniques, which can easily alter the thicknesses of the design.

Consequently, to improve on the previous results, several actions need to be taken:

1. Improve the performance of the geometric discriminators, for their performance dramatically impacts the generator's performance.
2. Isolate the new geometric discriminators with the mechanical conditions as done in chapter 3 with the Nbr_{bars} to improve our understanding of the correlations between the different geometrical variables (Fig. 5.7 in section 5.3.5).
3. Retrain DL-AM-TO with the new geometrical discriminators (section 5.6).
4. Put DL-AM-TO in action by printing several geometries of the same design to validate our hypothesis, which states that DL-AM-TO accelerates the whole DfAM, not simply the design phase.

5.4 Improving the geometrical discriminator's performance

The geometrical constraints are scalar values; thus, previously, we have chosen a regression-like architecture. Unfortunately, this architecture only convenes some constraints; th_{min} discriminator needed to be more precise. Hence, we decided to convert the regression problem into an ordinal regression one; it is a special case of classification problems. Instead of predicting the scalar value of a constraint, we will predict an interval in which it falls. The geometrical constraints that were concerned are len_{max} , th_{min} , and Θ_{min} .

The len_{max} constraint was divided into twenty-one classes: $(0.349, 0.4]$; $(0.4, 0.45]$; $(0.45, 0.5]$; $(0.5, 0.55]$; $(0.55, 0.6]$; $(0.6, 0.65]$; $(0.65, 0.7]$; $(0.7, 0.75]$; $(0.75, 0.8]$; $(0.8, 0.85]$; $(0.85, 0.9]$; $(0.9, 0.95]$; $(0.95, 1.0]$; $(1.0, 1.05]$; $(1.05, 1.1]$; $(1.1, 1.15]$; $(1.15, 1.2]$; $(1.2, 1.25]$; $(1.25, 1.3]$; $(1.3, 1.35]$; $(1.35, \sqrt{2}]$ with 0.349 and $\sqrt{2}$ being the minimum and maximum values of the len_{max} constraint, respectively. As we have described in chapter 4, the len_{max} and th_{min} are measured with respect to the unit measure, with it being the width of the design space, which is equal its height in our work.

The th_{min} constraint was divided into nine classes: $(0.0005, 0.01]$; $(0.01, 0.02]$; $(0.02, 0.03]$; $(0.03, 0.04]$; $(0.04, 0.05]$; $(0.05, 0.06]$; $(0.06, 0.07]$; $(0.07, 0.08]$; $(0.08, 0.14]$ with 0.0005 and 0.14 being the minimum and maximum values of the th_{min} constraint, respectively.

The Θ_{min} constraint was divided into fourteen classes: $(0, 5]$; $(5, 10]$; $(10, 15]$; $(15, 20]$; $(20, 25]$; $(25, 30]$;

(30, 35]; (35, 40]; (45, 50]; (50, 55]; (55, 60]; (60, 65]; (65, 70] with 0° and 70° being the minimum and maximum values of the Θ_{min} constraint, respectively. It is important to highlight here that we exclude the rotational data augmentation for two reasons. First, the Θ_{min} is the only geometrical value affected by the rotation of the design space. Also, we realized that the distribution of Θ_{min} values after rotational data augmentation became not uniform and did not include all the value intervals for the designs present in GMCAD; thus, this data augmentation will affect the representativeness of some classes in the dataset and deteriorate the model’s performance. Second, we train DL-AM-TO with no data augmentation; hence, the Θ_{min} discriminator is only needed to predict on non-rotated designs with Θ_{min} ranging from 0 to 70° .

To comply with the problem’s formulation as a classification instead of a regression, the discriminators’ architectures will only differ on the last fully connected layer; the output is a binary vector of K nodes with K being the number of classes of the variable described above; i.e., $K = 21$ for the len_{max} model, $K = 9$ for the th_{min} model, $K = 14$ for the Θ_{min} model.

5.4.1 Training

The geometric discriminators are trained via the state-of-the-art conditional ordinal regression for neural networks (CORN) framework [8]. The advantage of this approach over all previous versions of ordinal regression DL-based models [120, 121] is the rank consistency (Fig. 5.8).

The widely used training loss function in conventional classification is the cross entropy loss [122]. Unfortunately, this loss function is sub-optimal in our case, for it does not consider the order of the classes, and hence, it does not quantify the distance between the classes. In other terms, a prediction Θ_{min} of class 1 instead of class 2 is less problematic than a prediction Θ_{min} of class 5 instead of class 2. That is because the $\Delta\Theta_{min}$ of the former is 5° versus 15° for the latter. Thus, ordinal regression is used as an intermediate between regression and classification.

With ordinal regression, classes or labels range from 0 to K with $K =$ the total number of labels. We define $y_{ik} \in 0, 1$ such that $y_{ik} = 0$ if $y_{ik} < r_k$ and $y_{ik} = 1$ if $y_{ik} > r_k$ with r_k representing the k^{th} class and the rank $r_k \in 1, 2, \dots, K$.

CORN’s objective is to estimate conditional probabilities $f_{ik} = \hat{P}(y_i > r_k | y_i > r_{k-1})$ with $y_i > r_k \subseteq y_i > r_{k-1}$; in other terms, if a data point x_i has a $y_i = \Theta_{min} \in (10^\circ, 15^\circ]$ i.e. $y_i > r_2$, hence, definitely $y_i > r_1$ or $y_i > (5^\circ, 10^\circ]$.

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

The conditional probability $\hat{P}(y_i > r_k)$ is the product of all conditional probabilities over all subsets; $\hat{P}(y_i > r_k) = \prod_{j=1}^K f_j(x_i)$ with $f_j(x_i) \in [0, 1]$.

It is important to note here that estimating the conditional probabilities does not require all the training data points; only a subset is needed; to estimate $\hat{P}(y_i > r_5 | y_i > r_4)$, we need the subset of data points with $y_i > r_4$. Afterward, we minimize the binary cross entropy (loss function used for classification) on the data points of this subset.

Consequently, we decompose the training dataset into subsets with:

$$\begin{aligned} S_1 &= \text{all } (x_i, y_i) \quad \text{with } i = 1, \dots, N \quad \text{and } N = \text{Total number of training data points} \\ S_2 &: (x_i, y_i) | y_i > r_1 \\ &\dots \\ S_{K-1} &: (x_i, y_i) | y_i > r_{K-2} \end{aligned} \tag{5.1}$$

S_1 is the biggest subset (the whole training dataset), while S_{K-1} is the smallest subset. In other words, $|S_1| \geq |S_2| \dots \geq |S_k| \geq \dots \geq |S_{K-1}|$ with $|S_k|$ the size of the k^{th} subset.

The loss function as formulated by Shi et al. [8] is the following:

$$L(Z, y) = \frac{-1}{\sum_{j=1}^{K-1} |S_j|} \sum_{j=1}^{K-1} \sum_{i=1}^{|S_j|} [\log(\sigma(z_i)) \cdot \mathbb{1}_{y_i > r_j} + (\log(\sigma(z_i)) - z_i) \cdot \mathbb{1}_{y_i \leq r_j}] \tag{5.2}$$

Figure 5.9 shows how CORN predicts the class y_i of an input x_i . Additionally, figure 5.10, reprinted from [8], illustrates the loss computation based on the conditional training subsets.

5.4.2 training dataset

To further improve the geometric discriminator's accuracy, we have increased the number of training data samples per geometric variable to ensure that every class (i.e., every interval of values) is well represented. The distribution of the intervals of len_{max} , Θ_{min} and th_{min} values in the training set for every geometric discriminator is shown in figure 5.11.

5.4.3 Results

Tab 5.1 shows the global macro accuracy per geometrical discriminator. On the test set, the global accuracy score of the th_{min} is 93.7%, of len_{max} is 86.1%, and of Θ_{min} is 91.2%.

However, when it comes to multi-class classification, the global macro accuracy score can be deceiving, and a better approach is to use the micro scores, i.e., scores per class. Thus, figure 5.12 plots the classification results for each geometrical variable on the train, validation, and test sets. The red line

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

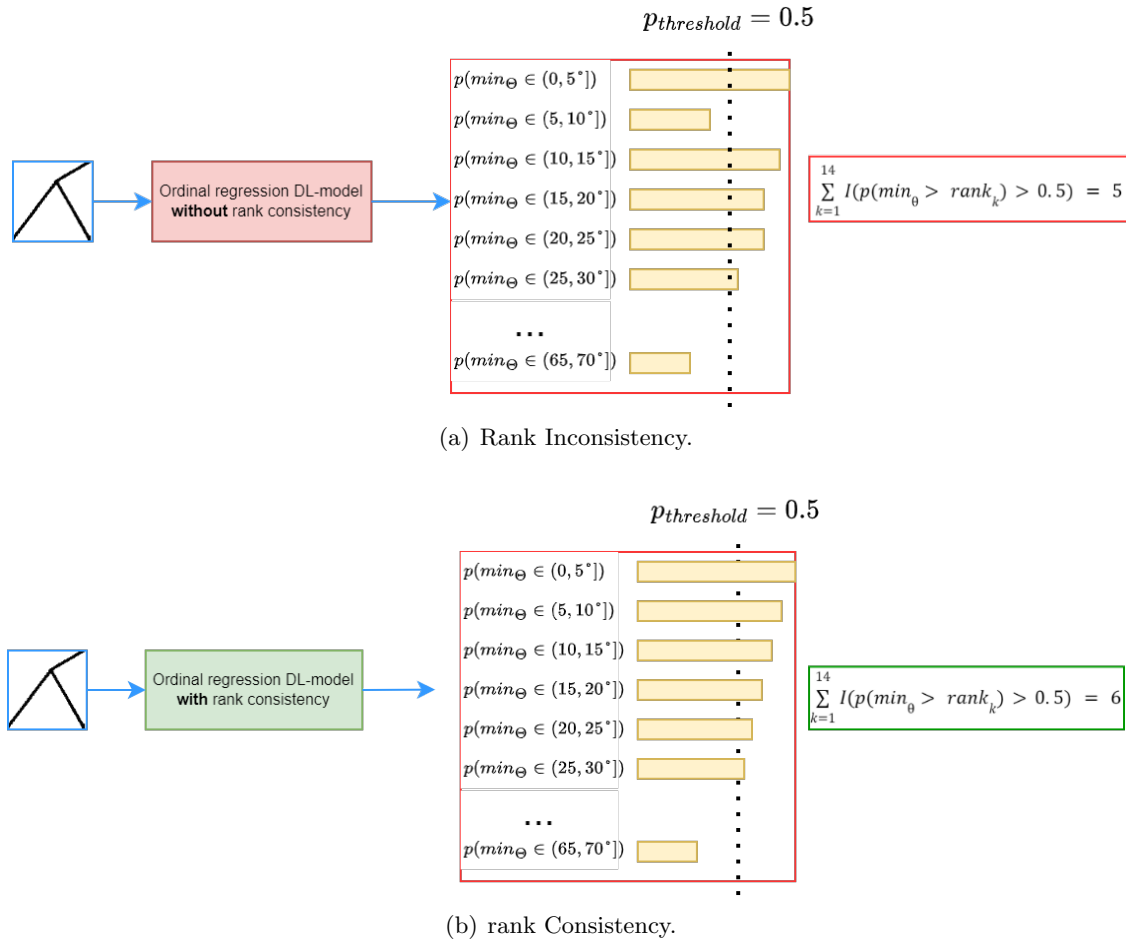


Figure 5.8: Ordinal regression models' rank consistency.

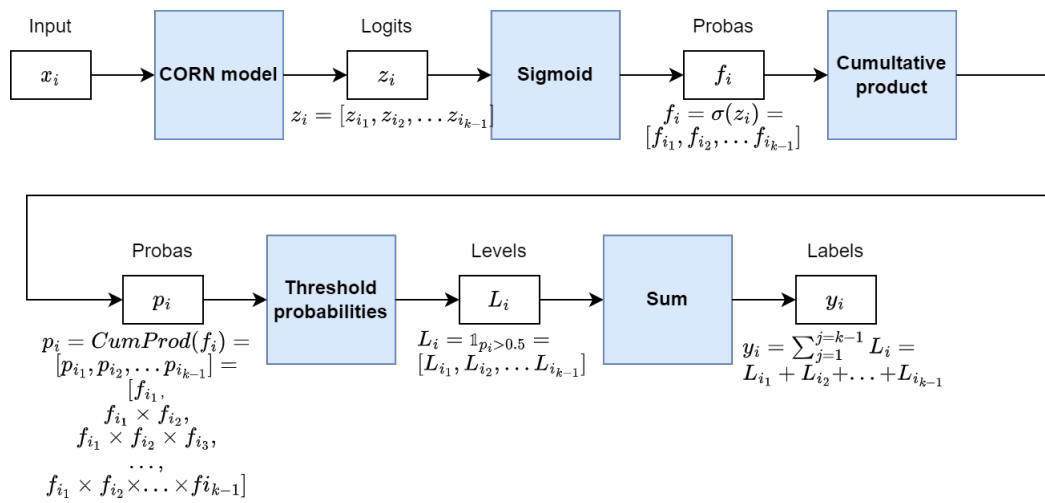


Figure 5.9: A Visual explanation of how CORN model predicts the ordinal class y_i of an input x_i .

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

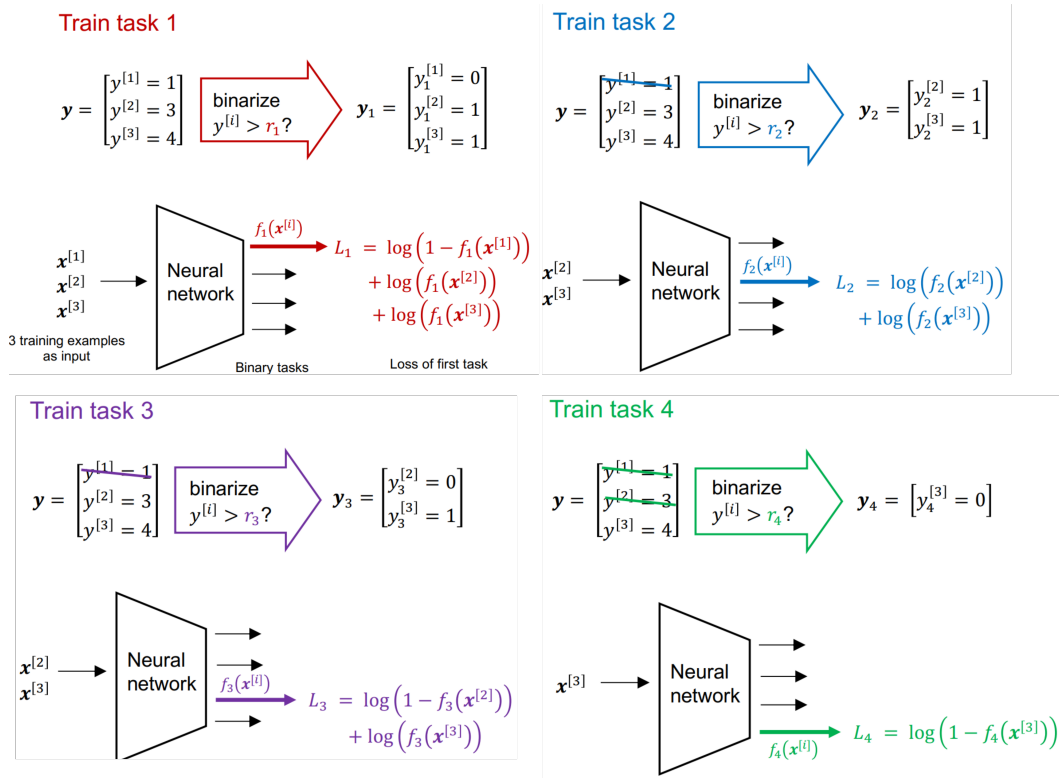
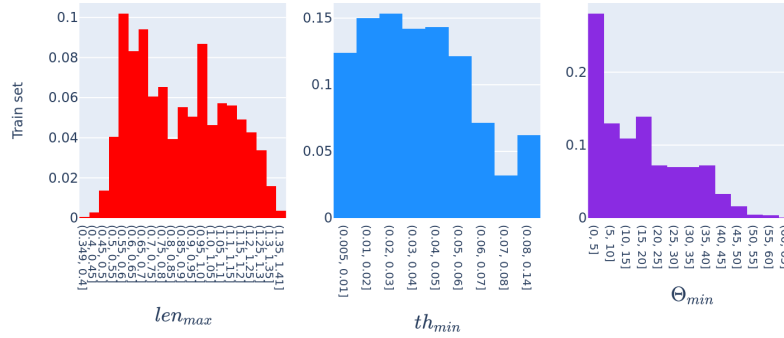
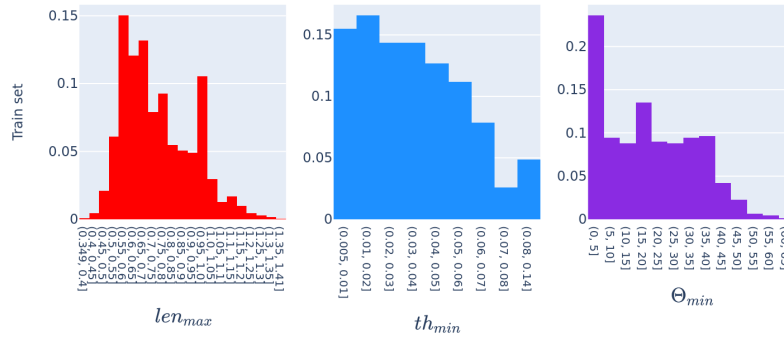


Figure 5.10: A Visual explanation of how the CORN loss is computed using the conditional training subsets. Reprinted from [8]. Assuming three training examples $x^{[1]}$, $x^{[2]}$, $x^{[3]}$ with three rank labels $y = [y^{[1]}, y^{[2]}, y^{[3]}] = [1, 3, 4]$, the overall loss $L(X, y) = \sum_i \frac{1}{|y_i|} \sum_i L_i = \frac{1}{3+2+2+1} (L_1 + L_2 + L_3 + L_4)$.

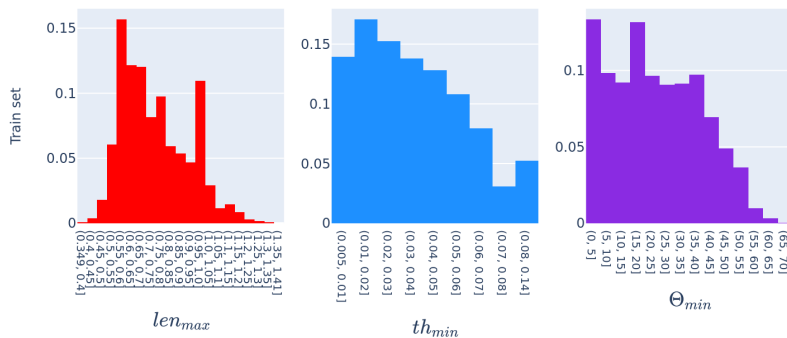
5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE



(a) Train set of len_{max} CORN discriminator.



(b) Train set of th_{min} CORN discriminator.



(c) Train set of Θ_{min} CORN discriminator.

Figure 5.11: Distribution of the geometric intervals of len_{max} , th_{min} , & Θ_{min} , from left to right respectively, in the training sets of the len_{max} , th_{min} , & Θ_{min} CORN discriminators, from top to down respectively.

corresponds to the predictions equal to the ground truth. The green lines correspond to the admissible misclassified predictions; an admissible misclassification is when the model classifies a point belonging to the class k into the class $k + 1$ or class $k - 1$.

As we can see, some classes are confused with classes outside the admissible misclassification margins. To estimate the percentage of confusions between the true class k and the admissible $k - 1$ and $k + 1$ classes and the percentage of misclassification outside these areas, we plot the confusion matrices of all three models (figures 5.13, 5.14, and 5.15).

For the th_{min} discriminator, the highest percentage of admissible confusion is 16.77%, for the class (0.07, 0.08]; it is confused with the class (0.08, 0.14], which is admissible. Moreover, the class (0.07, 0.08] is the least represented class in the training set (figure 5.11(b) shows the distribution of the classes of th_{min}). As for the remaining classes, this percentage never exceeds the 8%. The highest percentage of inadmissible confusion is of 1.8%, for the class (0.06, 0.07]; with this class being confused by 0.53% with the class (0.03, 0.04], 0.11% with the class (0.03, 0.04], and 1.16% with the class (0.08, 0.14].

For the len_{max} discriminator, the least accurate CORN discriminator, the highest percentage of admissible confusion is 30%, for the class (0.45, 0.5]. Similarly, the highest percentage of inadmissible confusion is 5.17% for the class (0.45, 0.05].

For the Θ_{min} discriminator, the highest percentage of admissible confusion is 28.6%, for the class (55°, 60°]; it seems that angles belonging to (55°, 60°] are highly confused with the class (60°, 65°]. As a matter of fact, this confusion is understandable, for these two classes are less represented with respect to the others (figure 5.11(c) shows the distribution of the classes of Θ_{min}). The highest percentage of inadmissible confusion is 1.68%, for the class (5°, 10°].

Finally, the percentage of inadmissible predictions are 1.23% for len_{max} discriminator, 0.92% for the Θ_{min} discriminator, and 0.4% for the th_{min} discriminator. Thus, all three geometrical discriminators are precise enough to forward the training of DL-AM-TO.

In addition to the approach detailed in chapter 2 (figure 2.1), we have decided to quantify the confidence of these DL models for each class predicted in order to validate the geometrical discriminators further. Uncertainty quantification is booming in the research and industrial worlds, for it is a way to quantify the reliability of a DL model within its deployment environment, an essential step for fully adopting DL models in industrial engineering systems.

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

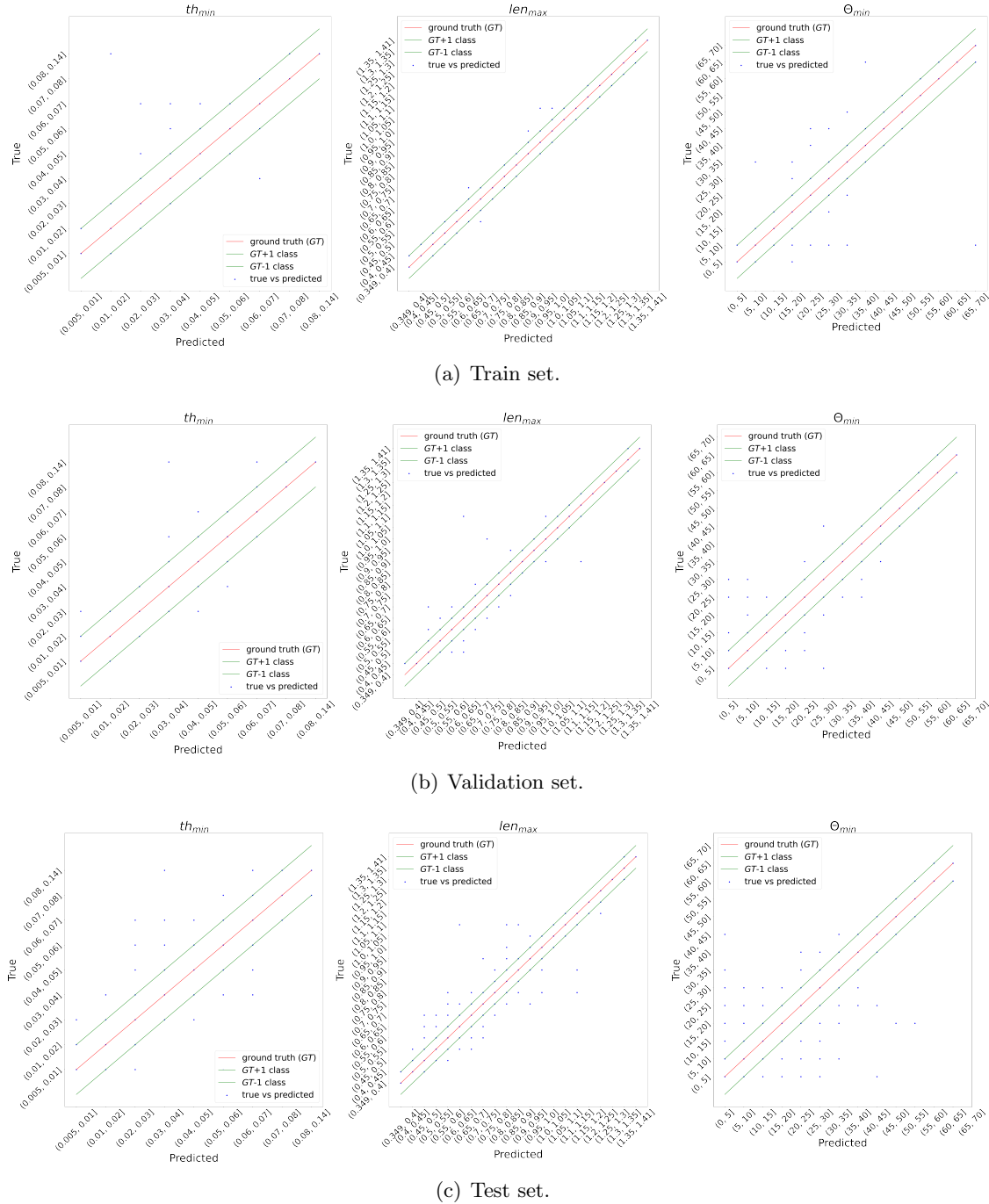


Figure 5.12: Performance of the geometric discriminators showing the predicted vs the true classes of th_{min} , len_{max} , & Θ_{min} , from left to right, respectively, for the train, validation, and test sets. The red line corresponds to the predictions being equal the true class. The green lines correspond to the previous and the following class; they represent the admissible misclassified predictions.

