Department of Precision and Microsystems Engineering

Layer Thickness Control in 3D Fabrication Sequence Optimization for Multi-Axis Additive Manufacturing

Riccardo Di Girolamo

Report no : 2024.101
Coach : MSc. Kai Wu
Professor : Dr. Jun Wu
Date : 6th December 2024

TU Delft Delft University of Technology

Challenge the future

Layer Thickness Control in 3D Fabrication Sequence Optimization for Multi-Axis Additive Manufacturing

Riccardo Di Girolamo
Delft University of Technology

To obtain the degree of Master of Science at the Delft University of Technology, to be defended publicly on Friday, December 6, 2024, at 1:00 PM.

Student number: 5847052

Project duration: September, 2023 – December, 2024

Thesis committee:

Dr. Jun Wu, TU Delft, supervisor

Dr. Can Ayas, TU Delft

Dr. Kunal Masania, TU Delft

MSc. Kai Wu, TU Delft, daily supervisor

TU Delft

Acknowledgment

I would like to express my sincere gratitude to my daily supervisor, Kai, who provided invaluable support during challenging moments in the implementation phase of this project. His guidance pulled me out of trouble more than once, and he was always available for feedback, no matter the situation. His commitment truly made a difference.

I also want to thank my supervisor, Jun, for not only helping me with the technical aspects of the project but also for sharing his experiences in the academic world. Jun's advice went beyond the project itself, giving me insight into what it truly means to be a researcher.

My friends, both near and far, family, and even my "acquired family" at K161, deserve a huge thank you. To those in Delft and those supporting me from afar, you made this experience brighter and always kept my spirits high. I couldn't have asked for better support.

Reflecting on my thesis, I am happy to have chosen a challenging topic—perhaps I aimed too high in hindsight, but I wouldn't have it any other way. I am especially grateful to my supervisor for allowing me the freedom to pursue side experiences, such as working as a teaching assistant and participating in Google Summer of Code. Though these commitments slowed down my thesis progress, they enriched my learning journey immensely.

There are undoubtedly many things I might have done differently, but the mistakes and challenges were all part of the process. Without them, I wouldn't be here reflecting on my growth. I am thankful for the opportunity to experience what it means to be a researcher, to take on a project, make it my own, and learn so much along the way.

Riccardo di Girolamo
November 2024

Abstract

Additive Manufacturing (AM) plays a crucial role in the revolution towards Industry 4.0, by enabling the direct translation of digital 3D models into physical objects while reducing process steps and minimizing human intervention. While conventional AM machines are generally limited to three-axis movement, multi-axis AM equipment extends the manufacturing flexibility by enabling the fabrication of freeform layers, thus creating new opportunities to improve part quality, though at the cost of increased complexity in process planning. Wire Arc Additive Manufacturing (WAAM) is a multi-axis technique for producing large metal components, with potential applications in maritime, aerospace and civil infrastructure. However, this potential is hindered by factors such as deformation during fabrication, which compromises part precision and can lead to process failure. Recently, a computational approach has been developed to reduce distortion by optimizing the fabrication sequence. While promising, the optimized sequence is characterized by large variations in layer thickness, rendering them non-manufacturable. This research proposes numerical methods to evaluate and restrict layer thickness in fabrication sequence optimization, ensuring uniform thickness within each layer and a consistent average thickness across the entire sequence. A 3D computational framework has been developed, integrating latest advancements in sequence optimization. To address the intensive computation in 3D, this framework features a parallel implementation using the PETSc library. This framework enables the numerical assessment of the method's performance and provides a foundation for future experimental validation.

Contents

1 Introduction

THE rapid advancements in manufacturing technologies are paving the way for Industry 4.0, where digitization and automation are key drivers. Additive Manufacturing (AM) fits seamlessly within this revolution by enabling the direct translation of digital 3D models into physical objects, significantly reducing process steps and minimizing human intervention [2], [3].

Technically, AM is defined as "*the process of joining materials to make objects from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing methodologies like machining*" [4]. By depositing material only where necessary, AM reduces waste compared to traditional methods, while its layer-by-layer approach offers unparalleled geometric freedom. These features make AM ideal for producing topology-optimized parts, which often feature complex geometries unattainable through conventional manufacturing techniques. Notable applications are found in the automotive, aerospace, and medical sectors, where lightweight yet strong components are in high demand [5]. However, the reliance on planar layers in traditional AM presents a limitation. The additional degrees of freedom in Multi-Axis Additive Manufacturing systems make it possible to transcend planar fabrication, allowing for the production of free-form layers. Although this added flexibility complicates process planning, it also provides opportunities to enhance part quality. Multi-axis robotic systems are capable of depositing both polymers and metals, as seen in material extrusion for polymers [6], and Wire Arc Additive Manufacturing (WAAM) for metals [7].

Process planning for Additive Manufacturing is a complex, multi-scale process [8]. At the *part scale*, it involves defining the overall construction sequence and assembly order. At the *track scale*, the focus shifts to tool-path planning, which determines the precise paths and movements of the fabrication tool. Finally, at the *bead scale*, control adjustments are made to parameters such as temperature and build speed. Each scale provides opportunities to optimize and improve manufacturing quality.

Several studies have focused on Multi-Axis process planning at the part scale. Chakraborty et al. [9] developed a slicing method which by using non-planar layers in the final stage of manufacturing improves the surface finish, overcoming the characteristic stair-stepping effect. Fang et al. [6] designed a slicing algorithm for 5-axis machines, optimizing layer shapes and sequence to reinforce parts against specific load cases, leveraging the anisotropic properties of AM. Similarly, Zhang et al. [10] optimized slicing for 6-axis machines, using curved layers to achieve, concurrently, support-free builds, smoother surface finishes, and stronger parts.

A particular challenge in Metal AM is process-induced deformation, occurring both during fabrication and upon part separation from the substrate [7], [11]. These deformations compromise part precision—a difficult issue to correct—and can even result in complete process failure. Research on deformations in Metal AM has primarily focused on Powder Bed Fusion (PBF) technologies. For example, Xu et al. [5] proposed a 2D topology optimization method incorporating self-supports and residual stress constraints, demonstrating that reduced residual stresses lead to lower deformations and emphasizing the importance of optimal build direction. Similarly, Misiun et al. [12] examined PBF simulation methods for designing topology-optimized structural parts, addressing in-process deformations that risk recoater collisions and global deformations following part separation from the base plate. While effective, these approaches are primarily design-phase methods, relying on topology optimization, which limits their applicability when geometry is predetermined, and the focus shifts to manufacturing.

Process-induced deformations in Direct Energy Deposition (DED) technologies, such as Wire Arc Additive Manufacturing (WAAM), have been relatively underexplored. Multi-axis AM systems, including WAAM, offer additional degrees of freedom that can be leveraged to adjust layer shapes and deposition sequences at the *part scale*—analogous to optimizing scan strategies in PBF at the *track scale*. These adjustments enable more uniform thermal distributions, reducing stress accumulation and minimizing deformation. Wang et al. [1] were among the first to explore this approach, developing a method that optimizes layer shapes and sequences to minimize process-induced deformation. Importantly, this method is applied in the post-design phase without altering the part's geometry or requiring additional material. Moreover, it is compatible with other optimization techniques, such as topology or tool-path optimization. An example of this compatibility is seen in Wang et al.'s [13] concurrent approach, which combines structural layout optimization with fabrication sequence optimization.

While promising, the method proposed by Wang et al. [1], [13] often produces fabrication sequences that do not meet the manufacturability limits of Additive Manufacturing equipment. This research proposes an extension aimed at enhancing the manufacturability of optimized sequences by focusing on layer thickness control, ensuring each layer maintains uniform thickness within itself, and that a consistent average thickness is achieved across the entire sequence. In Direct Energy Deposition equipment, achieving precise deposition control requires continuous adjustments to process parameters such as travel speed and power to maintain consistent quality. Ensuring uniformity of layer thickness within each layer is therefore favorable for effective quality control and to align with the equipment's capabilities. Building on the 2D formulation proposed by Lansu [14], we developed an improved formulation for 3D optimization, and assessed its performance through numerical experiments. We also incorporated the latest regularization technique introduced by Wang et al. [15]. Additionally, we created a parallel implementation using the PETSc library, providing a valuable tool for future experimental validation.

This report is structured as follows, Section 1 begins with an introduction to the context and motivations behind the study, outlining the core challenges in multi-axis additive manufacturing and highlighting the gaps in existing approaches. Section 2 builds on this by reviewing the relevant background knowledge, offering a comprehensive examination of process planning, advanced slicing techniques, and recent developments in fabrication sequence optimization. This sets the stage for Section 3, where the methodology is presented in depth, covering the 3D optimization framework, including the discretization of parts, the representation of the pseudo-time field, and the process of layer extraction. In Section 4, we delve into the implementation details, focusing on the parallel computation strategy using the PETSc library, along with the process simulation validation. Section 5 then presents and discusses the numerical results, providing insights into optimization behavior across different part geometries, exploring the impact of key parameters, and assessing computational performance. Finally, Section 6 draws together the key findings, addresses the current limitations, and outlines future research directions that could build on the advancements made in this work.

2 Background Knowledge and Related Work

This section provides the foundational knowledge and a review of related work necessary to contextualize this research. First, in Subsection 2-A, the challenges and process planning complexities of Multi-Axis Additive Manufacturing, with a particular focus on Wire Arc Additive Manufacturing (WAAM), are introduced. This includes an analysis of process-induced deformations and existing strategies to mitigate these issues. Next, Subsection 2-B reviews slicing methods for Multi-Axis Additive Manufacturing, highlighting notable techniques that utilize free-form layers to enhance part quality and mechanical properties. In Subsection 2-C, the original Fabrication Sequence Optimization method is introduced, along with recent advancements focused on enhancing the manufacturability of the resulting sequences. Finally, Subsection 2-D provides an overview of a PETSc-based parallel framework, which serves as the foundation for the implementation developed in this research to address large-scale optimization challenges.

2-A Challenges and Process Planning in Multi-Axis Additive Manufacturing

As introduced earlier, Additive Manufacturing (AM) offers significant advantages over traditional methods, particularly in producing complex geometries with minimal material waste. This capability is crucial for efficiently creating topology-optimized parts, which are highly sought after in the automotive, aerospace, and medical sectors, where lightweight yet durable components are essential [16], [3]. Multi-Axis Additive Manufacturing equipment further extends this capability by transcending planar layer limitations, increasing process flexibility and creating new opportunities to optimize part quality, though at the cost of more complex process planning. This section describes the characteristics of Multi-Axis additive manufacturing equipment, introduces Multi-Axis metal AM, and analyzes the quality challenges it presents. Additionally, relevant strategies from the literature for predicting and managing these quality issues will be discussed.

Conventional AM machines are generally limited to three-axis movement, with the z-axis primarily serving to transition between planar layers; hence, these systems are often designated as *2.5D*. Multi-axis equipment, by contrast, refers to systems that extend beyond *2.5D* fabrication. By leveraging additional degrees of freedom, one type of fabrication—known as multi-directional AM—still uses planar layers but allows the build direction to change between layers. This method already offers several advantages over traditional *2.5D* fabrication, such as reducing the stair-stepping effect and enabling support-free fabrication. Fully utilizing multi-axis capabilities enables the fabrication of complex geometries through *freeform layers*, which have been shown to improve surface quality [9], enhance mechanical performance [6], and elevate overall part quality. Examples of multi-axis AM systems are shown in Figure 1. While conventional planar slicing, and even adaptive slicing, are relatively straightforward tasks, the Fabrication Sequence Planning for multi-axis additive manufacturing is significantly more complex. For instance, achieving a specific curvature within a layer often requires its thickness to be varied. However, the extent to which layer thickness can be varied within a layer depends on the capabilities of the specific equipment being used. Moreover, both in multi-directional AM and free-form fabrication, there is a risk of collision between the print head and the already fabricated part—an important consideration to keep in mind during Fabrication Sequence planning.

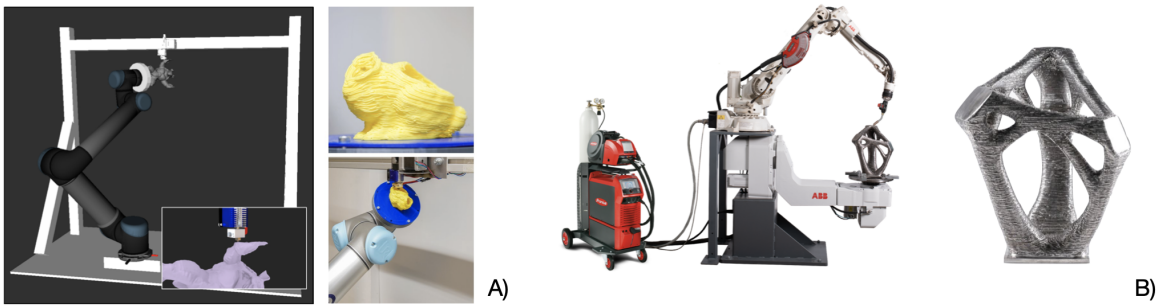


Fig. 1: Examples of Multi-Axis Additive Manufacturing Systems. A) A six-degree-of-freedom setup using Fused Deposition Modeling, demonstrating the fabrication of an object (in yellow) with free-form layers (Dai et al. [17]). B) A Wire Arc Additive Manufacturing setup, consisting in a welding torch attached to a six-degree-of-freedom manipulator. The manufactured part illustrates the characteristic poor surface finish of this method (Image obtianed from the MX3D website: [18]).

As can be evinced, Additive Manufacturing processes, and Multi-Axis AM processes in particular, are complex,

involving an intricate interplay of many parameters, which makes process planning a challenging task. In this context, process planning is inherently multi-scale [8], requiring detailed control across various levels. At the part-scale, exemplified by fabrication sequence planning, the primary focus is on the overall construction sequence and assembly order of the component. Moving to the track-scale, the emphasis shifts to tool-path planning, where the precise paths and movements of the fabrication tool are meticulously determined. At the bead-scale, the process involves even finer control, focusing on the adjustment of specific parameters such as temperature and build speed. Each scale, from part to bead, is crucial, and can be potentially optimized to improve manufacturing quality.

As noted in the introduction, a metal additive manufacturing technology compatible with multi-axis systems is Wire Arc Additive Manufacturing (WAAM) [7]. WAAM, a variant of Direct Energy Deposition (DED), uses an electric arc as the heat source and wire as the feedstock material. The wire is melted and deposited precisely where needed, building the part layer by layer. WAAM equipment typically consists of a welding torch mounted on a multi-axis robotic arm, allowing for the printing of non-planar layers [16]. This method is particularly effective for large-scale metal components due to its high deposition rate and relatively low equipment costs, making it well-suited for rapid production. While WAAM parts exhibit good mechanical properties, post-processing is often required to achieve a smooth surface finish. A significant challenge, particularly prominent in metal additive manufacturing are *process-induced deformations*. Such deformations can compromise part precision and, in severe cases, lead to complete process failure. To mitigate this problem, and advance metal AM toward becoming a fully viable production technology [3], it is essential first to understand the underlying mechanisms behind it. In the review article by Xie et al. [11] different researches have been analyzed to understand the common causes of deformations and residual stresses across different AM technologies, including Selective Laser Melting, Fused Deposition Modeling, Stereolithography, and Digital Light Processing. The primary cause of this issue in additive manufacturing has been identified as non-uniform material expansion/contraction. This is influenced by the material's thermal and mechanical properties, as well as by the specific deposition or scan strategy employed. High energy input in additive manufacturing processes has been linked to increased residual stress and a higher occurrence of deformations. This correlation helps explain why metal Additive Manufacturing in particular is more prone to deformation. In their study on Direct Energy Deposition technology with a focus on Wire Arc AM, Jafari et al. [8] observed that the high energy input in Wire Arc AM leads to the melting of substantial material amounts. As this big portion of material solidifies and shrinks, it contributes to accumulation of stress and distortions. From this stress state, the part would naturally try to reach an equilibrium by macroscopic distorting the deposited material, or, if the structure cannot macroscopically distort, the stresses may lead to microscopic deformations (such as yielding or cracking). Residual stresses might remain also after the completion of the manufacturing process. These often result in component distortion, which becomes evident when the component is released from the substrate, as displayed in Figure 2.

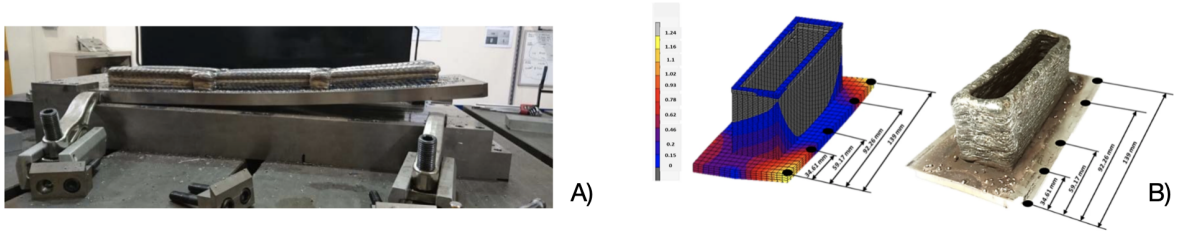


Fig. 2: Example of process-induced distortion in Wire Arc Additive Manufacturing. A) Distortion of the part and substrate observed at the end of the process, after being unclamped (Xu et al. [19]). B) Visual comparison between simulated and experimental distortion of a part, observed post-process after unclamping. A sequentially coupled simulation was used to model the process (Manurung et al. [20]).

Different methods have been proposed in literature to reduce process-induced residual stresses and deformations. Xu et al. [5] introduced an innovative approach in their study by proposing a 2D topology optimization that integrates self-supports and aggregated residual stress constraints for a Powder Bed Fusion process. Their findings revealed an intriguing correlation: imposing a limit on the maximum residual stress not only achieved its intended purpose but also inadvertently reduced residual distortion. Furthermore, they identified that selecting an optimal build direction was crucial in mitigating stresses without adversely impacting the stiffness performance of the structure. In a similar study, Misiun et al. [12] focused on identifying an effective Powder Bed Fusion process simulation method to design topology optimized structural parts with

enhanced manufacturability. Their research addressed both in-process deformations, which could lead to re-coater collisions, and global deformations occurring post-separation of the part from the base plate. The optimized designs often incorporated modified shapes and support-like structures. Notably, the study observed that deformations during the process were counterbalanced by additional distortions post-separation. Another innovative approach to mitigate deformation involves adjusting the part's geometry during pre-processing, informed by an understanding of the specific process and deformation patterns. This technique has been successfully implemented by Relativity Space in building fully additive manufactured rockets [21]. While the methodologies proposed by Xu et al. [5] and Misiun et al. [12] are promising, they primarily operate in the design phase and involve topology optimization, making them less applicable when the geometry is predetermined and the focus is solely on manufacturing. Additionally, both methods result in the creation of support structures, which, despite their benefits in stress and deformation reduction, lead to increased material wastage. It is also noteworthy that most research focusing on deformation issues, including these studies, is centered on Powder Bed Fusion technology, with relatively limited exploration in the field of Direct Energy Deposition.

An important step in the presented studies, is to perform computational simulations to predict, control, and potentially exploit these side-effects of the process [22]. However, as previously discussed, the occurrence of deformations in metal additive manufacturing processes is influenced by a complex interplay of factors. These include specific process parameters and the unique mechanical and thermal properties of the materials employed. Such intricacies make process simulation particularly challenging. There are three primary methods for simulating Additive Manufacturing processes: Fully Coupled Simulation, Sequentially Coupled Simulations, and Mechanical Simulations (usually with the Inherent Strain Method).

- In **Fully Coupled Simulations**, thermal and mechanical equations are solved simultaneously. This approach accounts for how temperature variations impact material properties and, in turn, how these changes influence temperature distributions. Such simulations are highly accurate, capturing the interdependencies between thermal and mechanical phenomena. However, they are computationally demanding. At times, these simulations also encompass multi-physics aspects, including fluid dynamics, to predict the behavior of materials in the molten pool [23].

- In **Sequentially Coupled Simulations**, the process begins with thermal analysis, where a time-evolving temperature field is calculated. These thermal results are then utilized as inputs for subsequent mechanical analysis [24]. The equations in these simulations are solved in a sequential order, thereby capturing the coupling between thermal and mechanical phenomena in a unidirectional manner. While the results of sequentially coupled simulations are generally less accurate than those from fully coupled simulations, they offer greater computational efficiency.

- The **Inherent Strain Method** instead, does not require a thermal simulation, but only focuses on the mechanical aspect. It was firstly developed to predict deformations in metal welding by Umeda et al. [25], and later adapted to AM. It considers the thermal shrinkage as a prescribed mechanical strain associated to the deposition of each layer [12]. Equivalent nodal forces are computed from these strains and thus the macroscopic displacement field can be derived. This method is very efficient but has low accuracy. However, time-wise often it is the only feasible option for optimization problems. The Inherent strain values are usually obtained through experimentation or accurate fully coupled simulations.

The studies by Xu et al. [5] and Misiun et al. [12], along with most research addressing optimization problems to minimize process-induced deformations, primarily utilized the Inherent Strain Method due to its computational efficiency.

To understand the inherent strain method, consider the melting and solidification process of a material portion. As it cools, the material solidifies and undergoes volumetric shrinkage, as schematically illustrated in Figure 3 for a two-dimensional case [1]. This shrinkage is represented by a thermal strain tensor, $\underline{\varepsilon}^*$, whose values, as mentioned earlier, can be obtained through experimental data or detailed process simulations. This tensor is then used to compute the element's equivalent nodal forces, which are subsequently assembled into the right-hand side of the Finite Element Method (FEM) linear system. The Equation used is $\boldsymbol{f}_e(\underline{\varepsilon}^*) = [\boldsymbol{D}]^\mathsf{T}[\boldsymbol{C}]\underline{\varepsilon}^*$ where $\boldsymbol{D}$ is the strain–displacement relation matrix, and $\boldsymbol{C}$ denotes the elasticity matrix. Anisotropic shrinkage can also be accounted for by scaling the values of the inherent strain tensor according to the deposition direction [1].

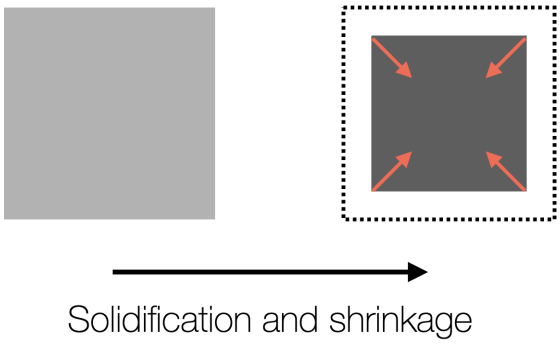


Fig. 3: Schematic illustration of material cooling and the resulting shrinkage process.

2-B Overview of Relevant Slicing Methods

As outlined in Section 2-A, Process Planning in Additive Manufacturing operates across multiple scales, with slicing being a fundamental component at the part scale. Traditional slicing algorithms divide the design into planar layers. However, Multi-Axis Additive Manufacturing systems enable the division of the part into non-planar, free-form layers. Although this additional flexibility adds complexity to process planning, it provides opportunities to improve part quality and address challenges associated with planar deposition, such as reducing or even eliminating the reliance on support structures. This section reviews notable slicing algorithms for multi-axis systems documented in the literature.

1) Reinforced FDM: Multi-Axis Filament Alignment with Controlled Anisotropic Strength
One inherent challenge in Additive Manufacturing, closely tied to its layer-wise fabrication, is the anisotropy of mechanical properties in the manufactured part. Additively Manufactured parts, indeed, exhibit anisotropic mechanical properties, with higher strength within the layer plane (in-plane strength) and reduced strength between layers (out-of-plane strength) due to weaker interlayer bonding. This anisotropy is present in both planar and free-form layer fabrication. However, free-form layers offer the opportunity to mitigate this limitation by tailoring layer shapes and orientations. Specifically, the tangent to each layer can be locally aligned with the principal stress directions, allowing the anisotropy to be leveraged for reinforcement. By optimizing the deposition paths and layer geometries, the part can better withstand specific mechanical loads, enhancing its overall structural performance. This is exactly what Fang et al. [6] did in their study on *Reinforced FDM: Multi-Axis Filament Alignment with Controlled Anisotropic Strength*. This study is particularly interesting for its approach to encoding the Fabrication Sequence and for its consideration of multiple objectives—not only to ensure part strength but also to prioritize manufacturability. In fact, the study emphasizes that manufacturability is considered more crucial than reinforcement itself.

