Check for updates

ASME Journal of Mechanical Design
Online journal at:
https://asmedigitalcollection.asme.org/mechanicaldesign



Concurrent Build Direction, Part Segmentation, and Topology Optimization for Additive Manufacturing Using Neural Networks

Hongrui Chen
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: hongruic@andrew.cmu.edu

Aditya Joglekar
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: aajoglek@andrew.cmu.edu

Kate S. Whitefoot
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: kwhitefoot@cmu.edu

Levent Burak Kara[1]
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: lkara@cmu.edu

Without an explicit formulation to minimize support structures, topology optimization may create complex shapes that require an intensive use of support material when additively manufactured. We propose a neural network-based approach to topology optimization that aims to reduce the use of support structures in additive manufacturing. Our approach uses a network architecture that allows the simultaneous determination of an optimized: (1) part segmentation, (2) the topology of each part, and (3) the build direction of each part that collectively minimize the amount of support structure. Through training, the network learns a material density and segment classification in the continuous 3D space. Given a problem domain with prescribed load and displacement boundary conditions, the neural network takes as input 3D coordinates of the voxelized domain as training samples and outputs a continuous density field. Since the neural network for topology optimization learns the density distribution field, analytical solutions to the density gradient can be obtained from the input–output relationship of the neural network. We demonstrate our approach on several compliance minimization problems with volume fraction constraints, where support volume minimization is added as an additional criterion to the objective function. We show that simultaneous optimization of part segmentation along with the topology and print angle optimization further reduces the support structure, compared to a combined print angle and topology optimization without segmentation. [DOI: 10.1115/1.4062663]

Keywords: artificial intelligence, computer-aided design, design for manufacturing, generative design

1 Introduction

Additive manufacturing has become a popular method of manufacturing topology optimized components. Depositing materials layer by layer, additive manufacturing can manufacture topology optimization designs that are challenging or impossible for traditional subtractive machining. However, additive manufacturing requires support structures for overhang regions, which are common in the resulting shapes of topology optimization designs. Support structures require additional materials that do not contribute to the final product while taking up extra time during fabrication. Moreover, extra effort is required to remove support material afterward. It is possible to reduce support material without modifying the topology by reorienting the print direction or segmenting the part into smaller components [1]. However, this approach is limited to the extent that it can eliminate support material. Simultaneously optimizing build orientation and segmentation with the part's topology could result in further elimination of support structures.

Previously, optimization for reducing support structures has been implemented in both solid isotropic material with penalization (SIMP), level-set method [2,3], and part consolidation [1]. Wang and Qian further developed the SIMP-based method by adding simultaneous build orientation optimization [4]. While overhang surfaces occur at the boundary of geometry, Heaviside projection is required to push SIMP results closer to 0 and 1 for a clear boundary. Chandrasekhar and Suresh's work on neural network-based topology optimization demonstrated the possibility of direct topology optimization using neural network [5]. A neural network is a universal function approximator. The SIMP density field can be directly represented by neural network activation functions with the additional benefit of a crisp and differentiable boundary. Nie et al. demonstrated that print orientation optimization for each subassembly obtained after part decomposition can provide a greater reduction in support structure compared to print direction

[1]Corresponding author.

Contributed by the Design Automation Committee of ASME for publication in the Journal of Mechanical Design. Manuscript received September 19, 2022; final manuscript received May 22, 2023; published online June 23, 2023. Assoc. Editor: Shikui Chen.

 Copyright © 2023 by ASME

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

optimization for the large part [1]. Support structures may be further reduced by adding segmentation to the simultaneous print direction and overhang optimization. We develop upon the neural network approach used by Chandrasekhar and Suresh and extend Wang and Qian's method of overhang detection, to achieve concurrent build direction, part segmentation, and topology optimization to reduce support structures.

Our approach combines print direction, segmentation, and topology optimization through a modular framework using neural networks. Within this modular framework, topology and segmentation is handled through radial basis function neural networks (RBNN) whereas print direction is determined through a single layer perceptron which serves as an optimizer that fits within the modular neural network framework. This modular neural network architecture allows combined optimization by selectively turning on and off different modules. The RBNN learns a density field for which it receives 3D coordinates as input and outputs density values. From this input–output relationship, we can obtain an analytical solution to the density gradient which directly correlates to the surface gradient used for overhang minimization. Finally, through combined build direction, segmentation, and topology optimization, we achieve superior results compared to previous methods.

The implementation of this work can be found at online.[2]

Our main contributions are as follows:

- An analytical solution to the density gradient obtained from the input–output relationship of the neural network.
- Combined build direction, part segmentation, and topology optimization.
- A modular neural network architecture that allows combined optimization and selectively turning on and off different modules.

2 Related Work

Our review focuses on studies that highlight additive manufacturing oriented topology optimization and integration of machine learning and topology optimization.