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

Table 5.1: Macro average accuracy score of th_{min} , len_{max} , and Θ_{min} .

Geometrical Discriminator	Train	Validation	Test
th_{min}	99.3%	93.1%	93.7%
len_{max}	98.3%	86.8%	86.1%
Θ_{min}	98.1%	91.8%	91.2%

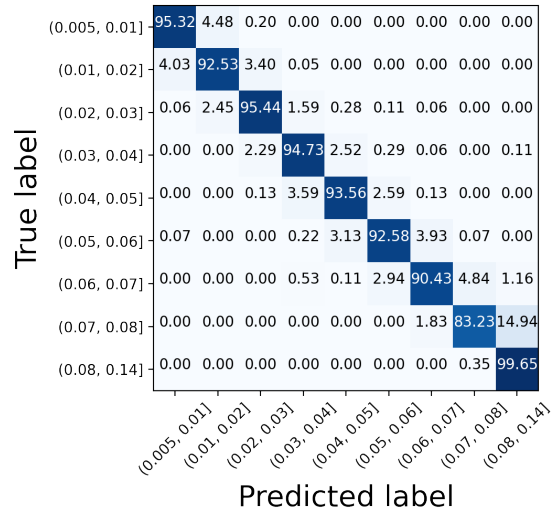


Figure 5.13: The confusion matrix of the th_{min} CORN discriminator computed on the test set.

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

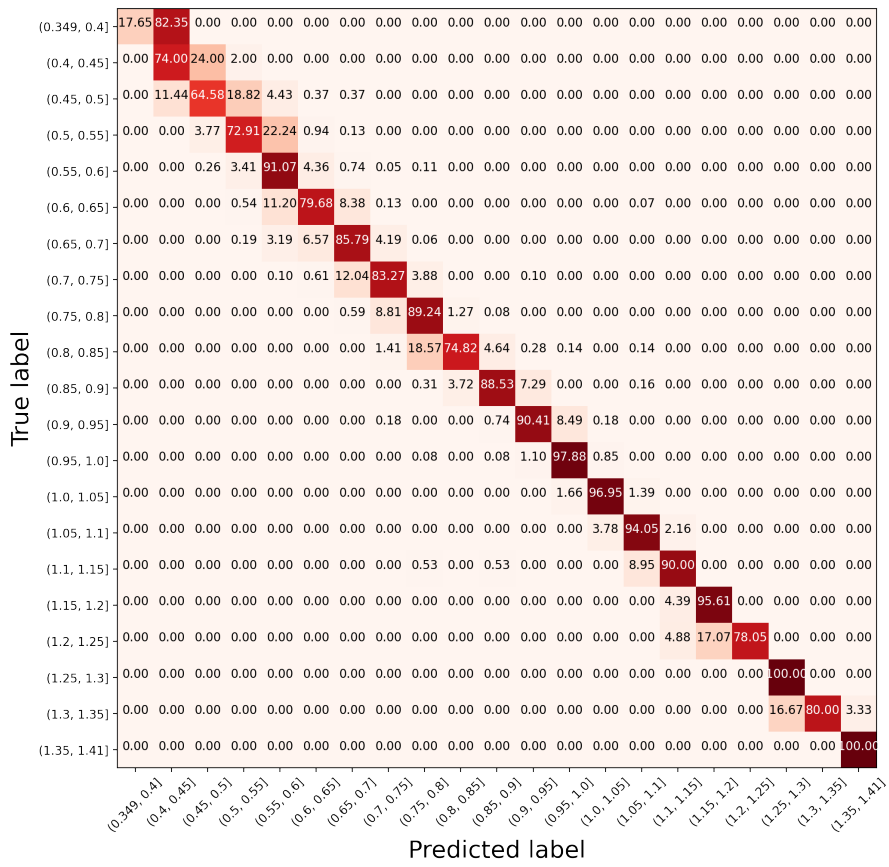
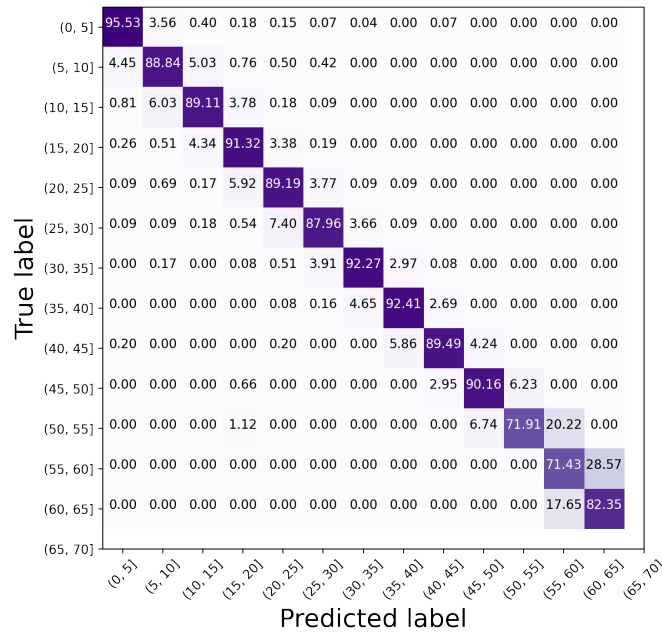


Figure 5.14: The confusion matrix of the len_{max} CORN discriminator computed on the test set.

Figure 5.15: The confusion matrix of the Θ_{min} CORN discriminators computed on the test set.

5.4.4 Uncertainty quantification of the DL geometrical discriminators

Uncertainty in NN originates essentially from two sources: the network’s architecture and training (loss function, learning rate, number of epochs, batch size, Etc.) and the data forwarding the learning [123]. The literature on uncertainty quantification identifies two main types of uncertainty: data uncertainty, known as aleatoric uncertainty; wrong labeling, low image resolution, inaccurate measurements, and data not covering sufficiently the real-world’s variability; and model uncertainty, known as epistemic uncertainty; the number of parameters in a NN’s architecture, the learning rate, and the number of iterations leading to an under or over-fitted model, Etc. [124, 125]. Aleatoric uncertainty is irreducible due to the measurements’ precision and information loss; the training data scarcely represents the entire sample space. On the other hand, model uncertainty represents the uncertainty caused by defects in the model, either by errors in the learning procedure, an insufficient architecture, or a lack of knowledge due to unknown samples or poor coverage of the training data set [123].

The diversity of the uncertainty sources in deep NN makes it impossible to eradicate it. Despite the impossibility of its eradication, the literature proposes several approaches to estimate it. Gawlikowski et al. (2021)[123] propose a classification into four groups of uncertainty quantification: determinis-

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR’S PERFORMANCE

tic approaches[126, 127], Bayesian approaches[128, 129, 130], ensemble methods[131], augmentation methods during test time[132]. Augmentation methods are inappropriate in our case because a rotation or translation of a design could change the geometrical variable’s value; the Θ_{min} constraint is rotation-variant. The deterministic and Bayesian approaches were implemented and tested by our intern Steve Baggio Sedjro NOUATIN.

In brief, Bayesian inference in NN is based on the fundamental idea that the NN’s weights are represented by probability distributions over possible values. Thus, rather than learning a single fixed value as in deterministic NN, Bayesian networks try to learn distributions over the weights of the NN. The Bayesian methods that were explored are the Monte Carlo Dropout (MCD) [129], the Bayes By Backprop (BBB)[128] and the preconditioned Stochastic Gradient Langevin Dynamics (pSGLD). Monte Carlo Dropout is considered an approximation of Bayesian inference in NN. It consists in inserting a dropout layer before each model layer.

Gal and Ghahramani showed that dropout before each weight layer of DNN is mathematically equivalent to an approximation by variational inference of a Gaussian process. The dropout layers are active not only during training but also during prediction to ensure that the model outputs are stochastic. The Bayes By Backprop method proposes to use variational inference to approximate the a posteriori distribution of the model parameters.

An alternative to variational inference in Bayesian NN is using Stochastic Gradient Markov Chain Monte Carlo methods, particularly pSGLD for scalability reasons, to generate an a posteriori sample of the model parameters [130].

Evidential NNs are complementary approaches to Bayesian networks for uncertainty quantification. This technique is based on the belief functions of the Dempster-Shafer theory [133], and on the theory of subjective logic [134]. An evidential NN assigns a Dirichlet distribution to the class probabilities for classification use cases. They treat the NN’s predictions as subjective opinions and thus learn the function that gathers the evidence leading to these opinions by a deterministic NN from the data. Subjective logic assigns a positive belief value and an uncertainty value to each class so that the sum of the belief values and uncertainty equals one [126]. Evidential NN in classification are, therefore, classical NN whose softmax layer is replaced by a positive output activation layer (e.g., ReLU) to guarantee a non-negative output, which is taken as an evidence vector for the predicted Dirichlet distribution.

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

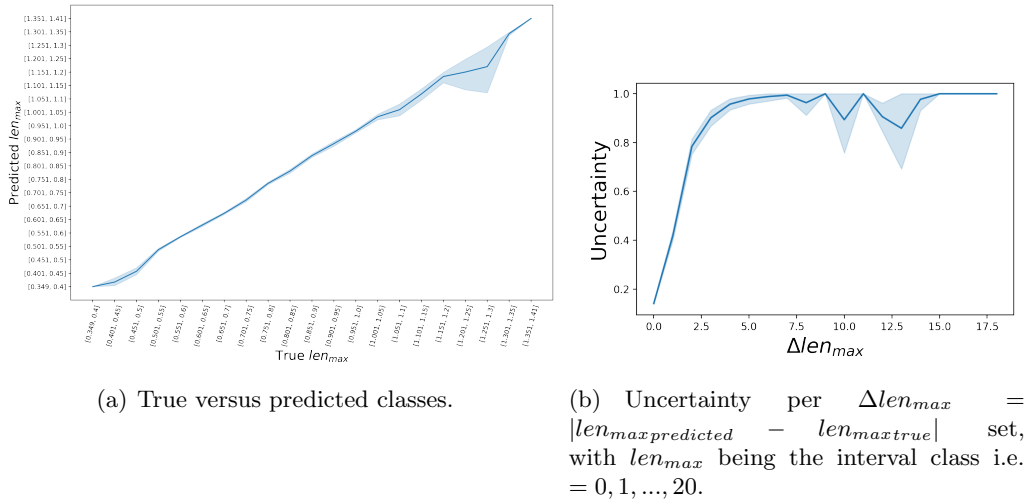


Figure 5.16: Uncertainty quantification computed over the predictions of the len_{max} geometrical discriminator on the test set.

A detailed explanation of these methods and their results can be found in the supplementary material, Steve’s final year project report. These methods converged to the same result. However, their implementation, training, and usage costs widely varied. In the following, the uncertainties computed using the Evidential NN method are reported for the three geometrical CORN discriminators (len_{max} , Θ_{min} , and th_{min}) for evidential NN shows the adequate method when it comes to accuracy, DL model’s number of parameters, training to convergence time, storage, and prediction time.

The results of the uncertainty quantification are illustrated in two ways: the true versus predicted classes plot and the uncertainty versus the prediction error plot; the prediction error in the three cases is the difference between the true and predicted class. In all three figures 5.16, 5.17, and 5.18, the true versus predicted class plot is almost a straight line demonstrating that the discriminators’ prediction precision. It should be noted that figure 5.16(a) shows that the class [1.251, 1.3] is confused by its preceding class [1.201, 1.25], which is admissible. Similarly, figure 5.17(a) shows that the class [60°, 65°] is confused with its preceding class [55°, 60°]. On the other hand, figures 5.16(b), 5.17(b), and 5.18(b) show the evolution of the uncertainty value outputted by the networks versus the prediction error. For all three discriminators, the uncertainty value predicted increased with the prediction error. To sum up, the geometrical discriminators are now quantitatively reliable with evidential NN; they can now forward DL-AM-TO’s training.

5.4. IMPROVING THE GEOMETRICAL DISCRIMINATOR'S PERFORMANCE

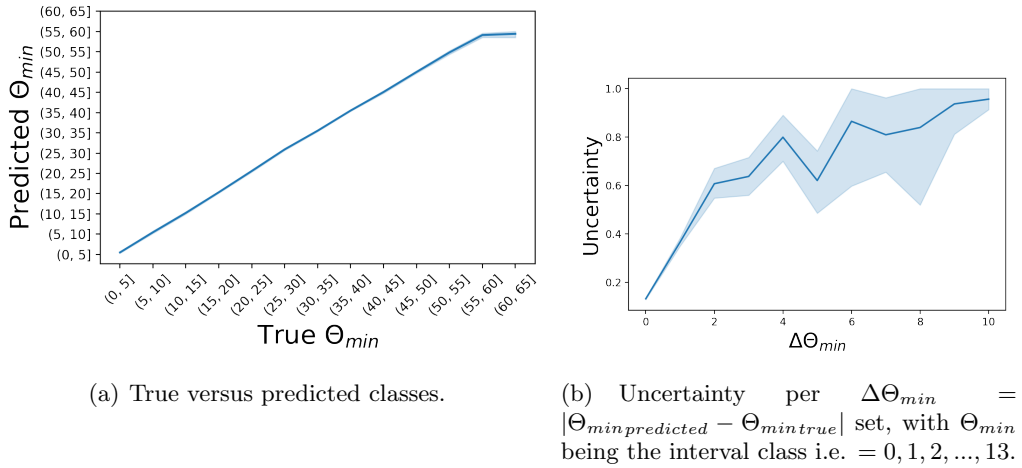


Figure 5.17: Uncertainty quantification computed over the predictions of the Θ_{min} geometrical discriminator on the test set.

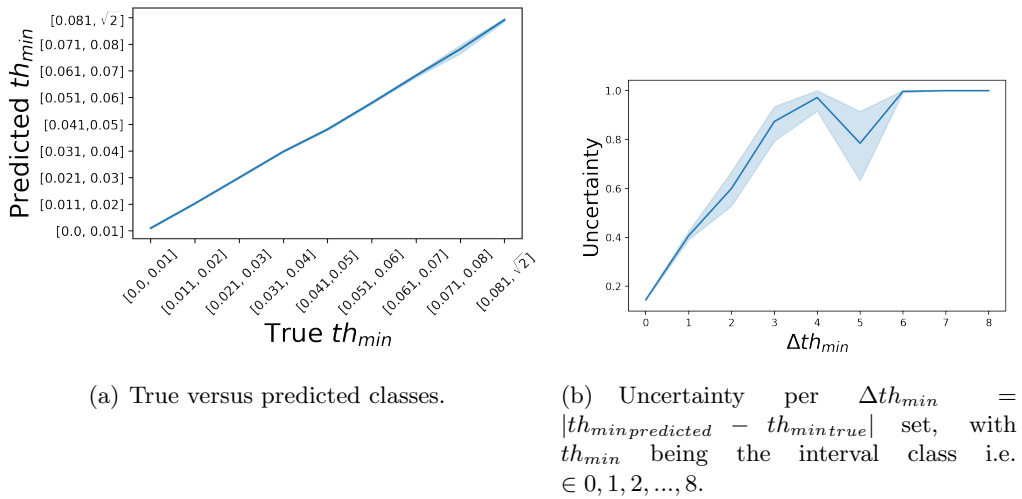


Figure 5.18: Uncertainty quantification computed over the predictions of the th_{min} geometrical discriminator on the test set.

5.5 Training a model per geometrical variable

In order to understand the relationship between the geometrical constraints, we will need to isolate every geometrical condition and train the model, and then we will study the influence of changing one geometrical variable on the other geometrical variables.

The geometrical variables that will be considered are len_{max} , Θ_{min} , and Nbr_{bars} . The th_{min} is omitted for thickening, or thinning bars do not have any other influence on the lengths of these bars, their number, or their angles.

The training consists of the generator (DL-AM-TO), which takes as input the mechanical conditions $(BC_x, BC_y, F_x, F_y, V)$ and the geometrical condition in hand and outputs a 2D design validating the input conditions. The generated design is compared to the ground truth design, passes through the geometrical discriminator to predict its geometrical condition, and is penalized if it does not conform to the input condition.

When the training is completed, the experiment consists of generating several designs conforming with the same mechanical constraints but different values of this geometrical condition. In other words, the engineer will have several geometries for a fixed set of mechanical constraints and will be able to choose the geometry that fits the manufacturing-geometrical constraint the printer needs.

5.5.1 Training with Θ_{min}

The training consists of replicating the same procedure detailed in section 5.2.1 but with one geometrical constraint and thus one geometrical discriminator, here the Θ_{min} 's.

The idea here is to understand the correlation between Θ_{min} versus the other geometrical variables; len_{max} and Nbr_{bars} .

For this study, we have used the training dataset built to train the Θ_{min} discriminator (Fig.5.11(c)). As mentioned previously, we will test the ability of the generator to propose several geometries for the same mechanical constraints. Figure 5.19 shows the generated designs after having scanned several len_{max} values for three fixed sets of mechanical conditions (Fig.5.19(a), Fig.5.19(b), and Fig.5.19(c)). For every sub-figure, there are three design representations. Each sub-figure corresponds to a design space subject to a fixed set of mechanical constraints with increasing len_{max} constraint. The first representation, in every subfigure's first line, corresponds to the image-like geometries generated by the

DL model after applying threshold, denoising, and smoothing computer vision algorithms. The second representation, in the second line, corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).

As we can clearly see, the geometries change to adapt to the increasing Θ_{min} constraint. However, we observe some irregularities sometimes.

For example, in Fig.5.19(a), the DL model struggles to generate a geometry with more than 45° no matter how much we increase the input Θ_{min} . The mechanical constraints seem to have a higher dominance in this case.

In Fig.5.19(b), we can see that the model changed the geometry radically to comply with the increasing Θ_{min} ; to comply with the increasing Θ_{min} , the geometry passes through several modifications where at a certain point the Θ_{min} is no longer compliant with the increasing input value (the seventh and eighth designs in Fig.5.19(b)).

As for Fig.5.19(c), the fifth and sixth designs do not comply with the input Θ_{min} . However, this is compensated from the seventh to the ninth design proposed.

A simple procedure would be to eliminate the designs not complying with the Θ_{min} constraint and only present to the user the valid propositions, as shown in Fig.5.20.

This model aspires to understand the impact of controlling one geometrical variable over the others that are not controlled.

Fig.5.20(a) shows that the len_{max} variable is kept constant and the Nbr_{bars} variable fluctuates with no clear relationship with the Θ_{min} 's change. In Fig.5.20(b), the average len_{max} is 0.62, the generated designs have $len_{max} \pm 0.1$ from 0.62. However, we can clearly remark that the Nbr_{bars} decreases in the design when we increase Θ_{min} . This same behavior is identified in Fig.5.20(c). Thus, we can conclude that Θ_{min} is inversely proportional to Nbr_{bars} . Indeed, a higher value of Θ_{min} means highly tilted bars; thus, horizontal internal transmission bars need to be eliminated, which explains this relationship between the two variables. For the len_{max} , we see no obvious relationship with Θ_{min} ; hence, for the moment being, we assume they are independent.

In conclusion, our model proposes to engineer several geometries creatively with different Θ_{min} constraints in a fraction of a second. This alleviates him/her the need to come up with these shapes him/herself and helps him/her choose the right design complying with the Θ_{min} constraint directly

without the need to get stuck in a loop of modifying the geometry and testing until convergence.

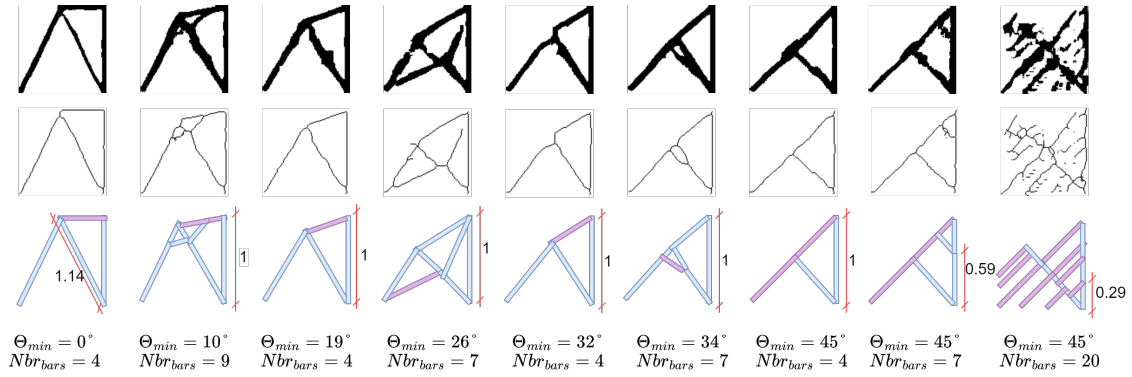
5.5.2 Training with len_{max}

Figure 5.21 shows the generated designs after having scanned several len_{max} values for three fixed sets of mechanical conditions. For every sub-figure, there are three design representations. The first representation, in every sub-figure's first line, corresponds to the image-like geometries generated by the DL model after applying threshold, denoising, and smoothing computer vision algorithms. The second representation, in the second line, corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).

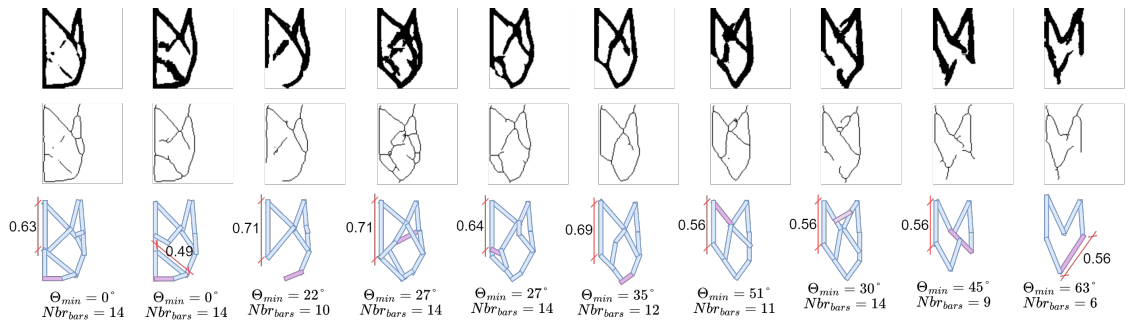
In all three figures, we can see an evolution in the geometry of the design to comply with the increasing value of the len_{max} condition. However, the sensitivity of the DL model to the geometrical input value varies from one structure to another. Designs in Fig.5.21(a) show the best sensitivity; the geometries evolve in a way to ensure increasing len_{max} constraint. More complex structures can be harder to adjust; the structure of designs of Fig.5.21(b) is an example. In order to comply with the increasing value of len , the model needed to widen the design; thus, starting from the fifth design, we can see the shape becoming wider, nevertheless losing the conformity with the constraint ($0.57 \leq 0.6$). However, this inconvenience comes with the benefit of complying with the constraint in the sixth and seventh designs. In figure 5.21(c), we can see that the DL model was creative and abstracted the notion of the len_{max} constraint; the first two shapes do not exist in the training database.

As for the relationship between len_{max} and the other geometrical constraints, we can validate that the len_{max} condition is inversely proportional to the Nbr_{bars} constraint; if we would like to eliminate bridges (long bars hanging), it is better to add a bar to support them. For the Θ_{min} , there is no clear rule. It depends heavily on the outer shape, which is defined by mechanical constraints. When this shape happens to be flexible while always complying with mechanical constraints, Θ_{min} decreases with increasing len_{max} (figures 5.21(b) and 5.21(c)). It is important to highlight that in section 5.5.1 where the controllable variable was len_{max} , there were no obvious relation between len_{max} and Θ_{min} . However, when Θ_{min} is the controllable variable, we have identified that it is inversely proportional to len_{max} but not in all cases, only in cases where the outer structure defined by the mechanical constraints allows it. In other terms, len_{max} and Θ_{min} are conditionally dependent with respect to

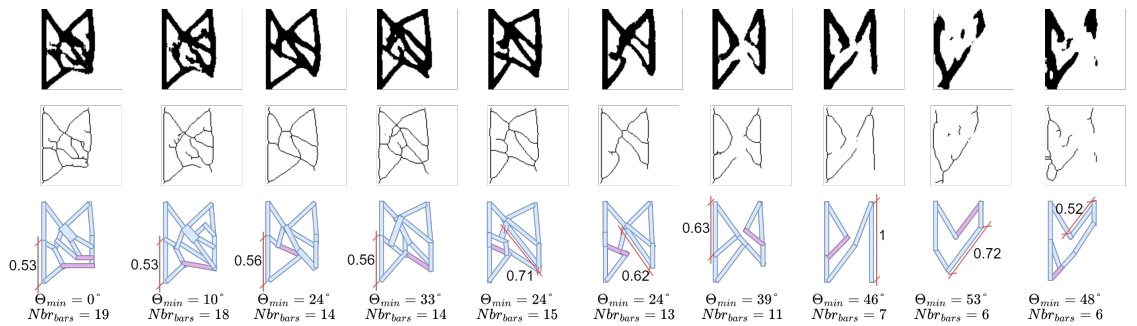
5.5. TRAINING A MODEL PER GEOMETRICAL VARIABLE



(a) First set of designs.