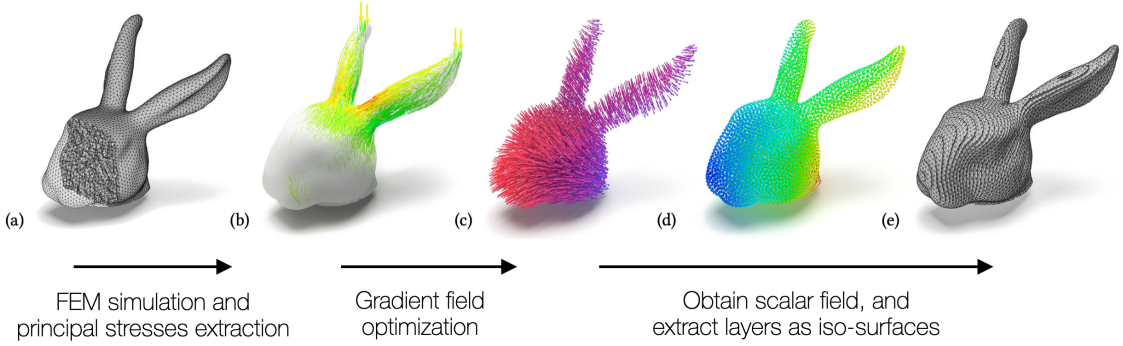


Fig. 4: (a) The bunny-head model $\mathcal{H}$ is represented by a tetrahedral mesh $\mathcal{T}$. (b) Principal stress values are visualized using color mapping to highlight stress distribution within the model. (c) A vector field $V(\boldsymbol{x})$ is optimized according to reinforcement principles and fabrication constraints to ensure both structural strength and manufacturability. (d) A scalar field $G(\boldsymbol{x})$ is subsequently generated by enforcing the gradient $\nabla G(\boldsymbol{x})$ to align with the optimized vector field $V(\boldsymbol{x})$. (e) Preliminary curved layers are produced by extracting iso-surfaces from the scalar field $G(\boldsymbol{x})$, forming the basis for the fabrication sequence. Image source: Fang et al. [6].

In this method, the Fabrication Sequence is represented by a scalar field $G(\boldsymbol{x})$, from which layers can be extracted as iso-surfaces, as illustrated in Figure 4 (d) and (e). The field $G(\boldsymbol{x})$ is optimized to ensure that the part withstands a specific applied load.

The method begins by discretizing the part $\mathcal{H}$ into a tetrahedral mesh $\mathcal{T}$. A commercial FEA software (Abaqus) is then used to simulate the application of the load that the part must endure. This simulation outputs a stress distribution, represented as a 3×3 stress tensor $\sigma(e)$ for each tetrahedral element $e \in \mathcal{T}$, centered at the positions $\boldsymbol{x}_e$.

From these stress tensors, the principal stresses $\sigma_1, \sigma_2, \sigma_3$ and their associated eigenvectors are obtained through eigenvalue decomposition, where $\sigma_1 > \sigma_2 > \sigma_3$. Among all tetrahedrons in $\mathcal{T}$, a subset $\mathcal{T}^*$, referred to as the critical region, is identified. An element belongs to this subset if it meets two conditions: $|\sigma_1| > k_1|\sigma_{\text{max}}|$ and $|\sigma_2| > k_2|\sigma_3|$, where k_1 and k_2 are tunable parameters. Here, σ_{max} represents the maximum principal stress magnitude across all elements in the domain. This filtering process isolates regions within $\mathcal{H}$ where stresses exceed a critical threshold, and the first two principal stresses are sufficiently larger than the third.

For each tetrahedron in $\mathcal{T}^*$, the direction τ_{min}, corresponding to the smallest principal stress σ_3, is identified. The condition for the optimal scalar field gradient is then defined as $\nabla G(\boldsymbol{x})$ being as much as possible parallel to τ_{min}, leading to the objective formulated in Equation 1.

$$E_s = \sum_{e \in \mathcal{T}^*} V_e \|\nabla G(\boldsymbol{x}_e) \times \tau_{\text{min}}(e)\|^2, \tag{1}$$

where V_e is the element's volume. Instead of optimizing $G(\boldsymbol{x})$ directly, a vector field $\boldsymbol{V}(\boldsymbol{x})$ is optimized, represented in discrete form by storing one vector for each element $e \in \mathcal{T}$. Although the optimization method involves additional complexities beyond the scope of this research, an essential aspect to consider is *how layer thickness is controlled to ensure manufacturability*. Indeed, due to the limitation of currently available hardware, the thickness of each layer that can be Additively Manufactured has a limited range $[d_{min}, d_{max}]$. This is considered in the optimization through the additional objective in Equation 2,

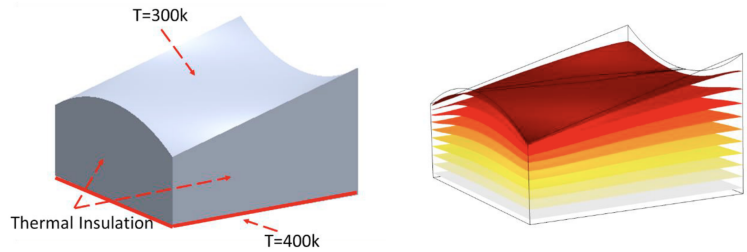
$$E_t = \sum_{e \in \mathcal{T}} V_e \left(\|\nabla G(\boldsymbol{x}_e)\| - c\right)^2 \quad \rightarrow \quad \{g_i\} = \arg\min \sum_{e \in \mathcal{T}} w_e \|\nabla G(\boldsymbol{x}_e) - \hat{\boldsymbol{v}}_e\|^2, \quad \hat{\boldsymbol{v}}_e = c\frac{\boldsymbol{v}_e}{\|\boldsymbol{v}_e\|}. \tag{2}$$

In the above equation, g_i represents the nodal values of $G(\boldsymbol{x})$. The equation at the right of the arrow actually illustrates how the objective is formulated: the difference between the gradient of the field, $\nabla G(\boldsymbol{x})$, and the optimized vector field, $\boldsymbol{V}(\boldsymbol{x})$, is minimized. The vector field $\boldsymbol{V}(\boldsymbol{x})$ is normalized and scaled by a factor c, which ensures that both the direction and layer thickness are accounted for in the optimization. This approach effectively aligns the gradient of the scalar field $G(\boldsymbol{x})$ with the desired vector field $\boldsymbol{V}(\boldsymbol{x})$, controlling layer orientation and thickness simultaneously.

2) Curved Layer Slicing based on Isothermal Surface

Another notable slicing technique is *Curved Layer Slicing based on Isothermal Surface* by Shan et al. [26]. Their research aims to leverage the increased degrees of freedom provided by Multi-Axis Additive Manufacturing machines to enhance the surface finish of parts. The method generates curved layers by solving a heat transfer problem, as outlined in Equation 3. The solution to this problem creates a smooth and continuous temperature field, which is well-suited for representing a fabrication sequence.

In this approach, two types of boundary conditions are applied: Dirichlet boundary conditions are imposed on the top and bottom surfaces, constraining these surfaces to be treated as single layers, while Neumann boundary conditions are applied to the sides, designating them as thermally insulated. Similar to the method proposed by Fang et al. [6], layers in this approach are represented as iso-surfaces extracted from a scalar field. However, since this scalar field is derived from solving a heat equation, its gradient is not uniform. Consequently, the threshold values for extracting iso-surfaces must be carefully selected to ensure a consistent distance between layers, thereby achieving uniform layer thickness. The strategy to select these threshold values can be obserevd in Equation 4.



$$\rho C_p \frac{\partial T}{\partial t} = k\Delta T \quad (3)$$

$$T_i = T_{i-1} - d \cdot |\Delta T| \quad (4)$$

Fig. 5: Visualization of the boundary conditions applied to the domain and the resulting fabrication sequence produced by extracting the iso-surfaces from the temperature field. Image source: Shan et al. [26].

An important aspect observed in Equation 4 is that, given a target layer thickness value d, the average gradient magnitude within a layer can be used to approximate the threshold value T_i for the next iso-surface, ensuring that it lies at a distance d from the previous one.

2-C Fabrication Sequence Optimization

As outlined in the previous section, Direct Energy Deposition—and specifically, Wire and Arc Additive Manufacturing technologies—can leverage Multi-Axis Equipment to go beyond planar layers, enabling the deposition of free-form layers. Although this approach requires more complex process planning, it presents a significant opportunity. By adjusting layer shapes and deposition sequences, similar to optimizing scan strategies in Powder Bed Fusion, a more uniform thermal distribution within the part can be achieved, thereby minimizing deformation and stress buildup issues highlighted in section 2-A. Building on these principles, Wang et al. [1] provides foundational research on Fabrication Sequence Optimization for Multi-Axis AM processes, focusing on optimizing the shape and sequence of free-form layers to minimize process-induced deformations. This section discuss one study targeting the mitigation of thermal-induced deformations through optimized fabrication sequences [1], covering also a recently introduced regularization technique [15], which has also been integrated into the present study.

1) Fabrication Sequence Optimization Base Formulation

To understand the *Fabrication Sequence Optimization* method, the first step is to understand how the additive manufacturing process sequence is represented. The process begins by defining a design domain, which is discretized into a 2D grid of square elements. Within this discretized domain, the structural layout of the part to be manufactured is encoded in a density vector, $\boldsymbol{\rho}$, where each element is assigned a binary density value — 1 or 0 — to indicate the presence or absence of material at a given point within the domain. A continuous scalar field called the *pseudo-time field*, denoted by $\boldsymbol{t}$, is then defined over the previously established structural layout. Here, each element posses a pseudo-time variable $t_e \in [0, 1]$, where this interval is normalized: 0 represents the start and 1 the end of the fabrication process. The values $t_e \in [0, 1]$, one for each element in the domain, are taken as design variables in this optimization.

In the research by Wang et al. [1], the optimization objective is to minimize process-induced deformations. This requires simulating the manufacturing process at each iteration step. Following the options outlined in Section 2-A, and in line with the majority of the literature, Wang's study employs the Inherent Strain Method. Moreover, elements are not treated individually but are grouped into layers, allowing the simulation to model the deposition of entire layers simultaneously. This approach is less accurate but more computationally efficient, aligning with similar techniques presented in the literature [22]. The number of process steps, and thus of layers N is prescribed. To delineate these layers, the pseudo-time field $\boldsymbol{t}$ is divided into intervals $[T_{j-1}, T_j]$, with $j = 1, \ldots, N$ and $T_j = \frac{j}{N}$, where N is the total number of layers. An element is included in a particular layer if its pseudo-time variable falls within the time interval associated with that layer. To partition the pseudo-time field $\boldsymbol{t}$, and thus extract the layers in a differentiable way, thus suitable for gradient-based optimization methods, a smooth Heaviside Projection displayed in Equation 5 is used.

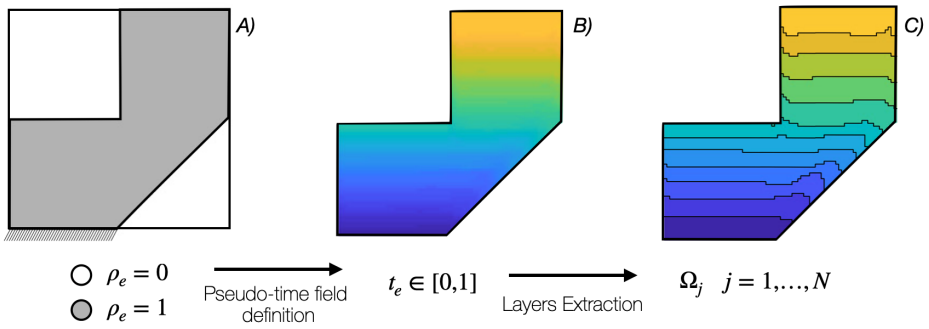


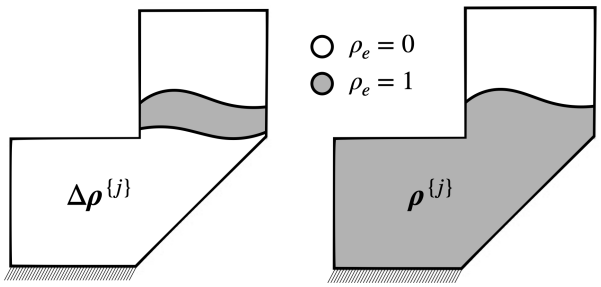
Fig. 6: Visualization of three steps in the Fabrication Sequence Optimization process. A) Encoding of the part's structural layout in the density vector $\boldsymbol{\rho}$. B) Definition of the pseudo-time field $\boldsymbol{t}$ based on the established structural layout. C) Extraction of an N-layer sequence from the time field through equispaced time intervals.

$$\bar{t}_e^{[T]} = 1 - \frac{\tanh(\beta_t T) + \tanh(\beta_t (t_e - T))}{\tanh(\beta_t T) + \tanh(\beta_t (1 - T))} \tag{5}$$

In the above Equation, T represents the threshold value of the projection, while β_t represents its sharpness.

This projection function is used to compute two density vectors: $\boldsymbol{\rho}^{\{j\}}$ and $\Delta\boldsymbol{\rho}^{\{j\}}$. Here, $\boldsymbol{\rho}^{\{j\}}$ represents the entire structure manufactured up until the jth time-step, meaning every element deposited before time-step T_j has a value of $\rho_e \approx 1$. In contrast, $\Delta\boldsymbol{\rho}^{\{j\}}$ represents the density distribution of the single layer deposited at the jth time-step, with non-zero density values assigned to elements where t_e lies within the interval $[T_{j-1}, T_j]$.

These two vectors $\boldsymbol{\rho}^{\{j\}}$ and $\Delta\boldsymbol{\rho}^{\{j\}}$ are then used to assemble respectively the stiffness matrix $\boldsymbol{K}$ and the forcing vector $\boldsymbol{f}$, through a penalization scheme.



$$\begin{aligned} \boldsymbol{\rho}^{\{j\}} &= \boldsymbol{\rho} \cdot \bar{\boldsymbol{t}}^{\{j\}}, \\ \Delta\boldsymbol{\rho}^{\{j\}} &= \boldsymbol{\rho} \cdot (\bar{\boldsymbol{t}}^{\{j\}} - \bar{\boldsymbol{t}}^{\{j-1\}}) \end{aligned} \tag{6}$$

$$\boldsymbol{K}_e(\rho_e) = (E_{\min} + (1 - E_{\min})\rho_e^p)\, \boldsymbol{K}_0, \tag{7}$$

$$\Delta\underline{\varepsilon}^{*\{j\}}(t)\big|_e = \left(\Delta\rho^{\{j\}}(t)\big|_e\right)^q \underline{\varepsilon}^*. \tag{8}$$

Fig. 7: Visualization of the density vectors $\boldsymbol{\rho}^j$ and $\Delta\boldsymbol{\rho}^j$, representing, respectively, the material deposited up to and including the j-th time interval and the material being deposited in the j-th time interval.

As shown in Equation 7, each element begins with the same elemental stiffness matrix $\boldsymbol{K}_0$, which is then scaled according to its density value $\rho_e \in \boldsymbol{\rho}^{\{j\}}$. Similarly, Equation 8 illustrates the penalization applied to the elemental inherent strains, where each element starts with the same inherent strain vector $\underline{\varepsilon}^*$, subsequently scaled by the density value $\Delta\rho^{\{j\}}\big|_e \in \Delta\boldsymbol{\rho}^{\{j\}}$. The parameters p and q serve as the penalization factors for the stiffness and force vectors, respectively, and they satisfy the condition $p \leq q$. This penalization scheme effectively applies forces that approximate the thermal loads generated by the deposition of layer j onto the structure deposited up to the j-th manufacturing step.

As can be evinced from Equation 7, all elements of the domain contribute to the stiffness matrix $\boldsymbol{K}^{\{j\}}$, the void and not-yet-fabricated elements are treated as *quiet elements*. This means they possess a very low stiffness value to prevent the matrix from becoming singular. Consequently, the dimensions of the equilibrium equation remain constant. The Equilibrium Equation 9 is applied thus to the entire domain. Assuming small displacement, the total displacement field $\boldsymbol{u}$, is determined by aggregating the incremental displacements $\boldsymbol{\Delta u}^{\{j\}}$.

$$\boldsymbol{K}^{\{j\}}\boldsymbol{\Delta u}^{\{j\}} = \boldsymbol{f}^{\{j\}}, \quad j = 1, \ldots, N, \rightarrow \boldsymbol{u} = \sum_{j=1}^{N} \boldsymbol{\Delta u}^{\{j\}} \tag{9}$$

With the displacement field at hand the objective function can be computed as in Equation 10, where $\boldsymbol{Q}$ is a matrix which combine the nodes displacements to compute the desired distortion metric.

$$J = \boldsymbol{u}^{\boldsymbol{T}}\boldsymbol{Q}\boldsymbol{u} \tag{10}$$

This optimization includes two types of constraints: a *Continuity Constraint* and a *Speed Constraint*. The Continuity Constraint enforces continuity in the pseudo-time field $\boldsymbol{t}$, preventing the formation of local minima and maxima, which would render the results non-manufacturable. Local minima would indicate areas requiring support structures, while local maxima would represent enclosed voids. The Speed Constraint ensures that an equal volume of material is deposited within equal time spans, effectively preventing layers from being emptied during optimization.

Once the objective, constraints, and their sensitivities are calculated, the optimizer—here, the Method of Moving Asymptotes—computes [27] the updated design variables $\boldsymbol{t}$.

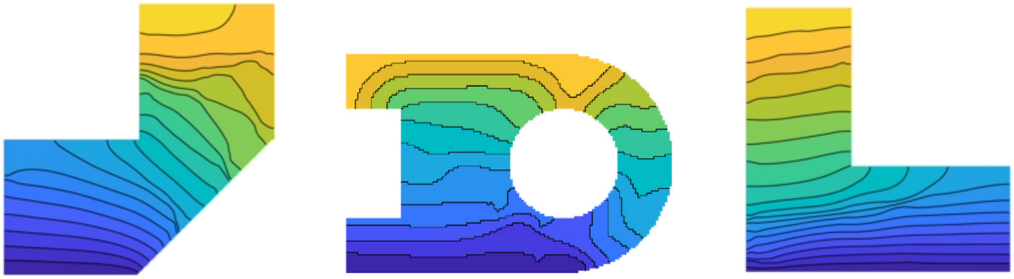


Fig. 8: Examples of optimized fabrication sequences obtained using the formualtions from Wang et al. [1], [15]. The sequences present visibly irregular layer shapes.

2) Fabrication Sequence Optimization with Layer Thickness Control

Analyzing the results of the formulation presented in Wang et al. [1], [15], as shown in Figure 8, reveals that although the distortion objective is minimized, the resulting layers are too irregular in shape to be realistically manufacturable. To address this, Lansu [14] proposed a modified formulation that improves control over layer thickness and uniformity.

To the distortion objective presented in Equation 10, the modified formulation adds a secondary objective, referred as *Layer Uniformity Objective* which can be observed in Equation 11.

$$L = \alpha \sum_{j=1}^{N} L_j \quad \text{with} \quad L_j = \sum_{e \in \mathcal{T}} \Delta\rho_e^{\{j\}} \left(M_e - m_j\right)^2 \tag{11}$$

In the equation, $\Delta\rho_e^{\{j\}} \in \Delta\boldsymbol{\rho}^{\{j\}}$ is computed in the same way as in Equation 6. Here, M_e represents the gradient magnitude value for element e, and m_j is the average gradient value for layer j. This objective, similar to the one presented in the research by Fang et al. [6], aims to minimize a quantity similar to the variance of the gradient magnitude across each layer, thereby promoting uniformity.

Two additional constraints are introduced: a *Layer Thickness Constraint* and a *Minimum Volume Constraint*. The Layer Thickness Constraints include $2 \times N$ individual constraints, two for each layer. As shown in Equation 12, the thickness of a layer is approximated by using the average gradient magnitude value m_j and the time interval $[T_{j-1}, T_j]$ corresponding to the jth layer. This approach is similar to the method used in the Isothermal Surface Slicing Technique by Shan et al. [26].

$$d_{min} \leq d_j \leq d_{max} \quad \text{with} \quad j = 1, \ldots, N \tag{12}$$

$$d_j = (T_j - T_{j-1})\frac{1}{m_j} \quad \text{with} \quad j = 1, \ldots, N \tag{13}$$

However, adding the *Layer Thickness Constraints* alongside the *Speed Constraints* over-constrains the optimization, as fixing both the layer volume and layer thickness inadvertently constrains the "length" dimension as well. To accommodate both constraints, the time steps T_j, where $j = 1, \ldots, N$, are also treated as design variables. This adjustment necessitates an additional constraint—the *Minimum Volume Constraint*—to prevent overlap between the time steps values.

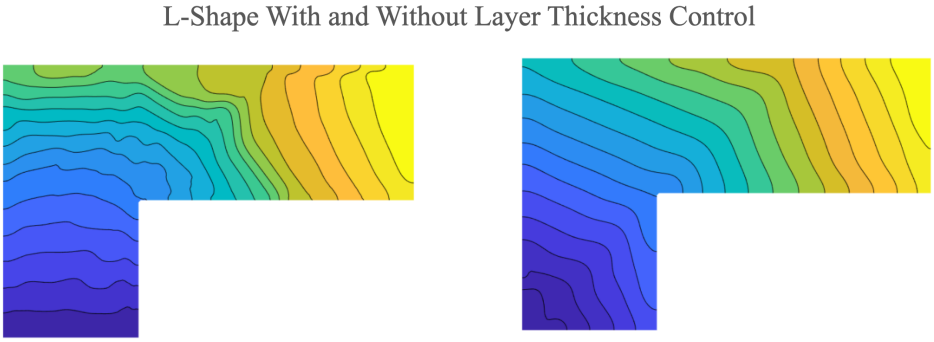


Fig. 9: Comparison between two optimized Fabrication Sequences: one obtained without layer thickness control (left) and the other obtained using the modified formulation from Lansu [14] for layer thickness control (right).

In Lansu's work [14], the method appears to successfully control layer thickness in the presented results (Figure 9); however, the results are limited, raising concerns about the robustness of the approach. In the basic formulation, the role of each constraint was clearly defined, but in this modified formulation, the individual roles are less transparent, making it difficult to determine whether any constraints might be counteracting one another. For example, in the original formulation, the speed constraint was used to prevent layers from shrinking. However, in this modified approach, it is unclear why the speed constraint is still necessary, as the same effect seems intuitively to be achieved by the layer thickness constraint.

3) Fabrication Sequence Optimization with Thermal Regularization
From the results of the original formulation by Wang et al. [1], it can be observed that the Continuity Constraint does not always successfully prevent the formation of local minima and maxima, rendering some of those results non-manufacturable. To address this, a recent study by Wang et al. [15] introduced an improved approach using a regularization technique to more effectively enforce these critical properties in the field.

The regularization involves indirectly determining the pseudo-time field $\boldsymbol{t}$ by computing it as the solution to a Heat Equation, rather than optimizing $\boldsymbol{t}$ directly. This approach leverages the inherent properties of the heat equation's solution (Equation 14) to naturally enforce the required properties on the pseudo-time field. Thus, the new design variables are the diffusivities $\boldsymbol{\kappa}$ defined on the structural layout in the same way as the pseudo-time field $\boldsymbol{t}$. For the boundary conditions, a Dirichlet temperature condition is imposed on the base plate, while conduction is restricted to occur only within the structural layout. This setup imposes a Neumann boundary condition with no heat conduction on the outer boundaries of the geometry.

$$\nabla \cdot (\kappa \nabla \tau) - \alpha_T \tau = 0. \tag{14}$$

In the above Equation, τ represents the *virtual temperature* field, $\kappa(x, y, z)$ denotes the spatially varying *thermal diffusivity*, modeling how quickly heat diffuses through the material, and the term $\alpha_T \tau$ is a drain term, modeling heat loss in proportion to the virtual temperature. Fixing the base plate virtual-temperature at a value of 1 the pseudo time-field can be easily obtained from the virtual temperature field thorugh $\boldsymbol{t} = \mathbf{1} - \boldsymbol{\tau}$.

2-D Overview of The PETSc Topology Optimization Framework

Since this research primarily focuses on 3D large-scale Fabrication Sequence optimizations, a parallelized implementation is necessary. Given the similarity between the Fabrication Sequence optimization method and density-based topology optimization, existing tools from the literature on topology optimization can be repurposed effectively. Specifically, the fully parallel, open-source topology optimization framework using PETSc developed by Aage et al. [28] will be utilized. This section provides background information on the advantages of parallel computing and introduce the just cited framework.

1) Introduction to Parallel Computing

Parallel computing is fundamentally rooted in the concept that executing multiple arithmetic operations simultaneously, rather than sequentially, enhances performance [29]. Initially, parallel computing was predominantly utilized to expedite computations with high time and memory requirements. The potential for speed-up was initially described by Amdahl's law in 1967, which stated that the improvement in performance due to parallelization is limited by the fraction of the task that cannot be parallelized. However, Gustafson's law, introduced in 1988, presented a different perspective by focusing on increasing the problem size while maintaining a constant overall execution time. This approach implies that for larger problems, the impact of the non-parallelizable portion becomes less significant, allowing for better scalability and effective use of parallel computing resources.

Historically, parallel computing systems were largely dedicated to complex scientific problems. Today, however, they are integral to a wide range of computing devices available in the market [30].

Parallel computing systems are generally classified based on their memory organization: Shared Memory Systems and Distributed Memory Systems [31]. In Shared Memory Systems, while data sharing between processors is more straightforward, each processor must coordinate with others for memory access. This coordination often requires complex synchronization and can lead to delays, particularly when multiple processors access the same memory area at the same time. In contrast, Distributed Memory Systems comprise a network of nodes, each with its own memory, which allows for data transfers between nodes. While this arrangement mitigates direct access conflicts, it introduces potential delays due to inter-node communication.

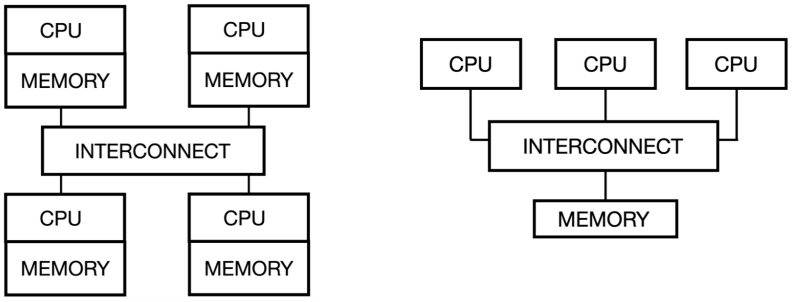


Fig. 10: Representation of Distributed Memory parallel architecture (on the left) and Shared Memory architecture (on the right)

Parallelization significantly improves memory usage efficiency, which in turn further influences and reduces overall computation time. By distributing tasks, data is processed in smaller segments across multiple processors or cores, allowing each processor to manage a smaller, more manageable memory portion. This approach helps to overcome the limitations of memory capacity per processor, while effectively increasing the system's overall memory capacity. Furthermore, as processors have become faster, largely due to increased transistor density (Moore's Law), they can process data at a much faster rate than what memory systems can supply. Thus, often a single processor is delayed waiting for data from memory. Parallel processing alleviates this bottleneck effect. With multiple processors operating together, more data can be handled simultaneously, thereby minimizing periods of inactivity [32].

In the ideal scenario of parallel computing, a process is fully divisible into independent sub-processes that run concurrently. This allows for theoretical maximum efficiency or linear speedup, where doubling the processors

could halve the computation time. However, achieving this is challenging due to communication overhead and synchronization issues.

2) Fully Parallelized Topology Optimization Framework using PETSc
A well known framework in the field of structural optimization is the one by Aage et al. [28], which provides a realatively easy to use and portable code, based on the open-source library PETSc, made for high performance and scientific computing. Although the optimization problem in this thesis research is not directly Topology Optimization, it shares many similarities in terms of formulation, which render the cited framework a valuable starting point for the implementation.

PETSc is an acronym for Portable and Extendable Toolkit for Scientific Computing. It includes most of the necessary building-blocks for writing, analyzing, and optimizing large-scale numerical simulations [33]. The main provided tools are sparse matrices, vectors, iterative linear solvers, non-linear solvers and time-stepping scheme, eliminating the need to write low-level math libraries and speed up the development process [28]. PETSc is mainly based on distributed memory systems and uses Message Passing Interface to manage communications between the nodes.

The framework developed by Aage et al. [28], is represented in Figure 11. In the picture, the red boxes represent fundamental parts of the framework, thus require very little to no modifications:

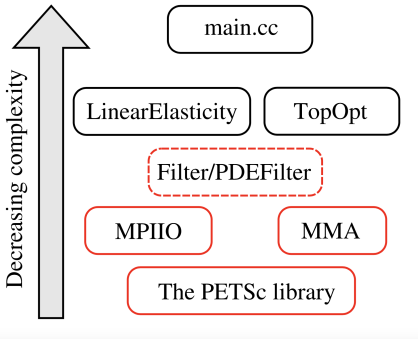


Fig. 11: Layout of the optimization framework. The names in the boxes refer to the main program and to the classes containing physics, optimization settings, filtering, output, MMA and the PETSc library. The arrow to the left indicates the complexity of the different components, i.e. the higher in the diagram the simpler the code

- The **MMA** Class contains a fully parallelized implementation of the Method of Moving Asymptotes (MMA) (Svanberg 1987 [27]) as descripted by Aage and Lazarov [34]
- MPIIO is an output class to dump arbitrary field data into a single binary file
- The PETSc library contains the lower-level building blocks of the framework.

While the Linear Elasticity class in the framework is primarily designed for compliance minimization problems and thus requires modifications for this research, it highlights the importance of using an efficient linear system solver. Such a solver should prioritize not only simplicity but also aim to reduce the overall wall clock time of computations [35]. The preferred choice are iterative solvers because they require less communications to arrive at the solution since they mainly perform operations like vector addition and scalar multiplication, (matrix-vector multiplications break down in those) which rely on local data [34].

3 Methodology

This section offers a comprehensive and detailed view of the 3D Fabrication Sequence Optimization with Layer Thickness Control. First, an overview of all steps is provided, which can be followed with the help of Figure 12. Then, each step is described in detail in subsequent subsections. Finally, the complete formulation is assembled and presented. Following the pioneering work on Fabrication Sequence Optimization by Wang et al. ([13][1][15]), the goal of this optimization is to determine a manufacturable fabrication sequence to produce a given part through multi-axis Additive Manufacturing, while minimizing process-induced deformations. With optimized fabrication sequence it is meant individuating the optimal shape and sequence of layers, while excluding considerations about tool-path planning. Particular attention is given to the manufacturability of the layers; therefore, a modification of Lansu's [14] *Layer Thickness Control* formulation has been investigated.

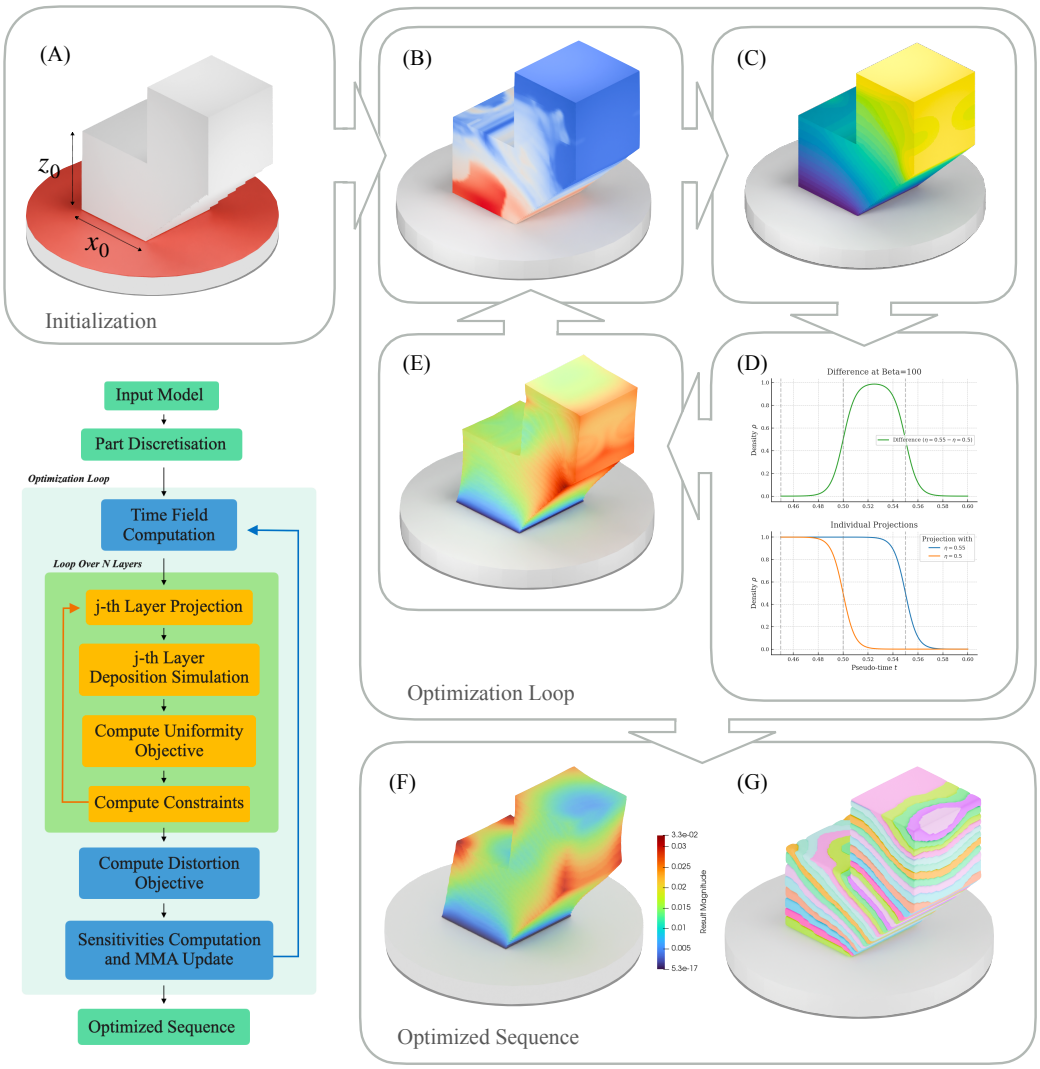


Fig. 12: Schematic representation of the Fabrication Sequence Optimization Method. A) The part geometry is defined, discretized, and boundary conditions for the process simulation are applied. B), C), D), and E) illustrate the steps within the optimization loop. B) The diffusivity field $\boldsymbol{\kappa}$, defined on the structural layout, is used to compute the pseudo-time field $\boldsymbol{t}$ (C) by solving the heat equation 15. D) Each layer is extracted from the pseudo-time field using a smooth Heaviside projection and utilized to compute the stiffness matrices $\boldsymbol{K}$ and thermal load vectors $\boldsymbol{f}$ for process simulation. E) The total displacement field is calculated and used to compute the desired distortion measures. The loop continues until convergence or the maximum iteration limit is reached. F) and G) depict the optimized displacement field and fabrication sequence, respectively. Steps included in the flowchart but not depicted in the visualization are the computations of objectives and constraints, along with their sensitivities, and the subsequent update step using the Method of Moving Asymptotes.

3-A Method Overview

The optimization process begins with the *Part Discretization* phase (Section 3-B), where the part design is encoded in a structured grid of hexaedral elements. Each grid element is designated as either filled or empty, representing the material distribution. This results in a vector, $\boldsymbol{\rho}$, where each element has a value of 1 (filled) or 0 (empty), corresponding to the material state. This operation is commonly referred to as *Voxelization*. In this phase the fixed base-plate boundary conditions is set as well. (Figure 12 A)

Next, the *Fabrication Sequence* is defined based on the discretized structural layout (Section 3-C). The sequence is represented as a scalar field (pseudo-time field $\boldsymbol{t}$) encoding the deposition order of the elements, where each element is assigned a normalized time value, $t_e \in [0, 1]$, with 0 marking the start and 1 the end of the process. To ensure a realistic additive manufacturing sequence, the pseudo-time field must exhibit critical properties such as the absence of local minima. The *Thermal Regularization* method by Wang et al. [15] is employed to achieve these requirements. This method, instead of defining directly the t_e values on the grid, computes the time field $\boldsymbol{t}$ by solving a heat equation with spatially varying thermal diffusivity $\boldsymbol{\kappa}$. Local maxima are permissible in the field but only if they occur on the boundaries of the part geometry (Figure 12 B and C).

Unlike more common slicing methods ([26], [6]), in this method the number of manufacturing layers, N, is predetermined. Layers are extracted from the pseudo-time field using a Smooth Heaviside projection (Section 3-D), enabling the field to be segmented in a differentiable manner. The threshold values of the projection functions are evenly spaced between 0 and 1, defined as $T_j = \frac{j}{N}$ for $j = 0, 1, \ldots, N$. (Figure 12 D)

During manufacturing, each element's addition causes structural deformation. However in this research, elements are considered *lumped* together in layers, and each layer's deposition is simulated using the inherent strain method (Section 3-E, Figure 12 E). The total process displacement field vector, obtained by summing each layer's contribution $\boldsymbol{U} = \sum_j \Delta \boldsymbol{U}_j$ is then used to compute the objective function (Section 3-F). The objective function can represent different distortion measures, such as minimizing displacement at specific points or ensuring surface flatness.

Given the technological limitations of additive manufacturing equipment in industry, it is crucial to control the shape of the layers to ensure they align with the equipment's capabilities. This research addresses this aspect by extending previous formulations from the literature. Specifically, a second objective is introduced to minimize the non-uniformity of thickness within each layer (Section 3-G), along with a constraint to directly control the average layer thickness value (Section 3-H).

3-B Part Discretization and Density Vector Definition

This optimization method shares similarities with traditional density-based topology optimization, where the design domain is discretized into a structured grid of trilinear hexahedral elements. However, in this approach, the part geometry is predefined rather than evolving during optimization. For simple part's geometries, like the ones used in this research's numerical examples, the geometry can be directly defined in the structured grid through a mathematical function. An example is shown in Figure 13.

The part is thus encoded in a binary density vector $\boldsymbol{\rho}$ associated with the structured grid, in which values of 1 indicate material presence and values of 0 represent void. The boundary conditions relative to the manufacturing process simulation are also defined at this stage, with the build plate acting as a fixed constraint on the corresponding nodes of the discretized part.

3-C Fabrication Sequence Representation

The pseudo-time field is defined on the discretized part geometry and encodes the material deposition sequence during the manufacturing process. Each element is assigned a continuous, normalized time value $t_e \in [0, 1]$, where 0 represents the start and 1 the completion of manufacturing. This method of representing the fabrication sequence has been previously utilized in the literature, notably by Wang et al. [1], [13] and Fang et al. [6]. The complete sequence of deposition is thus encoded in the pseudo-time field vector $\boldsymbol{t}$ associated to the structured grid.

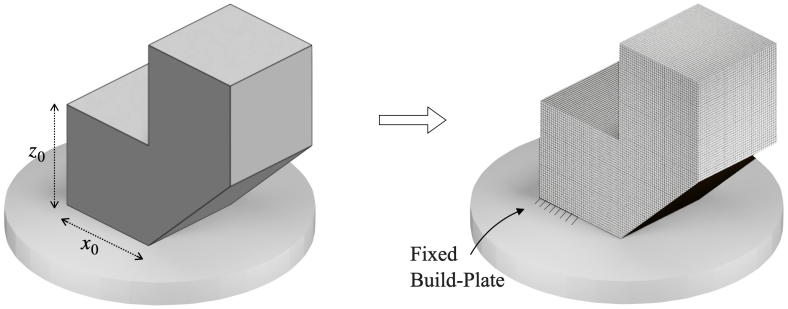


Fig. 13: Visualization of the part discretization in the structured grid of hexaedral elements. In this example the density function $\rho(x, y, z)$ is defined as follows: it takes the value 1.0 if $z < z_0$ or $x > x_0$, and $z > (x - z_0 - \epsilon)$. Otherwise, $\rho(x, y, z)$ is equal to 0.0. The fixed build plate conditions applies to all the nodes on the bottom face of the geometry.

In realistic multi-axis additive manufacturing, to avoid the use of support structures, new material can only be deposited on either the build plate or on already deposited material. This imposes key mathematical constraints on the time field, specifically the *absence of local minima*. Local minima would imply material being deposited in mid-air. Instead, local maxima in the field are acceptable, but only if lying on the boundary of the part geometry, since otherwise they would represent inaccessible enclosed voids.

To ensure the pseudo-time field $\boldsymbol{t}$ meets these requirements, it is not directly taken as a design variable like in Wang et al. [1], but is instead derived from the solution of a heat equation. This solution consists in a temperature field $\boldsymbol{T}$ defined on the nodes of the grid, which is then used to compute the elemental pseudo-time values. It is important to distinguish this temperature field (which for clarity will be called *virtual*) from the thermal field arising from the actual metal additive manufacturing process. This regularization technique was developed in Wang et al. [15] , and inspired by research in computer graphics where it is used as a computationally efficient way of approximating a geodesic distance field. The considered heat equation in differential form is:

$$\nabla \cdot (\kappa \nabla \tau) - \alpha_T \tau = 0, \tag{15}$$

where τ here is the *virtual temperature* field, $\kappa(x, y, z)$ represent a spatially varying *thermal diffusivity*, modeling how quickly heat diffuses through the material, while the term $\alpha_T \tau$ is a drain term, which models loss of heat proportionally to the virtual temperature.

The inclusion of the drain term $\alpha_T \tau$ is advantageous as it eliminates the need for inhomogeneous Dirichlet boundary conditions. This avoids having to predefine the position of the last deposited material in the manufacturing process, as is done, for instance, in the *Isothermal Surface Curved Layer Slicing* method by Shan et al. [26]. The heat equation is subject to the following boundary conditions: On the boundary adjacent to the build plate, Γ_0, the temperature is held constant at $\tau_{\Gamma_0} = 1$. All other boundary segments are thermally insulated, meaning $(\nabla \tau \cdot \mathbf{n})_{\Gamma \setminus \Gamma_0} = 0$, where $\nabla \tau$ is the spatial gradient of the temperature field and $\boldsymbol{n}$ denotes the normal vector to the boundary.

The diffusivity field $\kappa(x, y, z)$ varies spatially and in the discretization it is represented by individual scalar values assigned to each element within the domain. However, only the diffusivity values $\boldsymbol{\kappa}$ of elements that fall within the structural layout defined by $\boldsymbol{\rho}$ are designated as design variables. Elements outside this layout, where $\rho_e = 0$, are assigned a diffusivity of $\kappa_e = 1e - 3$, mainly restricting heat conduction to the structural layout, thus approximating the thermal insulation boundary condition explained above.

Equation 15 is discretized using the Finite Element Method resulting in the linear form,

$$\boldsymbol{K}_T \boldsymbol{T} = \boldsymbol{b}, \tag{16}$$

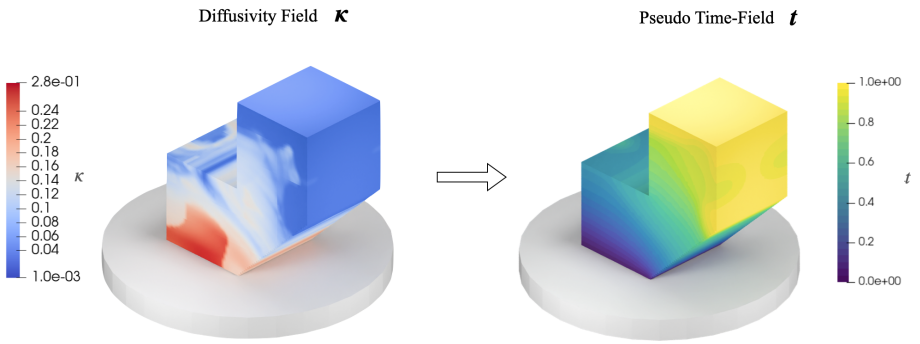


Fig. 14: Visualization of the diffusivity field $\kappa(x, y, z)$ (left) and the pseudo-time field $\boldsymbol{t}$ (right), both defined over the structural layout. The pseudo-time field is derived from the diffusivity field by solving Equation 15. Both fields are shown at an arbitrary iteration of the optimization loop.

where $\boldsymbol{T}$ is the nodal virtual temperature vector. $\boldsymbol{b}$ is the thermal load vector encoding the boundary conditions. $\boldsymbol{K}_T$ is the global system matrix.

$$\boldsymbol{K}_{\mathrm{T}} = \sum_{\{e\}} \boldsymbol{K}_{\mathrm{T}}^{e} + \sum_{\{e\}} \boldsymbol{K}_{\mathrm{C}}^{e}, \tag{17}$$

$\boldsymbol{K}_T$ is assembled from $\boldsymbol{K}_{\mathrm{T}}^{e}$ which correspond to the thermal diffusivity, and $\boldsymbol{K}_{\mathrm{C}}^{e}$ which correspond to the drain term as shown in Equation 17. These two elemental matrices are derived from the shape functions N_i and N_j by using:

$$\boldsymbol{K}_T^e = \int_{\Omega_e} \kappa_e \left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y} + \frac{\partial N_i}{\partial z}\frac{\partial N_j}{\partial z} \right) d\Omega, \tag{18}$$

$$\boldsymbol{K}_C^e = \int_{\Omega_e} \alpha_T N_i N_j d\Omega. \tag{19}$$

The solution of this linear system of equations (Equation 16) gives the nodal virtual temperature distribution $\boldsymbol{T}$, from which the pseudo-time field $\boldsymbol{t}$ can be computed as $\boldsymbol{t} = \mathbf{1} - \boldsymbol{\tau} = \mathbf{1} - \boldsymbol{G}\boldsymbol{T}$, where $\boldsymbol{G}$ is a sparse transformation matrix used to convert nodal values to elemental values. For all the numerical experiments in this research, all the elements of the design variable vector $\boldsymbol{\kappa}$ are initialized to $\kappa_e = 0.1$. The drain term α_T instead, is derived from,

$$\alpha_T = \frac{\beta}{{l_c}^2}, \tag{20}$$

where β is a non-dimensional parameter, and ${l_c}^2$ is the square of a characteristic length of the design domain. From previous research [15], β in the range $\beta \in [0.1, 1]$, have been found to lead to better convergence in the optimization.

To obtain a smoother virtual temperature field $\boldsymbol{T}$ and, consequently, a smoother time field $\boldsymbol{t}$, an averaging step can be introduced instead of directly using the heat diffusivities to solve the heat equation. This approach smooths the heat diffusivity by averaging its values over elements within a specified radial distance. For each element, the averaged diffusivity is computed based on its neighbors within the defined radius. This is formalized as:

$$\boldsymbol{\kappa}_{\text{avg}} = \boldsymbol{H}\boldsymbol{\kappa}, \tag{21}$$

where $\boldsymbol{H}$ is a square matrix constructed to apply the averaging operation. Each entry in $\boldsymbol{H}$ is assembled to represent the influence of neighboring elements within the specified radial distance on the diffusivity of a given element.

To summarize, as can be observed in the complete optimization method overview Figure 12, at each iteration of the optimization the updated design variables $\boldsymbol{\kappa}$ are used to solve the Heat Equation 15, and obtain the new Fabrication Sequence $\boldsymbol{t}$, which is then used, in the next steps to simulate the fabrication process and evaluate the layer thickness constraints.

3-D Layer Extraction via Heaviside Projection

In this research, the focus is on optimizing the manufacturing process at the part scale. Thus, it was chosen to treat groups of elements as lumped layers rather than individually, thereby simulating the collective impact of adding these elements, at each stage of the manufacturing, on the overall objective.