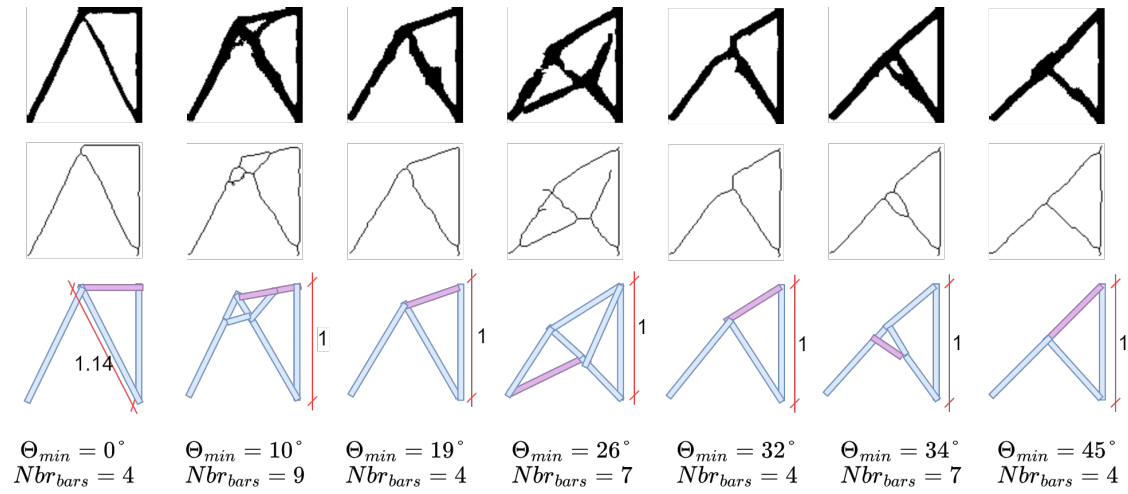
(b) Second set of designs.



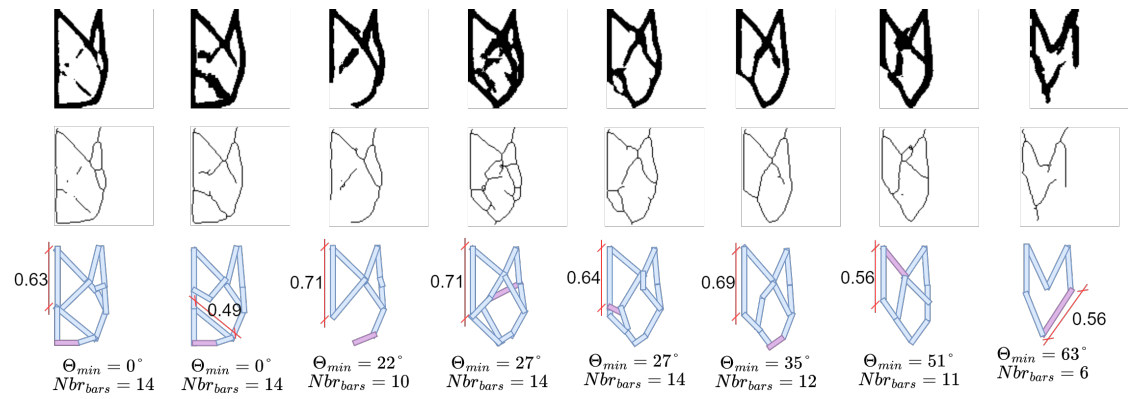
(c) Third set of designs.

Figure 5.19: Geometries proposed by the DL model trained to tailor Θ_{min} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different len_{max} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing's computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).

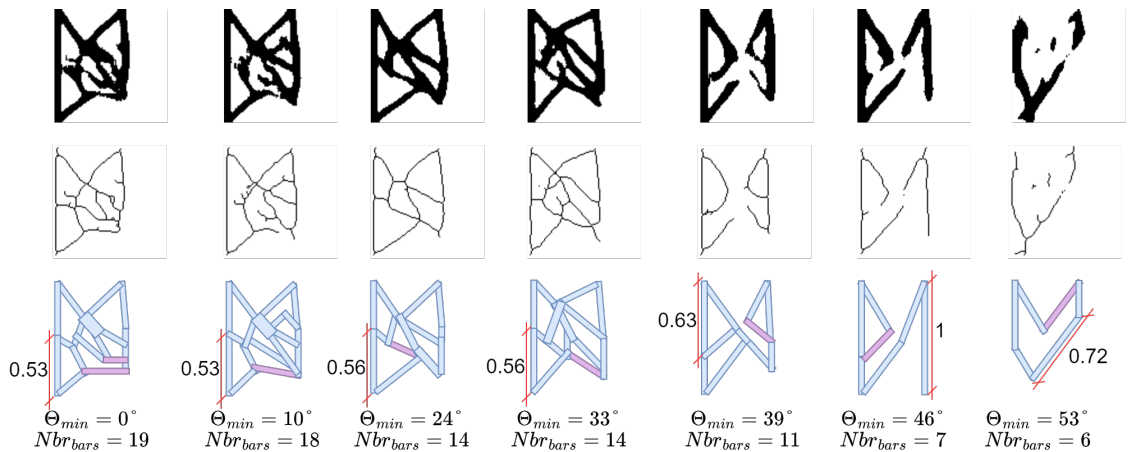
5.5. TRAINING A MODEL PER GEOMETRICAL VARIABLE



(a) First set of designs.



(b) Second set of designs.



(c) Third set of designs.

Figure 5.20: Geometries proposed by the DL model trained to tailor Θ_{min} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different len_{max} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing's computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).¹²⁸

the mechanical constraints.

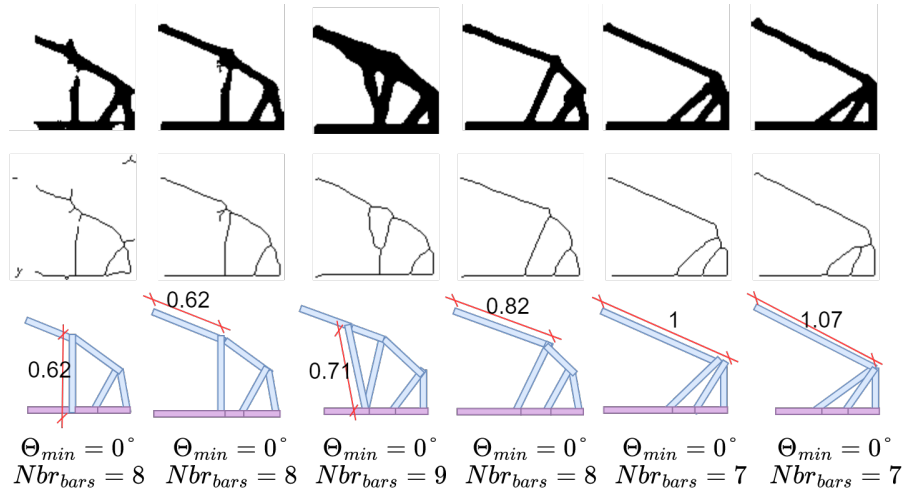
5.5.3 Training with Nbr_{bars}

Figure 5.22 shows the generated designs after having scanned several Nbr_{bars} values for three fixed sets of mechanical conditions. For every sub-figure, there are three design representations. The first representation, in every subfigure's first line, corresponds to the image-like geometries generated by the DL model after applying threshold, denoising, and smoothing computer vision algorithms. The second representation, in the second line, corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).

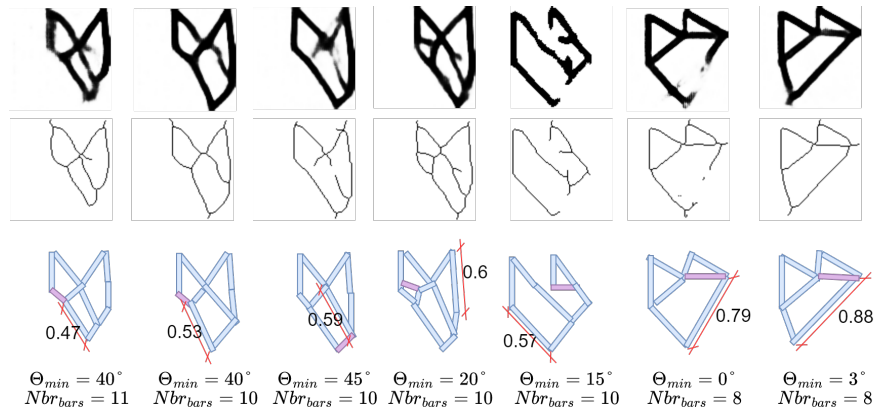
This last experiment is important since it will decide whether the conclusion about the relationships between Nbr_{bars} and Θ_{min} and len_{max} deduced in sections 5.5.1 and 5.5.2 still hold when the controlling variable is Nbr_{bars} .

All three figures 5.22(a), 5.22(b), and 5.22(c) validate the inverse proportionality between len_{max} and Nbr_{bars} deduced in section 5.5.2. Similarly, figures 5.22(a) and 5.22(b) validate the inverse proportionality between Θ_{min} and Nbr_{bars} . However, in figure 5.22(c), we can notice that the Θ_{min} is constant with increasing Nbr_{bars} . In other terms, the Nbr_{bars} and Θ_{min} 's inverse proportionality relationship is conditioned on the general structure of the design, which is defined by the mechanical constraints. This behavior encourages us to revisit this type of structure with the DL model trained to control the Θ_{min} to check how the model modifies the geometry of this type of design. This is done in figure 5.23. As we can clearly see, the model has trouble changing the geometry to comply with the Θ_{min} constraint; however, to finally comply, we can see the geometry migrating to a geometry that it does, knowing that it might not be the best fit with respect to the mechanical constraints. Hence, we conclude that the Nbr_{bars} and Θ_{min} 's inverse proportionality relationship is conditioned on the general structure of the design, which is defined by the mechanical constraints, and more importantly, that the mechanical constraints are still predominant when it comes to the general geometry of the design. As a matter of fact, in some cases, the optimal geometry does not exist. Nonetheless, this particular structure is better optimized by adding bars to support hanging features.

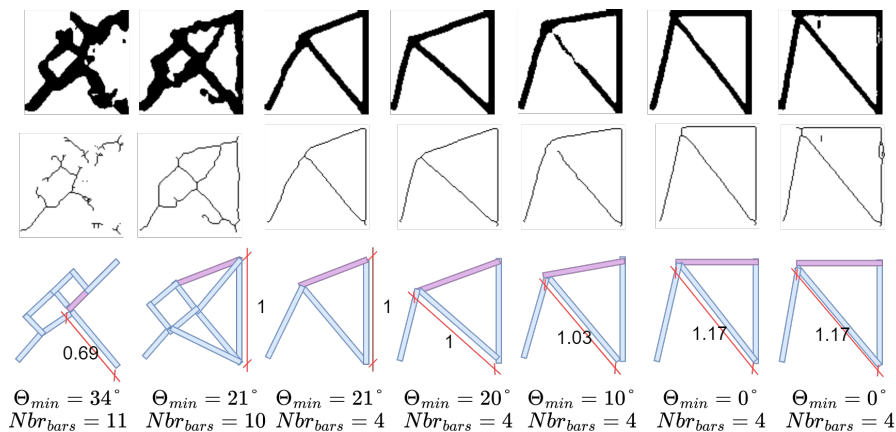
5.5. TRAINING A MODEL PER GEOMETRICAL VARIABLE



(a) First set of designs.



(b) Second set of designs.



(c) Third set of designs.

Figure 5.21: Geometries proposed by the DL model trained to tailor len_{max} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different len_{max} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing computer vision algorithms. The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).

In conclusion, while training several DL models, each controlling a different geometrical variable, We still have seen a consistency in their learning of the geometrical information. We have deduced that the len_{max} and Nbr_{bars} are inversely proportional, the Nbr_{bars} and Θ_{min} are inversely proportional when the general geometry defined by the mechanical constraints allows it, and finally, len_{max} and Θ_{min} are conditionally dependent with respect to the mechanical constraints. In the next section, we will examine their behavior when controlled simultaneously along the mechanical constraints.

5.6 Second variant of DL-AM-TO

With the findings in sections 5.4 and 5.5, we retrained DL-AM-TO with CORN geometrical discriminators. We added a new element to the training loss function, the structural similarity loss (L_{SSIM}) implemented in the pytorch library piq². L_{SSIM} 's formula and implementation details are described in the following reference <https://piq.readthedocs.io/en/latest/modules.html#piq.SSIMLoss>.

5.6.1 Results

This section will be divided into two parts. The first part describes the overall performance of DL-AM-TO over 12340 designs, and the second part will examine its capacity to tailor a design's geometry corresponding to a variation in the geometrical input value.

5.6.2 DL-AM-TO's overall performance

Figure 5.24 shows the distribution of the aesthetics metrics, the Structural Similarity ($SSIM$), Peak Signal to Noise ratio ($PSNR = 10\log_{10} \times \frac{R^2}{MSE}$ in decibel dB , with R the maximum fluctuation in the input image data type, here 1; the generated designs are images with values ranging from 0 to 1.0), and the reconstruction error (the Mean Squared Error, MSE) between the generated designs and the ground truth designs for three variants. We compute the metrics with the raw outputs, i.e., we compare DL-AM-TO's output to the ground truth before the application of any transformation, then, we recompute the metrics with the designs thresholded, and finally, we compare the ground truth designs' skeletons to the generated designs' ones. The average $SSIM$ computed on the raw generated designs is 0.72 and 0.75 on the skeletons. The average $PSNR$ is 14dB versus 15dB on the skeletons. The average MSE is 0.051 versus 0.033 on the skeletons. It is important to note that the gap between

²https://piq.readthedocs.io/en/latest/usage_examples.html

5.6. SECOND VARIANT OF DL-AM-TO

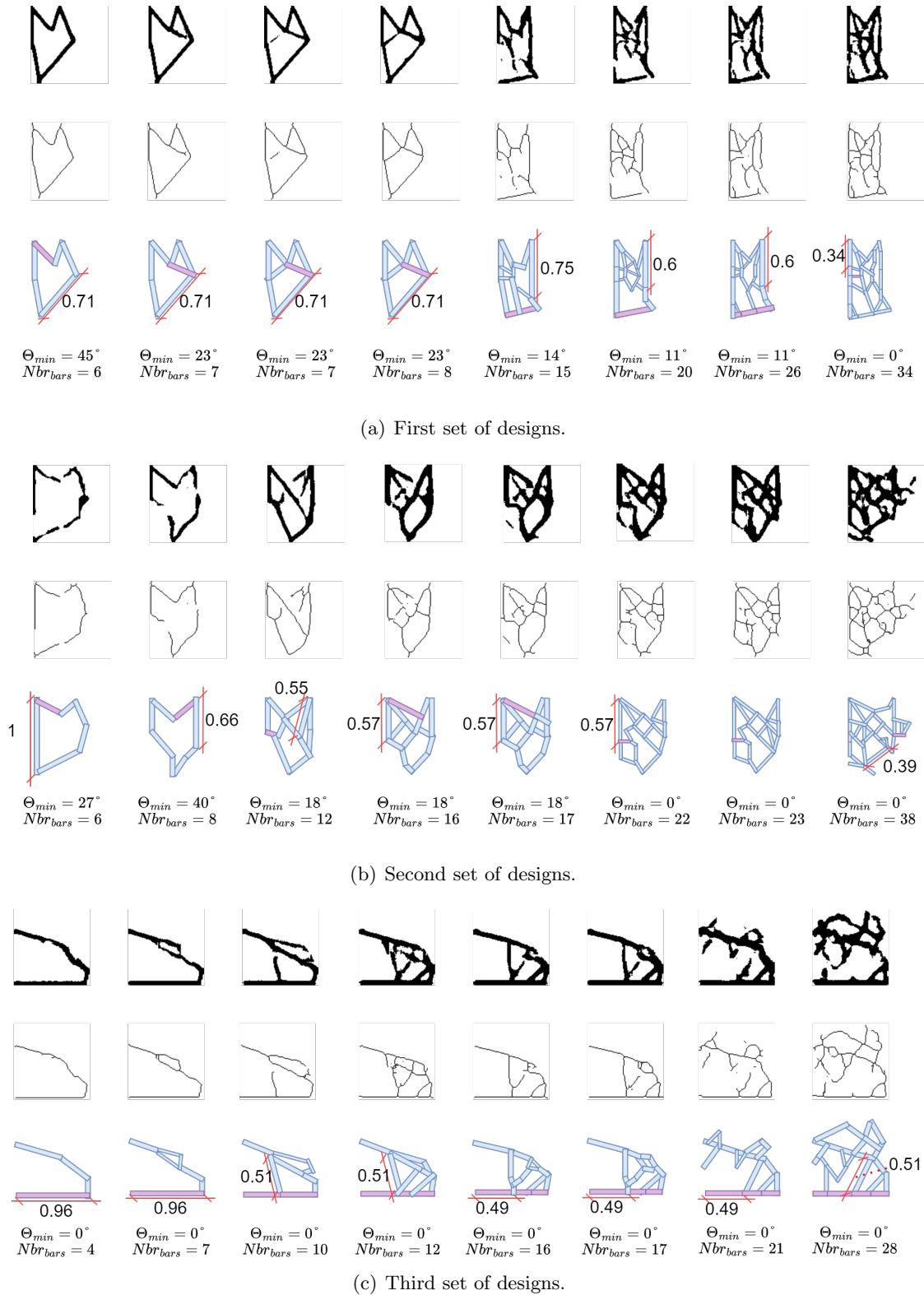


Figure 5.22: Geometries proposed by the DL model trained to tailor Nbr_{bars} . Every figure shows a list of geometries corresponding to a fixed set of mechanical conditions and different Nbr_{bars} condition. The first line corresponds to the geometries generated by DL after the application of threshold, denoising and smoothing computer vision algorithms.¹³² The second line corresponds to the skeletons. Finally, the third line corresponds to the geometries drawn manually; the beams colored in purple are the beams with the minimum overhang (Θ_{min}).



Figure 5.23: Particular geometries generated by a DL model trained to control the Θ_{min} along the mechanical constraints.

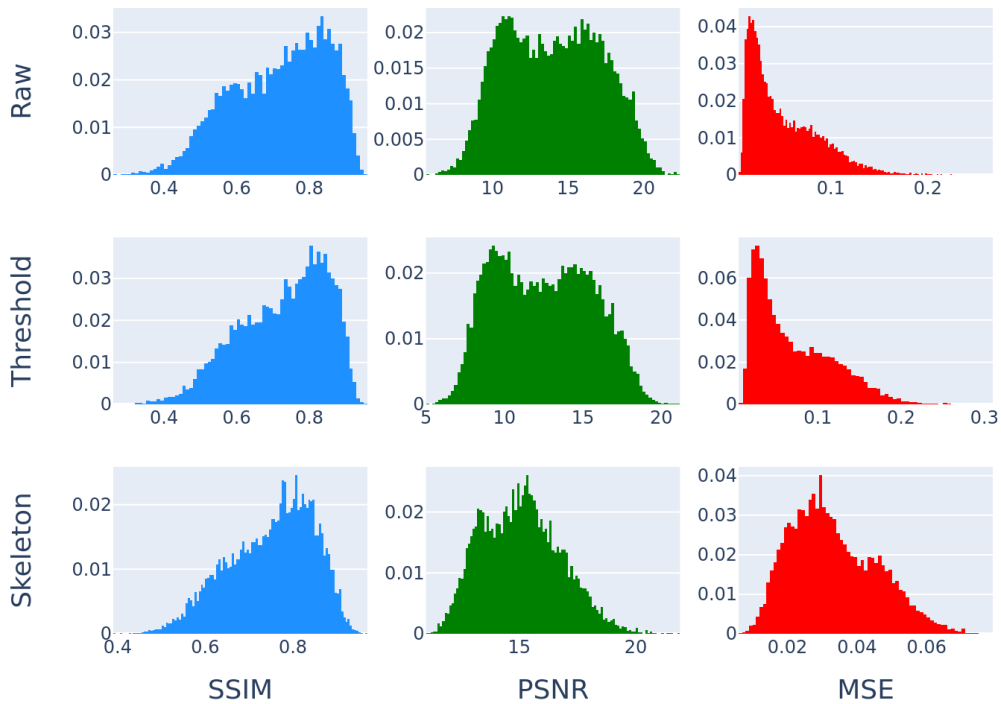


Figure 5.24: Distribution of the Structural Similarity ($SSIM$), Peak Signal to Noise ratio ($PSNR$), and the reconstruction error (the Mean Squared Error, MSE), from left to right respectively, computed on the test set, for three variants of the designs: raw, threshold, and skeletons. the raw designs are DL-AM-TO's outputs without any transformation.

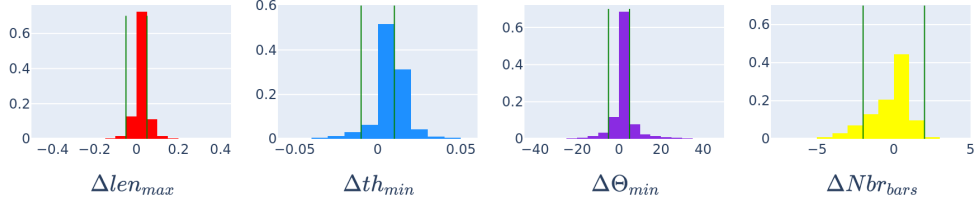


Figure 5.25: Distribution of the geometrical conformity metrics, $\Delta_{len_{max}}$, $\Delta_{th_{min}}$, $\Delta_{\Theta_{min}}$, and $\Delta_{Nbr_{bars}}$, from left to right respectively, with $\Delta_X = X_{generated} - X_{input}$; $X \in len_{max}, th_{min}, \Theta_{min}$, and Nbr_{bars} . The green line corresponds to $\Delta_{len_{max}} = \pm 0.05units$, $\Delta_{th_{min}} = \pm 0.01units$, $\Delta_{\Theta_{min}} = \pm 5^\circ$, and $\Delta_{Nbr_{bars}} \pm 2bars$ in the $\Delta_{len_{max}}$, $\Delta_{th_{min}}$, $\Delta_{\Theta_{min}}$, and $\Delta_{Nbr_{bars}}$ plots, respectively.

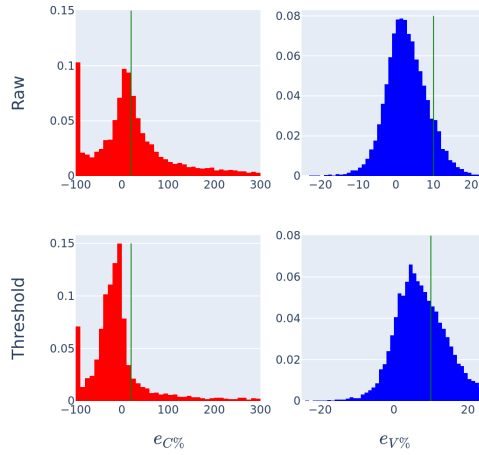


Figure 5.26: Distribution of the mechanical metrics, the relative error of compliance ($e_C\%$) and the relative error of volume fraction ($e_V\%$). The green line corresponds to $e_C\% = 20\%$ and $e_V\% = 10\%$ in the $e_C\%$ and $e_V\%$ plots, respectively.

the aesthetic metrics of the generated designs and the skeletons' is minimal. The gap in the $SSIM$ is reduced to 0.03 versus 0.2 with the first variant of DL-AM-TO, which validates our assumption that the geometrical discriminators' performance highly influences the model's performance. Moreover, as we have mentioned earlier, the global geometry of the design tells a lot about the placement of the loads and boundary conditions; thus, having a high $SSIM$ implies the conformity of the designs with these two mechanical constraints (F and BC).

Figure 5.26 plots the distribution of the mechanical metrics, the relative errors of compliance³ and volume fraction, as defined in chapter 3 section 3.3.2; these values are computed over the generated designs without any transformation (raw) and after the application of the threshold. 44.4% of the generated designs show a compliance that is less or equal to 1.2 times the ground truth designs' compliance (i.e. $C_{generated} \leq 1.2 \times C_{ground-truth}$). This percentage is improved by 42% to achieve 63% after applying the threshold. This result is similar to what we obtained in chapter 3; we resolved the high compliance issue by adding a compliance discriminator. 89.5% of the generated designs show a $V_{generated} \leq 1.1 \times V_{input}$. However, after applying the threshold, the percentage decreases to 66.2%. To circumvent to this setback, a better approach would be to use a local customized threshold per design.

Lastly, we examine the geometrical conformity of DL-AM-TO to the geometrical input values. Figure 5.25 shows the distribution of $\Delta_X = X_{generated} - X_{input}$; $X \in len_{max}, th_{min}, \Theta_{min},$ and Nbr_{bars} . For the Nbr_{bars} constraint, the same metric used in section 5.3.5 and chapter 3 section 3.3.2 is used here to check the generated designs' conformity. If the absolute difference between the Nbr_{bars} of the generated design and the input Nbr_{bars} is less than 2, the generated design is considered compliant with the Nbr_{bars} constraint, such that the Nbr_{bars} is predicted via the regression-based Nbr_{bars} discriminator. For the remaining geometrical constraints, a generated design is geometrically compliant if its geometrical value is, at most, distant by \pm one class from the input value, such that the class value is predicted by the corresponding CORN geometrical discriminator. The distance between two consecutive classes of the len_{max} equals 0.05 units. The distance between two consecutive classes of the th_{min} equals 0.01 units. The distance between two consecutive classes of the Θ_{min} equals 5° . Thus, 96.3% of the generated designs are compliant with the len_{max} constraint, 89.3% with the th_{min} constraint, 87.9% with the Θ_{min} constraint, and 88.7% with the Nbr_{bars} constraint.