N time intervals are defined as $[T_{j-1}, T_j]$ for $j = 1, \ldots, N$, where N is the predetermined total number of layers. An element is assigned to a particular layer if its pseudo-time t_e value falls within the corresponding time interval of that layer. However, to formulate this segmentation operation in a manner suitable for gradient-based optimization methods, a differentiable Heaviside projection displayed in Equation 22 was employed.

$$\bar{t}_e^{\{j\}} = 1 - \frac{\tanh(\beta_t T_j) + \tanh(\beta_t (t_e - T_j))}{\tanh(\beta_t T_j) + \tanh(\beta_t (1 - T_j))} \tag{22}$$

The bar above $t_e \rightarrow \bar{t}_e$ indicates the value of the pseudo-time variable after the projection operation. In Equation 22, two important quantities are present: β_t, which controls the sharpness of the projection, and T_j, which controls the threshold value of the projection. The projection and the effect of its paramters can be visualized in Figure 15

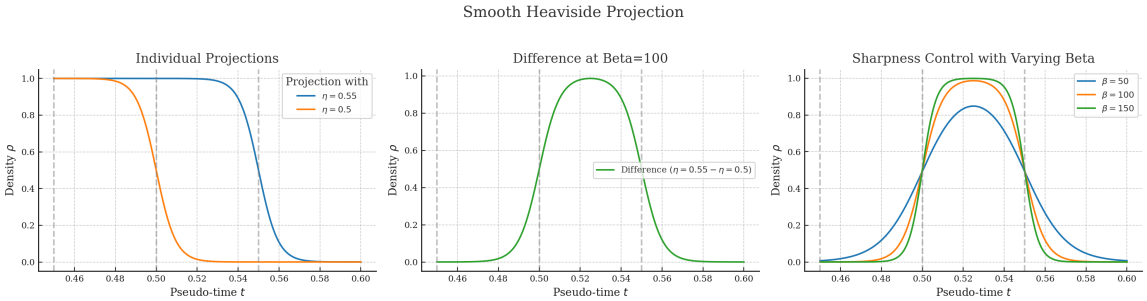


Fig. 15: Visualization of the Smooth Heaviside Projection function from Equation 22. The left image shows two projections with different threshold values, while the middle image demonstrates how these projections are combined to isolate a single layer. The right image illustrates the effect of varying sharpness values on the projection.

From the segemntation operation two density vectors are computed $\boldsymbol{\rho}^{\{j\}}$ and $\Delta\boldsymbol{\rho}^{\{j\}}$ where,

$$\begin{aligned} \boldsymbol{\rho}^{\{j\}} &= \boldsymbol{\rho} \odot \bar{\boldsymbol{t}}^{\{j\}}, \\ \Delta\boldsymbol{\rho}^{\{j\}} &= \boldsymbol{\rho} \odot (\bar{\boldsymbol{t}}^{\{j\}} - \bar{\boldsymbol{t}}^{\{j-1\}}), \end{aligned} \tag{23}$$

$\boldsymbol{\rho}^{\{j\}}$ is the density vector associated to the portion of the structure manufactured up until the j-th stage, while $\Delta\boldsymbol{\rho}^{\{j\}}$ is the density vector associated with a single layer. The symbol $\odot$ represent the point wise multiplication operation. In the layer extraction process there are two exceptions for the first and the last layer. For the first layer $\Delta\boldsymbol{\rho}^{\{1\}}$ is computed as $\Delta\boldsymbol{\rho}^{\{1\}} = \boldsymbol{\rho} \odot \bar{\boldsymbol{t}}^{\{j\}}$. For the last layer, $\Delta\boldsymbol{\rho}^{\{N\}}$ is computed as $\Delta\boldsymbol{\rho}^{\{N\}} = \boldsymbol{\rho} \odot (\mathbf{1} - \bar{\boldsymbol{t}}^{\{j-1\}})$, while $\boldsymbol{\rho}^{\{N\}} = \mathbf{1}$.

3-E Process Simulation using the Inherent Strain Method

Recalling that the primary objective in this optimization, is to minimize the thermally induced deformation during the additive manufacturing process, the entire process must be simulated at each optimization iteration, and the total distortion of the part must be computed. Although the method formulation can work with any of the process simulation approaches introduced in the background knowledge Section 2-A, the Inherent Strain

Method was chosen in this research due to its computational efficiency and its widespread use in similar applications within the literature. However, this method involves significant approximations, making the overall optimization primarily qualitative at this stage.

Taking a single element as example, as it gets deposited during the Additive Manufacturing process it transforms from a liquid to a solid state, and undergoes shrinkage. The *Inherent Strain Method* models this effect purely mechanically by assigning a thermal strain tensor in 3D, denoted as $\underline{\boldsymbol{\varepsilon}}^* = \{\varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{zx}\}^\mathsf{T}$, to each newly deposited element. These strain values are derived either empirically or through detailed process simulations. For the numerical test in this study the inherent strain are arbitrarily set to $\underline{\boldsymbol{\varepsilon}}^* = \{-1e^{-2}, -1e^{-2}, -1e^{-2}, 0, 0, 0\}^\mathsf{T}$. From these strains, equivalent nodal forces acting upon the nodes of the elements can be computed.

$$\boldsymbol{f}_e(\underline{\boldsymbol{\varepsilon}}^*) = [\boldsymbol{D}]^\mathsf{T}[\boldsymbol{C}]\underline{\boldsymbol{\varepsilon}}^*, \tag{24}$$

where $\boldsymbol{D}$ is the strain–displacement relation matrix, and $\boldsymbol{C}$ denotes the elasticity matrix.

The quest of computing the contribution of each layer addition to the total deformation during the process translates to solving a set of N linear systems formulated as:

$$\boldsymbol{K}^{\{j\}}\boldsymbol{\Delta U}^{\{j\}} = \boldsymbol{f}^{\{j\}}, \quad j = 1, \ldots, N \tag{25}$$

where $\boldsymbol{\Delta U}^j$ is the displacement vector associated with the addition of the j-th layer, $\boldsymbol{K}^j$ is the stiffness matrix, and $\boldsymbol{f}^j$ is the forcing vector assembled from the equivalent nodal forces of the strains (Equation 24), both relative to the j-th layer. The total deformation $\boldsymbol{U}$ can be thus computed by summing the contributions from each layer $\Delta\boldsymbol{U}^{\{j\}}$,

$$\boldsymbol{U} = \sum_{j=1}^{N} \Delta\boldsymbol{U}^{\{j\}}. \tag{26}$$

To assemble the stiffness matrix $\boldsymbol{K}^{\{j\}}$ associated with the addition of the j-th layer, the Solid Isotropic Material with Penalization (SIMP) method is used. In this approach, every element has the same base stiffness matrix $\boldsymbol{K}_0$, which is scaled according to its projected density value through:

$$\boldsymbol{K}_e^{\{j\}}(\rho_e^{\{j\}}) = \left(E_{\text{min}} + (1 - E_{\text{min}})(\rho_e^{\{j\}})^p\right)\boldsymbol{K}_0, \tag{27}$$

where $\rho_e^{\{j\}}$ belongs to $\boldsymbol{\rho}^{\{j\}}$ previously calculated through Equation 23. Here, p is a penalization factor, typically set to a value between 3 and 4, which increases the stiffness contrast between solid and void elements.

The forcing vector $\boldsymbol{f}^{\{j\}}$ associated with the j-th layer addition, is similarly assembled from each element's equivalent nodal forces vector through a penalization operation dependent on its projected density value:

$$\Delta\underline{\boldsymbol{\varepsilon}}^{*\{j\}}(t)\big|_e = \left(\Delta\rho^{\{j\}}(t)\big|_e\right)^q \underline{\boldsymbol{\varepsilon}}^*, \tag{28}$$

$$\boldsymbol{f}_e = [\boldsymbol{D}]^\mathsf{T}[\boldsymbol{C}]\Delta\underline{\boldsymbol{\varepsilon}}^{*\{j\}}\big|_e. \tag{29}$$

One important feature of this process simulation method is that, Equation 25 can be solved in parallel for the addition of each layer as demonstrated from Munro et al. [22] in the analysis of a linearity assumption, initially proposed for simplifying the simulation of powder-bed processes with only planar layers.

3-F Distortion Objective Calculation

Once the total *Displacement Vector* $\boldsymbol{U}$ is obtained from the process simulation, it can be used to compute the *Distortion Objective* J through:

$$J = \boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{Q}\boldsymbol{U} \tag{30}$$

where $\boldsymbol{Q}$ is a matrix that combines the displacement values of specific nodes based on the type of objective being considered. Different distortion objectives are possible, two examples inspired from previous research in 2-Dimensions by Wang et al. [1] are demonstrated below.

Obj-1. Minimizing the displacement of a single or group of nodes. The square of the displacement magnitude of a set of nodes is computed by

$$J_1 = \sum_{i}^{m}[(u_i^x)^2 + (u_i^y)^2 + (u_i^z)^2], \tag{31}$$

where subscript i refers to the index of the m nodes of interest, and the superscripts x, y, and z indicate the x, y, and z components of the displacement vector, respectively.

Obj-2. The flatness of a horizontal edge is computed by

$$J_2 = \frac{1}{n(\mathcal{I})}\sum_{i\in\mathcal{I}}\left(u_i^z - \frac{1}{n(\mathcal{I})}\sum_{i\in\mathcal{I}}u_i^z\right)^2, \tag{32}$$

where $\mathcal{I}$ denotes the set of nodes that are sampled from the face of interest. The number of nodes $n(\mathcal{I})$ is equal to or larger than 3. The flatness is tested with different numbers of sample nodes.

3-G Layer Uniformity Objective Calculation

Minimizing process-induced deformation, is the primary target of this optimization method. However, unlike polymer-based Fused Deposition Modeling (FDM) machines, achieving precise deposition control in WAAM is more challenging, and requires a continuous adjustments to process parameters. Therefore, maintaining uniformity in layer geometry is favorable to ensure that the optimized layers align with the equipment's manufacturing capabilities.

To address this, the formulation incorporates a secondary objective to promote layer uniformity by minimizing thickness disparities within each layer. Instead of directly calculating the exact layer thickness at every point, which is complex, this objective is achieved by minimizing the sum of squared deviations (Equation 33) of the pseudo-time field gradient magnitude from its mean. Reducing these deviations ensures a more consistent gradient, resulting in a more uniform layer geometry. The generalized form of the sum of squared deviations from the mean is:

$$L = \sum_{j=1}^{l}(x_j - \mu)^2, \tag{33}$$

where x_j represents the individual data points (e.g., gradient magnitudes), μ is their mean, and l is the total number of data points. This formulation minimizes the weighted sum of squared deviations from the mean, thereby reducing variation and promoting uniformity in the system.

The *Uniformity Objective* formulation is thus as follows,

$$L = \sum_{j=1}^{N} L_j \quad \text{with} \quad L_j = (\Delta\boldsymbol{\rho}^{\{j\}})^{\top}(\boldsymbol{M} - \boldsymbol{m}_j)^2 \tag{34}$$

Here, $\Delta\boldsymbol{\rho}^{\{j\}}$ represents the density distribution associated with the j-th layer and it is computed from Equation 23, $\boldsymbol{M}$ represents the vector of gradient magnitudes, and $\boldsymbol{m_j}$ is a vector of the same dimension as

$\boldsymbol{M}$, with each element equal to the average gradient magnitude m_j in the j-th layer. This formulation aims at minimizing the variance of the gradient magnitudes within each layer.

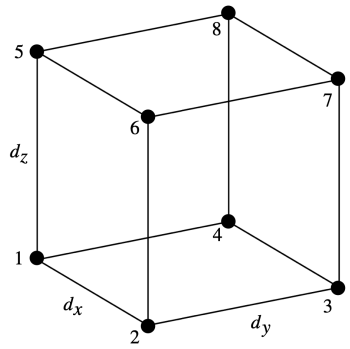


Fig. 16: Visualization of one hexaedral element and its node numbering

$$\begin{aligned} &\boldsymbol{w}_x = \frac{1}{8}[-1,1,1,-1,-1,1,1,-1] \\ &\frac{\partial T}{\partial x}\Big|_e = \boldsymbol{w}_x \cdot \boldsymbol{T}_e \\ &\boldsymbol{w}_x \rightarrow \text{assembly} \rightarrow \boldsymbol{G_x} \end{aligned} \tag{35}$$

Since the magnitude of the gradient, which is independent of sign, is of interest, the nodal virtual temperatures $\boldsymbol{T}$ can be conveniently used to compute it, instead of the pseudo-time field $\boldsymbol{t}$. For each element, the gradient magnitude is determined using its nodal virtual temperature values, as illustrated in Figure 16 and its associated Equation 35. This computation can be expressed in vector form as:

$$\boldsymbol{M} = \sqrt{\boldsymbol{V}_x \odot \boldsymbol{V}_x + \boldsymbol{V}_y \odot \boldsymbol{V}_y + \boldsymbol{V}_z \odot \boldsymbol{V}_z} \tag{36}$$

$$\boldsymbol{V}_x = \boldsymbol{G}_x\boldsymbol{T} \quad \boldsymbol{V}_y = \boldsymbol{G}_y\boldsymbol{T} \quad \boldsymbol{V}_z = \boldsymbol{G}_z\boldsymbol{T} \tag{37}$$

where $\boldsymbol{G}_x$, $\boldsymbol{G}_y$, and $\boldsymbol{G}_z$ are matrices used to combine the nodal virtual temperature value to compute respectively the x, y, and z components of the gradient. The symbol $\odot$ represent the point wise multiplication operation. As example, the gradient computation for the x component is displayed in Equation 35.

To compute instead the average gradient's magnitude over a layer m_j, the following expression is used:

$$m_j = \frac{\boldsymbol{M}^\top \Delta\boldsymbol{\rho}^{\{j\}}}{\mathbf{1}^\top \Delta\boldsymbol{\rho}^{\{j\}}} \tag{38}$$

where $\Delta\boldsymbol{\rho}^{\{j\}}$ is computed through Equation 23.

3-H Optimization Constraints

Together with the uniformity objective, to improve the manufacturability of the resulting optimized Fabrication Sequences, a *Layer Thickness Constraint* is introduced. This constraint aims at controlling the average thickness value of each layer between a lower and upper boundary d_{min} and d_{max}. These two values will be dependent on the Additive Manufacturing equipment used. The constraint is formulated as follows:

$$d_{min} \leq d_j \leq d_{max} \quad \text{with} \quad j = 1, \ldots, N \tag{39}$$

$$d_j = (T_j - T_{j-1})\frac{1}{m_j} \quad \text{with} \quad j = 1, \ldots, N \tag{40}$$

where m_j is the average gradient magnitude value for the j-th layer, while T_j and T_{j-1} define the j-th time interval. Thus, d_j effectively represents an approximation of the layer thickness. In the numerical experiments conducted in this research, the minimum and maximum bounds for the *Layer Thickness Constraint*, d_{min} and d_{max}, were derived from a target layer thickness value, $d_{\text{target}} \pm 10\%$. While the number of layers, N, is computed by dividing a reference domain length by d_{target}.

3-I Complete Formulation

To summarize, the Fabrication Sequence Optimization, for distortion minimization, with layer thickness control, and using the heat equation for regularization is formulated as follows:

Objective Function:

$$\min_{\boldsymbol{\kappa}} \quad J(\boldsymbol{\kappa}) = \mathbf{U}^\top \mathbf{Q} \mathbf{U} + \alpha \sum_{j=1}^{N} (\Delta \boldsymbol{\rho}^{\{j\}})^\top (\boldsymbol{M} - \boldsymbol{m}_j)^2 \tag{41}$$

subject to: (42)

State Equations:

$$\mathbf{K}^{\{j\}}(\boldsymbol{\kappa}) \Delta \mathbf{U}^{\{j\}} = \mathbf{f}^{\{j\}}(\boldsymbol{\kappa}), \quad \forall j \in \{1, \ldots, N\}, \tag{43}$$

$$\mathbf{K}_{\mathrm{T}}(\boldsymbol{\kappa}) T = \mathbf{b}, \tag{44}$$

Constraints:

$$g_{j1}(\boldsymbol{\kappa}): \quad -\frac{(T_j - T_{j-1})}{m_j} + d_{min} \leq 0, \tag{45}$$

$$g_{j2}(\boldsymbol{\kappa}): \quad \frac{(T_j - T_{j-1})}{m_j} - d_{max} \leq 0, \quad \forall j \in \{1, \ldots, N\}, \tag{46}$$

Design Variables Range:

$$1e^{-3} \leq \kappa_e \leq 1, \quad \forall e. \tag{47}$$

The objective function J consists of two components: the first addresses the *distortion objective*, as discussed in Section 3-F, and the second focuses on the *layer uniformity objective*, introduced in Section 3-G. A parameter α is included to control the relative importance of the uniformity objective compared to the distortion objective.

There are two types of state equations. The first type pertains to the computation of process-induced deformations. For each of the N manufacturing stages, a linear system must be solved. From the solutions to these N systems, the displacement contributions from each layer, $\Delta \boldsymbol{U}^{\{j\}}$, are obtained and used to compute the total deformation $\boldsymbol{U}$, as explained in 3-E. The second state equation is the Heat Equation 15, solved at each optimization iteration, and used to obtain a monotonic pseudo-time field $\boldsymbol{t}$ from the diffusivity design variables $\boldsymbol{\kappa}$ through the thermal regularization method, as detailed in 3-C.

There is one type of constraint, which is the layer thickness control constraint. Its purpose is to ensure that the approximated thickness for each layer remains within the specified range, $d_{min} \leq d_j \leq d_{max}$. This results in two inequalities per layer, leading to a total of $2N$ constraint functions.

The optimization problem is solved using a gradient-based numerical method, specifically the Method of Moving Asymptotes (MMA) [27]. As a gradient-based approach, this requires the calculation of sensitivities for both the objective function and the constraints. Since there are two FEM solves involved between the design variables and the objective function, two adjoint steps need to be performed. The complete sensitivity analysis is detailed in Appendix A.

4 Implementation and Validation

In this section, the implementation of the method is described, focusing on the rationale for adopting parallelization, assessing parallel performance, and presenting part of the code validation process.

While previous research on Fabrication Sequence Optimization [1], [15], [14] focused primarily on 2D examples, this study aims to extend and evaluate the optimization method within 3D applications. Transitioning from 2D to 3D significantly increases the problem's scale, resulting in a large-scale computational challenge [34], thus involving hundreds of thousands of elements or more. This scale intensifies both computation time and memory requirements, quickly exceeding the capacity of standard workstations. To address these demands, a parallelized implementation suitable for high-performance computing infrastructures was developed [31].

As a foundation for developing the fully parallel implementation of Fabrication Sequence Optimization with Layer Thickness Control, existing solutions from the literature were leveraged. Specifically, Aage et al.'s [28] well-known Density-based Topology Optimization framework in PETSc was used, as it offers a relatively easy-to-use and portable code, built on the PETSc library for high-performance scientific computing. Although the optimization problem in this research is not Density-based Topology Optimization, its formulation shares similarities, making this framework a valuable starting point for the implementation.

Following the Topology Optimization in PETSc framework as example, the adopted parallelization strategy is domain decomposition with Message Passage Interface (MPI). This strategy involves dividing the entire discretized domain into smaller segments, with each processor or computing unit dedicated to handling computations for a specific segment. The more each computation remain independent from the others, the less the necessity for inter-processor communication will be, limiting the communication overhead that can hinder overall performance, as introduced in Section 2-D1. PETSc is mainly based on distributed memory systems and uses Message Passing Interface to manage communications between the nodes [33].

4-A Overview of Key Components and Their Interactions

This section provides a high-level overview of the main components in the optimization framework, focusing on each class's role and its interactions within the overall system. To streamline the implementation, several components from the original framework were either fully or partially reused. For instance, the *MMA* class, which implements a parallelized version of the Method of Moving Asymptotes [27], and the *MPIIO* class, which manages binary output files, were directly integrated. Other elements, such as parts of the *Initialization* class and sections of the *LinearElasticity* class for Finite Element Analysis, were adapted to meet specific requirements.

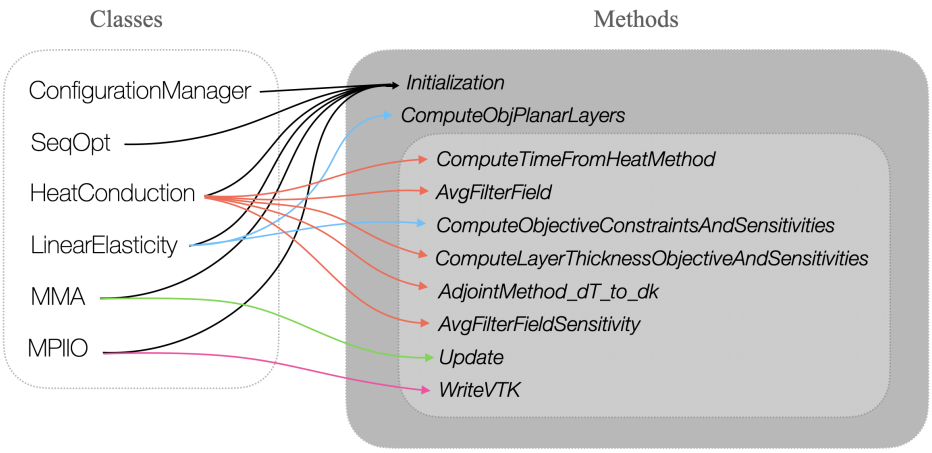


Fig. 17: Overview of Classes and Key Methods in the Optimization Framework

In the `main` function, essential objects for the optimization process are instantiated, and the optimization loop is orchestrated. Within this loop, various methods from different classes are invoked to handle each substep of

the optimization. The `main` function also logs iteration details, manages the continuation of the parameter β, and outputs intermediate results to a binary file every 15 iterations.

The `ConfigurationManager` class defines the primary optimization parameters and updates them through runtime options if needed. It also sets up the results directory, generating a unique name based on the chosen parameters. Additionally, the class provides the `WriteIterationRecordFile` method, which saves quantitative data—such as objective values, constraint values, and execution timings—in a `.json` file.

The `SeqOpt` class maintains essential PETSc vector objects used in the optimization, including fields like `rho` ($\boldsymbol{\rho}$), `k` ($\boldsymbol{\kappa}$), and `t` ($\boldsymbol{t}$), as well as sensitivity vectors. This class is responsible for creating a structured grid and defining the structural layout, stored in the `rho` vector. Additionally, it initializes the pseudo-time field vector based on elements' Euclidean distances in the z-direction, which is crucial for simulating planar layers before entering the optimization loop. The class also provides methods for scaling objective and constraint values before passing them to the `MMA` optimizer.

The `HeatConduction` class is responsible for solving the heat equation to produce the virtual temperature field $\boldsymbol{T}$, defined on grid nodes via the `ComputeTimeFromHeatMethod`. This temperature field is used to directly calculate the layer uniformity objective and layer thickness constraints, and thus the class also manages these computations. Specifically, the `ComputeLayerThicknessObjectiveAndSensitivity` method evaluates both the layer uniformity objective and thickness constraints, along with their sensitivities, while the `AdjointMethod_dT_to_dk` method maps sensitivities from the temperature field $\boldsymbol{T}$ to the design variables $\boldsymbol{\kappa}$. This class also provides two methods, `AvgFilterField` and `AvgFilterFieldSensitivity`, used to average the design variables $\boldsymbol{\kappa}$ at the start of each optimization iteration.

The `LinearElasticity` class manages the objects and methods related to process simulation. Its main method, `ComputeObjectiveConstraintsAndSensitivities`, takes the pseudo-time field as input, executes the FEM routine to simulate the process, and computes the displacement field $\boldsymbol{U}$. This displacement field is then used to derive distortion measures, and the adjoint method is applied to obtain sensitivities of the objective with respect to the pseudo-time field variables, which will then be mapped back to the design variables $\boldsymbol{\kappa}$ through the same `AdjointMethod_dT_to_dk` cited above.

4-B Process Simulation Validation

To validate the correctness of the process simulation code, two tests based on relevant literature were performed. The first test involved running a process simulation and comparing the results to a reference implementation—the 2D MATLAB implementation used by Wang et al. [1] in the original Fabrication Sequence Optimization work. The second test, similar in approach, compared the simulation results to a 3D reference from the work of Munro et al. [22].