³The compliance was computed using the FE method.

To sum up, DL-AM-TO’s overall performance is promising. It integrates mechanical and geometrical manufacturing-related constraints at the same level and generates aesthetically plausible and mechanically and geometrically valid designs in a fraction of a second. However, to improve the design’s compliance, we could retrain DL-AM-TO with an additional compliance discriminator or run a few iterations of SIMP on the shape outputted by DL-AM-TO. The latter hybrid approach is the most efficient for two reasons. (1) Retraining DL-AM-TO is costly in time and computational power. (2) SIMP method’s setback makes it unable to modify a shape after it was identified; it can only optimize on it; this is beneficial in our case for a shape compliant with the geometrical-manufacturing constraints is already identified by DL-AM-TO. Nevertheless, this shape sometimes shows disconnected or blurry bars or intermediate density values, causing poor mechanical performance. Thus, applying a few iterations of SIMP could improve this mechanical performance. This conclusion is so essential for us to re-validate our assumption that DL will help compensate for the difficulties faced by robust mathematical FE-TO when it comes to integrating different natures of constraints but could not replace it, especially since, until now, we are not able to validate its outcome fully.

5.6.3 Tailoring a design’s geometry with DL-AM-TO

In this section, we conduct four experiments to test DL-AM-TO’s ability to tailor a design’s geometry while still compliant with the mechanical constraints. In each experiment, we fix all input constraints except one geometrical constraint. We will scan a range of values for this constraint and examine how the design’s geometry changes to adapt to these changes.

The key takeaways from section 5.5 is that the Nbr_{bars} constraint is the only constraint with an obvious correlation with the other geometrical constraints. We will check how DL-AM-TO generates designs to comply as much as possible with this constraint while being able to adjust the design’s geometry to comply with the other geometrical constraint. Adding more constraints should affect the outcome of the DL model; in section 5.5, the models were trained to control one variable while the others were free to change, here they are not anymore. We wait for one of the two phenomena to happen: one geometrical constraint will be predominant over the others, or we will see a lower number of designs proposed, i.e., a lower potential design space but conformity to all the constraints.

Figure 5.28 shows three sets of designs with increasing len_{max} . In figure 5.28(a), the Nbr_{bars} and the Θ_{min} are relatively constant, $7 \pm 1bar$ except for the fourth geometry and $31 \pm 6^\circ$ except for the sixth

geometry.

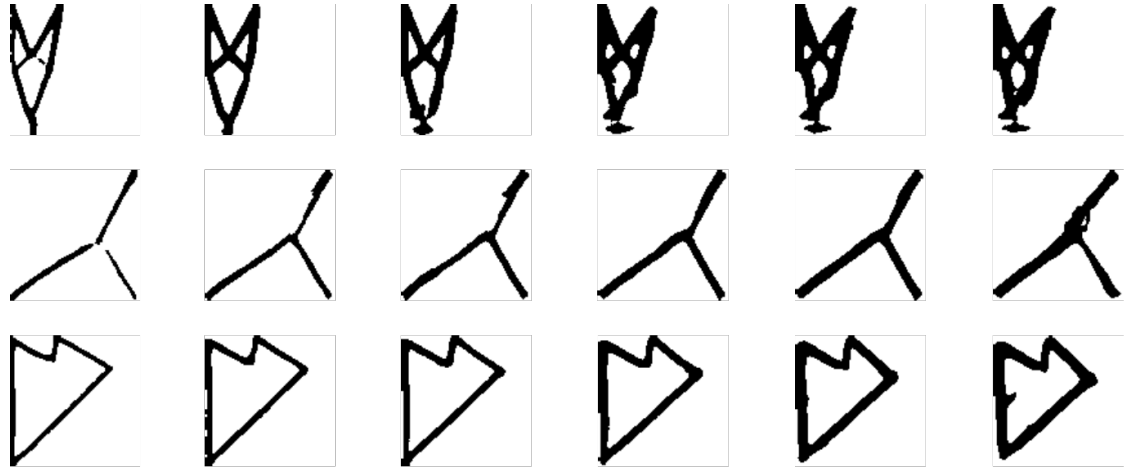
In figure 5.28(b), the Nbr_{bars} is higher for small len_{max} value, which is expected, to minimize the len_{max} , longer bars need to be intercepted by other bars, whereas Θ_{min} fluctuates between 0 and 24°. In figure 5.28(c), the Nbr_{bars} and the Θ_{min} are relatively constant, ± 1 bar except for the second geometry.

Thus, changing len_{max} is still inversely proportional to the Nbr_{bars} , however, since DL-AM-TO controls both variables simultaneously, the change in the Nbr_{bars} is modest when the geometry allows it unlike what we observed in section 5.5.2. Similarly, Θ_{min} is more or less constant versus the changing len_{max} . It is important to point out that the Nbr_{bars} and Θ_{min} radically change together (i.e., violate the general rule) and not separately. This validates partially the conclusion deduced in section 5.5.1; there is a relationship between Nbr_{bars} and Θ_{min} , however, it is not here an inverse proportionality relationship.

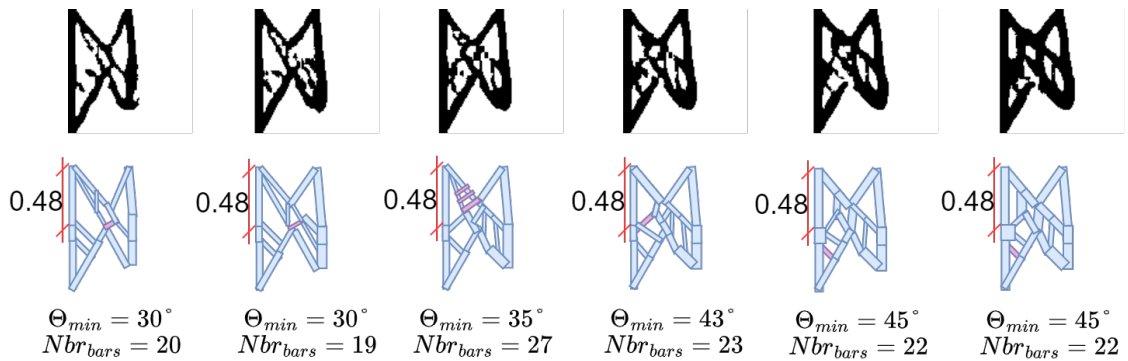
Figure 5.29 shows three sets of designs with increasing Θ_{min} . In figures 5.29(a) and 5.29(b), the Nbr_{bars} and the len_{max} are relatively constant, $Nbr_{bars_{input}} \pm 2$ bar except for the first geometry and $len_{max_{input}} \pm 0.05$ units. Nonetheless, the third set (Fig.5.29(b)) shows a lower conformity for the Nbr_{bars} and the len_{max} ; the difference between the input and the generated design's Nbr_{bars} varies between -3 and +8 bars and the len_{max} varies between -0.2 and +0.4.

Figure 5.30 shows DL-AM-TO's response to changing only the Nbr_{bars} variable. As mentioned in section 3.4.5, chapter 3, changing the Nbr_{bars} requires a changing in the volume fraction in order to ensure conformity; hence the V was adjusted with the increasing Nbr_{bars} . As we can clearly see increasing the Nbr_{bars} induces the decrease of the Θ_{min} and the len_{max} , with the len_{max} being the least impacted. In other terms, the Nbr_{bars} is the dominant geometrical variable. Changing it induces higher chances of non-conformity to the other geometrical variables. Nonetheless, it is the least relevant to AM. Thus, the engineer could sacrifice it on the advantage of being compliant with Θ_{min} , and len_{max} , which are core AM constraints.

Figure 5.27 shows the DL-AM-TO's response to changing the th_{min} . The same rule found in section 3.4.5, chapter 3 for Nbr_{bars} applies for the th_{min} ; thicker designs need more material i.e. a higher V and reciprocally; hence the V was adapted accordingly here too. As we can see, the bars thicken with the increasing value of th_{min} with a conformity with all the other constraints (Fig.5.27(a)), and for more complex geometries (i.e. Fig.5.27(b)) additional bars start appearing at a point so that after they



(a) Three set of designs, each subject to the same mechanical constraints, input Nbr_{bars} , input Θ_{min} , & input len_{max} with an increasing value of th_{min} .



(b) The generated designs are subject to the same mechanical constraints, input $Nbr_{bars} = 13$, input $\Theta_{min} = 33^\circ$, & input $len_{max} = 0.6u$ and an increasing value of th_{min} .

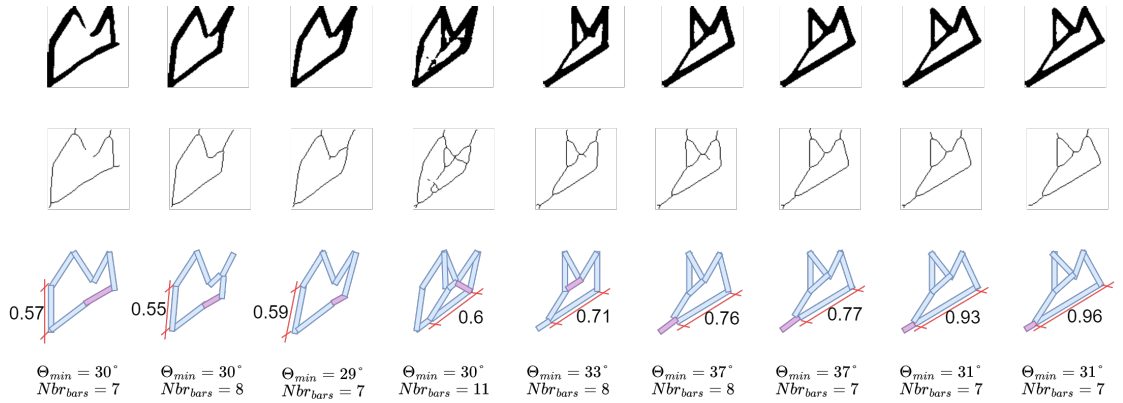
Figure 5.27: Tailoring th_{min} .

end up merging for high values of th_{min} , thus inducing a variation in the other geometrical variables.

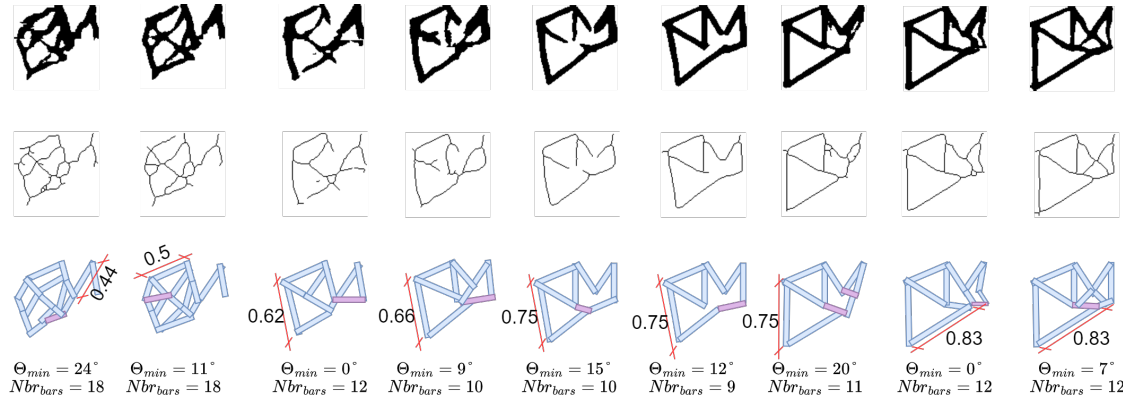
5.7 Printing designs generated by DL-AM-TO

Now that we have shown the capabilities of DL-AM-TO when tailoring a design's geometry to comply with input geometric manufacturing-related constraints, it is time to go to the printing phase. We will proceed by drawing the CAD of the design outputted by SIMP (i.e., the output geometry does not control any geometrical manufacturing constraints). Then, we generate several variations of this design with DL-AM-TO by changing one geometrical variable, and we draw them; FreeCAD[9], an open-source 3D modeling software, is used for drawing the CADs. These designs are then exported as

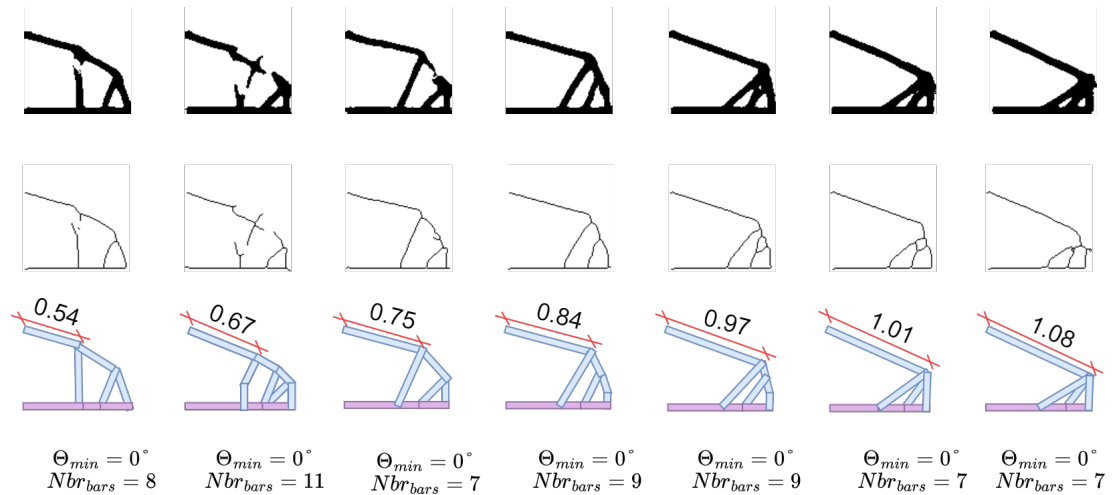
5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



(a) First set of designs. Input $Nbr_{bars} = 7$ & input $\Theta_{min} = 31^\circ$.



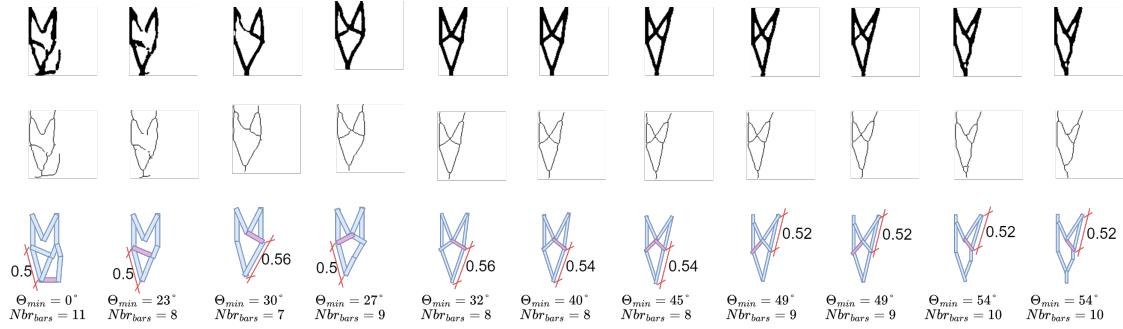
(b) Second set of designs. Input $Nbr_{bars} = 9$ & input $\Theta_{min} = 21^\circ$.



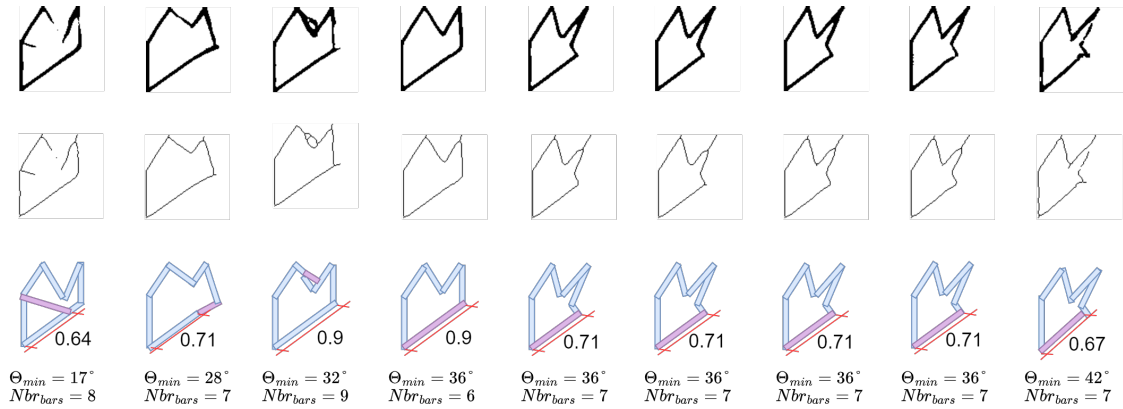
(c) Third set of designs. Input $Nbr_{bars} = 8$ & input $\Theta_{min} = 0^\circ$.

Figure 5.28: In this figure, we show the same mechanical constraints and changed the len_{max} .

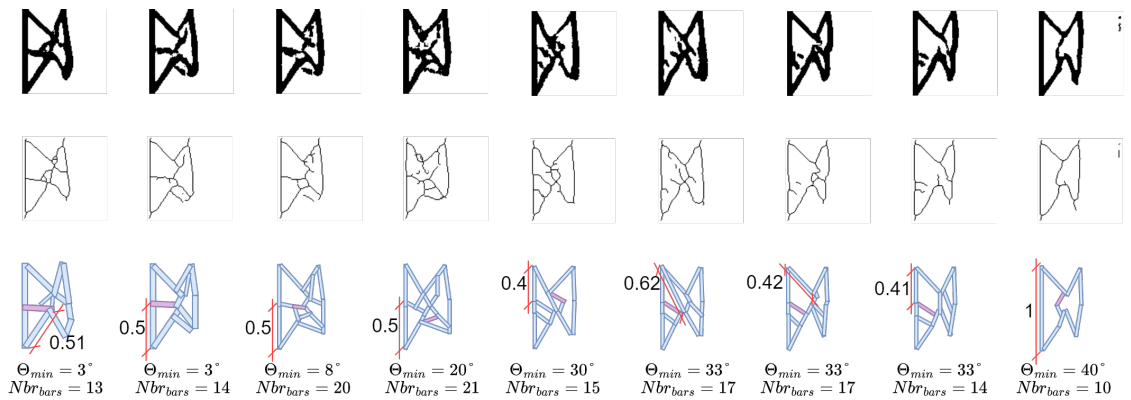
5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



(a) First set of designs. Input $Nbr_{bars} = 8$ & input $len_{max} = 0.51u$.



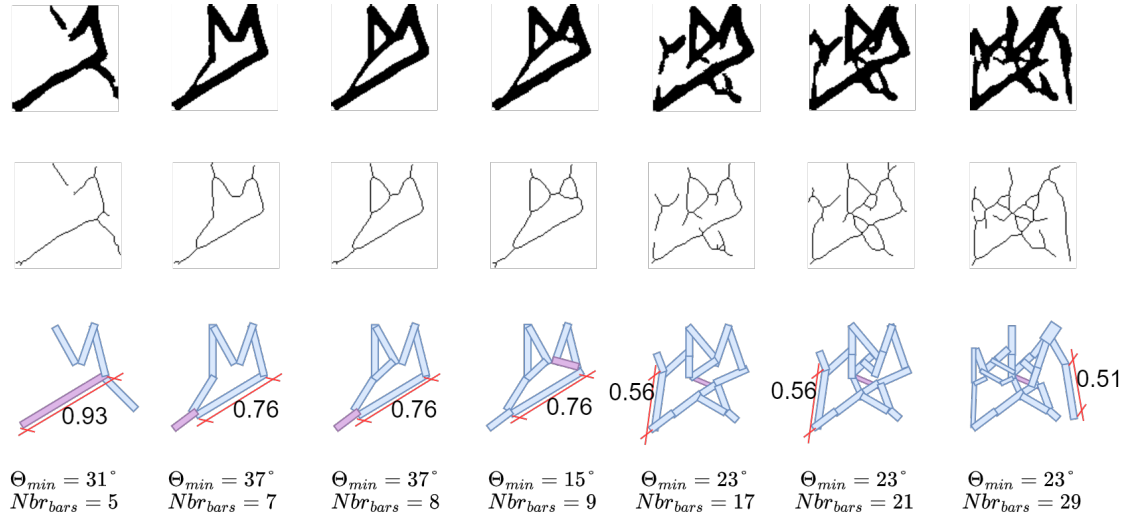
(b) Second set of designs. Input $Nbr_{bars} = 7$ & input $len_{max} = 0.7u$.



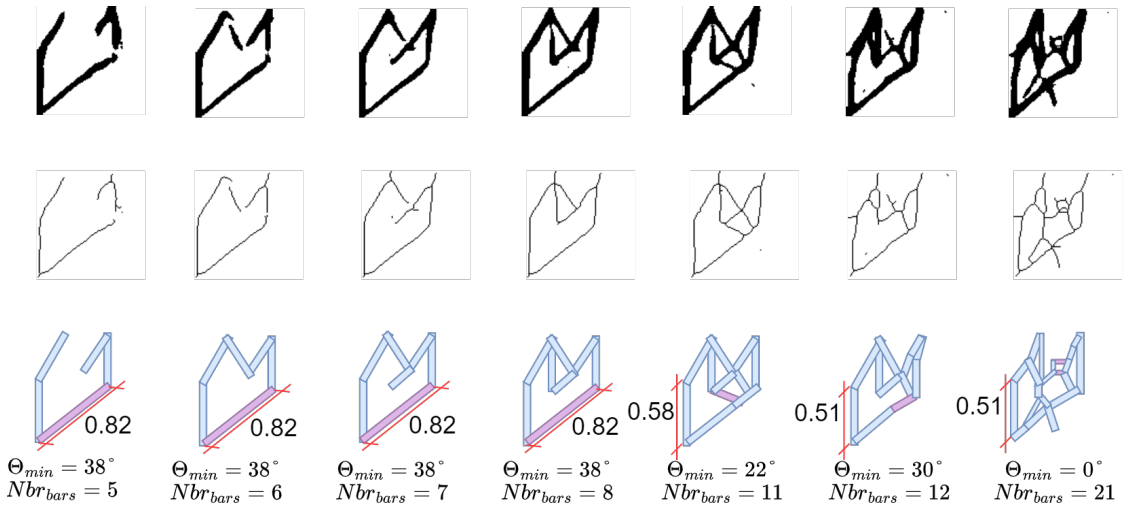
(c) Third set of designs. Input $Nbr_{bars} = 13$ & input $len_{max} = 0.6u$.

Figure 5.29: In this figure, we show the same mechanical constraints and changed the Θ_{min} .

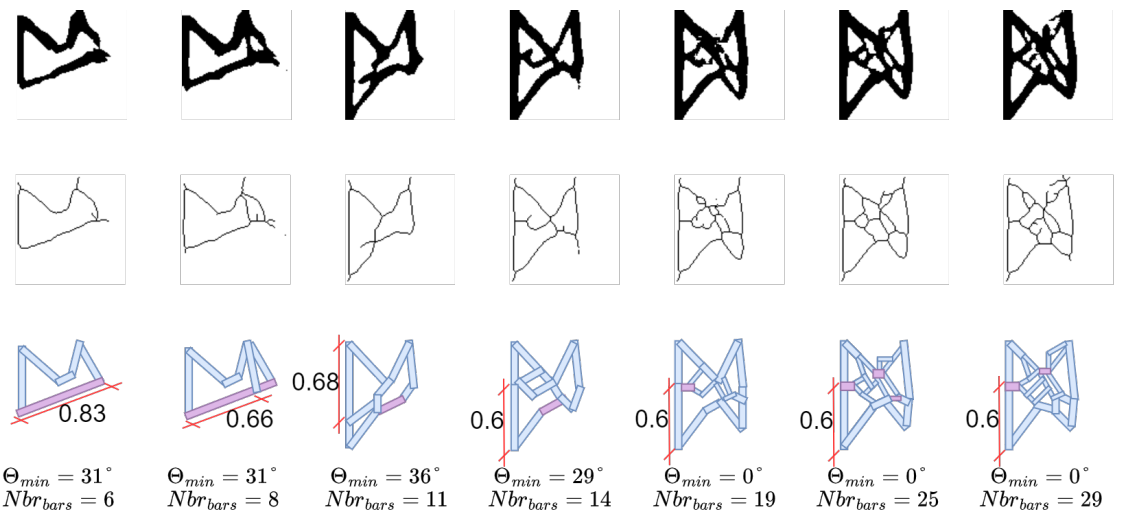
5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



(a) First set of designs. Input $\Theta_{min} = 31^\circ$ & input $len_{max} = 1.02u$.