1) First Test

The first test was executed by simulating the additive manufacturing process of the V-shape from Wang et al. [1]. In MATLAB, the domain with dimensions $(x : 100\,\mathrm{mm}, y : 100\,\mathrm{mm})$ was discretized using a structured grid of 100×100 elements. In the 3D PETSc implementation, the 2D shape was extruded in the third dimension to $(x : 100\,\mathrm{mm}, y : 100\,\mathrm{mm}, z : 10\,\mathrm{mm})$ with a corresponding discretization of $100 \times 100 \times 10$ elements. The process was simulated with $N = 10$ planar lumped layers. The material properties were defined as Young's modulus $E = 110\,\mathrm{GPa}$, Poisson's ratio $\nu = 0.3$, and an inherent strain vector $\varepsilon = \{-1 \times 10^{-2}, -1 \times 10^{-2}, 0.0, 0.0, 0.0, 0.0\}$.

As observed in Figure 18, the displacement fields produced by the two implementations are very similar. Although primarily qualitative, this comparison validates the accuracy of the PETSc implementation. The slight discrepancies in displacement values may be caused by minor differences in the shape of the overhang region. The PETSc results is 3D, but only a 2D side-view is shown in the figure.

2) Second Test

In the second test the result taken as reference is from the work of Munro et al. [22]. In the reference study, the simulation focuses on an Additive Manufacturing (AM) process for creating a 100 mm³ cube, divided into 1000 elements arranged in a 10x10x10 grid. This manufacturing process is carried out in 10 planar layers,

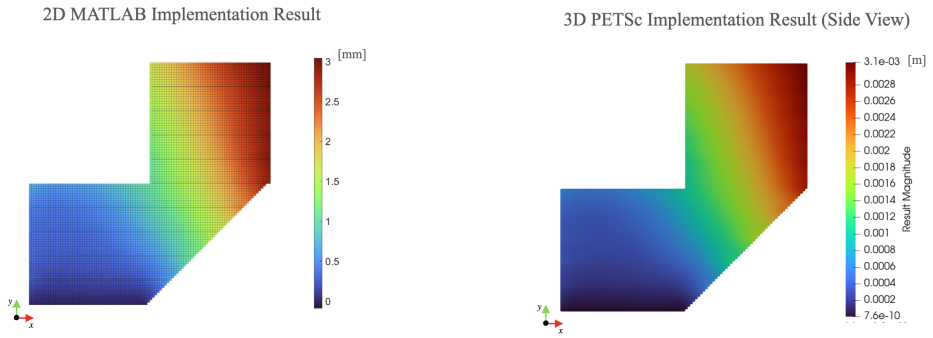


Fig. 18: Comparison of the displacement field obtained using the MATLAB implementation and the PETSc implementation. The MATLAB colorbar is in millimeters, while the PETSc results are displayed in meters.

with the base surface being fixed. The simulation employs the inherent strain method, assuming material properties of $E = 125$ GPa and $\nu = 0.333$. The inherent strain, denoted by ε, is isotropic, meaning each element shrinks uniformly across the part in every direction. The specific value of ε is defined in the code as $\{-5e^{-3}, -5e^{-3}, -5e^{-3}, 0.0, 0.0, 0.0\}$, indicating uniform shrinkage in the x, y, and z directions without any shear strain.

The reference study utilizes Abaqus (CalculiX) for the implementation, in particular employing the C3D8R element type, which consists of hexahedral elements with linear shape functions and a reduced integration scheme. Similar to the optimization method considered in this research, the layers in the referenced study are considered lumped, and thus the simulation is comprised of 10 steps corresponding to the addition of the 10 discrete planar layers.

The same numerical experiment was performed using the PETSc implementation developed our research, but with some differences in implementation. In this research's process simulation, the layers are not treated discretely; instead, they are extracted from the pseudo-time field through a smooth Heaviside projection. An additional difference is that a full integration scheme is used in this research implementation. For reference, the C3D8R elements used by Munro et al. utilize reduced integration and hourglass control [36], [37].

The results from this method's implementation are compared to the reference paper in Figure 19.

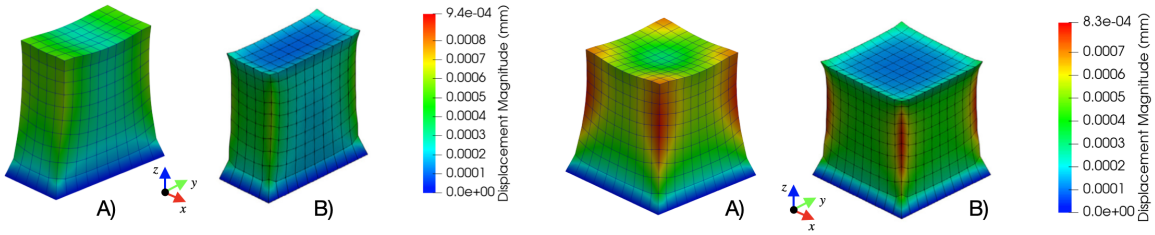


Fig. 19: Comparison between the results of Munro et al. [22] and those obtained through the PETSc implementation. The results designated by **B)** represent the reference results, while those designated by **A)** are produced using the implementation developed in this research.

The results shown in Figure 19 appear to match, confirming the validity of the PETSc implementation. However, slight differences in the deformation shape are visible, likely attributable to the use of reduced integration with hourglass control in Munro's work [22], instead of the full integration used in this research's implementation.

4-C Sensitivity Analysis Validation

In order to verify the correctness of the implemented analytical sensitivity calculation (detailed in Appendix A), a finite difference test has been carried out. The finite difference test compares, for different perturbation steps h, the percentage error between the analytically computed sensitivity and the one computed through

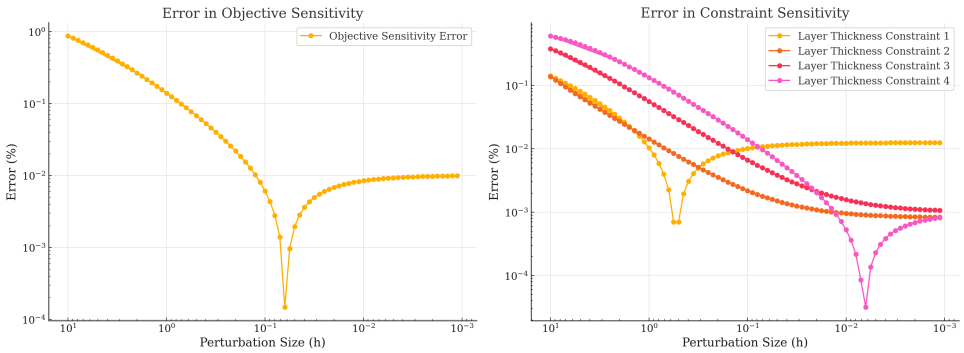


Fig. 20: Plots of the finite differences sensitivities check

finite difference. The perturbation is applied to the diffusivity value κ_e of a single element in the domain. The expression used to compute the error is provided in the following Equation 48.

$$\text{Error} = \frac{|S_{\text{analytical}} - S_{\text{finite difference}}|}{|S_{\text{analytical}}|} \times 100, \tag{48}$$

A reduced-dimensional problem has been used with a full density domain of size $2 \times 2 \times 2$ and a discretization of $10 \times 10 \times 10$ elements. The number of layers N is set to 4, with β_t fixed at 10 and $\alpha = 1$. The perturbation steps range between 10^{1} and $10^{-3.95}$, covering 4 decades with 20 logarithmically spaced substeps per decade. The plots in Figure 20 successfully validate the sensitivity analysis. It is important to note that, while there are 8 constraints in total, their sensitivities occur in identical pairs. As a result, only 4 unique constraints are displayed. The objective represent the combined distortion and uniformity objective.

5 Numerical Results

The method was implemented in C++ utilizing the PETSc library. This section presents a series of numerical experiments aimed at characterizing the method's behavior. Following the approach outlined in the *Part Discretization* section, trilinear hexahedral elements were used. The analysis will primarily focus on qualitative aspects, with material properties chosen for illustrative purposes. Specifically, all the part dimensions, and displacement magnitudes in the plots are defined in meters $[m]$, while the material is defined by a Young's modulus of $E = 1[\text{Pa}]$, a Poisson's ratio of 0.3, and isotropic inherent strains of $\underline{\varepsilon}^* = \{-1 \times 10^{-2}, -1 \times 10^{-2}, -1 \times 10^{-2}, 0.0, 0.0, 0.0\}$. The final subsection is dedicated to an analysis of parallel performance, emphasizing *scalability*.

5-A Characterizing Optimization Behavior Across Various Part Geometries

The first three numerical results, conducted with similar optimization settings, illustrate the method's behavior across different part geometries. The parts shown in Figure 21 — labeled from left to right as V-Shape, Bracket Shape, and Asymmetric Shape — were inspired by Wang et al. [1] to highlight varying characteristics of the method. In each case, a Dirichlet boundary condition is applied to the nodes on the base face of the part, fixing them to the build plate (shown in red in Figure 21). The distortion metric used across all examples follows the formulation in Equation 31, with the selected node group m visualized in aquamarine in Figure 21. Every experiment is run for a total of 400 iterations.

Additionally, a continuation scheme for the projection sharpness parameter β_t was employed across all examples, initialized at 10 and increased by 20 every 15 iterations. Both the *Layer Uniformity Objective* and *Distortion Objective* are normalized by dividing their values by their respective values from the first iteration and multiplying the result by 100, ensuring that both objectives start at a value of 100. Subsequently, the relative weight parameter α from Equation 41 is applied to the normalized *Layer Uniformity Objective* as shown in Equation 41. In this subsection's experiments α is always set to 1.

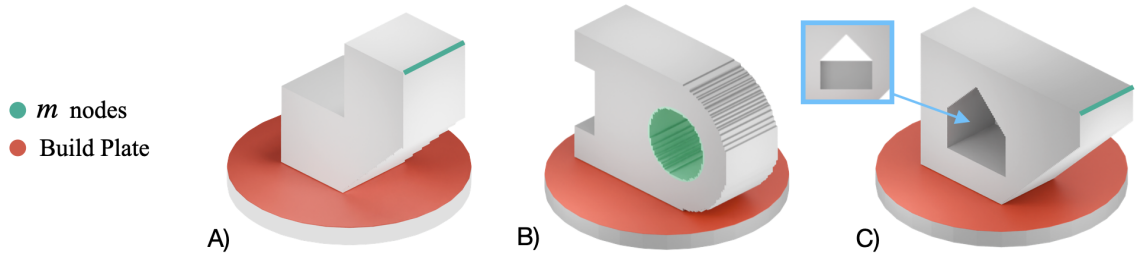


Fig. 21: Display of geometries used in the first three numerical experiments: A) *V-Shape*, B) *Bracket Shape*, and C) *Asymmetric Shape*. The image highlights the base-plate (in red), which imposes a fixed boundary condition on the nodes at the bottom face of each geometry. Nodes considered in the distortion measurements used as objectives are shown in aquamarine.

1) V-Shaped Component

The first test was conducted on a *V-Shape* model, inspired by one of the 2D geometries employed by Wang et al. [1], but extruded into the third dimension. The domain measures $2 \times 1 \times 2$, discretized into $80 \times 40 \times 80$ elements, with nodes on the build plate constrained as fixed. In this example, the distortion measure considers the displacement of the nodes belonging to the top-right edge, as illustrated in Figure 21A. The target layer thickness is set to $d_{\text{tar}} = 0.1$, making $N = 25$ a reasonable choice for the number of prescribed layers for the optimization. This estimate is based on the part's dimensions, following a similar approach to the formula introduced by Wang et al. [1], $N \approx \frac{l_c}{d_{tar}}$, in which l_c is the characteristic length of the structural layout. The relationship between d_{tar} and the upper and lower thickness bounds, d_{max} and d_{min}, is detailed in Section 3-H. To ensure a fair comparison between planar and optimized layer fabrication, only 20 layers are used in the planar case, as fewer layers with thickness $d_{\text{tar}} = 0.1$ fit along the z-axis.

As shown in Figure 22, the method successfully produces a fabrication sequence that minimizes the distortion objective while maintaining uniform layer thickness. Specifically the distortion objective is reduced from 3.9×10^{-1} in the planar fabrication case, to 7.9×10^{-4} in the optimized case. Additionally, the average

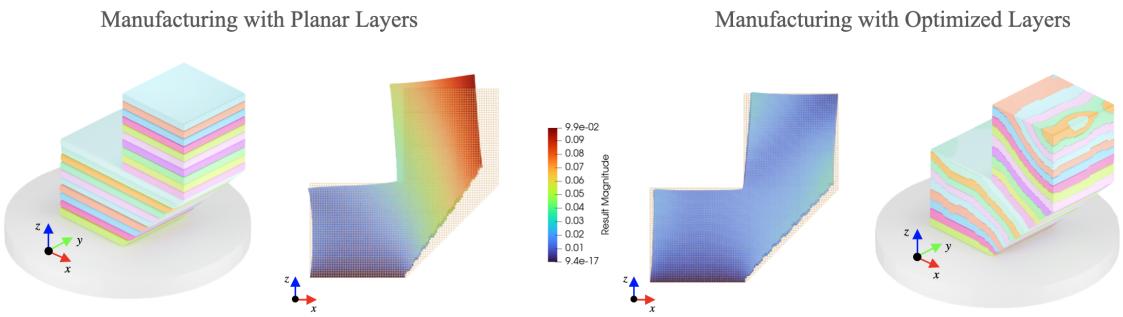


Fig. 22: Comparison between two fabrication sequences: planar layers on the left and optimized layers on the right. The displacement fields from the process simulation demonstrate the effectiveness of the optimized fabrication sequence in minimizing the displacement of the top edge nodes, compared to the planar fabrication case.

thickness of each layer is controlled within specified upper and lower bounds. Variations in thickness within each layer are minimized but permitted to achieve curved layer shapes. A close examination of the optimized sequence reveals a distinctive feature of this method. Unlike previous formulations, this approach allows layer volumes to diminish. Consequently, only 24 of the 25 prescribed layers are present in the optimized sequence. This reduction does not compromise the process simulation, as a small constant (1×10^{-9}) is added to the element density values ρ_e during layer extraction via Heaviside projection, to avoid numerical problems, while rendering disappearing layers practically non-influential on the displacement field.

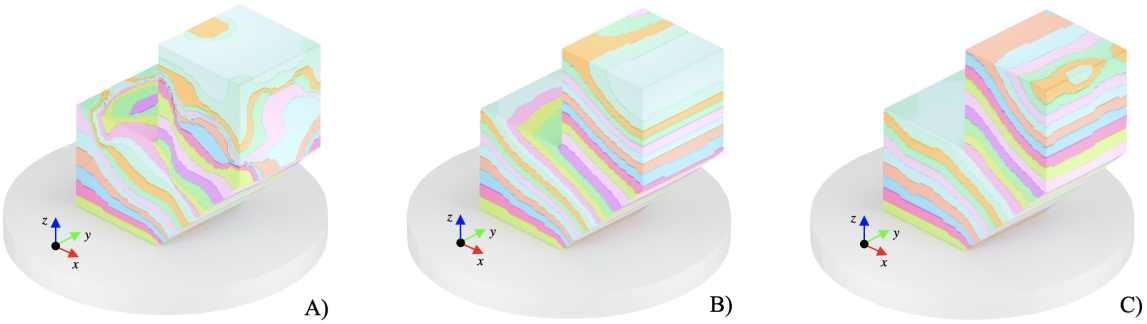


Fig. 23: Comparison of optimized fabrication sequences obtained using different method formulations. **A)** Sequence generated with the original formulation by Wang et al. [1], with added thermal regularization as per Wang et al. [15]. **B)** Sequence obtained using the A) formulation, though with the addition of the uniformity objective (Equation 34). **C)** Sequence produced using the complete formulation presented in this study.

To better understand the effect of each component in this optimization problem formulation (Section 3-I), two additional numerical experiments with different formulations were conducted on the same geometry:

The first employed formulation (Formulation 1), is the original approach by Wang et al. [1], with the addition of thermal regularization from Wang et al. [15]. In this version, the *Continuity Constraint* is removed, as the continuity of the scalar field is inherently maintained by the thermal regularization. This formulation does not employ *layer thickness constraints*; instead, *Speed Constraints* are used. These constraints ensure that the same amount of material is deposited within equal time intervals, as represented by Equation 49, where V^* denote the total volume of the component. Given that the number of layers is prescribed as N, the cumulative volume up to the j-th layer is $\frac{j}{N}V^*$. This constraint is also employed to avoid the shrinkage o layers.

$$V^j(t) = \sum_e \rho_e^j(t_e) \cdot v_e = \frac{j}{N}V^*, \quad j = 1, \ldots, N, \tag{49}$$

As observed in Figure 23A, while this formulation successfully minimizes the objective to 8.3×10^{-5}, it produces a sequence that is challenging to manufacture. The layers exhibit highly irregular geometry with

substantial thickness variations, which, additionally, complicates post-processing for visualization purposes.

The second formulation (Formulation 2) closely resembles the one used for the first example shown in Figure 23A, but with the addition of the *Uniformity Objective* introduced in Equation 34. The resulting sequence, displayed in Figure 23B, shows significantly more regular layers compared to the first case. While the thickness within each layer is now uniform, there is a high variability in the average thickness between layers across the entire sequence, allowing different layers to have different average thickness values. The third formulation (Formulation 3), instead, is the complete one, thus employing the *Uniformity Objective* and the *Layer Thickness Constraint*, but not the *Speed Constraints*. The results, are shown both in Figure 22 and Figure 23.

To quantitatively assess the impact of layer thickness constraints in the complete formulation (Formulation 3), a comparative analysis was performed against the formulation that includes only the uniformity objective (Formulation 2). In this analysis, the average thickness value and the thickness variance for each layer were geometrically computed in a post-processing step and compared between the two formulations. Specifically, layer boundaries were extracted as isosurfaces from the two optimized fabrication sequences. From each isosurface, a group of 1,000 points was sampled. For every sampled point, the distance to the closest point on the subsequent isosurface in the sequence was calculated. These distances were then used to compute the average thickness and thickness variance, providing a quantitative measure of layer uniformity.

This process is schematically illustrated in Figure 24, where two consecutive isosurfaces are denoted as Ω_j and Ω_i. To prevent erroneous thickness calculations in cases where the two surfaces have significantly different areas, an approximation was applied: if the distance between two points exceeds 30% of the target thickness d_{tar}, the point is discarded, and a new point is sampled to maintain accuracy. From the plot in Figure 24, it can be observed that, in the case without thickness constraints, the variance of the thickness within individual layers is comparable to that in the complete formulation. However, the average layer thickness varies significantly across the sequence, highlighting the need for thickness constraints to ensure consistency throughout the fabrication process. In this analysis, the first and last layers are excluded due to challenges in accurately extracting their bounding isosurfaces. However, this exclusion does not compromise the validity of the analysis.

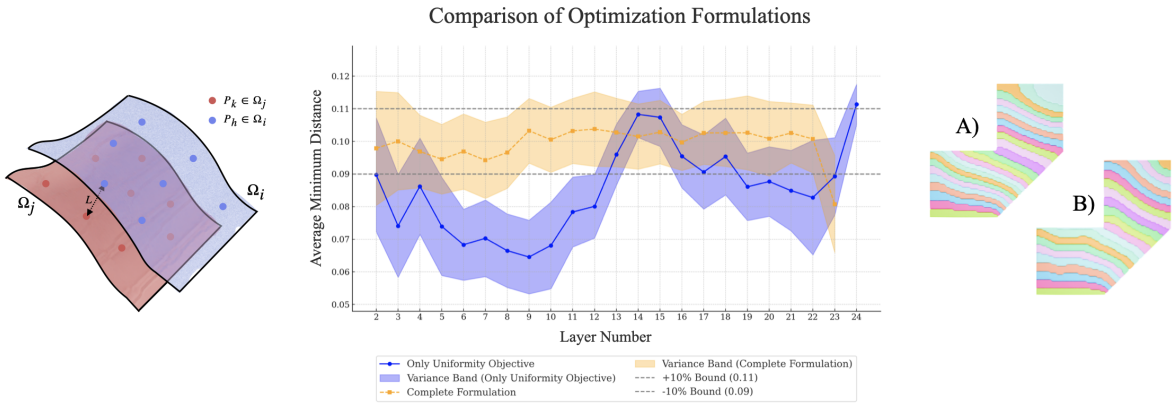


Fig. 24: Comparison between Formulation 2 and Formulation 3. The left image provides a schematic representation of the distance calculation process between consecutive isosurfaces. The middle image presents a plot comparing the average thickness and variance for each layer in the two formulations. The right image shows a side view of the resulting sequences, with A corresponding to Formulation 2 and B to Formulation 3.

2) Bracket-Shaped Component

A second test was performed on a 2D *Bracket Shape* model from Wang et al. [1], extruded into the third dimension. The domain measures $3 \times 1 \times 2$, discretized with $120 \times 40 \times 80$ elements. Nodes on the build plate are constrained as fixed. The number of prescribed layers is set to $N = 30$ with a target thickness of $d_{tar} = 0.07$. The distortion measure in this example considers the displacement of the nodes belonging to the bracket hole, as shown in Figure 21B.

The distortion objective was successfully minimized from 1.98×10^{1} in the planar fabrication case to 6.12×10^{-1} , as shown in the displacement fields in Figure 25. The convergence of the optimization process is depicted

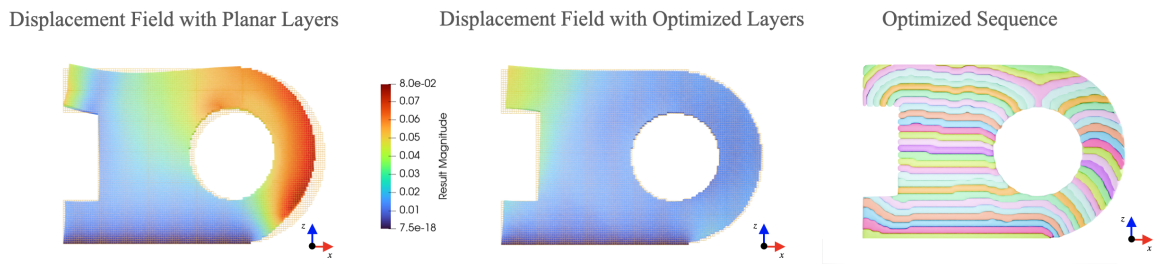


Fig. 25: Comparison of displacement fields computed through process simulation for two Fabrication Sequences: Planar layers (left) and optimized free-form layers (middle). Side view of the optimized fabrication sequence for the bracket shape (right)

in Figure 26, which displays the trends for both the Uniformity and Distortion objectives. The values in the plot are normalized relative to the highest value assumed by their respective objectives during the optimization. However, note that the initial value (corresponding to iteration 1) in the plot does not correspond to the planar fabrication case. The visualization in Figure 26 enable a qualitative analysis of the manufacturability of the resulting sequence. Each layer appears to have a uniform thickness within itself, with thickness values consistent across the whole sequence, satisfying the *Uniformity Objective* and *Layer Thickness Constraints*. Also in this case, while the total number of layers for optimization is predefined at $N = 30$, the optimized sequence includes only 28 layers.

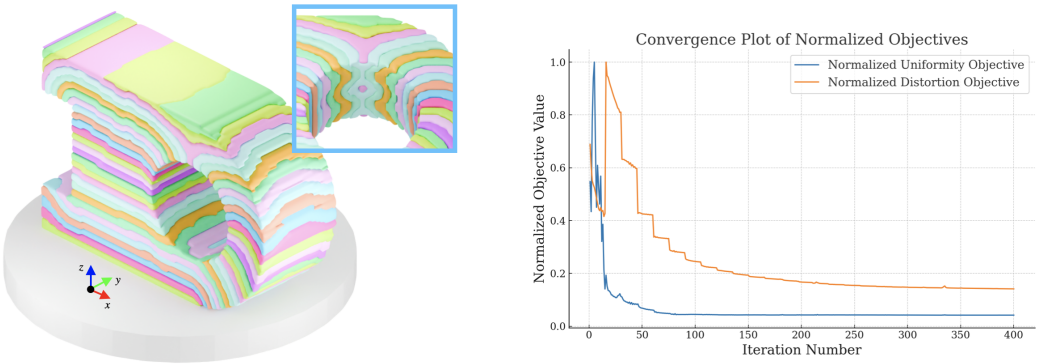


Fig. 26: Visualization of the optimized fabrication sequence, highlighting the features of the last layers (left). Convergence plot of the Uniformity and Distortion objectives (right), both normalized by their respective maximum values recorded during the optimization iterations.