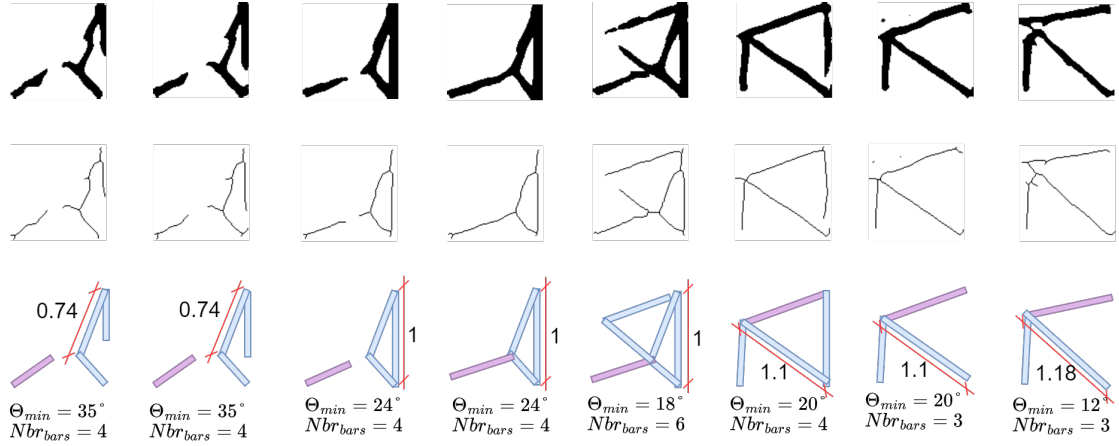
(b) Second set of designs. Input $\Theta_{min} = 36^\circ$ & input $len_{max} = 0.7u$.



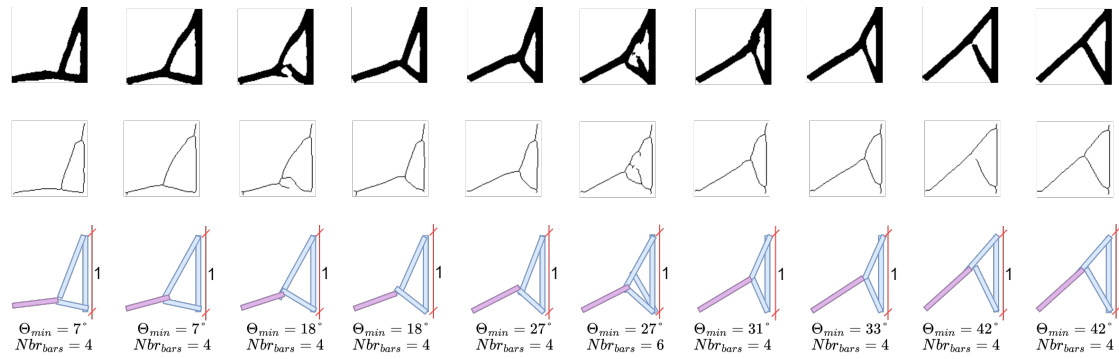
(c) Third set of designs. Input $\Theta_{min} = 24^\circ$ & input $len_{max} = 0.67u$.

Figure 5.30: In this figure, we show the same mechanical constraints and changed the Nbr_{bars} .

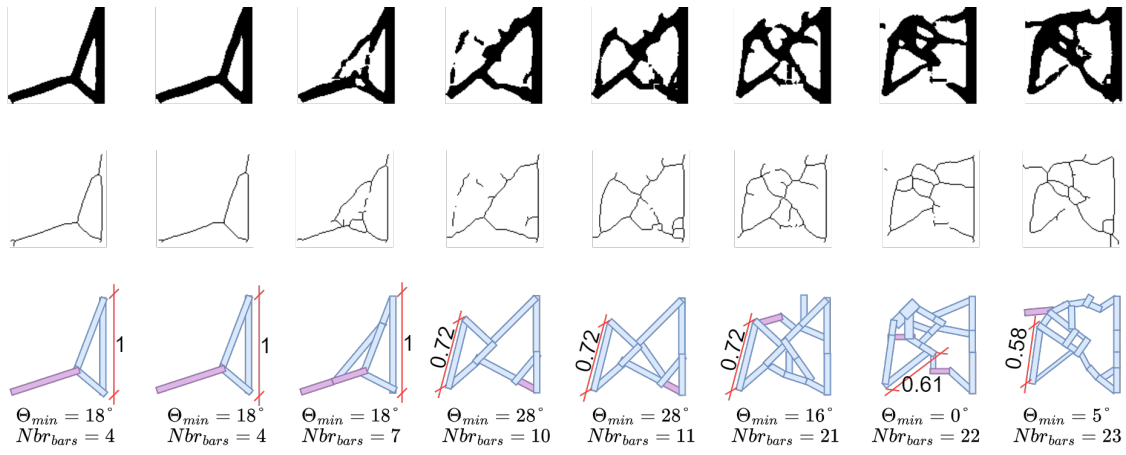
5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



(a) Changing len_{max} .



(b) Changing Θ_{min} .



(c) Changing Nbr_{bars} .

Figure 5.31: In this figure, we show the same mechanical constraints and changed three different geometrical constraints each at a time. Input $Nbr_{bars} = 5$, input $len_{max} = 1u$, & input $\Theta_{min} = 18^\circ$.

.stl files and analyzed by Ultimaker Cura software [10]; an open-source slicing application for 3D printers. The latter takes the .stl design and the 3D printer parameters (the printer's type, its $\Theta_{minprinter}$, which is $\pi/2 - \Theta_{min}$ defined in our work, the percentage of infill, the build plate adhesion, Etc.), adds supports to the structure if needed, and outputs an estimation of the material needed's build time (B_T), mass M (in grams g), and length L (in meters m). Finally, it generates a .gcode file containing the building steps to be accomplished by the 3D printer, a Creality Ender 3⁴, to build the design. The mechanical performance, represented by the maximum displacement (u_{max}) in mm is computed with Patran[135] Nastran[136].

We will study the geometrical variables via their increasing order of importance in the manufacturing phase; thus, start with the first geometrical variable we have handled, the Nbr_{bars} , then, by the len_{max} , and finally to end with the most influential geometrical-manufacturing constraint, the Θ_{min} . The objective is to validate DL-AM-TO's contribution. Acceleration forwarded by DL-AM-TO. DL-AM-TO allows the generation of several geometries for the same mechanical constraints to comply with the manufacturing criteria. With DL-AM-TO, the engineer's search for the best candidate design that complies with all the required specifications (mechanical and manufacturing) is accelerated; (s)he will not need to get stuck in a loop of updating the geometry and analyzing it. Furthermore, with an adequate geometry, the material usage, the supports, the post-processing, and hence the build cost and time are reduced.

5.7.1 Printing geometries of different Nbr_{bars} constraint

For this experiment, we have chosen the second, the fifth, and the eighth geometries in figure 5.30(a). The input Θ_{min} is 31° . Thus, the $\Theta_{minprinter}$ is set to 59° . The geometries' Nbr_{bars} are 7, 17, and 29 respectively.

Looking at the the DL-based designs (where the Nbr_{bars} can be changed) without supports (figure 5.32(b)), we notice that adding 10 bars increases the material needed by 60% (5g to 8g), the printing time by 25% (72 minutes to 90 minutes); the second and third design in figure 5.32(b)). However, adding 22 bars costs an increase of the material needed by 80% and the build time of 46%; the second and fourth designs in figure 5.32(b)). It is important to note that the designs in the figure are not

⁴https://www.creality3dofficial.eu/products/ender-3s1-pro-3d-printer-eu?gclid=Cj0KCQjw1bqZBhDXARIsANTjCPL-wLdAh-KYxcGhegr5sp0-_YWukzzqWy9Qfv5HPRVkf9Y0E7AdCEaAs54EALw_wcB

printable without supports; the printer's nozzle did not succeed in sticking the material filament to the build plate; the filaments got tangled, and the printing failed. This brings us to figure 5.32(c) where supports were added. As we can see, the support structures for the geometry with 7 bars constitutes about 50% of the material printed (the support material between the second design in Fig.5.32(b) and the second design in Fig.5.32(c) = $10 - 5 = 5g$) and increases the build time by 54%. For the geometry with 17 bars, the supports consist of 50% of the total material printed, and the build time increase totals to 48%. For the geometry with 29 bars, the supports structure is the least (22% of the material printed), and the build time increases by 21%.

It is interesting to compare the DL-based geometries of Fig 5.32(c) after the addition of support structures. As a matter of fact, adding 22 bars has increased the material by only 9% (10g to 11g) and the printing time by 15.3% (111 minutes to 128 minutes). In other terms, if the end part can tolerate more material (for the additional bars) and its surface must be preserved from the post-processing that supports needed, increasing the Nbr_{bars} is the easiest solution.

Mechanically, designs with additional bars are more resilient; the maximum displacement of the DL-based design with 29 bars ($u_{max} = 0.565mm$) is ten times less than the DL-based one with 7 bars ($u_{max} = 5.68mm$). More importantly, the DL-based designs where additional bars were added (the third and fourth design) exhibit a better mechanical performance than the one proposed by SIMP.

In conclusion, increasing the Nbr_{bars} decrease the need for support; the added bars play the role of support and enhance the mechanical resilience, with a slight increase in the build time. For parts where the support structure can damage the surface and the functionality, increasing the Nbr_{bars} is a good choice for the increase in build time, and material is admissible (around 10%).

DL-AM-TO allows changing the Nbr_{bars} in geometry, a task hardly feasible with SIMP, and proposes geometries that are mechanically resilient; their maximum displacement is of the same order of magnitude as those outputted by SIMP.

5.7.2 Printing geometries of different len_{max} constraint

For this experiment, we have chosen the first, third, and seventh geometries in figure 5.28(c). The input Θ_{min} is 0° . However, the $\Theta_{minprinter}$ is set to 45° , for if set to 90° , no supports are added by Cura; hence, we could not see the influence of long bars on the acceleration of printing phase. The DL-based geometries' len_{max} are $0.54u$, $0.75u$, and $1.08u$ respectively.

5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO

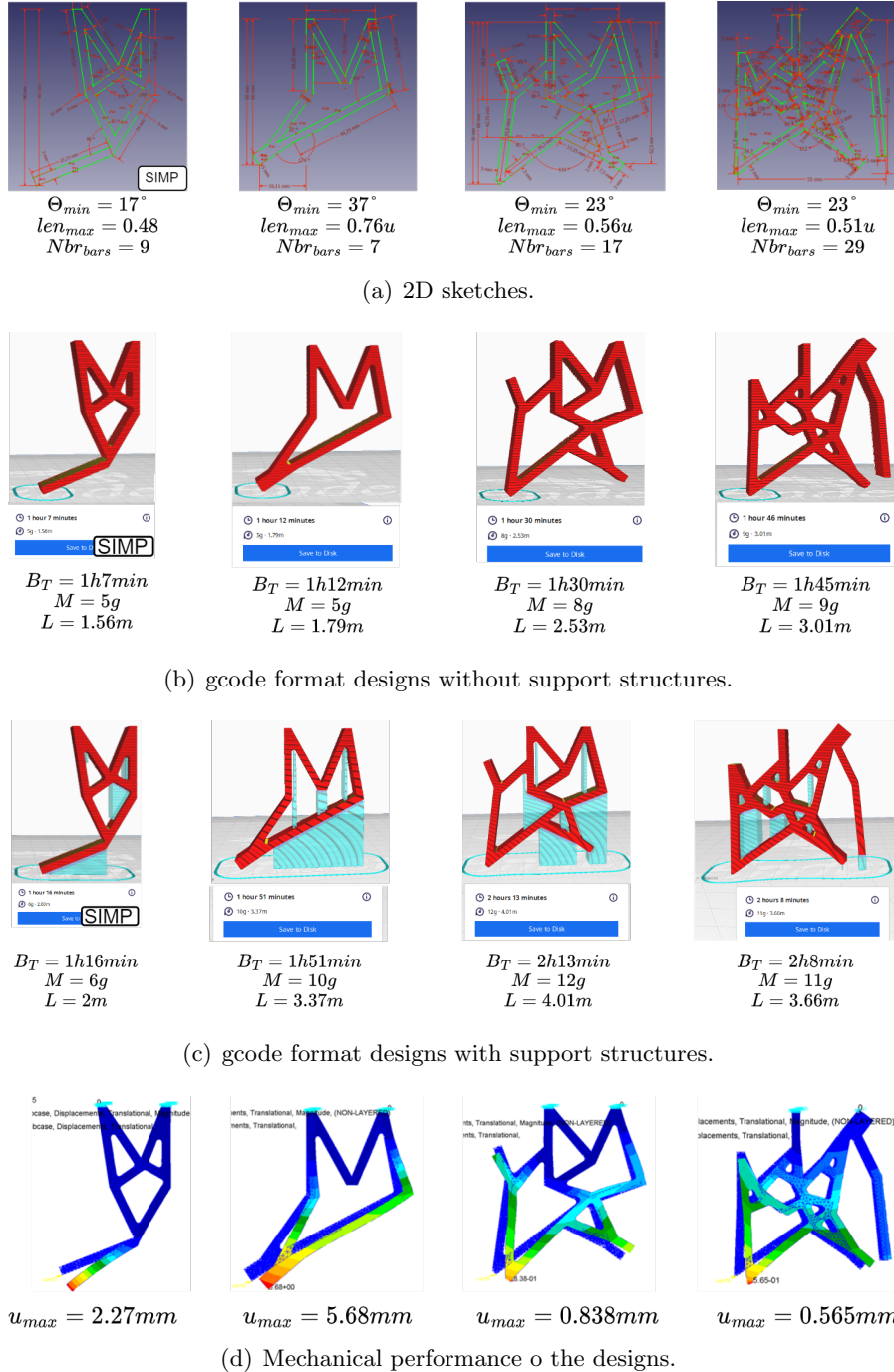


Figure 5.32: In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (without and with the supports) and the estimated design’s mass and material filament’s length needed, outputted by Cura[10], and finally the printed end-design. The designs’ Nbr_{bars} are 7, 17, and 29, from left to right, respectively. The $\Theta_{minprinter}$ is set to 59° for which support structures are added.

Figures 5.33(b) and 5.33(c) show that increasing the len_{max} does not impact the the material needed nor the build time (4% increase in build time). For this type of structure, DL-based designs and the SIMP design are similar geometrically; the manufacturing time is almost similar, and mechanically; the maximum displacement of DL-AM-TO's designs are of the same order of magnitude as the SIMP design.

5.7.3 Printing geometries of different Θ_{min} constraint

Θ_{min} is the most influential constraint in manufacturing. As mentioned in section 2.1 in chapter 2, a design violating this constraint has hanging features and hence needs support structures. The latter yields additional material, slows the build time, damages the part's accuracy and finishing, and induces the need for post-processing, further delaying the fabrication [40]. Consequently, we dedicated two experiments to this constraint.

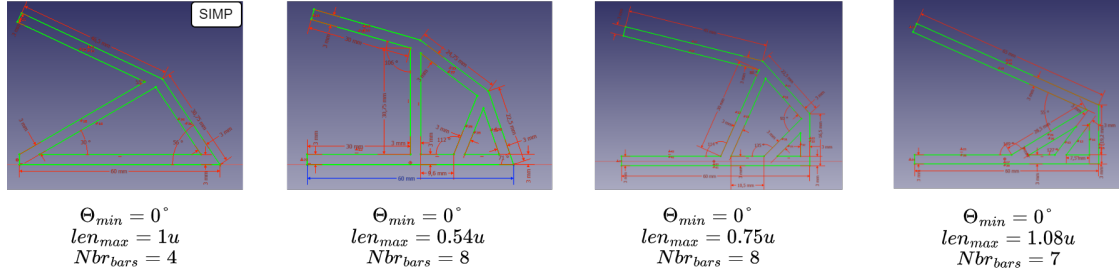
For the first experiment, we have chosen the first, third, seventh, and eleventh geometries in figure 5.29(a). The $\Theta_{minprinter}$ is set to 45° . The geometries' Θ_{min} are 0° , 30° , 45° , and 54° .

Comparing DL-based designs in figure 5.34(b) to the ones in figure 5.34(c) shows the decrease in the need for support structures when the geometry's Θ_{min} constraint better complies with the printer's minimum overhang; the support structure's mass decreases from 50% ($\frac{9g-6g}{6g} \times 100 = 50\%$, the second design in figures 5.34(b) and 5.34(c)) to 8% ($\frac{5g-4g}{4g} \times 100 = 25\%$, the last design in figures 5.34(b) and 5.34(c)), which induces the decrease in the build time by 45.6% ($\frac{1h39min-1h8min}{1h8min} \times 100 = 45.6\%$, the second and last design in figure 5.34(c)).

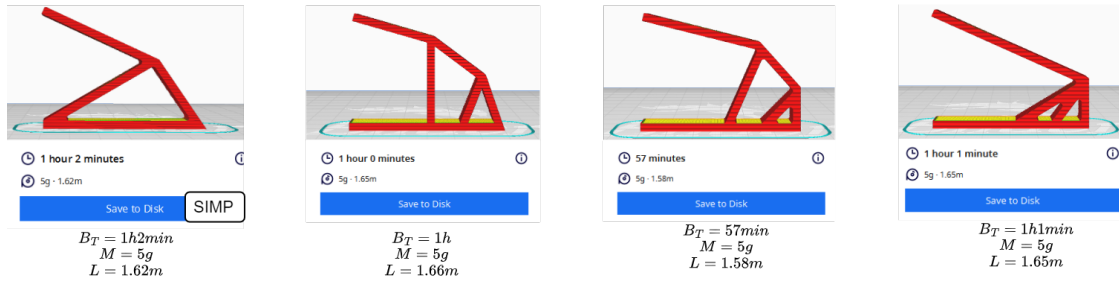
From a mechanical point of view, DL-AM-TO generates designs with the same order of magnitude when it comes to u_{max} (figure 5.34(d)) and allows the geometry's modification to comply with the manufacturing constraint. As a matter of fact, the best design, geometrically and mechanically, is the fourth design in figure 5.34(d) that was generated by DL-AM-TO; it shows the same u_{max} as the SIMP one, however, it is $1.57\times$ lighter ($7g$ versus $11g$), and hence $1.57\times$ cheaper, and is printed $1.3\times$ faster ($1h18min$ versus $1h54min$).

For the second Θ_{min} experiment, we have chosen the first, fifth, and tenth geometries in figure 5.31(b). As we can clearly see, DL-AM-TO proposes a different shape from SIMP but that is mechanically similar to the one suggested by SIMP (figure 5.35(d)). More importantly, in addition of being compliant

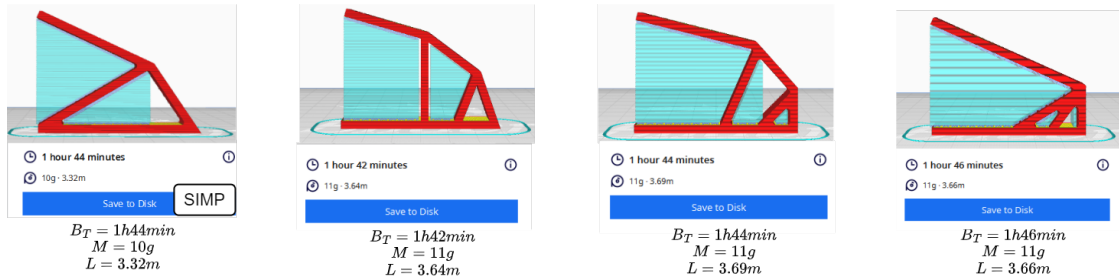
5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



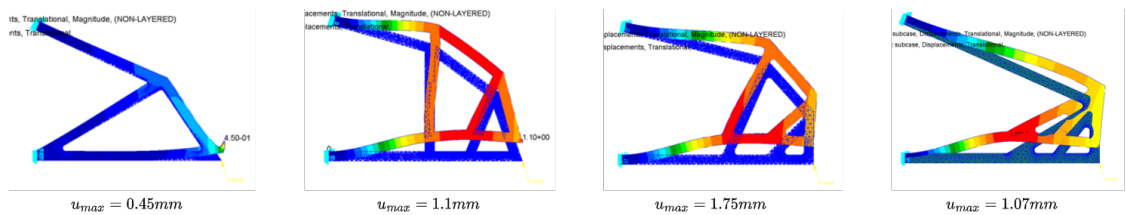
(a) 2D sketches.



(b) gcode format designs without support structures.



(c) gcode format designs with support structures.



(d) Mechanical performance of the designs.

Figure 5.33: In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (without and with the supports) and the estimated design's mass and material filament's length needed, outputted by Cura[10], the displacements computed by Patran-Nastran in mm , and finally the printed end-design. The designs' len_{max} are $0.54u$, $0.75u$, and $1.08u$, from left to right, respectively. The $\Theta_{minprinter}$ is set to 45° .

with the Θ_{min} constraint, DL-AM-TO's designs are cheaper; less material due to the need of less support material (13g for the SIMP design versus 9g for the fourth DL-based design, figure 5.35(c)); and can be printed faster; (1h59min for the SIMP design versus 1h30min for the fourth DL-based design, figure 5.35(c)).

In conclusion, DL-AM-TO tailors the design's geometry to comply with the Θ_{min} constraint while keeping a decent mechanical performance. On average, it accelerates the printing phase by 1.4× and saves about 40% in material and costs. It is essential to highlight that with SIMP, modifying the geometry is hardly feasible; SIMP does not allow the proposition of several geometries by a simple change of the input Θ_{min} constraint.

5.7.4 DL-AM-TO accelerates the DfAM process and is a lighter module in industrial design software

In this section, we compare the traditional DfAM process via the one forwarded by DL-AM-TO. For this comparison, we will consider the DfAM process as the following: the design phase, the CAD drawing phase, the FEA phase, and the printing phase; in other terms, a DfAM process needs $t_{design} + t_{CAD} + t_{FEA} + t_{print}$ time.

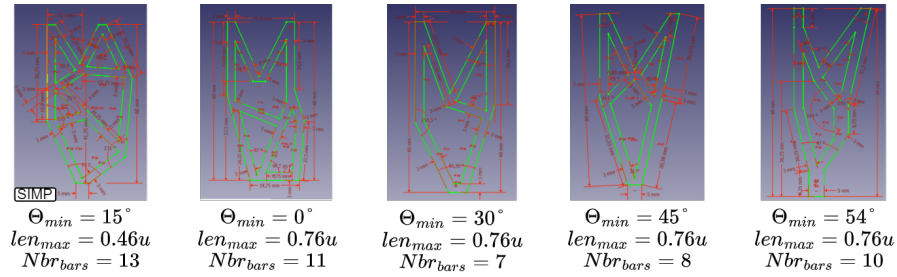
The DfAM process with our approach, DL-AM-TO, differs from the traditional process in the design phase, which impacts the printing phase. Consequently, the comparison can be reduced to these two phases, i.e., t_{DfAM} can be reduced to $t_{design} + t_{print}$ for the sake of the comparison. As computed in chapter 3, DL-AM-TO is 3500× faster than SIMP (Tab.3.2) and is independent of the input's complexity; Tab.3.3 shows that DL-AM-TO becomes 32800× ($656s/0.02s = 32800$) faster than SIMP and decreases the computational costs by 27225% ($\frac{620.28-2.27}{2.27} \times 100$) when the the number of loads increases to 10.

Furthermore, as demonstrated in section 5.7.3, DL-AM-TO's accelerates the printing phase up to 1.4×, for the geometry proposed by DL-AM-TO can account for the Θ_{min} manufacturing constraint while this is hardly feasible by SIMP.

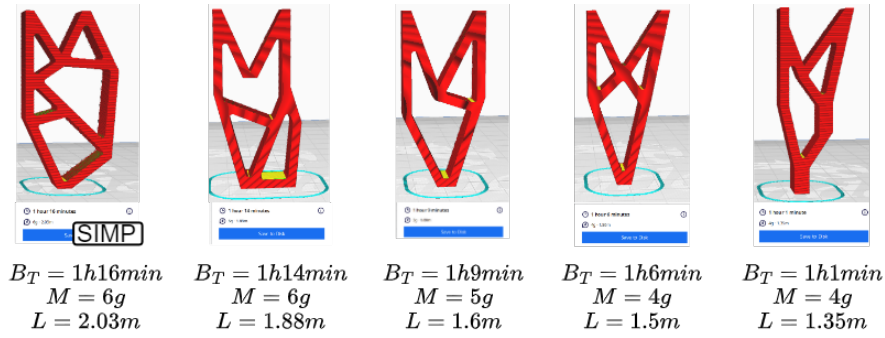
Finally, the DfAM process with DL-AM-TO needs $\frac{t_{design}}{3500} + \frac{t_{print}}{1.4}$ with t_{design} and t_{print} being the time needed in the design (SIMP here) and printing phase of a traditional DfAM process.

In our case simplistic mechanical conditions were considered thus t_{design} varied between 140s i.e. 2min to 620s i.e. 10min and the size of the parts was chosen for a faster printing phase; from figures 5.34(c) and 5.35(c) t_{print} , including the supports, is about 1h56min on average. Thus, t_{DfAM} can be further

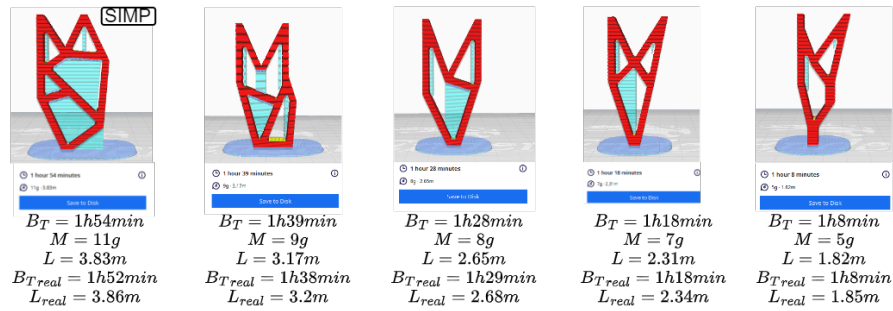
5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



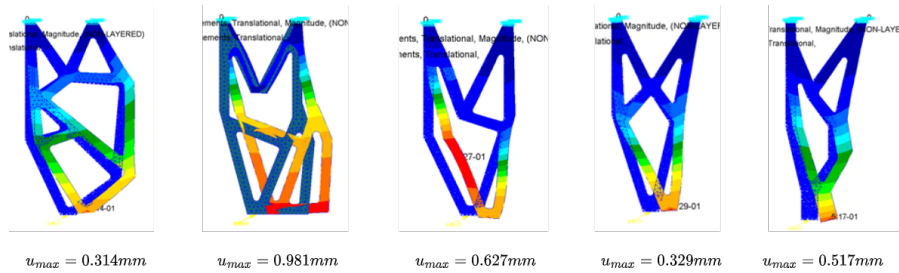
(a) 2D sketches.



(b) gcode format designs without support structures.



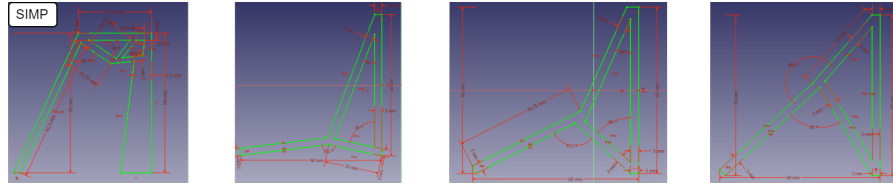
(c) gcode format designs with support structures.



(d) Mechanical performance of the designs.

Figure 5.34: In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (without and with the supports) and the estimated design's build time B_T , mass M , and material filament's length needed L , outputted by Cura[10]; $B_{T_{real}}$ and L_{real} are the build time and filament length measured in the printing phase (figure 5.36), the mechanical performance illustrated with the displacements in mm outputted by Patran-Nastran, and finally the printed end-design. The designs' Θ_{min} are 0° , 30° , 45° , and 54° , from left to right, respectively. The $\Theta_{minprinter}$ is set to 45° .

5.7. PRINTING DESIGNS GENERATED BY DL-AM-TO



$$\Theta_{min} = 6^\circ$$

$$len_{max} = 1.025$$

$$Nbr_{bars} = 8$$

$$\Theta_{min} = 7^\circ$$

$$len_{max} = 1$$

$$Nbr_{bars} = 4$$

$$\Theta_{min} = 27^\circ$$

$$len_{max} = 1$$

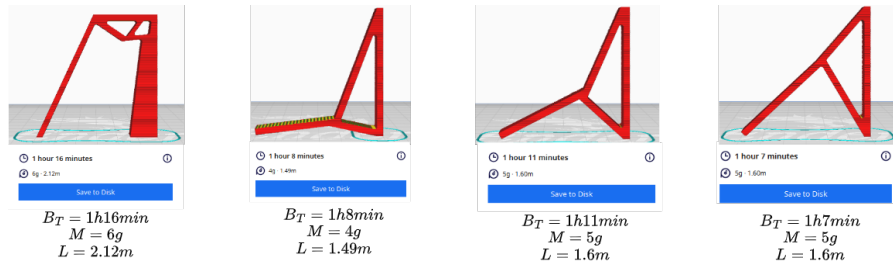
$$Nbr_{bars} = 4$$

$$\Theta_{min} = 42^\circ$$

$$len_{max} = 1$$

$$Nbr_{bars} = 4$$

(a) 2D sketches.



$$B_T = 1h16min$$

$$M = 6g$$

$$L = 2.12m$$

$$B_T = 1h8min$$

$$M = 4g$$

$$L = 1.49m$$

$$B_T = 1h11min$$

$$M = 5g$$

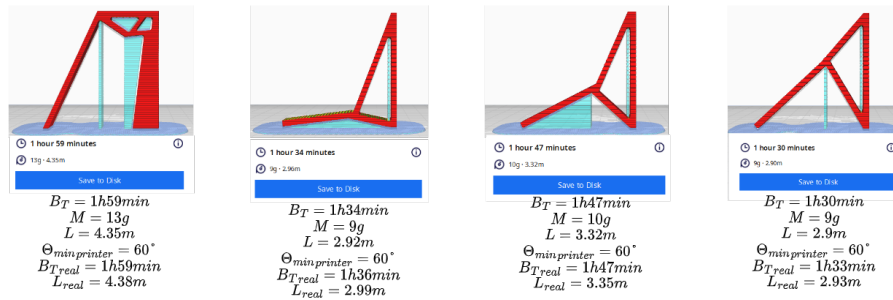
$$L = 1.6m$$

$$B_T = 1h7min$$

$$M = 5g$$

$$L = 1.6m$$

(b) gcode format designs without support structures.



$$B_T = 1h59min$$

$$M = 13g$$

$$L = 4.35m$$

$$\Theta_{minprinter} = 60^\circ$$

$$B_{Treal} = 1h59min$$

$$L_{real} = 4.38m$$

$$B_T = 1h34min$$

$$M = 9g$$

$$L = 2.92m$$

$$\Theta_{minprinter} = 60^\circ$$

$$B_{Treal} = 1h36min$$

$$L_{real} = 2.99m$$

$$B_T = 1h47min$$

$$M = 10g$$

$$L = 3.32m$$

$$\Theta_{minprinter} = 60^\circ$$

$$B_{Treal} = 1h47min$$

$$L_{real} = 3.35m$$

$$B_T = 1h30min$$

$$M = 9g$$

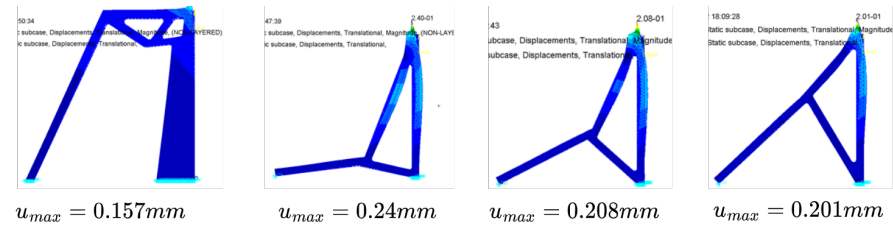
$$L = 2.9m$$

$$\Theta_{minprinter} = 60^\circ$$

$$B_{Treal} = 1h33min$$

$$L_{real} = 2.93m$$

(c) gcode format designs with support structures.



$$u_{max} = 0.157mm$$

$$u_{max} = 0.24mm$$

$$u_{max} = 0.208mm$$

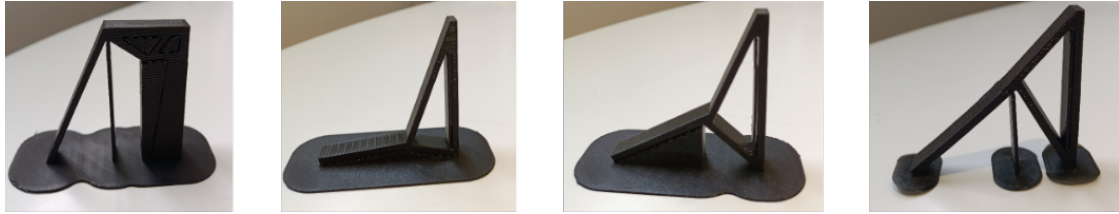
$$u_{max} = 0.201mm$$

(d) Mechanical performance of the designs.

Figure 5.35: In this figure, we show the 2D sketches drawn in FreeCAD[9], the gcode format design (with the supports) and the estimated design's build time B_T , mass M , and material filament's length needed L , outputted by Cura[10] with $\Theta_{minprinter} = 60^\circ$; B_{Treal} and L_{real} are the build time and filament length measured in the printing phase (figure 5.36), and finally the mechanical performance illustrated with the displacements in mm outputted by Patran-Nasteran.



(a) Designs of figure 5.34 printed.



(b) Designs of figure 5.35 printed.

Figure 5.36: Designs of figures 5.34 and 5.35 printed by the 3D printer Creality Ender 3.

reduced to the printing phase. Furthermore, since DL-AM-TO provides a draft design that already complies with manufacturing criteria, it reduces the number of trials the designer has to test and avoids him/her from getting stuck in a loop of n iterations. Hence, DL-AM-TO accelerates the DfAM process by up to $1.4n$ times with $n \geq 1$.

Furthermore, DL-AM-TO is computationally cheaper than SIMP. DL-AM-TO costs for any input's complexity $1.97GFLOPs$ (Giga floating point operations, figure 5.37), while SIMP's $GFLOPs$ vary from 68 to 656 (Tab. 3.3).

These results come to validate the hypotheses 2 and 3. Indeed, DL-AM-TO accelerates the DfAM process by integrating both mechanical and geometrical manufacturing-related constraints at the same level. Furthermore, its constant computational cost, in terms of storage and floating operations, makes it advantageous over FE-based design methods; it could be implemented into industrial design software as a lighter generative design module.

5.8 Discussion

To summarize, in this chapter, we demonstrate the effectiveness of our approach, DL-AM-TO, and validate the hypotheses listed in chapter 1. DL-AM-TO is a data-driven TO method that integrates the mechanical and geometrical manufacturing-related constraints concurrently at the same level.

5.8. DISCUSSION

Layer	Type	Output Shape	Param #			
resnet_12	Resnet_12	(-1, 512, 1, 1)	0			
├b1	First_residual_conv	(-1, 32, 50, 50)	0			
	├downsample	Conv2d	(-1, 32, 50, 50)	2,592		
		├residual_block	Sequential	(-1, 32, 50, 50)	18,561	
			├identity_map	Sequential	(-1, 32, 50, 50)	1,153
├b2	Residual_convolution	(-1, 64, 25, 25)	0			
	├downsample	Conv2d	(-1, 64, 25, 25)	32,768		
		├residual_block	Sequential	(-1, 64, 25, 25)	74,242	
			├identity_map	Sequential	(-1, 64, 25, 25)	4,353
├b3	Residual_convolution	(-1, 128, 13, 13)	0			
	├downsample	Conv2d	(-1, 128, 13, 13)	73,728		
		├residual_block	Sequential	(-1, 128, 13, 13)	295,938	
			├identity_map	Sequential	(-1, 128, 13, 13)	16,897
├b4	Residual_convolution	(-1, 256, 7, 7)	0			
	├downsample	Conv2d	(-1, 256, 7, 7)	294,912		
		├residual_block	Sequential	(-1, 256, 7, 7)	1,181,698	
			├identity_map	Sequential	(-1, 256, 7, 7)	66,561
├b5	Residual_convolution	(-1, 512, 4, 4)	0			
	├downsample	Conv2d	(-1, 512, 4, 4)	1,179,648		
		├residual_block	Sequential	(-1, 512, 4, 4)	4,722,690	
			├identity_map	Sequential	(-1, 512, 4, 4)	264,193
├bridge	Residual_convolution	(-1, 512, 1, 1)	0			
	├downsample	Conv2d	(-1, 512, 1, 1)	4,194,304		
		├residual_block	Sequential	(-1, 512, 1, 1)	4,722,690	
			├identity_map	Sequential	(-1, 512, 1, 1)	264,193
├b6_prime	Residual_Transpose_c	(-1, 512, 4, 4)	0			
	├upsample	ConvTranspose2d	(-1, 512, 4, 4)	4,194,304		
		├residual_block	Sequential	(-1, 512, 4, 4)	4,722,690	
			├identity_map	Sequential	(-1, 512, 4, 4)	264,193
├b6	Residual_Transpose_c	(-1, 256, 7, 7)	0			
	├upsample	ConvTranspose2d	(-1, 256, 7, 7)	1,179,648		
		├residual_block	Sequential	(-1, 256, 7, 7)	1,181,698	
			├identity_map	Sequential	(-1, 256, 7, 7)	66,561
├b7	Residual_Transpose_c	(-1, 128, 13, 13)	0			
	├upsample	ConvTranspose2d	(-1, 128, 13, 13)	294,912		
		├residual_block	Sequential	(-1, 128, 13, 13)	295,938	
			├identity_map	Sequential	(-1, 128, 13, 13)	16,897
├b8	Residual_Transpose_c	(-1, 64, 25, 25)	0			
	├upsample	ConvTranspose2d	(-1, 64, 25, 25)	73,728		
		├residual_block	Sequential	(-1, 64, 25, 25)	74,242	
			├identity_map	Sequential	(-1, 64, 25, 25)	4,353
├b9	Residual_Transpose_c	(-1, 32, 50, 50)	0			
	├upsample	ConvTranspose2d	(-1, 32, 50, 50)	32,768		
		├residual_block	Sequential	(-1, 32, 50, 50)	18,690	
			├identity_map	Sequential	(-1, 32, 50, 50)	1,153
├b10	Residual_PixelSuffle	(-1, 1, 100, 100)	0			
	├0	Residual_PixelSuffle	(-1, 8, 100, 100)	1,891		
	├1	Conv2d	(-1, 1, 100, 100)	8		

Trainable params: 29,819,800						
Non-trainable params: 0						
Total params: 29,819,800						

Model size (params + buffers): 113.81 Mb						
Framework & CUDA overhead: 2397.38 Mb						
Total RAM usage: 2511.19 Mb						

Floating Point Operations on forward: 1.97 GFLOPs						
Multiply-Accumulations on forward: 986.13 MMACs						
Direct memory accesses on forward: 1.02 GDMAs						

Figure 5.37: DL-AM-TO’s architecture and computational cost detailed. This figure shows the number of DL-AM-TO’s trainable parameters, their memory usage in Mega bytes (*Mb*), their number of floating point operations (*GFLOPs*, giga *FLOPs*), their Multiply-Accumulate operations (*MMAC*, mega *MACs*), and Direct Memory Access operations (*GDMAs*, giga *DMAs*) on the model’s forward pass. This information is outputted by <https://pypi.org/project/torchscan/>, a python torch library to summarize a torch model’s computational cost.

DL-AM-TO was trained via a five-discriminator GAN framework, one traditional discriminator, and four geometrical discriminators. The latter were validated via uncertainty quantification methods to ensure that DL-AM-TO is trained and penalized correctly.

DL-AM-TO creatively tailors the design's geometry to comply with the manufacturing criteria without deteriorating its mechanical performance. It accelerates the DfAM process by suggesting to the engineer a design compliant with the input constraints, particularly the manufacturing ones, which prevents the designer from getting stuck in later phases of the DfAM process, the CAD drawing, and mechanical testing of the design. It accelerates the printing phase, for the part needs fewer support structures. Four experiments were carried out to validate DL-AM-TO's designs; the designs were drawn by FreeCAD, mechanically tested with Patran-Nastran, and printed with Cura and the 3D printer Creality Ender 3.

DL-AM-TO validates the three hypotheses proposed in chapter 1.

AM constraints that can hardly be analytically formulated like Nbr_{bars} , len_{max} , and more importantly, Θ_{min} were integrated into DL-AM-TO at the same level as the mechanical constraints (hypothesis 1). DL-AM-TO accelerates the whole DfAM process and not only the design phase by generating a first design draft complying with geometrical-manufacturing and mechanical constraints and avoiding repetitive iterations of design updating and evaluating (hypothesis 2).

Finally, DL-AM-TO's computational advantages make a lighter generative design module to be implemented in industrial design software in the future (hypothesis 3).

In conclusion, our approach benefits from the best of the AM-FE-TO and the DL-TO approaches. It compensates for the difficulties faced in AM-FE-TO when integrating informal manufacturing constraints into the design phase with DL techniques, particularly CNN. Also, convergence is better served for DL-AM-TO was trained on converged designs. It is mesh-independent, scalable, and fast, which qualifies it as a light, generative design module to be implemented in industrial design software. Finally, it accelerates the whole DfAM process and not only the design phase (Tab.5.2).

5.8. DISCUSSION

Advantages	Approaches to accelerate the DfAM process				
	Formulating AM design rules	Integration of AM constraints to FE-TO	ML and DL-assisted FE-TO	Full DL-TO	Our approach
Speed	+	+	++	+++	+++
Scalability			+	++	++
Convergence		-	+	++	++
Mesh-Independency	-	-	+	++	++
Geometrical control at the conceptual level		+			++
Light generative design module in industrial software		-	+	++	++
DfAM process acceleration	+	++	+	++	+++

Table 5.2: Our approach versus the state of the art. A blank cell means that the concerned characteristic does not apply for the method. A “+” sign defines the presence of the characteristic; the higher the number of the “+” sign, the better the method is when it comes to this characteristic. A “-” sign defines the absence of the characteristic.

5.9 French summary

Avec la création de GMCAD, nous avons pu explorer l’approche DL-AM-TO détaillée dans la section 2.1. Son objectif est de compenser les difficultés rencontrées par les méthodes de TO basées sur les éléments finis quand à l’intégration des contraintes géométriques (la minimum épaisseur de barre (th_{min}), la maximum longueur (len_{max}), le minimum surplomb (Θ_{min}), le nombre de barres (Nbr_{bars})) liées à la fabrication et celles mécaniques (les conditions aux bords BC , les forces F , et la fraction volumique V) au même niveau conceptuel. Suivant l’approche DL-TO développée dans le chapitre 3, DL-AM-TO est entraînée dans le cadre d’un GAN composé d’un générateur (DL-AM-TO) et de cinq discriminateurs : le discriminateur contradictoire traditionnel et quatre discriminateurs géométriques, un compteur de barres, des prédicteurs th_{min} , len_{max} et Θ_{min} (Fig.5.1). L’architecture de DL-AM-TO est basée sur les Resnet [104] pour des raisons de performance, à la différence de DL-TO dont l’architecture était basée sur les ResUnet [6]. Les trois nouveaux discriminateurs sont des DL modèles de régression dont l’architecture est basée sur Inception v4 [85]; ils prennent un design 2D en entrée et prédisent une valeur scalaire en sortie correspondant à la contrainte géométrique. La première variante de DL-AM-TO n’a pas montré la meilleure performance. Nous avons identifié les raisons de cette performance. La raison primaire est la performance des discriminateurs géométriques qui pénalisent le générateur ; un discriminateur fournissant une valeur erronée pénalise le générateur via la fonction de perte par une valeur non informative et erronée.

Par conséquent, les trois nouveaux discriminateurs géométriques ont été analysés et améliorés. Nous avons transformé le problème de régression vers un problème de classification ou ce qu’on appelle une classification ordinaire. En effet, nous avons découpé l’intervalle des valeurs de chaque variable géométrique en K mini-intervalles ; au lieu de prédire la valeur scalaire de la variable géométrique, nous prédisons la classe qui représente le mini-intervalle. Mais, cette nouvelle formulation du problème n’est pas une classification traditionnelle. Dans cette dernière, la fonction de perte ne considère pas la notion de distance entre les classes, sauf que cette notion est très importante dans notre cas ; prédire que le surplomb d’un design appartient à la classe 0 (qui représente l’intervalle $[0\check{r}, 5\check{r}]$) quand il appartient à classe 10 ($[55\check{r}, 60\check{r}]$) doit être pénalisée plus sévèrement que lorsque la classe prédite est 1 ($[5\check{r}, 10\check{r}]$). Par suite, la méthode de régression ordinaire conditionnelle pour les réseaux de neurones, CORN [8], est adoptée. De plus, nous avons augmenté la base d’entraînement pour chaque discriminateur afin d’uniformiser la distribution des classes et garantir une meilleure précision pour

la prédiction. Les trois discriminateurs sont ensuite entraînés et validés à deux étapes. La première étape consistait à prédire sur une base de test et remonter les métriques de précision et accuracy. La deuxième étape était d'utiliser les réseaux bayésiens de neurones pour quantifier les incertitudes sur les prédictions des discriminateurs. Plusieurs méthodes ont été implémentées et testées par notre stagiaire Steve Nouatin. Toutes ont convergées sur le même résultat. Pour des raisons de couts computationnels pendant l'entraînement et l'inférence, nous avons gardé la méthode évidentielle [126]. Le résultat de cette méthode est que les trois discriminateurs étaient précis globalement et leur confusion était limitée aux classes limitrophes (la classe k était confuse par la classe $k \pm 1$), ce qui est acceptable. Avant de passer à l'entraînement de DL-AM-TO, nous avons effectué une étude d'ablation afin de comprendre l'effet de chaque variable géométrique sur les autres quand elle est seule contrôlée par le générateur DL. L'objectif était de, en premier plan, s'assurer que le modèle DL est capable de comprendre l'effet de chaque variable sur la géométrie, et en second plan, d'explorer les corrélations entre les variables géométriques, quand elles sont contrôlées chacune à part ; avec plus de degrés de liberté, avant de passer vers DL-AM-TO où elles seront contrôlées en parallèle.

En d'autres termes, nous avons entraînés trois modèles DL avec les contraintes mécaniques et à chaque fois une et une seule variable géométrique en entrée. La seule contrainte qui a été exclue de cette ablation est l'épaisseur minimale des barres parce qu'elle n'a pas un effet à modifier radicalement la géométrie du design et par suite influencer les autres variables géométriques. Les aboutissements de cette étude étaient les suivants.

Pour le modèle qui contrôlait le surplomb en entrée : Θ_{min} est inversement proportionnel à Nbr_{bars} . En effet, une valeur plus élevée de Θ_{min} signifie que les barres sont fortement inclinées, il faut donc éliminer les barres de transmission interne qui sont souvent des barres horizontales, ce qui explique cette relation entre les deux variables. Pour le len_{max} , nous ne voyons pas de relation évidente avec le Θ_{min} , donc, pour le moment, nous supposons qu'ils sont indépendants.

Pour le modèle qui contrôlait la longueur maximale des barres : la condition len_{max} est inversement proportionnelle à la contrainte Nbr_{bars} ; si l'on veut éliminer les ponts (longues barres pendantes), il vaut mieux ajouter une barre pour les soutenir. len_{max} et Θ_{min} sont inversement proportionnels mais à condition que les contraintes mécaniques appliquées le permettent.

Pour le modèle qui contrôlait le nombre de barres : la condition len_{max} est inversement proportionnelle à la contrainte Nbr_{bars} . La relation de proportionnalité inverse de Nbr_{bars} et de Θ_{min} est conditionnée

par la structure générale du design, qui est définie par les contraintes mécaniques, et surtout par le fait que les contraintes mécaniques sont toujours prédominantes en ce qui concerne la géométrie générale du design.

Après cette étape, nous avons ré-entraîné DL-AM-TO, en utilisant la même architecture de Resnet et en ajoutant la fonction de perte de similarité structurelle à sa fonction de perte initiale. Les designs générés par DL-AM-TO ont montré une meilleure qualité que la première variante, justifiant notre hypothèse quant à l'effet des discriminateurs sur la convergence et par suite la performance du générateur.

De plus, nous avons testé la capacité de DL-AM-TO d'adapter la géométrie d'un design vis-à-vis des modifications des contraintes géométriques en entrée. Nous avons figé les contraintes mécaniques et trois contraintes géométriques et avons varié la quatrième contrainte géométrique. Vu qu'on dispose de quatre contraintes géométriques, cette expérience a été réalisée quatre fois. Les issues de chacune de ces expériences sont les suivantes. Il est essentiel de noter que pour les contraintes de Nbr_{bars} et th_{min} , la contrainte de fraction volumique V a été ajustée en parallèle ; augmenter le nombre de barres ou leur épaisseur demandent plus de matières, cet aspect est contrôlé par V .

Quand la longueur maximale de barre est balayée, nous constatons que les contraintes de len_{max} et Nbr_{bars} sont toujours inversement proportionnels. Cependant, comme DL-AM-TO contrôle les deux variables simultanément, le changement dans les Nbr_{bars} est modeste lorsque la géométrie le permet contrairement à ce que nous avons observé quand la len_{max} était la seule variable contrôlée. Le Θ_{min} est plus ou moins constant en fonction de la variation de len_{max} .

Quand le minimum surplomb est balayé, DL-AM-TO réussit plus ou moins à conserver les autres contraintes géométriques intactes. Quand le minimum épaisseur est balayée, DL-AM-TO génèrent des designs toujours respectant les autres variables géométriques. Toutefois, nous avons remarqué que l'augmentation de l'épaisseur de géométries complexes provoquait l'apparition de nouvelles barres et par suite des modifications des autres contraintes géométriques.

Quand le nombre de barres est balayé, nous remarquons toujours l'inverse proportionnalité entre Nbr_{bars} et les deux variables Θ_{min} et len_{max} , avec la variable len_{max} la moins impactée. Nous déduisons que la contrainte de barres est la plus dominante dans DL-AM-TO. Mais, pourvu qu'elle soit la moins importante quant à la fabrication additive (FA), le concepteur pourrait toujours sacrifier cette contrainte afin de se conformer aux autres plus pertinentes pour la FA.