To better appreciate the layers shapes, in Figure 27, 10 out of 30 manufacturing steps are displayed.

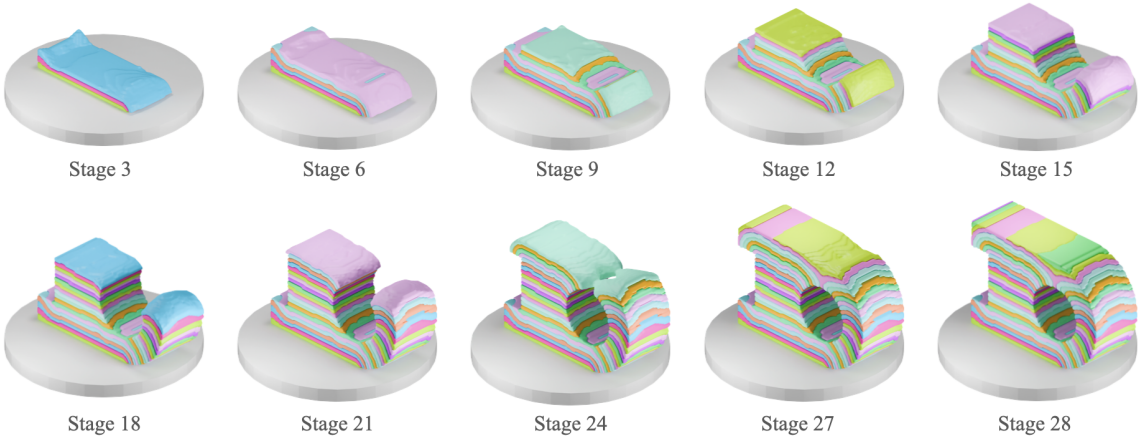


Fig. 27: Visualization of 10 different stages of the optimized fabrication sequence for the bracket shape

3) Asymmetric Shaped Component

Another notable result arises from the asymmetric shape (Figure 21C), selected specifically to test the method's performance on a more complex geometry. The domain of this part measures $3 \times 1 \times 2$ and is discretized with $120 \times 40 \times 80$ elements. Nodes on the build plate are constrained as fixed. The target layer thickness is $d_{tar} = 0.07$ and the number of prescribed layers is set to $N = 30$. In this example, distortion is measured by tracking the displacement of nodes along the top-right edge, as illustrated in Figure 21.

Figures 28 shows the optimized sequence. Similar to the previous example, the optimized sequence consists of only 28 layers instead of the prescribed 30. In the figure attention is drawn to several internal layers, where a prominent concavity is observed. Moreover, the figure displays the convergence plot for the optimization, normalized similarly to the one shown for the previous numerical experiment. The distortion objective is successfully minimized from the planar case 9.66×10^{-1} to 9.2×10^{-4} in the optimized case, as can be observed from the displacement field in Figure 29.

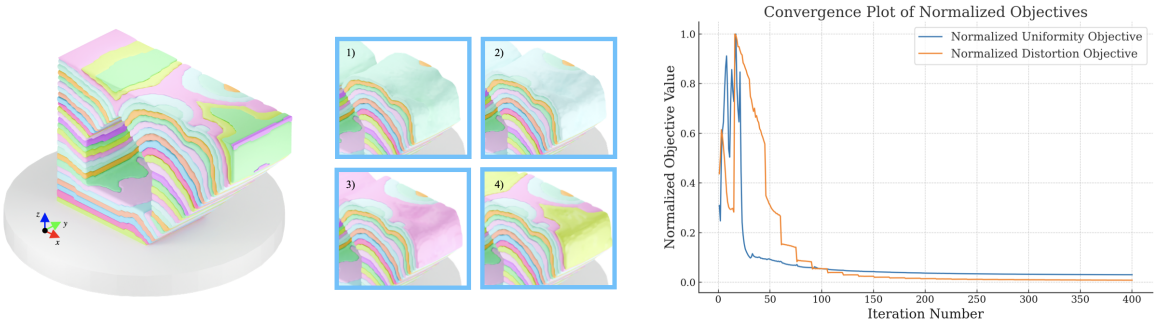


Fig. 28: *Left Image*: Optimized sequence of layers for the Asymmetric Shape. The *blue boxes* highlight features in the layers that may pose manufacturing challenges. The numbers represent the relative order of deposition of the highlighted layers. *Right Image*: Convergence plot of the Uniformity and Distortion objectives, both normalized by their respective maximum values recorded during the optimization iterations.

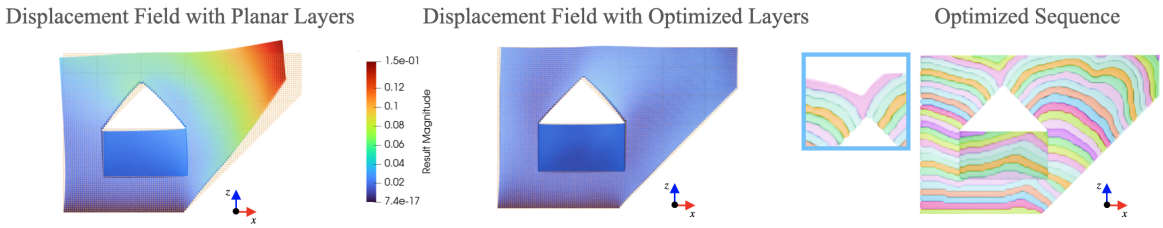


Fig. 29: Comparison of displacement fields computed through process simulation for two Fabrication Sequences: planar layers (left) and optimized free-form layer (middle). Side view of the optimized fabrication sequence for the asymmetric shape (right)

4) Comments on the manufacturability of the first three numerical examples

In the first three numerical examples presented, specific layer features have been highlighted using blue square boxes, as shown in Figures 28, 29, and 26. These features illustrate potential unresolved challenges in the Fabrication Sequence Optimization method. Specifically, in both the bracket shape and asymmetric shape cases, the optimizer generated layers with significant concavities (Figure 28) or layers requiring material deposition in tight spaces (Figures 29, 26). If not addressed appropriately, these scenarios could result in collisions between the printhead and the already deposited material during the manufacturing process. In this study, manufacturability aspect, other than layer uniformity, have not been considered.

5-B Parameter Study - Relative Weight Coefficient Between Objectives

As shown in Equation 41, the objective to be minimized consists of two components: the *Distortion Objective* and the *Uniformity Objective*. The relative weight between these components is controlled by the parameter α, where a higher value of α prioritizes uniformity over distortion minimization. The geometry used in this study is inspired by the reversed L-shape utilized by Lansu [14] to ensure continuity between the two studies. The domain of this part measures $3 \times 1 \times 2$ and is discretized with $120 \times 40 \times 80$ elements. The target layer thickness is $d_{tar} = 0.2$, with a prescribed number of layers $N = 20$. The parameter β_t is initialized at 30 and increased by 20 every 15 iterations with a continuation scheme. As shown in Figure 30, the build plate remains fixed, as in the previous numerical experiments. The nodes considered for the distortion objective are those belonging to the top-right and top-left edges, highlighted in aquamarine in the figure.

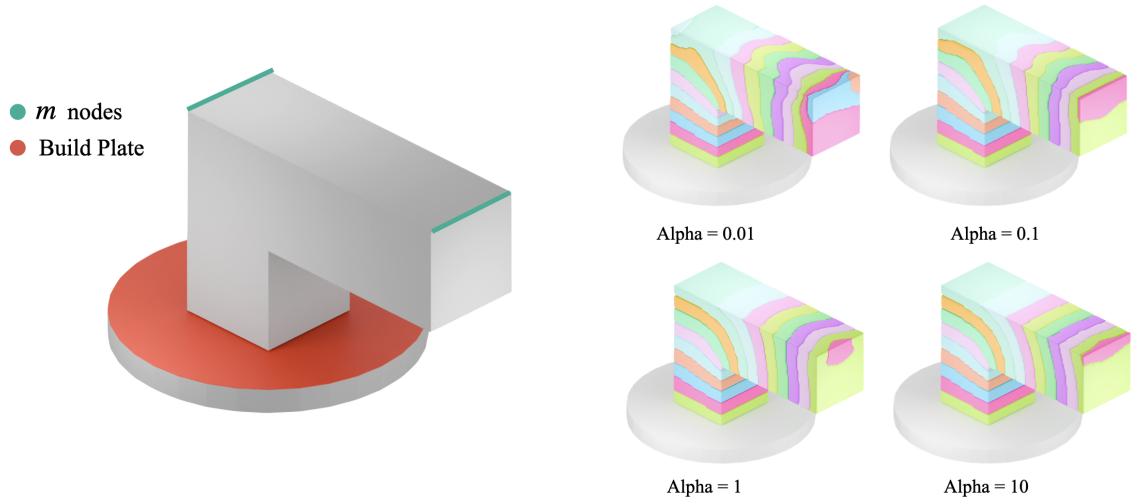


Fig. 30: Numerical experiment on the *L-shape* geometry to assess the influence of the α parameter on the optimization results. The boundary conditions and the nodes considered for the distortion objective are shown on the left, while the four optimization results corresponding to different α values are displayed on the right.

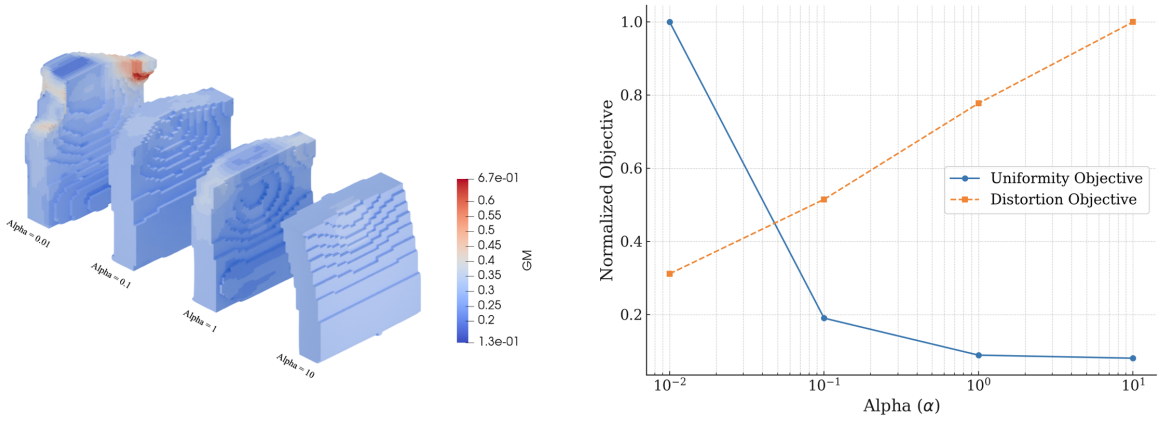


Fig. 31: (A) Visualization of the gradient magnitude field within layer 14. As α increases, the gradients become more uniform, resulting in a more regular layer shape. (B) Plot of the normalized *distortion objective* and normalized *uniformity objective* against the α parameter, highlighting the trade-off between uniformity and distortion minimization.

The results of this parameter study are presented in Figures 30 and 31. The distortion objective values for varying α demonstrate a clear trade-off between distortion minimization and layer uniformity as shown in Figure 31. For $\alpha = 0.01$, the distortion objective is 1.17×10^{-2}, increasing to 1.94×10^{-2} for $\alpha = 0.1$, 2.93×10^{-2} for $\alpha = 1$, and 3.77×10^{-2} for $\alpha = 10$. This trend indicates that as α increases, the optimization process prioritizes uniformity, resulting in higher distortion values. In comparison, the planar fabrication approach produces a significantly larger distortion objective of 3.54, confirming the effectiveness of the proposed optimization method in reducing deformation across all α values.

In the left image of Figure 31, the focus is on an example layer (layer 14). The visualization highlights the gradient magnitude distribution across the layer for different α values. For smaller α values, the variation is higher, leading to a less regular layer shape, while larger α values result in smoother gradients and more uniform shapes.

To further analyze this trend, a quantitative evaluation was conducted by measuring the average layer thickness geometrically, following a method similar to that described in Section 5-A1. The results, shown in Figure 32, indicate that the variance generally decreases with increasing α values for most layers, though some layers deviate from this trend. Based on these results, a reasonable compromise appears to be achieved by using α values between 0.1 and 1. From the figures, it can also be observed that, similar to previous cases, the final number of layers in most of the 4 sequences is 17, instead of the prescribed 20.

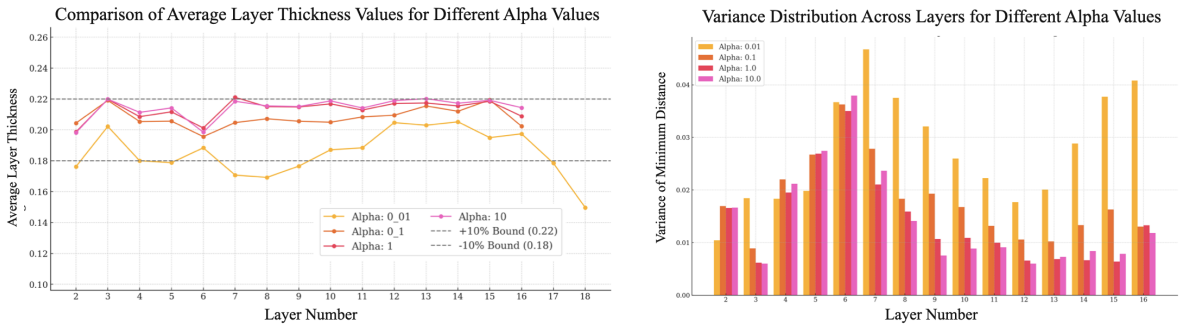


Fig. 32: Left: Plot of the geometrically computed average thickness value for each layer across various α values. Right: Plot of the geometrically computed layer thickness variance for each layer across various α values.

5-C Optimizing using fewer layers

In the proposed formulation, the *Uniformity Objective* (Equation 34) encourages uniformity in the gradient magnitude within each layer. For a given layer j corresponding to the time interval $[T_{j-1}, T_j]$, subdividing this interval into n smaller sub-intervals effectively defines additional layers. Due to the enforced gradient magnitude uniformity, the newly created layers will also exhibit consistent gradient values and, consequently, uniform thickness. This approach can be leveraged to perform Fabrication Sequence Optimizations with a reduced number of layers and subsequently, in a post-processing step, extract the desired number of layers with the required thickness from the optimized pseudo-time field.

To test this hypothesis, three numerical experiments were conducted using the same part geometry—specifically, the V-shape introduced in Section 5-A1. The part uses the same boundary conditions and distortion measure as in the experiment described in Section 5-A1. The domain of this part measures $2 \times 1 \times 2$ and is discretized with $80 \times 40 \times 80$ elements. The parameter β_t is initialized at 10 and increases by 20 every 15 iterations using a continuation scheme. The experiments varied the number of layers used during the optimization process: 10 layers, 20 layers, and 40 layers. The target layer thickness is $d_{\text{tar}} = 0.2$ for the 10 layers case, $d_{\text{tar}} = 0.1$ for the 20 layers case, and $d_{\text{tar}} = 0.05$ for the 40 layers case. After 400 optimization iterations, a post-processing step was applied to extract and simulate the process with 40 layers from each of the three optimized pseudo-time fields. The geometric properties of the layers and the corresponding distortion minimization performances are analyzed. The results can be observed in Figure 33 and Table I. For this analysis, the side view of the optimized Fabrication Sequence is sufficient.

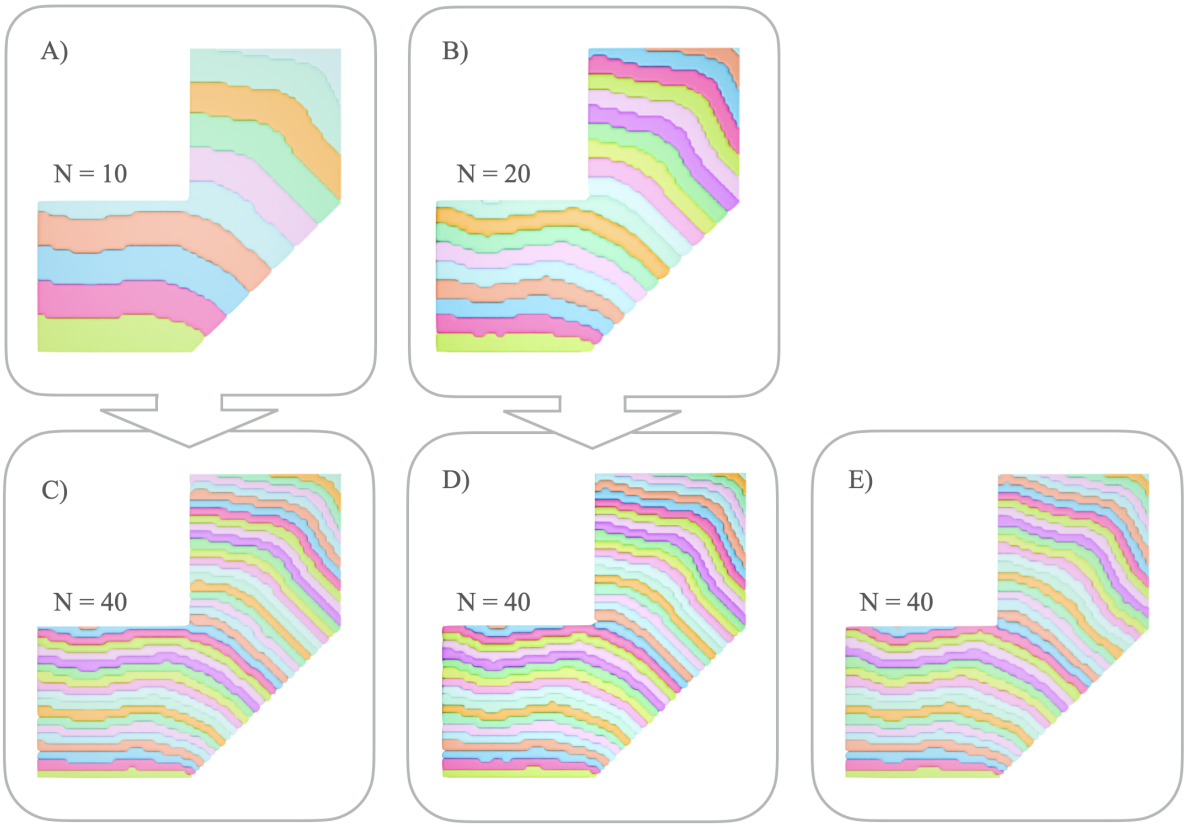


Fig. 33: Side view of optimized Fabrication Sequences obtained with different numbers of prescribed layers. A) and B) Sequence obtained with 10 prescribed layers and a sequence of 40 layers derived through a post-processing step from the 10-layer sequence. C) and D) Sequence obtained with 20 prescribed layers and a sequence of 40 layers derived through a post-processing step from the 20-layer sequence. E) Sequence optimized directly with 40 prescribed layers.

Figure 33 confirms the expectations. In all the analyzed cases, the sequences obtained through the post-processing step appear to be uniform and maintain a controlled thickness value in each layer. Table I presents quantitative metrics, from which it can be observed that the average thickness remains approximately constant across all three cases ($d = 0.048$). Interestingly, the average thickness variance in the sequences obtained through post-processing is actually lower. This observation can be explained by considering Equation 41, where the

Original N of Layers	Avg. Variance (40 Layers)	Avg. Thickness (m)	Obj. Value (40 Layers)	Orig. Obj. Value
Planar Case: 40	0.00	5.00×10^{-2}	7.04×10^{-1}	7.04×10^{-1}
10	3.38×10^{-3}	4.80×10^{-2}	1.43×10^{-2}	1.09×10^{-2}
20	3.65×10^{-3}	4.80×10^{-2}	4.31×10^{-3}	3.97×10^{-3}
40	3.82×10^{-3}	4.79×10^{-2}	2.00×10^{-3}	2.00×10^{-3}

TABLE I: Data representative of the optimized sequence properties, such as distortion objective value and geometrically approximated thickness.

Uniformity Objective term is computed by summing across all stages, the sum of squared deviations of the gradient's magnitude. Since the uniformity objective is firstly scaled to a constant value (100) before being multiplied by alpha, as the number of prescribed layers N increases during the optimization, the relative weight of each layer's uniformity objective diminishes in comparison to the distortion objective. The distortion objective obtained by simulating the process with 40 layers from each optimized sequence is 1.434×10^{-2} for the case optimized with 10 layers, 4.31×10^{-3} for the case optimized with 20 layers, and 2×10^{-3} for the case optimized with 40 layers, all significantly lower compared to planar fabrication with 40 layers which lead to a distortion objective of 7.04×10^{-1}.

5-D Code Performance Metrics

While PETSc accelerates the development process by abstracting certain implementation details, this abstraction reduces the user's direct control over the parallelization. Although the main objective was to enable large-scale 3D simulations to run within a reasonable timeframe, a deeper assessment of parallel performance can inform future improvements. Thus, a performance evaluation was conducted to gauge the quality of the current implementation, with a focus on speedup and scalability. These metrics reveal how effectively the code scales as computational resources increase for a fixed-size problem, shedding light on both the achievable gains and inherent limitations within the current parallel setup. All the numerical experiments were run on the same node consisting of 4 Intel Xeon Gold 6150 CPU for a total of 72 physical cores, running at 2.70 GHz on an x86_64 architecture, and equipped with a 25 MB L3 cache. As an initial performance measure, the numerical experiments detailed in Section 5-A, ran on the just described hardware, took approximately 13 hours for the V-Shape simulation and 30 hours for the other two shapes, each running for 400 iterations.

1) Problem Definition and Baseline Selection
The benchmark problem used in this analysis is the V-shaped part from Section 5-A1, discretized with $80 \times 40 \times 80$ elements. This setup is consistent with prior test example and chosen for computational relevance.

The code was executed across different processor counts, with $n \in \{1, 2, 4, 8, 16, 24, 32, 40, 48, 56, 64, 72\}$. Each run consisted of 20 optimization loop iterations, recording timing data at each step. To account for initialization effects (e.g., cache setup), we analyzed only the last 10 iterations, representing a more steady-state performance. While timing was collected for runs beginning with a single core ($n = 1$), our analysis uses 4 cores as the baseline for performance comparisons. This choice aligns with the practical considerations of parallel performance studies. According to theory[38], a serial optimized version of the code, run on a single core should ideally be used as a reference. However, in this case, where such a version is unavailable, we use 4 cores as a baseline to provide a more realistic view of code scaling in real-world scenarios. Running a parallelized code on a single core can introduce non-representative overheads related to setup and parallel configuration costs, which might lead to an overestimation of the performances.

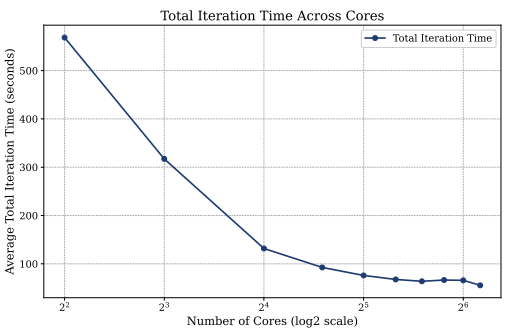


Fig. 34: Scalability plot showing total iteration time on the y-axis versus the number of processors on the x-axis. A strong scalability analysis assumes a fixed problem size.

n_p	$T(n_p)$ $[s]$	$S(n_p)$	$E(n_p)$
4	568.36	1.00	0.2500
8	317.19	1.79	0.2239
16	131.73	4.31	0.2697
24	92.54	6.14	0.2559
32	75.99	7.48	0.2337
40	67.68	8.40	0.2099
48	63.87	8.90	0.1854
56	66.47	8.55	0.1527
64	65.84	8.63	0.1349
72	55.49	10.24	0.1422

TABLE 2: Execution time, speedup, and efficiency values for an increasing number of processors with a fixed problem size.