De plus, nous avons mis DL-AM-TO à l'épreuve en reproduisant le process de design pour la FA avec quatre expériences. Une expérience où nous avons montré l'effet du Nbr_{bars} , une autre pour la contrainte de len_{max} et deux expériences pour la contrainte de Θ_{min} qui est la contrainte la plus impactante quant à la FA. Dans les quatre expériences, nous avons dessiné les CAO des designs proposés par DL-AM-TO et de celui issu de SIMP. Ces CAOs sont ensuite exportées en stl et passé vers le logiciel de découpe d'impression 3D, Cura ; ce logiciel ajoute les structures de support nécessaires et génère les étapes d'impression à passer à l'imprimante 3D, Creality Ender 3. L'analyse mécanique, le maillage et le calcul de déplacement maximal, a été produite par Patran-Nastran. Les résultats de ces expériences sont les suivantes.

1. L'augmentation des Nbr_{bars} diminue le besoin de support ; les barres ajoutées jouent le rôle de support, améliorent la résilience mécanique, avec une légère augmentation du temps de construction. Pour les pièces dont la structure de support peut endommager la surface et la fonctionnalité, choisir d'augmenter le nombre de barres est un bon choix car l'augmentation du temps de construction et du matériau est admissible (environ 10 %). DL-AM-TO permet de changer le Nbr_{bars} d'une géométrie, tâche difficilement réalisable avec SIMP, et propose des géométries mécaniquement résilientes, leur déplacement maximal est du même ordre de grandeur que ceux sortis par SIMP.
2. avec l'exemple considéré, nous n'avons pas pu conclure sur l'effet de la modification de la contrainte de len_{max} sur l'accélération de la phase d'impression.
3. DL-AM-TO adapte la géométrie du design pour respecter la contrainte Θ_{min} tout en gardant une performance mécanique similaire à celle de SIMP. En moyenne, elle accélère la phase d'impression de 1.4 fois et permet d'économiser environ 40 % de matériaux et de coûts. Il est essentiel de souligner qu'avec SIMP la modification de la géométrie est difficilement réalisable ; SIMP ne permet pas de proposer plusieurs géométries par un simple changement de la contrainte Θ_{min} en entrée.

Finalement, nous capitalisons sur la performance computationnelle de DL-AM-TO. DL-AM-TO fournit une ébauche de design qui est déjà conforme aux critères de fabrication, elle réduit le nombre d'essais que le/la concepteur(e) doit effectuer et lui évite de rester coincé dans une boucle de n itérations. Par conséquent, DL-AM-TO accélère le processus de DfAM jusqu'à $1,4n$ fois avec $n \geq 1$. En

outre, DL-AM-TO est moins coûteux en termes de calcul que SIMP. Pour toute complexité d'entrée, DL-AM-TO coûte 1,97 *GFLOPs* (giga floating point operations, figure 5.37), alors que les *GFLOPs* de SIMP varient de 68 à 656 (Tab. 3.3). Ces avantages computationnels la rendent un bon candidat comme module génératif dans des logiciels industriels de conception. En résumé, DL-AM-TO adapte de manière créative la géométrie de la conception afin de respecter les critères de fabrication sans détériorer ses performances mécaniques. Il accélère le processus de DfAM en suggérant à l'ingénieur une conception conforme aux contraintes d'entrée, en particulier celles de fabrication, ce qui évite au concepteur d'être bloqué dans les phases ultérieures du processus de DfAM, le dessin CAO et les tests mécaniques de la conception. Il accélère la phase d'impression, car la pièce nécessite moins de structures de support. Quatre expériences ont été réalisées pour valider les designs de DL-AM-TO. DL-AM-TO valide les trois hypothèses proposées dans le chapitre 1.

Les contraintes AM qui peuvent difficilement être formulées analytiquement comme Nbr_{bars} , len_{max} , et plus important encore, Θ_{min} ont été intégrées dans DL-AM-TO au même niveau que les contraintes mécaniques (hypothèse 1).

DL-AM-TO accélère l'ensemble du processus de DfAM, et pas seulement la phase de conception, en générant une première ébauche de conception conforme aux contraintes géométriques-fabrication et mécaniques et en évitant les itérations répétitives de mise à jour et d'évaluation de la conception (hypothèse 2). Enfin, les avantages computationnels de DL-AM-TO en font un module de conception générative plus léger à mettre en œuvre dans les logiciels de conception industrielle à l'avenir (hypothèse 3).

En conclusion, notre approche bénéficie du meilleur des approches AM-FE-TO et DL-TO. Elle compense les difficultés rencontrées dans AM-FE-TO lors de l'intégration des contraintes informelles de fabrication dans la phase de conception avec les techniques DL, notamment CNN. De plus, la convergence est mieux servie car DL-AM-TO a été entraînée sur des designs convergés. Elle est indépendante du maillage, et est évolutive et rapide, ce qui le qualifie comme un module de conception générative léger à mettre en œuvre dans un logiciel de conception industrielle. Enfin, elle accélère l'ensemble du processus de DfAM et pas seulement la phase de conception (Tab.5.2).

5.9. FRENCH SUMMARY

Conclusion and perspectives

Content

5.10 Conclusion	162
5.11 Perspectives	162

5.10 Conclusion

This thesis explores DL techniques in the DfAM’s design phase to compensate for the difficulties faced by traditional FE design approaches when integrating manufacturing constraints into the early design phase.

We started with a proof of concept approach, DL-TO. Then, it was trained on a dataset of designs and their mechanical constraints from an FE-based TO method, SIMP, and a first geometrical constraint, the number of bars. DL-TO validated the concept that DL can learn geometrical features at the same level as the mechanics. Furthermore, it showed a generalization power and accounted for additional spatial constraints, passive and active elements, without being trained on this task and when SIMP failed to do so. This approach encouraged us to improve DL-TO to integrate more concrete and DfAM-impactful manufacturing constraints and build our objective approach, Deep Learning Additive Manufacturing Topology Optimization, DL-AM-TO. Ergo, a dataset comprising 2D designs in an image and CAD format alongside their mechanical and geometrical constraints, was built, GMCAD. GMCAD has been published for the public for research purposes only. With GMCAD, DL-AM-TO was trained. DL-AM-TO demonstrated its capacity to generate 2D designs complying with input mechanical and geometric manufacturing constraints, particularly the minimum overhang, the maximum bar length, the minimum bar thickness, and the number of bars. DL-AM-TO tailors the design’s geometry creatively while always maintaining a similar mechanical performance to FE-based TO, SIMP. It validated all three hypotheses proposed in chapter 1. It integrated four geometric-manufacturing constraints that are hardly analytically formulated (hypothesis 1). It alleviates the designer from getting stuck in a loop of updating and testing the CAD and hence accelerates the whole DfAM process up to $1.4n$ times, with n being the number of trials in the loop (hypothesis 2). Its computational and operational costs are constant and cheaper than traditional FE-based design approaches. Therefore, DL-AM-TO is a good candidate as a lighter and faster generative module in industrial design software (hypothesis 3).

5.11 Perspectives

To recall SIMP’s principal setback: the general shape is usually identified by SIMP at the very early iterations, which makes it difficult to modify it to account for other changes imposed by other

constraints. Hence, the output shape ends up a local minimum [64] (section 1.2.1). Arguably, the level-set approach resolves the principal setback of density-TO. Nevertheless, density-TO, SIMP mainly, is the most implemented in industrial-commercial software, which justifies developing a DL approach trained on SIMP designs. SIMP is the simplest and most easily implemented TO algorithm. Thus, the designs driving GMCAD's consolidation and DL-AM-TO's training are created based on SIMP. Certainly, any new data coming from other TO algorithms can be rather used to train DL-AM-TO. In the future, GMCAD could be augmented with 3D CAD data, allowing us to improve DL-AM-TO to generate 3D designs. However, this upgrade comes with challenges. Indeed, DL-AM-TO requires three challenging and time-consuming steps: data preparation, model architecture, and training. As we have seen, creating the training dataset GMCAD was not straightforward; it needed two DL models to be trained and tested and thousands of designs to be drawn and stored. Thus, going to 3D will require more sophisticated DL models and larger storage space. Moreover, choosing the suitable model's architecture is critical; the hyperparameters of a DL model are numerous (the number of layers, the number of features map, the activation functions, Etc.), the set of possible values for each hyperparameter is vast, and trying all these combinations is nearly impossible. As a matter of fact, every entry to DL-AM-TO, in its 2D version, is a matrix of $9 \times 101 \times 101$, and the complexity of DL-AM-TO's architecture; $29.82e6$ trainable parameters (figure 5.37), training took about twelve hours to converge and required a significant number of computational powers, precisely eight Graphical Processing Units (GPUs), accordingly going to 3D would require a more complex model's architecture and more extensive computational powers.

Indeed, the asset of DL-AM-TO is that, unlike FE-TO, the initial guess can be easily tailored to account for geometric manufacturing-related constraints. Moreover, it increases the domain space of potential designs, which can be leveraged by a shape/size optimization afterward and provides creativity and rapidity to the DfAM. While its training involves expensive computational powers, its usage afterward necessitates constant computational and operational powers no matter the input's complexity, unlike FE-based approaches. Consequently, the acceleration advantage of DL-AM-TO over FE-SIMP can seem moderate in the short term but will definitely pay off in the long run. Besides, these computational advantages favor DL-AM-TO to be industrialized in the future as a lighter, faster, and computationally efficient generative module in industrial design software.

Nevertheless, DL-AM-TO should never fully replace robust mathematical FE approaches. The optimal

approach is to proceed via a hybrid one, i.e., we start with DL-AM-TO to obtain a first draft, which is geometrically compliant with manufacturing criteria, and then we optimize this shape via traditional FE TO methods to enhance its mechanical specs. With this approach, we overcome the TO's principal setback and supply TO with a manufacturing-friendly geometry to be optimized mechanically. This approach is to be considered in future work.

Furthermore, additional manufacturing constraints could be integrated into DL-AM-TO, like buckling and thermal distortion, and more importantly, custom industry-related rules due to the chosen flexible training framework based on the GANs. Nonetheless, this would require labeling GMCAD's designs to account for these new rules either by modeling buckling/thermal distortion via a mathematical simulation or training a DL model to learn these rules, which will be used to predict these rules on GMCAD's designs.

Bibliography

- [1] G. A. Adam and D. Zimmer, “Design for additive manufacturing—element transitions and aggregated structures,” *CIRP Journal of Manufacturing Science and Technology*, vol. 7, no. 1, pp. 20–28, 2014.
- [2] S. C. Subedi, C. S. Verma, and K. Suresh, “A review of methods for the geometric post-processing of topology optimized models,” *Journal of Computing and Information Science in Engineering*, pp. 1–17, 2020.
- [3] Z. Bi and X. Wang, *Computer aided design and manufacturing*. John Wiley & Sons, 2020.
- [4] E. Tyflopoulos and M. Steinert, “A comparative study of the application of different commercial software for topology optimization,” *Applied Sciences*, vol. 12, no. 2, p. 611, 2022.
- [5] W. Almasri, D. Bettebghor, F. Adjed, F. Ababsa, and F. Danglade, “Gmcd: an original synthetic dataset of 2d designs along their geometrical and mechanical conditions,” *Procedia Computer Science*, vol. 200, pp. 337–347, 2022.
- [6] Z. Zhang, Q. Liu, and Y. Wang, “Road extraction by deep residual u-net,” *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 749–753, 2018.
- [7] M. Leary, *Design for additive manufacturing*. Elsevier, 2019.
- [8] X. Shi, W. Cao, and S. Raschka, “Deep neural networks for rank-consistent ordinal regression based on conditional probabilities,” *arXiv preprint arXiv:2111.08851*, 2021.
- [9] J. Riegel, W. Mayer, and Y. van Havre, “FreeCAD,” 2016.
- [10] U. David Braam, “Cura,” <https://ultimaker.com/software/ultimaker-cura/>, 19 July 2022.

BIBLIOGRAPHY

- [11] Y. Saadlaoui, J.-L. Milan, J.-M. Rossi, and P. Chabrand, “Topology optimization and additive manufacturing: Comparison of conception methods using industrial codes,” *Journal of Manufacturing Systems*, vol. 43, pp. 178–186, 2017.
- [12] J. Wu, N. Aage, R. Westermann, and O. Sigmund, “Infill optimization for additive manufacturing—approaching bone-like porous structures,” *IEEE transactions on visualization and computer graphics*, vol. 24, no. 2, pp. 1127–1140, 2017.
- [13] J. Jiang, S. T. Newman, and R. Y. Zhong, “A review of multiple degrees of freedom for additive manufacturing machines,” *International Journal of Computer Integrated Manufacturing*, vol. 34, no. 2, pp. 195–211, 2021.
- [14] T. Pereira, J. V. Kennedy, and J. Potgieter, “A comparison of traditional manufacturing vs additive manufacturing, the best method for the job,” *Procedia Manufacturing*, vol. 30, pp. 11–18, 2019.
- [15] W. Gao, Y. Zhang, D. Ramanujan, K. Ramani, Y. Chen, C. B. Williams, C. C. Wang, Y. C. Shin, S. Zhang, and P. D. Zavattieri, “The status, challenges, and future of additive manufacturing in engineering,” *Computer-Aided Design*, vol. 69, pp. 65–89, 2015.
- [16] M. Zhou, Y. Liu, and Z. Lin, “Topology optimization of thermal conductive support structures for laser additive manufacturing,” *Computer Methods in Applied Mechanics and Engineering*, vol. 353, pp. 24–43, 2019.
- [17] K. Zhang, G. Cheng, and L. Xu, “Topology optimization considering overhang constraint in additive manufacturing,” *Computers & Structures*, vol. 212, pp. 86–100, 2019.
- [18] N. A. Sbrugnera Sotomayor, F. Caiazzo, and V. Alfieri, “Enhancing design for additive manufacturing workflow: Optimization, design and simulation tools,” *Applied Sciences*, vol. 11, no. 14, p. 6628, 2021.
- [19] O. M. Querin, M. Victoria, C. Alonso, R. A. Loyola, and P. M. Montrull, *Topology design methods for structural optimization*. Butterworth-Heinemann, 2017.
- [20] O. Sigmund and K. Maute, “Topology optimization approaches,” *Structural and Multidisciplinary Optimization*, vol. 48, no. 6, pp. 1031–1055, 2013.

BIBLIOGRAPHY

- [21] M. P. Bendsøe, “Optimal shape design as a material distribution problem,” *Structural optimization*, vol. 1, no. 4, pp. 193–202, 1989.
- [22] G. Allaire, F. Jouve, and A.-M. Toader, “A level-set method for shape optimization,” *Comptes Rendus Mathématique*, vol. 334, no. 12, pp. 1125–1130, 2002.
- [23] C. Mattheck, M. Scherrer, K. Bethge, and I. Tesari, “Shape optimization: an analytical approach,” *WIT Transactions on the Built Environment*, vol. 80, 2005.
- [24] O. Sigmund and J. Petersson, “Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima,” *Structural optimization*, vol. 16, no. 1, pp. 68–75, 1998.
- [25] M. P. Bendsøe and O. Sigmund, “Material interpolation schemes in topology optimization,” *Archive of applied mechanics*, vol. 69, no. 9-10, pp. 635–654, 1999.
- [26] J. W. Booth, J. Alperovich, T. N. Reid, and K. Ramani, “The design for additive manufacturing worksheet,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 50190. American Society of Mechanical Engineers, 2016, p. V007T06A041.
- [27] A. T. Gaynor and J. K. Guest, “Topology optimization considering overhang constraints: Eliminating sacrificial support material in additive manufacturing through design,” *Structural and Multidisciplinary Optimization*, vol. 54, no. 5, pp. 1157–1172, 2016.
- [28] A. M. Mirzendehtel and K. Suresh, “Support structure constrained topology optimization for additive manufacturing,” *Computer-Aided Design*, vol. 81, pp. 1–13, 2016.
- [29] G. Allaire, C. Dapogny, R. Estevez, A. Faure, and G. Michailidis, “Structural optimization under overhang constraints imposed by additive manufacturing technologies,” *Journal of Computational Physics*, vol. 351, pp. 295–328, 2017.
- [30] X. Guo, J. Zhou, W. Zhang, Z. Du, C. Liu, and Y. Liu, “Self-supporting structure design in additive manufacturing through explicit topology optimization,” *Computer Methods in Applied Mechanics and Engineering*, vol. 323, pp. 27–63, 2017.

BIBLIOGRAPHY

- [31] M. Langelaar, “An additive manufacturing filter for topology optimization of print-ready designs,” *Structural and multidisciplinary optimization*, vol. 55, no. 3, pp. 871–883, 2017.
- [32] Y. Mass and O. Amir, “Topology optimization for additive manufacturing: Accounting for overhang limitations using a virtual skeleton,” *Additive Manufacturing*, vol. 18, pp. 58–73, 2017.
- [33] W. Zhang and L. Zhou, “Topology optimization of self-supporting structures with polygon features for additive manufacturing,” *Computer Methods in Applied Mechanics and Engineering*, vol. 334, pp. 56–78, 2018.
- [34] Y. M. Yoely, O. Amir, and I. Hanniel, “Topology and shape optimization with explicit geometric constraints using a spline-based representation and a fixed grid,” *Procedia Manufacturing*, vol. 21, pp. 189–196, 2018.
- [35] Y. S. Han, B. Xu, L. Zhao, and Y. M. Xie, “Topology optimization of continuum structures under hybrid additive-subtractive manufacturing constraints,” *Structural and Multidisciplinary Optimization*, vol. 60, no. 6, pp. 2571–2595, 2019.
- [36] M. Bi, P. Tran, and Y. M. Xie, “Topology optimization of 3d continuum structures under geometric self-supporting constraint,” *Additive Manufacturing*, vol. 36, p. 101422, 2020.
- [37] B. Xu, Y. Han, L. Zhao, and Y. M. Xie, “Topological optimization of continuum structures for additive manufacturing considering thin feature and support structure constraints,” *Engineering Optimization*, pp. 1–22, 2020.
- [38] S. Li, S. Yuan, J. Zhu, C. Wang, J. Li, and W. Zhang, “Additive manufacturing-driven design optimization: Building direction and structural topology,” *Additive Manufacturing*, p. 101406, 2020.
- [39] C. Wang and X. Qian, “Simultaneous optimization of build orientation and topology for additive manufacturing,” *Additive Manufacturing*, p. 101246, 2020.
- [40] M. A. Matos, A. M. A. Rocha, and L. A. Costa, “Many-objective optimization of build part orientation in additive manufacturing,” *The International Journal of Advanced Manufacturing Technology*, pp. 1–16, 2020.

- [41] E. Fernández, K.-k. Yang, S. Koppen, P. Alarcón, S. Bauduin, and P. Duysinx, “Imposing minimum and maximum member size, minimum cavity size, and minimum separation distance between solid members in topology optimization,” *Computer Methods in Applied Mechanics and Engineering*, vol. 368, p. 113157, 2020.
- [42] S. Oh, Y. Jung, I. Lee, and N. Kang, “Design automation by integrating generative adversarial networks and topology optimization,” in *ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection, 2018.
- [43] I. Sosnovik and I. Oseledets, “Neural networks for topology optimization,” *Russian Journal of Numerical Analysis and Mathematical Modelling*, vol. 34, no. 4, pp. 215–223, 2019.
- [44] C. Wang, S. Yao, Z. Wang, and J. Hu, “Deep super-resolution neural network for structural topology optimization,” *Engineering Optimization*, pp. 1–14, 2020.
- [45] N. A. Kallioras, G. Kazakis, and N. D. Lagaros, “Accelerated topology optimization by means of deep learning,” *Structural and multidisciplinary optimization*, (under review), 2020.
- [46] S. Bi, J. Zhang, and G. Zhang, “Scalable deep-learning-accelerated topology optimization for additively manufactured materials,” *arXiv preprint arXiv:2011.14177*, 2020.
- [47] A. Chandrasekhar and K. Suresh, “Tounn: Topology optimization using neural networks,” *Structural and Multidisciplinary Optimization*, pp. 1–15, 2020.
- [48] Z. Nie, T. Lin, H. Jiang, and L. B. Kara, “Topologygan: Topology optimization using generative adversarial networks based on physical fields over the initial domain,” *arXiv preprint arXiv:2003.04685*, 2020.
- [49] E. Ulu, R. Zhang, and L. B. Kara, “A data-driven investigation and estimation of optimal topologies under variable loading configurations,” *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, vol. 4, no. 2, pp. 61–72, 2016.
- [50] Y. Yu, T. Hur, J. Jung, and I. G. Jang, “Deep learning for determining a near-optimal topological design without any iteration,” *Structural and Multidisciplinary Optimization*, vol. 59, no. 3, pp. 787–799, 2019.

BIBLIOGRAPHY

- [51] C. Sharpe and C. C. Seepersad, “Topology design with conditional generative adversarial networks,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 59186. American Society of Mechanical Engineers, 2019, p. V02AT03A062.
- [52] S. Rawat and M.-H. H. Shen, “A novel topology optimization approach using conditional deep learning,” *arXiv preprint arXiv:1901.04859*, 2019.
- [53] S. Hoyer, J. Sohl-Dickstein, and S. Greydanus, “Neural reparameterization improves structural optimization,” *arXiv preprint arXiv:1909.04240*, 2019.
- [54] D. W. Abueidda, S. Koric, and N. A. Sobh, “Topology optimization of 2d structures with nonlinearities using deep learning,” *Computers & Structures*, vol. 237, p. 106283, 2020.
- [55] M. Malviya, “A systematic study of deep generative models for rapid topology optimization,” 2020.
- [56] J. Rade, A. Balu, E. Herron, J. Pathak, R. Ranade, S. Sarkar, and A. Krishnamurthy, “Physics-consistent deep learning for structural topology optimization,” *arXiv preprint arXiv:2012.05359*, 2020.
- [57] Dassault Systemes, “Tosca abaqus,” <https://www.3ds.com/products-services/simulia/products/tosca/>, 2020.
- [58] Altair, “Altair inspire,” <https://solidthinking.com/product/inspire/>, 2018.
- [59] Autodesk, “Netfabb,” <https://www.autodesk.com/products/netfabb/>, 2021.
- [60] —, “Within,” <http://www.withinlab.com/>, 2015.
- [61] CAESS, “Protop,” <https://caess.eu/>, 2020.
- [62] M. Leary, L. Merli, F. Torti, M. Mazur, and M. Brandt, “Optimal topology for additive manufacture: A method for enabling additive manufacture of support-free optimal structures,” *Materials & Design*, vol. 63, pp. 678–690, 2014.

BIBLIOGRAPHY

- [63] M. Zhou, B. S. Lazarov, F. Wang, and O. Sigmund, “Minimum length scale in topology optimization by geometric constraints,” *Computer Methods in Applied Mechanics and Engineering*, vol. 293, pp. 266–282, 2015.
- [64] G. Allaire, C. Dapogny, and F. Jouve, “Shape and topology optimization,” in *Handbook of Numerical Analysis*. Elsevier, 2021, vol. 22, pp. 1–132.
- [65] R. Ranjan, R. Samant, and S. Anand, “Integration of design for manufacturing methods with topology optimization in additive manufacturing,” *Journal of Manufacturing Science and Engineering*, vol. 139, no. 6, 2017.
- [66] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding: A review,” *Neurocomputing*, vol. 187, pp. 27–48, 2016.
- [67] W. Rawat and Z. Wang, “Deep convolutional neural networks for image classification: A comprehensive review,” *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.
- [68] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.
- [69] A. Chandrasekhar and K. Suresh, “Length scale control in topology optimization using fourier enhanced neural networks,” 2020.
- [70] K. Zhang and G. Cheng, “Three-dimensional high resolution topology optimization considering additive manufacturing constraints,” *Additive Manufacturing*, vol. 35, p. 101224, 2020.
- [71] 3DS. Solidworks. [Online]. Available: <https://www.solidworks.com>
- [72] ——. Abaqus. [Online]. Available: <https://www.3ds.com/products-services/simulia/products/abaqus/>
- [73] Siemens. Siemens nx. [Online]. Available: <https://www.plm.automation.siemens.com/global/en/products/nx/>
- [74] ALTAIR. Optistruct. [Online]. Available: <https://www.altair.com/optistruct/>

BIBLIOGRAPHY

- [75] ANSYS. Ansys mechanical. [Online]. Available: <https://www.ansys.com/products/structures/ansys-mechanical>
- [76] B. Mahesh, “Machine learning algorithms-a review,” *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.
- [77] A. Shrestha and A. Mahmood, “Review of deep learning algorithms and architectures,” *IEEE access*, vol. 7, pp. 53 040–53 065, 2019.
- [78] P. Werbos, “Beyond regression:” new tools for prediction and analysis in the behavioral sciences,” *Ph. D. dissertation, Harvard University*, 1974.
- [79] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [80] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- [81] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “Squeezenet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size,” *arXiv preprint arXiv:1602.07360*, 2016.
- [82] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [83] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
- [84] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
- [85] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in *Thirty-first AAAI conference on artificial intelligence*, 2017.

BIBLIOGRAPHY

- [86] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, “Recurrent neural network based language model.” in *Interspeech*, vol. 2, no. 3. Makuhari, 2010, pp. 1045–1048.
- [87] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [88] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio, “Learning algorithms for the classification restricted boltzmann machine,” *The Journal of Machine Learning Research*, vol. 13, pp. 643–669, 2012.
- [89] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [90] N. Aloysius and M. Geetha, “A review on deep convolutional neural networks,” in *2017 international conference on communication and signal processing (ICCSP)*. IEEE, 2017, pp. 0588–0592.
- [91] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [92] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
- [93] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan,” *arXiv preprint arXiv:1701.07875*, 2017.
- [94] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, “Unrolled generative adversarial networks,” *arXiv preprint arXiv:1611.02163*, 2016.
- [95] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, “A review on generative adversarial networks: Algorithms, theory, and applications,” *IEEE Transactions on Knowledge and Data Engineering*, 2021.
- [96] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.