The total iteration time, $T(n_p)$ reported in Table 4, represents the average runtime required to complete an optimization loop iteration on n_p cores. As shown in Figure 34, $T(n_p)$ decreases as the number of cores increases. Starting at 568.36 seconds on 4 cores, the runtime reduces to 55.49 seconds on 72 cores—a reduction of approximately 90.2%. However, this reduction is nonlinear; instead, we observe diminishing returns at higher core counts. For instance, moving from 4 to 8 cores decreases the iteration time by 44.2% (from 568.36 to 317.19 seconds), whereas increasing from 32 to 64 cores results in only a 13.4% reduction (from 75.99 to 65.84 seconds).

This pattern of diminishing returns is typical in parallel computing, often attributed to the non-parallelizable (serial) portion of the algorithm and additional overhead, such as communication and synchronization costs. Beyond a certain point, adding more cores minimally impacts runtime reduction. This trend follows *Amdahl's Law* [39], which provides a speedup formula accounting for the serial portion of the code. According to *Amdahl's Law*, as the core count increases, the influence of non-parallelizable code becomes more significant,

leading to an *asymptotic limit on achievable speedup*. This behavior is evident in the *speedup* plot in Figure 35.

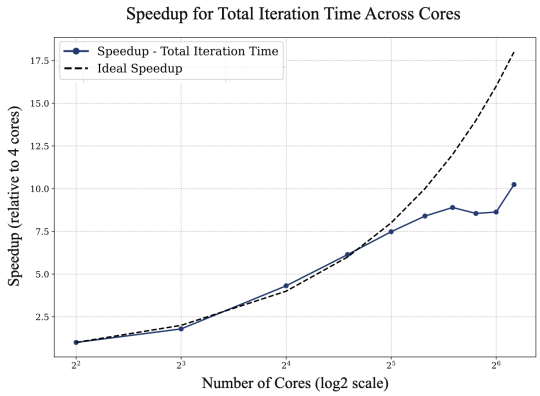


Fig. 35: Scalability plot showing speedup on the y-axis versus the number of processors on the x-axis.

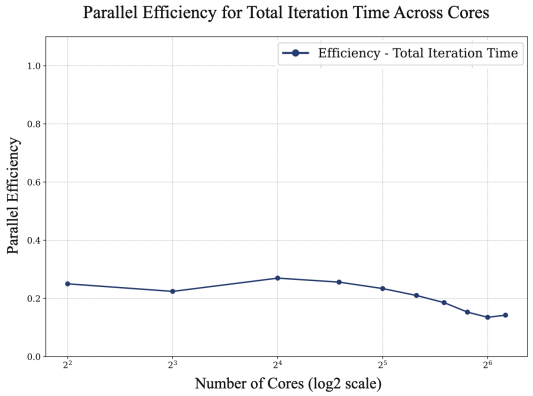


Fig. 36: Scalability plot showing efficiency on the y-axis versus the number of processors on the x-axis.

The speedup metric, $S(n_p)$, quantifies the performance gain from increasing the core count and is calculated as:

$$S(n_p) = \frac{T(4)}{T(n_p)} \tag{50}$$

where $T(4)$ is the runtime on the 4-core baseline, and $T(n_p)$ is the runtime on n_p cores. This metric provides insight into how effectively the code scales with more cores. Analyzing the plot in Figure 35, we observe that speedup initially follows a reasonable trend: at 8 cores, speedup is 1.79, close to the ideal 2.00, and at 16 cores, speedup reaches 4.31, slightly surpassing the ideal of 4.00. This suggests the code benefits substantially from parallelization up to 16 cores, likely due to efficient workload distribution and minimal parallel overhead. However, as core count increases, speedup begins to diverge from the ideal linear growth. For instance, at 32 cores, the speedup is only 7.48 compared to the ideal 8.00. This gap becomes even more pronounced at 72 cores, where speedup reaches just 10.24, falling significantly short of the ideal 18.00. This discrepancy indicates both the presence of serial code that limits scalability and the impact of overhead costs that grow with core count.

To better understand the scalability limiting factors of a parallel implementations, Karp and Flatt [40] introduced the *effective serial portion* metric f, defined by the formula:

$$f = \frac{\frac{1}{S(n_p)} - \frac{1}{n_p}}{1 - \frac{1}{n_p}} \tag{51}$$

This metric effectively reveals whether the decrease in efficiency is primarily caused by the non-parallelizable portion of the code or by other factors, such as load imbalance, communication costs, and synchronization overheads.

Table 3 presents the effective serial fraction f calculated using the Karp-Flatt metric for varying processor counts n_p and observed speedups $S(n_p)$. The serial fraction f decreases as the number of processors increases, stabilizing at around 0.081.

This behavior suggests that the primary limitation on further speedup lies in the serial portion of the code rather than in overhead from communication, synchronization, or load imbalance. As f stabilizes, adding more processors provides diminishing returns, since the remaining non-parallelizable part of the code constrains speedup gains. This observation aligns with Amdahl's Law, indicating that to achieve further improvements, efforts should focus on optimizing or reducing the serial fraction of the code. The stable serial fraction thus seems to highlight the inherent limits of parallelism for this particular workload.

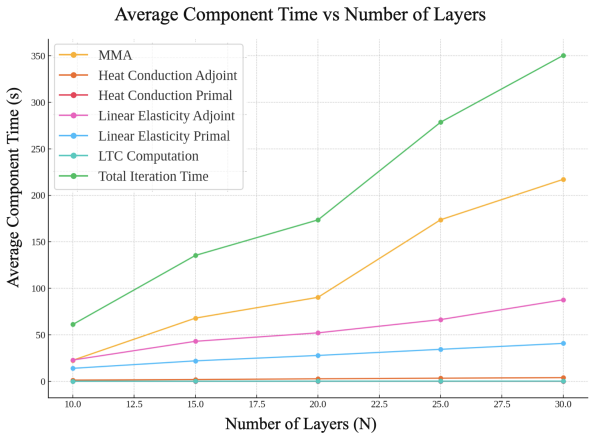


Fig. 37: Execution time of various code sub-components (y-axis) plotted against the number of layers used in the optimization (x-axis).

n_p	$S(n_p)$	f
8	1.79	0.496
16	4.31	0.181
24	6.14	0.126
32	7.48	0.106
40	8.40	0.092
48	8.90	0.086
56	8.55	0.084
64	8.63	0.083
72	10.24	0.081

TABLE 3: Speedup and effective serial fraction f, calculated using the Karp-Flatt metric for varying processor counts n_p.

Another test has been conducted to test the scalability of the implementation for increasing number of manufacturing process stages which can be visualized in the plot in Figure 37. As the number of layers N increases, runtime components respond differently, reflecting distinct scaling behaviors. *Heat Adjoint* and *Layer Thickness Control (LTC)* times per layer remain stable, indicating linear scaling, as these processes handle each additional layer with minimal added complexity. *Linear Elasticity* (Primal and Adjoint) times per layer also scale linearly with N, showing only a slight rise, suggesting manageable per-layer costs as N grows. In contrast, *MMA* time per layer grows significantly with N, showing superlinear scaling. This suggests that, unlike the other components, *MMA* complexity escalates with more layers, driving the total iteration time upward and indicating that this component may limit scalability as layers increase.

In Table 4 results are shown about a timing analysis focused on evaluating the component-level performance. Timings over the course of multiple iterations are used to obtain reliable averages, and understand the relative computational cost associated with each component, aiding in the identification of the distribution of computational effort across the optimization workflow. This timing experiment was run with 16 cores as this configuration demonstrated the highest computational efficiency.

Component	Avg Time (s)	Percent Total
Heat Primal Time	0.078	0.059%
Elasticity Primal Time	28.508	21.642%
Elasticity Adjoint Time	63.006	47.831%
LTC Time	0.135	0.102%
Heat Adjoint Time	2.482	1.884%
MMA Optimization Time	37.117	28.178%

TABLE 4: Table listing the execution time of different code subcomponents and the percentage of total iteration time each component takes.

In this implementation, a key aspect is the assembly of the *Stiffness Matrix* $\boldsymbol{K}$ for process simulation. Matrix assembly is resource-intensive, and here $2 \times N$ matrices need to be constructed. While storing these matrices for reuse in the adjoint solve step would be efficient, it would also lead to high memory demands. To balance memory use, only one stiffness matrix is stored at a time, which lowers memory requirements but slows down overall performance.

2) Comparison with Wang et al. [1], 3D MATLAB implementation performances

This subsection presents a direct comparison of execution time between the current PETSc-based implementation and the MATLAB implementation described in Wang et al. [1]. The asymmetric shape numerical example used in Wang's work has been adopted for this comparison. The part domain measures $3 \times 1 \times 2$ and is discretized into $60 \times 20 \times 40$ elements. Nodes on the build plate are fixed, and the number of prescribed layers is set to $N = 10$.

There are slight differences between the two optimization problems. Specifically, Wang's formulation, as

introduced in Section 2-C1, includes a *Continuity Constraint* and a *Manufacturing Speed Constraint*, whereas the current optimization employs only *Layer Thickness Constraints* and a modified objective incorporating the *Layer Uniformity* term. Additionally, Wang's optimization evaluates distortion based on the flatness of the top surface, while the current approach measures distortion as the displacement of the top-right and top-left edges, similar to the numerical experiment in Subsection 5-B.

Furthermore, the current approach incorporates the *Thermal Regularization Method*, which accounts for approximately 2% of the total computation time. However, as observed in Table 4, the majority of computation time is consumed by components shared between both approaches. This overlap makes the comparison of execution times between the MATLAB and PETSc implementations valid.

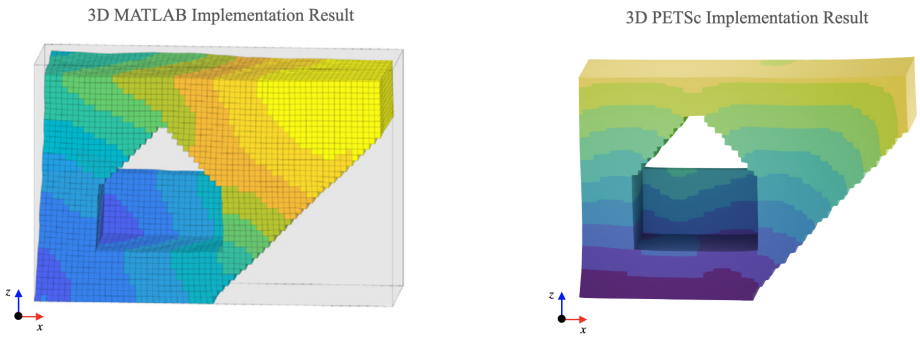


Fig. 38: Optimization results obtained using the MATLAB implementation and formulation from Wang et al. [1] (left) compared to the optimization results using the current method with the PETSc implementation (right).

In the study by Wang et al. [1], the authors report a total execution time of slightly less than $15\,\mathrm{h}$ for 400 optimization loop iterations. Using the PETSc implementation with 72 cores, the total execution time for the same number of iterations was approximately $1.31\,\mathrm{h}$. Compared to the reference execution time of $15\,\mathrm{h}$, this represents about 8.74% of the total, indicating a significant speedup, of approximately $11.44\times$. While the amount of core used is 72, from the knowledge gained in the efficiency study, and considering the smaller mesh dimensions, the same performances could have been probably achieved with a smaller number of processors.

6 Conclusion

This research presented a novel 3D computational framework for optimizing fabrication sequences in multi-axis additive manufacturing, specifically targeting layer thickness control and reduction of process-induced deformations. The proposed framework successfully produced fabrication sequences that minimize deformation while improving manufacturability. Compared to previously developed formulations, such as Lansu's 2D method [14], the approach presented here simplifies the formulation, increasing the confidence on its robustness.

The integration of the thermal regularization method for the pseudo-time field computation proved crucial in eliminating local minima, thus leading to improved convergence and manufacturability. Robustness was demonstrated across various part geometries, confirming that the method is versatile and can be readily adapted for different structural layouts. The parallel implementation using the PETSc library demonstrated a significant computational speed-up, allowing efficient evaluation of large-scale numerical experiments and establishing a solid foundation for future experimental validation.

A quantitative assessment of the parallel performance confirmed linear scaling of iteration time with respect to the number of prescribed layers, N. This linear relationship is critical since, in practical manufacturing scenarios, N could be much larger than the values used in this study, potentially posing a challenge from a computational perspective. Fortunately, two promising methods exist to mitigate this. First, leveraging the approach by Wang et al. [1], combined with Munro et al.'s [22] work, allows for parallel computation of the N linear system solves in the process simualtion step. This could be integrated with the domain decomposition approach already used in this research, potentially yielding substantial speed improvements. The second approach explored here involves optimizing the sequence with a reduced number of layers and subsequently increasing the layer count during a post-processing step. Although this approach results in marginally higher distortion objective values, it proved feasible, satisfying layer thickness constraints while maintaining good layer uniformity. Importantly, distortion values remained significantly lower than those observed for conventional planar fabrication.

The experiment with reduced layer count also highlighted possible improvements regarding objective scaling. Specifically, as the number of manufacturing stages increases, the relative influence of the each layer uniformity objective diminishes compared to the distortion objective. Future work should address ensuring scale-independent weighting to maintain uniformity control across varying manufacturing scenarios.

However, some limitations persist. Controlling the thickness of the last fabricated layer remains problematic, often resulting in greater variability. Additionally, practical considerations such as collision detection between the printhead and deposited material have not yet been addressed, potentially impacting manufacturability in real-world applications. The current layer uniformity objective is based on the sum of squared deviations of the time field gradient magnitudes from the mean value, a metric that tends to increase for layers with more elements, possibly biasing the optimization. Introducing a true variance metric may offer improvements in achieving uniformity across layers. Further, while the method showed robustness across different geometries, only a limited set was explored; more complex and topology-optimized components should be considered in future studies for more comprehensive numerical validation.

Future work will aim to address these challenges. Integrating collision detection into fabrication sequence planning would further improve manufacturability and enhance the method's applicability in real-world environments. Enhancing control over the final layer's thickness will also be essential for achieving consistent predictable behavior. A key next step involves experimental validation to establish the practical feasibility of the proposed optimization framework and to verify its advantages over conventional planning strategies. Lastly, moving beyond the inherent strain method towards more comprehensive process simulations could significantly improve the accuracy of deformation predictions.

A Appendix A
Sensitivity Analysis

A-A Sensitivity of the Distortion Objective

$L_{\text{dist}}(\boldsymbol{\kappa}) = \boldsymbol{u}^\top \boldsymbol{Q}\boldsymbol{u}$

Note: In the following equations, $\boldsymbol{u}$, $\boldsymbol{K}^{\{j\}}$, $\Delta\boldsymbol{u}^{\{j\}}$, and $\boldsymbol{f}^{\{j\}}$ are functions of $\boldsymbol{\kappa}$, but for brevity, the dependency is not shown explicitly.

The sensitivity is calculated using the adjoint method by adding an augmented Lagrangian term.

$L = L_{\text{dist}} + \boldsymbol{\lambda}^\top(\boldsymbol{K}_\mathsf{T}\boldsymbol{T} - \boldsymbol{b})$

In the equation above, $\boldsymbol{K}_T$ is the global system matrix relative to the static-heat transfer FEM, $\boldsymbol{T}$ is the nodal temperature vector, and $\boldsymbol{b}$ is the thermal load vector corresponding to the boundary condition.

$$\frac{\partial L}{\partial \kappa_e} = \frac{\partial L_{\text{dist}}}{\partial \boldsymbol{t}}\frac{\partial \boldsymbol{t}}{\partial \kappa_e} + \boldsymbol{\lambda}^\top\left(\frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e}\boldsymbol{T} + \boldsymbol{K}_\mathsf{T}\frac{\partial \boldsymbol{T}}{\partial \kappa_e}\right)$$

Recalling that $\boldsymbol{t} = \boldsymbol{1} - \boldsymbol{G}\boldsymbol{T}$:

$$\begin{aligned}\frac{\partial L}{\partial \kappa_e} &= -\left(\frac{\partial L_{\text{dist}}}{\partial \boldsymbol{t}}\right)^\top \boldsymbol{G}\frac{\partial \boldsymbol{T}}{\partial \kappa_e} + \boldsymbol{\lambda}^\top\left(\frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e}\boldsymbol{T} + \boldsymbol{K}_\mathsf{T}\frac{\partial \boldsymbol{T}}{\partial \kappa_e}\right)\\ &= \left(-\left(\frac{\partial L_{\text{dist}}}{\partial \boldsymbol{t}}\right)^\top \boldsymbol{G} + \boldsymbol{\lambda}^\top\boldsymbol{K}_\mathsf{T}\right)\frac{\partial \boldsymbol{T}}{\partial \kappa_e} + \boldsymbol{\lambda}^\top\frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e}\boldsymbol{T}\end{aligned}$$

By choosing $\boldsymbol{\lambda}$ such that the first term is equal to zero, the sensitivity expression can be simplified as follows:

$$-\left(\frac{\partial L_{\text{dist}}}{\partial \boldsymbol{t}}\right)^\top \boldsymbol{G} + \boldsymbol{\lambda}^\top\boldsymbol{K}_\mathsf{T} = 0 \rightarrow \frac{\partial L}{\partial \kappa_e} = \boldsymbol{\lambda}^\top\frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e}\boldsymbol{T}$$

To be able to solve the adjoint equation for the Lagrangian multiplier $\boldsymbol{\lambda}$, the derivative of L_{dist} with respect to the time field $\boldsymbol{t}$ needs to be calculated. To do so, another adjoint problem has to be solved, thus the expression is augmented with a new Lagrangian term $\boldsymbol{\gamma}$.

$$L = \boldsymbol{u}^\top\boldsymbol{Q}\boldsymbol{u} + \sum_{j=1}^{N}\boldsymbol{\gamma}_j^\top\left(\boldsymbol{K}^{\{j\}}\Delta\boldsymbol{u}^{\{j\}} - \boldsymbol{f}^{\{j\}}\right)$$

$$\begin{aligned}\frac{\partial L}{\partial t_e} &= 2\boldsymbol{u}^\top\boldsymbol{Q}\frac{\partial \boldsymbol{u}}{\partial t_e} + \sum_{j=1}^{N}\boldsymbol{\gamma}_j^\top\left(\frac{\partial \boldsymbol{K}^{\{j\}}}{\partial t_e}\Delta\boldsymbol{u}^{\{j\}} + \boldsymbol{K}^{\{j\}}\frac{\partial \Delta\boldsymbol{u}^{\{j\}}}{\partial t_e} - \frac{\partial \boldsymbol{f}^{\{j\}}}{\partial t_e}\right),\\ &= 2\boldsymbol{u}^\top\boldsymbol{Q}\sum_{j=1}^{N}\frac{\partial \Delta\boldsymbol{u}^{\{j\}}}{\partial t_e} + \sum_{j=1}^{N}\boldsymbol{\gamma}_j^\top\left(\frac{\partial \boldsymbol{K}^{\{j\}}}{\partial t_e}\Delta\boldsymbol{u}^{\{j\}} + \boldsymbol{K}^{\{j\}}\frac{\partial \Delta\boldsymbol{u}^{\{j\}}}{\partial t_e} - \frac{\partial \boldsymbol{f}^{\{j\}}}{\partial t_e}\right)\end{aligned}$$

Factoring the terms to isolate those that multiply the direct derivative of the state with respect to the design variables, we arrive at:

$$\frac{\partial L}{\partial t_e} = \sum_{j=1}^{N}\left(2\boldsymbol{u}^\top\boldsymbol{Q} + \boldsymbol{\gamma}_j^\top\boldsymbol{K}^{\{j\}}\right)\frac{\partial \Delta\boldsymbol{u}^{\{j\}}}{\partial t_e} + \sum_{j=1}^{N}\boldsymbol{\gamma}_j^\top\left(\frac{\partial \boldsymbol{K}^{\{j\}}}{\partial t_e}\Delta\boldsymbol{u}^{\{j\}} - \frac{\partial \boldsymbol{f}^{\{j\}}}{\partial t_e}\right)$$

From this expression, we can identify the adjoint problem, which leads us to a simpler expression for the sensitivities.

$$\boldsymbol{K}^{\{j\}}\boldsymbol{\gamma}_j + 2\boldsymbol{Q}^\top \boldsymbol{u} = 0 \rightarrow \boldsymbol{K}^{\{j\}}\boldsymbol{\gamma}_j = -2\boldsymbol{Q}^\top \boldsymbol{u}, \quad j = 1, 2, \ldots, N$$

$$\frac{\partial L}{\partial t_e} = \sum_{j=1}^{N} \boldsymbol{\gamma}_j^\top \left(\frac{\partial \boldsymbol{K}^{\{j\}}}{\partial t_e} \Delta \boldsymbol{u}^{\{j\}} - \frac{\partial \boldsymbol{f}^{\{j\}}}{\partial t_e} \right)$$

Recalling that the stiffness matrix for each element $\boldsymbol{K}_e^{\{j\}}$ is obtained through the Solid Isotropic Material with Penalisation (SIMP) method, and that the forcing vector for each element $\boldsymbol{f}_e^{\{j\}}$ is also obtained through a penalization scheme:

$$\boldsymbol{K}_e^{\{j\}} = \left(E_{\text{min}} + (1 - E_{\text{min}}) \times \left(\rho_e^{\{j\}} \right)^p \right) \boldsymbol{K}_0,$$

and

$$\boldsymbol{f}_e^{\{j\}} = [\boldsymbol{D}]^\mathsf{T}[\boldsymbol{C}]\Delta\underline{\boldsymbol{\varepsilon}}^{*\{j\}}\big|_e \quad \text{with} \quad \Delta\underline{\boldsymbol{\varepsilon}}^{*\{j\}}\big|_e = \left(\Delta\rho^{\{j\}}\big|_e \right)^q \underline{\boldsymbol{\varepsilon}}^*.$$

Since both $\rho_e^{\{j\}}$ and $\Delta\rho_e^{\{j\}}$ depend on t_e, the chain rule is applied for the derivatives with respect to t_e.

The derivative of $\boldsymbol{K}_e^{\{j\}}$ with respect to t_e is:

$$\frac{\partial \boldsymbol{K}_e^{\{j\}}}{\partial t_e} = \frac{\partial \boldsymbol{K}_e^{\{j\}}}{\partial \rho_e^{\{j\}}} \cdot \frac{\partial \rho_e^{\{j\}}}{\partial t_e}, \quad \text{where} \quad \frac{\partial \boldsymbol{K}_e^{\{j\}}}{\partial \rho_e^{\{j\}}} = p\,(1 - E_{\text{min}}) \left(\rho_e^{\{j\}} \right)^{p-1} \boldsymbol{K}_0.$$

The derivative of $\boldsymbol{f}_e^{\{j\}}$ with respect to t_e is:

$$\frac{\partial \boldsymbol{f}_e^{\{j\}}}{\partial t_e} = \frac{\partial \boldsymbol{f}_e^{\{j\}}}{\partial \Delta\rho_e^{\{j\}}} \cdot \frac{\partial \Delta\rho_e^{\{j\}}}{\partial t_e}, \quad \text{where} \quad \frac{\partial \boldsymbol{f}_e^{\{j\}}}{\partial \Delta\rho_e^{\{j\}}} = [\boldsymbol{D}]^\mathsf{T}[\boldsymbol{C}] \cdot q \left(\Delta\rho_e^{\{j\}} \right)^{q-1} \underline{\boldsymbol{\varepsilon}}^*.$$

The derivatives $\frac{\partial \rho_e^{\{j\}}}{\partial t_e}$ and $\frac{\partial \Delta\rho_e^{\{j\}}}{\partial t_e}$ can be trivially derived from the equations displayed in Section 3-D. $\frac{\partial \boldsymbol{K}_e^{\{j\}}}{\partial t_e}$ and $\frac{\partial \boldsymbol{f}_e^{\{j\}}}{\partial t_e}$ are used to assemble the necessary $\frac{\partial \boldsymbol{K}^{\{j\}}}{\partial t_e}$ and $\frac{\partial \boldsymbol{f}^{\{j\}}}{\partial t_e}$.

The derivative of the heat stiffness matrix $\frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e}$ instead can be trivially derived from 18.

In case an averaging step like the one in Equation 21 is used, thus the heat equation is solved using the averaged diffusivities $\boldsymbol{\kappa}_{\text{avg}}$, to obtain the sensitivities with respect to $\boldsymbol{\kappa}$ an extra matrix multiplication has to be performed:

$$\frac{\partial f}{\partial \boldsymbol{\kappa}} = \frac{\partial f}{\partial \boldsymbol{\kappa}_{\text{avg}}} \frac{\partial \boldsymbol{\kappa}_{\text{avg}}}{\partial \boldsymbol{\kappa}} \rightarrow \frac{\partial f}{\partial \boldsymbol{\kappa}_{\text{avg}}} \boldsymbol{H} = \boldsymbol{H}^\top \frac{\partial f}{\partial \boldsymbol{\kappa}_{\text{avg}}}$$

A-B Sensitivity of the Layer Uniformity Objective

The uniformity objective formulation, presented in the methodology section in Equation 34 is as follows,

$$L_{\text{unif}} = \sum_{j=1}^{N} L_j \quad \text{with} \quad L_j = \Delta\boldsymbol{\rho}^{\{j\}} \cdot (\boldsymbol{M} - \boldsymbol{m}_j)^2, \tag{52}$$

Where, $\Delta\boldsymbol{\rho}^{\{j\}}$ represents the density distribution associated with the j-th layer and it is computed from Equation 23, $\boldsymbol{M}$ represents the vector of gradient magnitudes, and $\boldsymbol{m_j}$ is a vector of the same dimension as $\boldsymbol{M}$, with each element equal to the average gradient magnitude m_j in the j-th layer. This formulation aims at minimizing the sum of squared deviations of the gradient magnitudes within each layer.

To compute the sensitivity of the *Uniformity Objective* with respect to the diffusivities $\boldsymbol{\kappa}$, the Adjoint method is necessary also in this case, Thus an augmented lagrangian term is added to the objective:

$$L = L_{\text{unif}} + \boldsymbol{\lambda}^\top (\boldsymbol{K}_\mathsf{T}\boldsymbol{T} - \boldsymbol{b})$$

Following similar step to above,

$$\frac{\partial L}{\partial \kappa_e} = \frac{\partial L_{\text{unif}}}{\partial \boldsymbol{T}} \frac{\partial \boldsymbol{T}}{\partial \kappa_e} + \boldsymbol{\lambda}^\top \left(\frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e} \boldsymbol{T} + \boldsymbol{K}_\mathsf{T} \frac{\partial \boldsymbol{T}}{\partial \kappa_e} \right)$$

$$\frac{\partial L}{\partial \kappa_e} = \left(\frac{\partial L_{\text{unif}}}{\partial \boldsymbol{T}} + \boldsymbol{\lambda}^\top \boldsymbol{K}_\mathsf{T} \right) \frac{\partial \boldsymbol{T}}{\partial \kappa_e} + \boldsymbol{\lambda}^\top \frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e} \boldsymbol{T}$$

By choosing $\boldsymbol{\lambda}$ such that the first term is equal to zero, the sensitivity expression can be simplified as follows:

$$\frac{\partial L_{\text{unif}}}{\partial \boldsymbol{T}} + \boldsymbol{\lambda}^\top \boldsymbol{K}_\mathsf{T} = 0 \rightarrow \frac{\partial L}{\partial \kappa_e} = \boldsymbol{\lambda}^\top \frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e} \boldsymbol{T}$$

To be able to solve the adjoint equation for the Lagrangian multiplier $\boldsymbol{\lambda}$, the derivative of L_{unif} with respect to the virtual temperture field $\boldsymbol{T}$ needs to be calculated.

$$\frac{\partial L_{\text{unif}}}{\partial \boldsymbol{T}} = \sum_{j=1}^{N} \frac{\partial L_j}{\partial \boldsymbol{T}}$$

$$\frac{\partial L_j}{\partial \boldsymbol{T}} = \left(\frac{\partial \Delta\boldsymbol{\rho}^{\{j\}}}{\partial \boldsymbol{T}} \right)^\top (\boldsymbol{M} - \boldsymbol{m}_j)^2 + \left\{ \left[2(\boldsymbol{M} - \boldsymbol{m}_j) \odot \frac{\partial \boldsymbol{M}}{\partial \boldsymbol{T}} \right]^\top \Delta\boldsymbol{\rho}^{\{j\}} \right\} - \left[2(\boldsymbol{M} - \boldsymbol{m}_j) \right]^\top \Delta\boldsymbol{\rho}^{\{j\}} \cdot \frac{\partial m_j}{\partial \boldsymbol{T}}.$$

The symbol $\odot$ represent the point wise multiplication operation.

1) Derivative of $\boldsymbol{M}$ with respect to $\boldsymbol{T}$
Remembering the definition of the elemental average gradient magnitude vector, where $\boldsymbol{G}_x$, $\boldsymbol{G}_y$, and $\boldsymbol{G}_z$ are matrices used to combine the nodal virtual temperature value to compute respectively the x, y, and z components of the gradient (as introduced in Section 3-G):

$$\boldsymbol{M} = \sqrt{\boldsymbol{u}}, \quad \boldsymbol{u} = \boldsymbol{V}_x \odot \boldsymbol{V}_x + \boldsymbol{V}_y \odot \boldsymbol{V}_y + \boldsymbol{V}_z \odot \boldsymbol{V}_z$$

$$\boldsymbol{V}_x = \boldsymbol{G}_x \boldsymbol{T}, \quad \boldsymbol{V}_y = \boldsymbol{G}_y \boldsymbol{T}, \quad \boldsymbol{V}_z = \boldsymbol{G}_z \boldsymbol{T}$$

$$\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{T}} = \left(\frac{1}{2\sqrt{\boldsymbol{u}}} \right) \odot 2(\boldsymbol{G}_x \boldsymbol{T} \odot \boldsymbol{G}_x + \boldsymbol{G}_y \boldsymbol{T} \odot \boldsymbol{G}_y + \boldsymbol{G}_z \boldsymbol{T} \odot \boldsymbol{G}_z)$$

2) Derivative of m_j with respect to $\boldsymbol{T}$
Remembering the definition of m_j where $\Delta\boldsymbol{\rho}^{\{j\}}$ is computed through Equation 23:

$$m_j = \frac{\boldsymbol{M} \cdot \Delta\boldsymbol{\rho}^{\{j\}}}{\mathbf{1} \cdot \Delta\boldsymbol{\rho}^{\{j\}}},$$

the derivative $\frac{\partial m_j}{\partial \boldsymbol{T}}$ can be decomposed in the computation of three terms, $\boldsymbol{A}$, $\boldsymbol{B}$ and C.

$$\frac{\partial m_i}{\partial \boldsymbol{T}} = \frac{\boldsymbol{A} - \boldsymbol{B}}{C}$$

Where:

$$\boldsymbol{A} = \left(\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{T}}^{\top} \Delta\boldsymbol{\rho}^{\{j\}} + \boldsymbol{M}^{\top} \frac{\partial \Delta\boldsymbol{\rho}^{\{j\}}}{\partial \boldsymbol{T}} \right) (\mathbf{1}^{\top} \Delta\boldsymbol{\rho}^{\{j\}})$$

$$\boldsymbol{B} = \left(\boldsymbol{M}^{\top} \Delta\boldsymbol{\rho}^{\{j\}} \right) \left(\frac{\partial \Delta\boldsymbol{\rho}^{\{j\}}}{\partial \boldsymbol{T}}^{\top} \mathbf{1} \right)$$

$$C = \left(\mathbf{1}^{\top} \Delta\boldsymbol{\rho}^{\{j\}} \right)^2$$

3) Derivative of $\Delta\boldsymbol{\rho}^{\{j\}}$ with respect to $\boldsymbol{T}$
Recalling the methodology section, $\Delta\boldsymbol{\rho}^{\{j\}}$ is a vector of the same dimension as $\boldsymbol{t}$, thus with one element for every element in the discretization, defined as:

$$\Delta\boldsymbol{\rho}^{\{j\}} = \boldsymbol{\rho} \odot \left(\bar{\boldsymbol{t}}^{\{j\}} - \bar{\boldsymbol{t}}^{\{j-1\}} \right)$$

thus:

$$\frac{\partial \Delta\boldsymbol{\rho}^{\{j\}}}{\partial \boldsymbol{T}} = \boldsymbol{\rho} \odot \left(\frac{\partial \bar{\boldsymbol{t}}^{\{j\}}}{\partial \boldsymbol{t}} \cdot \frac{\partial \boldsymbol{t}}{\partial \boldsymbol{T}} - \frac{\partial \bar{\boldsymbol{t}}^{\{j-1\}}}{\partial \boldsymbol{t}} \cdot \frac{\partial \boldsymbol{t}}{\partial \boldsymbol{T}} \right)$$

Given that $\boldsymbol{t} = 1 - \boldsymbol{G} \cdot \boldsymbol{T}$, the partial derivative of $\boldsymbol{t}$ with respect to $\boldsymbol{T}$ is:

$$\frac{\partial \boldsymbol{t}}{\partial \boldsymbol{T}} = -\boldsymbol{G}$$

Replacing this into the previous equation:

$$\frac{\partial \Delta\boldsymbol{\rho}^{\{j\}}}{\partial \boldsymbol{T}} = \boldsymbol{\rho} \odot \left[\left(\frac{\partial \bar{\boldsymbol{t}}^{\{j\}}}{\partial \boldsymbol{t}} - \frac{\partial \bar{\boldsymbol{t}}^{\{j-1\}}}{\partial \boldsymbol{t}} \right) \cdot (-\boldsymbol{G}) \right]$$

A-C Sensitivity of the Layer Thickness Constraint

Recalling the layer thickness constraint expression:

$$\begin{aligned} g_{j1}(\boldsymbol{\kappa}) : &\quad -d_j + d_{min} \leq 0, \\ g_{j2}(\boldsymbol{\kappa}) : &\quad d_j - d_{max} \leq 0, \quad \forall j \in \{1, \ldots, N\}, \end{aligned}$$

with

$$d_j = \frac{T_j - T_{j-1}}{m_j},$$

Similarly to the *Distortion Objective* and the *Uniformity Objective* sensitivities, in order to determine the sensitivity of the layer thickness control, the adjoint method must be employed. Following the steps detailed in the previous sections, the sensitivity expression for the constraints can be simplified to:

$$\frac{\partial g_j}{\partial \boldsymbol{T}} + \boldsymbol{\lambda}^\top \boldsymbol{K}_\mathsf{T} = 0 \quad \rightarrow \quad \frac{\partial g_j}{\partial \kappa_e} = \boldsymbol{\lambda}^\top \frac{\partial \boldsymbol{K}_\mathsf{T}}{\partial \kappa_e} \boldsymbol{T},$$

Thus, to solve the adjoint problem, the derivative of d_j with respect to $\boldsymbol{T}$ needs to be computed and is given by:

$$\frac{\partial d_j}{\partial \boldsymbol{T}} = -\frac{T_j - T_{j-1}}{m_j^2} \frac{\partial m_j}{\partial \boldsymbol{T}}.$$

Consequently, the derivatives of the constraints $g_{j1}(\boldsymbol{\kappa})$ and $g_{j2}(\boldsymbol{\kappa})$ with respect to $\boldsymbol{T}$ are:

$$\frac{\partial g_{j1}(\boldsymbol{\kappa})}{\partial \boldsymbol{T}} = -\frac{\partial d_j}{\partial \boldsymbol{T}} = \frac{T_j - T_{j-1}}{m_j^2} \frac{\partial m_j}{\partial \boldsymbol{T}},$$

$$\frac{\partial g_{j2}(\boldsymbol{\kappa})}{\partial \boldsymbol{T}} = \frac{\partial d_j}{\partial \boldsymbol{T}} = -\frac{T_j - T_{j-1}}{m_j^2} \frac{\partial m_j}{\partial \boldsymbol{T}},$$

where the calculation related to the term $\frac{\partial m_j}{\partial \boldsymbol{T}}$ has been detailed in the previous section.

A References

[1] W. Wang, F. van Keulen, and J. Wu, "Fabrication sequence optimization for minimizing distortion in multi-axis additive manufacturing," *Computer Methods in Applied Mechanics and Engineering*, vol. 406, p. 115899, Mar. 2023. [Online]. Available: http://dx.doi.org/10.1016/j.cma.2023.115899

[2] S. Vaidya, P. Ambad, and S. Bhosle, "Industry 4.0 – a glimpse," *Procedia Manufacturing*, vol. 20, p. 233–238, 2018. [Online]. Available: http://dx.doi.org/10.1016/j.promfg.2018.02.034

[3] K. Kanishka and B. Acherjee, "Revolutionizing manufacturing: A comprehensive overview of additive manufacturing processes, materials, developments, and challenges," *Journal of Manufacturing Processes*, vol. 107, p. 574–619, Dec. 2023. [Online]. Available: http://dx.doi.org/10.1016/j.jmapro.2023.10.024

[4] "ISO/ASTM 52900:2021, Additive manufacturing — General principles — Fundamentals and vocabulary," International Organization for Standardization, Geneva, CH, Tech. rep., 2021.

[5] S. Xu, J. Liu, and Y. Ma, "Residual stress constrained self-support topology optimization for metal additive manufacturing," *Computer Methods in Applied Mechanics and Engineering*, vol. 389, p. 114380, Feb. 2022. [Online]. Available: http://dx.doi.org/10.1016/j.cma.2021.114380

[6] G. Fang, T. Zhang, S. Zhong, X. Chen, Z. Zhong, and C. C. L. Wang, "Reinforced fdm: multi-axis filament alignment with controlled anisotropic strength," *ACM Transactions on Graphics*, vol. 39, no. 6, p. 1–15, Nov. 2020. [Online]. Available: http://dx.doi.org/10.1145/3414685.3417834

[7] T. A. Rodrigues, V. Duarte, R. M. Miranda, T. G. Santos, and J. P. Oliveira, "Current status and perspectives on wire and arc additive manufacturing (waam)," *Materials*, vol. 12, no. 7, p. 1121, Apr. 2019. [Online]. Available: http://dx.doi.org/10.3390/ma12071121

[8] D. Jafari, T. H. Vaneker, and I. Gibson, "Wire and arc additive manufacturing: Opportunities and challenges to control the quality and accuracy of manufactured parts," *Materials amp; Design*, vol. 202, p. 109471, Apr. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.matdes.2021.109471

[9] D. Chakraborty, B. Aneesh Reddy, and A. Roy Choudhury, "Extruder path generation for curved layer fused deposition modeling," *Computer-Aided Design*, vol. 40, no. 2, p. 235–243, Feb. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.cad.2007.10.014

[10] T. Zhang, G. Fang, Y. Huang, N. Dutta, S. Lefebvre, Z. M. Kilic, and C. C. L. Wang, "S 3 -slicer: A general slicing framework for multi-axis 3d printing," *ACM Transactions on Graphics*, vol. 41, no. 6, p. 1–15, Nov. 2022. [Online]. Available: http://dx.doi.org/10.1145/3550454.3555516

[11] D. Xie, F. Lv, Y. Yang, L. Shen, Z. Tian, C. Shuai, B. Chen, and J. Zhao, "A review on distortion and residual stress in additive manufacturing," *Chinese Journal of Mechanical Engineering: Additive Manufacturing Frontiers*, vol. 1, no. 3, p. 100039, Sep. 2022. [Online]. Available: http://dx.doi.org/10.1016/j.cjmeam.2022.100039

[12] G. Misiun, E. van de Ven, M. Langelaar, H. Geijselaers, F. van Keulen, T. van den Boogaard, and C. Ayas, "Topology optimization for additive manufacturing with distortion constraints," *Computer Methods in Applied Mechanics and Engineering*, vol. 386, p. 114095, Dec. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.cma.2021.114095

[13] W. Wang, D. Munro, C. C. L. Wang, F. van Keulen, and J. Wu, "Space-time topology optimization for additive manufacturing: Concurrent optimization of structural layout and fabrication sequence," *Structural and Multidisciplinary Optimization*, vol. 61, no. 1, p. 1–18, Nov. 2019. [Online]. Available: http://dx.doi.org/10.1007/s00158-019-02420-6

[14] M. Lansu, "Layer thickness control: Improving manufacturability in fabrication sequence optimization for multi-axis additive manufacturing," M.Sc Thesis, Delft University of Technology, 9 2023, report No. 2023.072.

[15] W. Wang, K. Wu, F. van Keulen, and J. Wu, "Regularization in space–time topology optimization for additive manufacturing," *Computer Methods in Applied Mechanics and Engineering*, vol. 431, p. 117202, Nov. 2024. [Online]. Available: http://dx.doi.org/10.1016/j.cma.2024.117202

[16] P. Durai Murugan, S. Vijayananth, M. Natarajan, D. Jayabalakrishnan, K. Arul, V. Jayaseelan, and J. Elanchezhian, "A current state of metal additive manufacturing methods: A review," *Materials Today: Proceedings*, vol. 59, p. 1277–1283, 2022. [Online]. Available: http://dx.doi.org/10.1016/j.matpr.2021.11.503

[17] C. Dai, C. C. L. Wang, C. Wu, S. Lefebvre, G. Fang, and Y.-J. Liu, "Support-free volume printing by multi-axis motion," *ACM Transactions on Graphics*, vol. 37, no. 4, p. 1–14, Jul. 2018. [Online]. Available: http://dx.doi.org/10.1145/3197517.3201342

[18] MX3D, "Robotic 3d metal printing solutions," 2024, accessed: 2024-10-31. [Online]. Available: https://mx3d.com

[19] F. Xu, V. Dhokia, P. Colegrove, A. McAndrew, S. Williams, A. Henstridge, and S. T. Newman, "Realisation of a multi-sensor framework for process monitoring of the wire arc additive manufacturing in producing ti-6al-4v parts," *International Journal of Computer Integrated Manufacturing*, vol. 31, no. 8, p. 785–798, Apr. 2018. [Online]. Available: http://dx.doi.org/10.1080/0951192X.2018.1466395

[20] Y. H. P. Manurung, K. P. Prajadhiana, M. S. Adenan, B. Awiszus, M. Graf, and A. Haelsig, "Analysis of material property models on waam distortion using nonlinear numerical computation and experimental verification with p-gmaw," *Archives of Civil and Mechanical Engineering*, vol. 21, no. 1, Feb. 2021. [Online]. Available: http://dx.doi.org/10.1007/s43452-021-00189-4

[21] SmarterEveryDay, "The genius of 3d printed rockets [video: 5:40-6:20]," YouTube, 2021, https://www.youtube.com/watch?v=kz165f1g8-E&t=340s.

[22] D. Munro, C. Ayas, M. Langelaar, and F. van Keulen, "On process-step parallel computability and linear superposition of mechanical responses in additive manufacturing process simulation," *Additive Manufacturing*, vol. 28, p. 738–749, Aug. 2019. [Online]. Available: http://dx.doi.org/10.1016/j.addma.2019.06.023

[23] V. Nain, "High-fidelity thermomechanical modeling of additive manufacturing processes: Methods and applications," Ph.D. dissertation, Université Paris-Saclay, 2022. [Online]. Available: https://theses.hal.science/tel-03962675v1/file/2022theseNainV.pdf

[24] E. R. Denlinger and P. Michaleris, "Effect of stress relaxation on distortion in additive manufacturing process modeling," *Additive Manufacturing*, vol. 12, p. 51–59, Oct. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.addma.2016.06.011

[25] Y. Ueda, K. Fukuda, K. Nakacho, and S. Endo, "A new measuring method of residual stresses with the aid of finite element method and reliability of estimated values," *Journal of the Society of Naval Architects of Japan*, vol. 1975, no. 138, p. 499–507, 1975. [Online]. Available: http://dx.doi.org/10.2534/jjasnaoe1968.1975.138_499

[26] Y. Shan, D. Gan, and H. Mao, "Curved layer slicing based on isothermal surface," *Procedia Manufacturing*, vol. 53, p. 484–491, 2021. [Online]. Available: http://dx.doi.org/10.1016/j.promfg.2021.06.081

[27] K. Svanberg, "The method of moving asymptotes—a new method for structural optimization," *International Journal for Numerical Methods in Engineering*, vol. 24, no. 2, p. 359–373, Feb. 1987. [Online]. Available: http://dx.doi.org/10.1002/nme.1620240207

[28] N. Aage, E. Andreassen, and B. S. Lazarov, "Topology optimization using petsc: An easy-to-use, fully parallel, open source topology optimization framework," *Structural and Multidisciplinary Optimization*, vol. 51, no. 3, p. 565–572, Aug. 2014. [Online]. Available: http://dx.doi.org/10.1007/s00158-014-1157-0

[29] B. Leasure, D. J. Kuck, S. Gorlatch, M. Cole, G. R. Watson, A. Darte, D. Padua, U. Banerjee, O. Schenk, K. Gärtner, D. Padua, H. Jay Siegel, B. Dalton Young, R. H. Campbell, U. undefinedatalyürek, C. Aykanat, J. Ajanovic, S. Schmid, R. Wattenhofer, E. N. M. Elnozahy, E. W. Speight, J. Li, R. Rajamony, L. Zhang, B. Arimilli, D. Padua, M. Gerndt, M. Gerndt, D. Padua, J. B. Dennis, B. Smith, G. Almasi, A. Stamatakis, D. Sangiorgi, D. Sangiorgi, D. Padua, J. A. Gunnels, J. Dongarra, P. Luszczek, B. Mohr, R. Eigenmann, P. Feautrier, C. Lengauer, D. Padua, P. Bose, J. F. JaJa, A. Gupta, R. De Nicola, D. Padua, D. Padua, D. Padua, R. S. Armen, E. R. May, M. Taufer, and A. Geist, *Parallel Computing*. Springer US, 2011, p. 1409–1416. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-09766-4_279

[30] R. Trobec, B. Slivnik, P. Bulić, and B. Robič, *Why Do We Need Parallel Programming*. Springer International Publishing, 2018, p. 3–7. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-98833-7_1

[31] T. Borrvall and J. Petersson, "Large-scale topology optimization in 3d using parallel computing," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, no. 46–47, p. 6201–6229, Sep. 2001. [Online]. Available: http://dx.doi.org/10.1016/s0045-7825(01)00216-x

[32] S. Mukherjee, D. Lu, B. Raghavan, P. Breitkopf, S. Dutta, M. Xiao, and W. Zhang, "Accelerating large-scale topology optimization: State-of-the-art and challenges," *Archives of Computational Methods in Engineering*, vol. 28, no. 7, p. 4549–4571, Jan. 2021. [Online]. Available: http://dx.doi.org/10.1007/s11831-021-09544-3

[33] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang, "PETSc Web page," 2018, http://www.mcs.anl.gov/petsc.

[34] N. Aage and B. S. Lazarov, "Parallel framework for topology optimization using the method of moving asymptotes," *Structural and Multidisciplinary Optimization*, vol. 47, no. 4, p. 493–505, Jan. 2013. [Online]. Available: http://dx.doi.org/10.1007/s00158-012-0869-2

[35] J. Alexandersen, O. Sigmund, and N. Aage, "Large scale three-dimensional topology optimisation of heat sinks cooled by natural convection," *International Journal of Heat and Mass Transfer*, vol. 100, p. 876–891, Sep. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.ijheatmasstransfer.2016.05.013

[36] C. D. Team, "Using the *step option in calculix 2.7," n.d., accessed: 2024-11-17. [Online]. Available: https://web.mit.edu/calculix_v2.7/CalculiX/ccx_2.7/doc/ccx/node27.html

[37] D. S. S. Corp., "Abaqus 2016 online documentation," 2016, accessed: 2024-11-17. [Online]. Available: http://130.149.89.49:2080/v2016/books/stm/default.htm

[38] A. Grama, A. Gupta, G. Karypis, and V. Kumar, *Introduction to Parallel Computing*, second edition ed. Addison-Wesley, 2003.

[39] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference on - AFIPS '67 (Spring)*, ser. AFIPS '67 (Spring). ACM Press, 1967. [Online]. Available: http://dx.doi.org/10.1145/1465482.1465560

[40] A. H. Karp and H. P. Flatt, "Measuring parallel processor performance," *Communications of the ACM*, vol. 33, no. 5, p. 539–543, May 1990. [Online]. Available: http://dx.doi.org/10.1145/78607.78614