BIBLIOGRAPHY

- [97] Z. Wang, Q. She, and T. E. Ward, “Generative adversarial networks in computer vision: A survey and taxonomy,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.
- [98] N. Aldausari, A. Sowmya, N. Marcus, and G. Mohammadi, “Video generative adversarial networks: a review,” *ACM Computing Surveys (CSUR)*, vol. 55, no. 2, pp. 1–25, 2022.
- [99] A. Kusiak, “Convolutional and generative adversarial neural networks in manufacturing,” *International Journal of Production Research*, vol. 58, no. 5, pp. 1594–1604, 2020.
- [100] J. J. Jeong, A. Tariq, T. Adejumo, H. Trivedi, J. W. Gichoya, and I. Banerjee, “Systematic review of generative adversarial networks (gans) for medical image classification and segmentation,” *Journal of Digital Imaging*, pp. 1–16, 2022.
- [101] J. J. Bird, C. M. Barnes, L. J. Manso, A. Ekárt, and D. R. Faria, “Fruit quality and defect image classification with conditional gan data augmentation,” *Scientia Horticulturae*, vol. 293, p. 110684, 2022.
- [102] C. Yinka-Banjo and O.-A. Ugot, “A review of generative adversarial networks and its application in cybersecurity,” *Artificial Intelligence Review*, vol. 53, no. 3, pp. 1721–1736, 2020.
- [103] L. Yang, G. Haijun, S. Dilip, and B. Stucker, “An investigation of thin feature generation in direct metal laser sintering systems,” in *2014 International Solid Freeform Fabrication Symposium*. University of Texas at Austin, 2014.
- [104] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [105] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [106] S. Kim, H.-g. Chi, X. Hu, Q. Huang, and K. Ramani, “A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks,” in *European Conference on Computer Vision*. Springer, 2020, pp. 175–191.
- [107] B. Starly, “Fabwave: 3d part repository,” <https://www.dimelab.org/fabwave>, 2020.

BIBLIOGRAPHY

- [108] A. Seff, Y. Ovadia, W. Zhou, and R. P. Adams, “Sketchgraphs: A large-scale dataset for modeling relational geometry in computer-aided design,” *arXiv preprint arXiv:2007.08506*, 2020.
- [109] K. Willis, Y. Pu, J. Luo, H. Chu, T. Du, J. Lambourne, A. Solar-Lezama, and W. Matusik, “Fusion 360 gallery: A dataset and environment for programmatic cad reconstruction,” 2020.
- [110] W. Almasri, D. Bettebghor, F. Ababsa, F. Danglade, and F. Adjed, “Deep learning architecture for topological optimized mechanical design generation with complex shape criterion,” in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 2021, pp. 222–234.
- [111] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, “Deconvolutional networks,” in *2010 IEEE Computer Society Conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2528–2535.
- [112] O. Sigmund, “A 99 line topology optimization code written in matlab,” *Structural and multidisciplinary optimization*, vol. 21, no. 2, pp. 120–127, 2001.
- [113] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [114] G. van den Burg and C. Williams, “On memorization in probabilistic deep generative models,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 916–27 928, 2021.
- [115] H. Smith and J. A. Norato, “Topology optimization with discrete geometric components made of composite materials,” *Computer Methods in Applied Mechanics and Engineering*, vol. 376, p. 113582, 2021.
- [116] H. A. Smith and J. A. Norato, “Geometric constraints for the topology optimization of structures made of primitives,” in *SAMPE Conference proceedings. Charlotte*. <https://doi.org/10.33599/nasampe/s>, vol. 19, 2019, pp. 19–1518.
- [117] J. A. Norato, “Topology optimization with supershapes,” *Structural and Multidisciplinary Optimization*, vol. 58, no. 2, pp. 415–434, 2018.

BIBLIOGRAPHY

- [118] B. S. Rupal, K. G. Mostafa, Y. Wang, and A. J. Qureshi, “A reverse cad approach for estimating geometric and mechanical behavior of fdm printed parts,” *Procedia Manufacturing*, vol. 34, pp. 535–544, 2019.
- [119] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [120] Z. Niu, M. Zhou, L. Wang, X. Gao, and G. Hua, “Ordinal regression with multiple output cnn for age estimation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4920–4928.
- [121] W. Cao, V. Mirjalili, and S. Raschka, “Rank consistent ordinal regression for neural networks with application to age estimation,” *Pattern Recognition Letters*, vol. 140, pp. 325–331, 2020.
- [122] K. Janocha and W. M. Czarnecki, “On loss functions for deep neural networks in classification,” *arXiv preprint arXiv:1702.05659*, 2017.
- [123] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher *et al.*, “A survey of uncertainty in deep neural networks,” *arXiv preprint arXiv:2107.03342*, 2021.
- [124] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya *et al.*, “A review of uncertainty quantification in deep learning: Techniques, applications and challenges,” *Information Fusion*, vol. 76, pp. 243–297, 2021.
- [125] E. Hüllermeier and W. Waegeman, “Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods,” *Machine Learning*, vol. 110, no. 3, pp. 457–506, 2021.
- [126] M. Sensoy, M. Kandemir, and L. M. Kaplan, “Evidential deep learning to quantify classification uncertainty,” *ArXiv*, vol. abs/1806.01768, 2018.
- [127] A. Amini, W. Schwarting, A. Soleimany, and D. Rus, “Deep evidential regression,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 927–14 937, 2020.

BIBLIOGRAPHY

- [128] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, “Weight uncertainty in neural network,” in *International conference on machine learning*. PMLR, 2015, pp. 1613–1622.
- [129] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.
- [130] C. Li, C. Chen, D. Carlson, and L. Carin, “Preconditioned stochastic gradient langevin dynamics for deep neural networks,” in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
- [131] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” *Advances in neural information processing systems*, vol. 30, 2017.
- [132] I. Kim, Y. Kim, and S. Kim, “Learning loss for test-time augmentation,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 4163–4174, 2020.
- [133] R. R. Yager and L. Liu, *Classic works of the Dempster-Shafer theory of belief functions*. Springer, 2008, vol. 219.
- [134] A. Jøsang, *Subjective logic*. Springer, 2016, vol. 3.
- [135] W. I. Moore, E. S. Donovan, and C. R. Powers, “Recent advances in msc/patran pre-processing software allows modeling of complex automotive lamp designs,” *Delphi Interior and Lighting Systems Anderson*, 1998.
- [136] R. H. MacNeal, *The NASTRAN theoretical manual*. Scientific and Technical Information Office, National Aeronautics and Space., 1970, vol. 221.
- [137] G. Allaire, F. Jouve, and A.-M. Toader, “Structural optimization using sensitivity analysis and a level-set method,” *Journal of computational physics*, vol. 194, no. 1, pp. 363–393, 2004.
- [138] S. Banga, H. Gehani, S. Bhilare, S. Patel, and L. Kara, “3d topology optimization using convolutional neural networks,” *arXiv preprint arXiv:1808.07440*, 2018.
- [139] S. Bobby, S. M. Spence, and A. Kareem, “Data-driven performance-based topology optimization of uncertain wind-excited tall buildings,” *Structural and Multidisciplinary Optimization*, vol. 54, no. 6, pp. 1379–1402, 2016.

BIBLIOGRAPHY

- [140] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction,” in *European conference on computer vision*. Springer, 2016, pp. 628–644.
- [141] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” *arXiv preprint arXiv:1603.07285*, 2016.
- [142] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.
- [143] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, “Category-specific object reconstruction from a single image,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1966–1974.
- [144] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” *arXiv preprint arXiv:1710.10196*, 2017.
- [145] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.
- [146] H. Mlejnek, “Some aspects of the genesis of structures,” *Structural optimization*, vol. 5, no. 1-2, pp. 64–69, 1992.
- [147] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [148] M. Y. Wang, X. Wang, and D. Guo, “A level set method for structural topology optimization,” *Computer methods in applied mechanics and engineering*, vol. 192, no. 1-2, pp. 227–246, 2003.
- [149] W. M. Wang, C. Zanni, and L. Kobbelt, “Improved surface quality in 3d printing by optimizing the printing direction,” in *Computer graphics forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 59–70.

- [150] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3d shapenets: A deep representation for volumetric shapes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.
- [151] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling,” in *Advances in neural information processing systems*, 2016, pp. 82–90.
- [152] X. Technologies. Ameba. [Online]. Available: <https://ameba.xieym.com/>
- [153] H. Zhao, K. Long, and Z.-D. Ma, “Homogenization topology optimization method based on continuous field,” *Advances in Mechanical Engineering*, vol. 2, p. 528397, 2010.
- [154] M. Zhou and G. Rozvany, “The coc algorithm, part ii: Topological, geometrical and generalized shape optimization,” *Computer methods in applied mechanics and engineering*, vol. 89, no. 1-3, pp. 309–336, 1991.
- [155] J. Zhu, J. Xie, and Y. Fang, “Learning adversarial 3d model generation with 2d image enhancer,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

BIBLIOGRAPHY

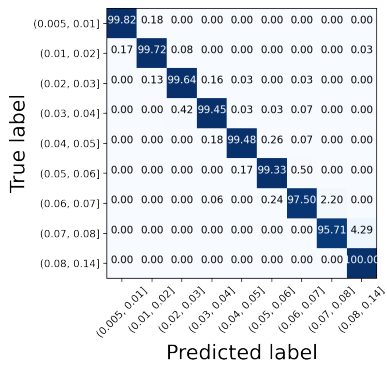
Appendix A

Appendices

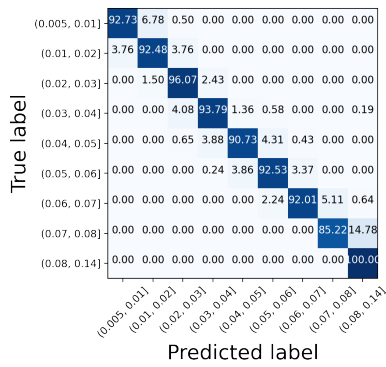
A.1 Geometrical discriminators' performance

This section includes the confusion matrices computed over the train, validation and test sets for the three CORN geometrical discriminators.

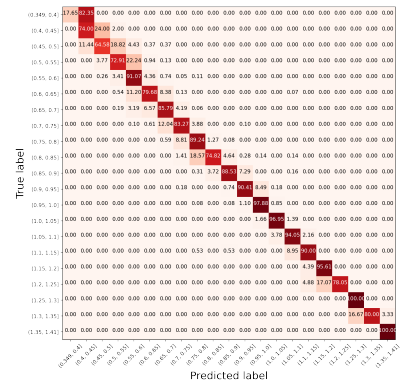
A.1. GEOMETRICAL DISCRIMINATORS' PERFORMANCE



(a) Train set.



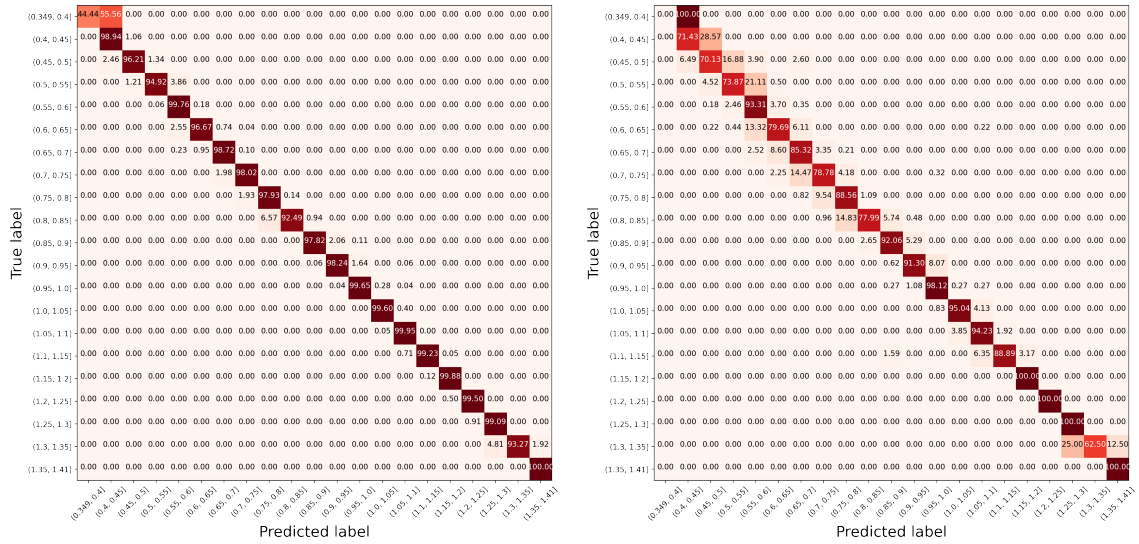
(b) Validation set.



(c) Test set.

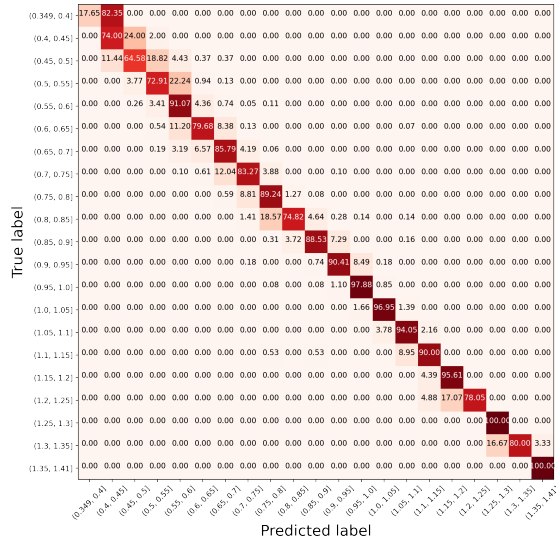
Figure A.1: The confusion matrices of the th_{min} geometric discriminator.

A.1. GEOMETRICAL DISCRIMINATORS' PERFORMANCE



(a) Train set.

(b) Validation set.



(c) Test set.

Figure A.2: The confusion matrices of the len_{max} geometric discriminator.

A.1. GEOMETRICAL DISCRIMINATORS' PERFORMANCE

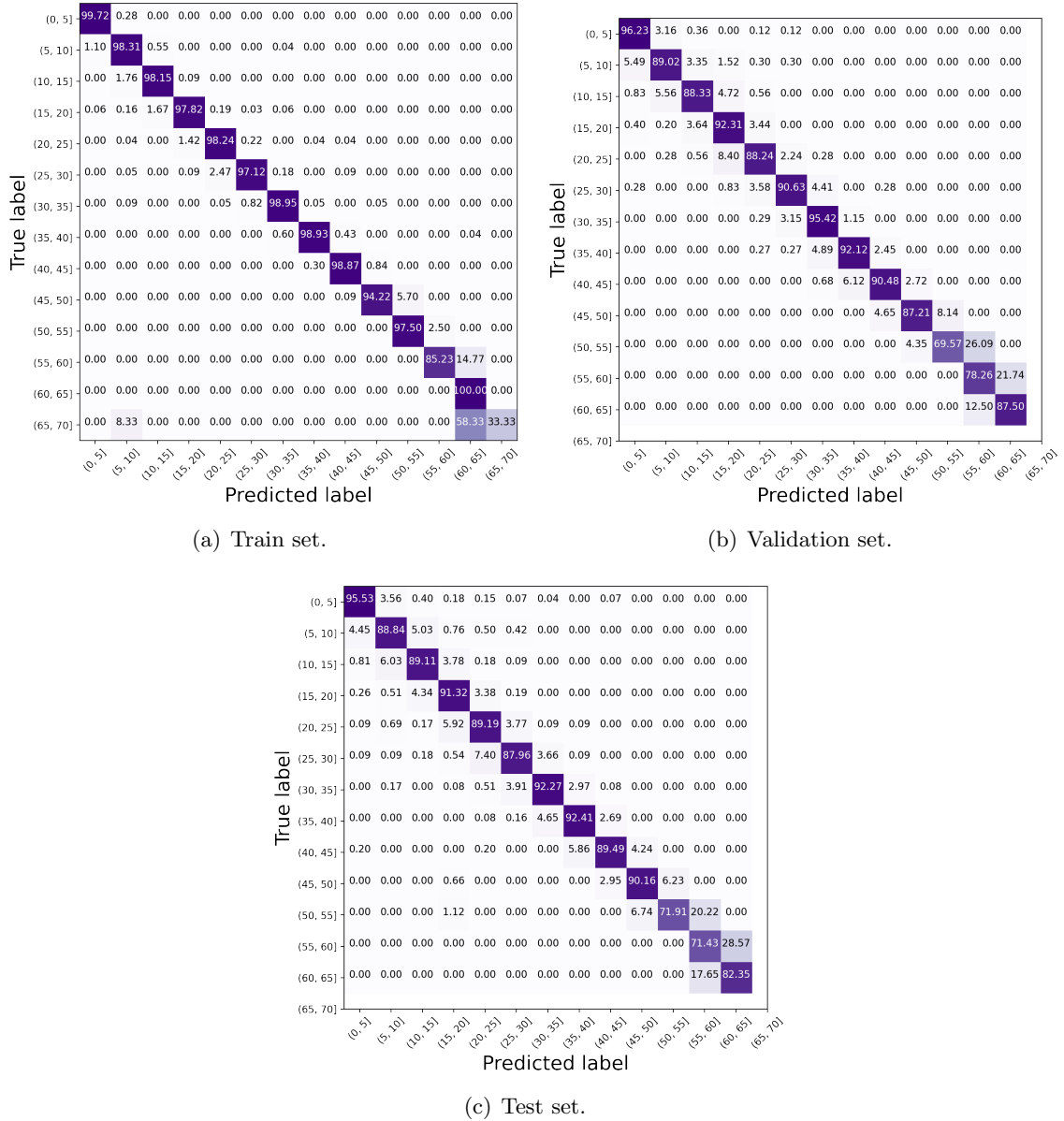


Figure A.3: The confusion matrices of the Θ_{min} geometric discriminator.

Abstract : The growing need for fast, organic, cost and material efficient products in the industrial world is driving research to develop new design methods. Topological optimization (TO) is one of them. TO takes mechanical constraints as input and generates optimal complex shapes in terms of material and mechanical performance. Additive manufacturing (AM) completes it; it allows the fabrication of any shape. However, this synergy between TO and AM is not ideal. Indeed, AM requires the conformity of the design to geometrical criteria like overhangs, thicknesses, Etc., which are difficult to integrate into TO. Moreover, TO is iterative and computationally expensive; it is based on finite elements (FE). Therefore, designing printable parts requires the interpretation of the shapes proposed by TO by experts, knowing that this can deteriorate the initial optimality. With TO software, generating a design takes seconds to days depending on the complexity of the mechanical conditions. This is acceptable if the Design for Additive Manufacturing (DfAM) process is limited to this step. However, TO, in its commercial form, does not consider AM constraints. The engineer must test several configurations to find the optimal, printable design and risks getting stuck in a design and performance test loop. With the flourishing role of AM in the industry, it is imperative to find a method that considers mechanical and geometric constraints at the same conceptual level to speed up the DfAM process. We find in the state of the art four approaches:

1. Formalizing AM design rules and guidelines for novice users. However, modifying the geometry of the design proposed by TO in the drawing phase often deteriorates its initial mechanical performance.
2. Integrating AM constraints into FE-TO. This helps to speed up the DfAM process. However, not all AM constraints can be defined analytically and are often contradictory. Moreover, this approach inherits the shortcomings of FE-TO: the general shape is identified in the first iterations, which prevents the method from modifying it to comply with the geometrical constraints, also, convergence is not guaranteed when the number of constraints increases.
3. Assisting FE-TO methods with Machine and Deep Learning (DL). This accelerates the TO phase of DfAM only and still inherits the defects of TO.
4. Replacing FE-TO with DL. This approach does not incorporate any FA criteria and does not avoid the loop in later phases of the DfAM process.

Our goal is to accelerate the whole DfAM process. Indeed, the design phase has the minimum cost and the maximum impact on the overall cost and quality of a part. But, provided that the acceleration of this phase is not sufficient, we propose DL-AM-TO which joins the best of the two approaches 2 and 4. The main contributions of the thesis are:

- The creation of a dataset of 2D designs with their mechanical and geometrical constraints of AM, GMCAD.
- The creation of DL-AM-TO which takes as input the mechanical and geometrical conditions of AM simultaneously and generates a 2D design, a difficult task to accomplish with FE-TO. It is based on convolutional neural networks, so it is fast and computationally inexpensive, independent of the design's dimensions and the complexity of the constraints. It tailors the design's geometry to meet the mechanical and geometrical requirements of AM, thus avoiding the design and test loop and accelerating the whole DfAM process up to 1.4 times.
- Any AM constraint or other business rule can be integrated into DL-AM-TO, not only those formulated analytically, thanks to the flexibility of DL models.
- DL-AM-TO can be industrialized as a lightweight generative design module in design software in the future.

Keywords : Topology Optimization, Additive Manufacturing, Design for Additive Manufacturing, Deep Learning, Generative Adversarial Networks, Convolutional Neural Networks.

Résumé : Le besoin croissant de produits organiques et efficaces en coûts, matériaux et temps en industrie, incite la recherche à développer des nouvelles méthodes de design. L'optimisation topologique (TO) en fait partie. TO prend des contraintes mécaniques en entrée et génèrent des formes complexes optimales en termes de matériau et de performance mécanique. La fabrication additive (FA) vient la compléter; elle permet la fabrication de n'importe quelle forme. Cependant, cette synergie entre TO et FA n'est pas idéale. En effet, la FA requiert la conformité du design à des critères géométriques de surplomb, d'épaisseurs, etc. difficilement intégrables à TO. De plus, TO, basée sur les éléments finis (FE), est itérative et coûteuse en calcul. Alors, concevoir des pièces imprimables nécessite l'interprétation par des experts des formes proposées par TO, sachant que cela peut détériorer l'optimalité initiale. Avec les logiciels de TO, générer un design prend des secondes à des jours selon la complexité des conditions mécaniques. Cela est acceptable si le process de conception pour FA (Design for Additive Manufacturing, DfAM) se limite à cette étape. Cependant, TO, dans sa forme commerciale, ne considère pas les contraintes FA. L'ingénieur(e) doit tester plusieurs configurations pour trouver le design optimal et imprimable et risque de se coincer dans une boucle de dessin et test de performance. Avec le rôle florissant de FA dans l'industrie, il est impératif de trouver une méthode considérant les contraintes mécaniques et géométriques au même niveau conceptuel pour accélérer le process DfAM. Nous trouvons dans l'état de l'art quatre approches:

1. La formalisation de règles de dessin de design de FA pour les utilisateurs novices. Cependant, modifier la forme du design proposé par TO dans la phase de dessin détériore souvent sa performance mécanique initiale;
2. l'intégration de contraintes FA dans FE-TO. Cela permet d'accélérer le process DfAM. Toutefois, les contraintes FA ne peuvent pas être toutes définies analytiquement et sont souvent contradictoires. De plus, elle hérite des défauts de FE-TO: la forme générale est identifiée dès les premières itérations, ce qui empêche la méthode de la modifier pour respecter les contraintes géométriques et la convergence n'est plus garantie lorsqu'il y en a plusieurs;
3. l'assistance des méthodes FE-TO avec du Machine et Deep Learning (DL). Cela accélère la phase TO du DfAM uniquement et hérite toujours des défauts de TO;
4. le remplacement de FE-TO par du DL. Cette approche n'intègre aucun critère FA et ne permet pas d'éviter la boucle dans les phases ultérieures du process DfAM. Notre objectif est d'accélérer l'ensemble du process DfAM. Certes, la phase de design a le coût minimal et l'impact maximal sur le coût global et la qualité d'une pièce. Mais, pourvu que l'accélération de cette phase ne soit pas suffisante, nous proposons DL-AM-TO qui rejoint le meilleur des deux approches 2 et 4.

Les principales contributions de la thèse sont:

- la création d'un dataset de designs 2D avec leurs contraintes mécaniques et géométriques de FA, GMCAD;
- la création de DL-AM-TO qui prend en entrée les conditions mécaniques et géométriques de FA simultanément et génère un design 2D, une tâche difficile à accomplir avec FE-TO. Elle est basée sur les réseaux convolutifs de neurones, ainsi, elle est dotée de rapidité, et de coûts de calcul avantageux et indépendants de l'échelle du design et de la complexité des contraintes. Elle taille la géométrie du design afin de respecter les conditions mécaniques et géométriques de FA et ainsi nous éviter la boucle de dessin et test et accélérer l'ensemble du process DfAM jusqu'à 1.4 fois;
- toute contrainte FA ou autre règle métiers peut être intégrée à DL-AM-TO, et pas seulement celles formulées analytiquement, grâce à la flexibilité des modèles DL;
- DL-AM-TO pourra être industrialisée comme un module de design génératif léger dans un logiciel de design à l'avenir.

Mots clés : Optimisation Topologique, Fabrication Additive, Design pour la Fabrication Additive, Apprentissage Profond, Réseaux génératifs antagonistes, Réseaux Convolutionnels de neurones.