Support structure minimization. A large body of work has investigated overhang edge detection and penalization for support structure minimization. Thompson et al. [6] explored support structures and their relationship to additive manufacturing costs and opportunities. Brackett et al. [7] first explored an approach to penalize overhang edges through edge detection in 2D. Several studies inferred surface gradients through approximation based on near neighbors. Leary et al. [8] created self-supporting structures with boundaries obtained through four or eight element connectivity. Overhang edges can also be obtained by scanning elements below. Gaynor and Guest [9] demonstrated the detection of three sets of overhang angles. Mhapsekar et al. [10] also used neighbor detection with additional height penalty. Overhang regions can also be excluded from layer-wise filtering [11]. Ven et al. [12] used front propagation to detect overhang surfaces. Zhang et al. [13] created self-supporting structures through polygon modifications. Support structure placement can be minimized while reducing the residual stress [14]. Most aforementioned methods lack accurate density gradient calculation due to the discretized grid of topology optimization. More emphasis on improving the accuracy of the density gradient calculation had been made through later research. Using the level set method, overhang control can be implemented by calculating the topological derivative [3]. Liu and To [15] demonstrated using level set for toolpath simulation to consider both raster direction and build direction. Zhang et al. [16] introduced an extra constraint to remove boundary oscillations in the density gradient formulation. Qian et al. [2] used linear interpolation between nodes of finite element mesh to obtain a more accurate density gradient. With density gradient, self-supporting structures, boundary slope control, and print angle optimization have been added for simultaneous optimization with topology [4,17,18]. We base our overhang detection method on Wang and Qian's work which integrates print angle from the vector dot product of the print angle with filtered density gradient with the distinction of (1) accurate, differentiable density gradient from the neural network which topology optimization is directly executed on and no filtering required and (2) additional height penalization which minimizes tall and slim support structures that increase the possibility of print failure.

Print direction and manufacturability optimization. Print direction is the direction in which a part is oriented during additive manufacturing. Chandrasekhar et al. [19] demonstrated the importance of print direction on structure performance by optimizing the print direction of fiber-reinforced additive manufactured components. Ulu et al. [20] trained a surrogate model from running FEA in different directions for which an optimal build orientation can be determined from gradient descent on surrogate model. Ulu et al. [21] also demonstrated the advantages of concurrent optimization of the total production cost. By small adjustment to the shape and print direction optimization, models can also be better suited for additive manufacturing [22]. Anisotropy of additive manufactured parts have also been integrated with segmentation to determine the segmentation and optimal print orientation [23]. Nie et al. [1] conducted optimization for minimum production cost considering part consolidation while demonstrating the tradeoff between no and full consolidation. Part layout, including rotation of each individual component, is optimized. Nie et al. demonstrated that for metal additive manufacturing, segmentation leads to a reduction in shadow volume. However, the topology of each component remains unchanged which motivates us to add part segmentation to our framework to further reduce shadow volume.

Topology optimization with machine learning. Both data-driven and real-time approaches have been explored for the application of topology optimization. Data-driven topology optimization aims to learn a neural network model from a database of topology optimization results which speeds up the process. Surrogate neural network model has been used for microstructure design [24]. Many other methods rely on convolutional neural networks (CNN) for their capabilities to learn from a large set of image data. Banga et al. [25] used a 3D encoder–decoder CNN to generate 3D topology results with a 4% reduction in computation time. Behzadi and Ilieş [26] used deep transfer learning with CNN. Zheng et al. [27] used U-net CNN for 3D topology synthesis. Machine learning can also generate an initial guess for topology optimization to speed up the convergence [28]. More accurate synthesis can be achieved from generative adversarial networks based on physical fields over the initial domain [29]. U-Net was also used for improving the manufacturablity of designs for metal additive manufacturing [30]. CNN demonstrated the capabilities of rapid topology synthesis but CNN alone cannot guarantee the mechanical performance of the result. A real-time data-driven hybrid approach trained the neural network during optimization to learned optimization sensitivities to further accelerate topology optimization [31]. Chandrasekhar and Suresh [5] first explored a real-time approach where the neural network directly optimizes the density field of SIMP. It guaranteed the mechanical performance of topology optimization as the density field is parameterized by the weights of the neural network. Chandrasekhar and Suresh [32] also explored Fourier projection-based neural network for length scale control. Application of multimaterial topology optimization is also explored with a similar concept [33]. Multimaterial topology optimization is achieved with a multilayer perceptron with a softmax layer attached at the end which serves to assign the material to each segment. Chandrasekhar and Suresh's [33] research demonstrated the benefit of a real-time neural network-based approach where boundaries can be differentiable. In our work, we further exploit this advantage for the application of minimizing overhang while leveraging the segmentation capabilities to partition the topology for print direction optimization of each individual segment.

[2]https://github.com/HongRayChen/seg-angle-topopt

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

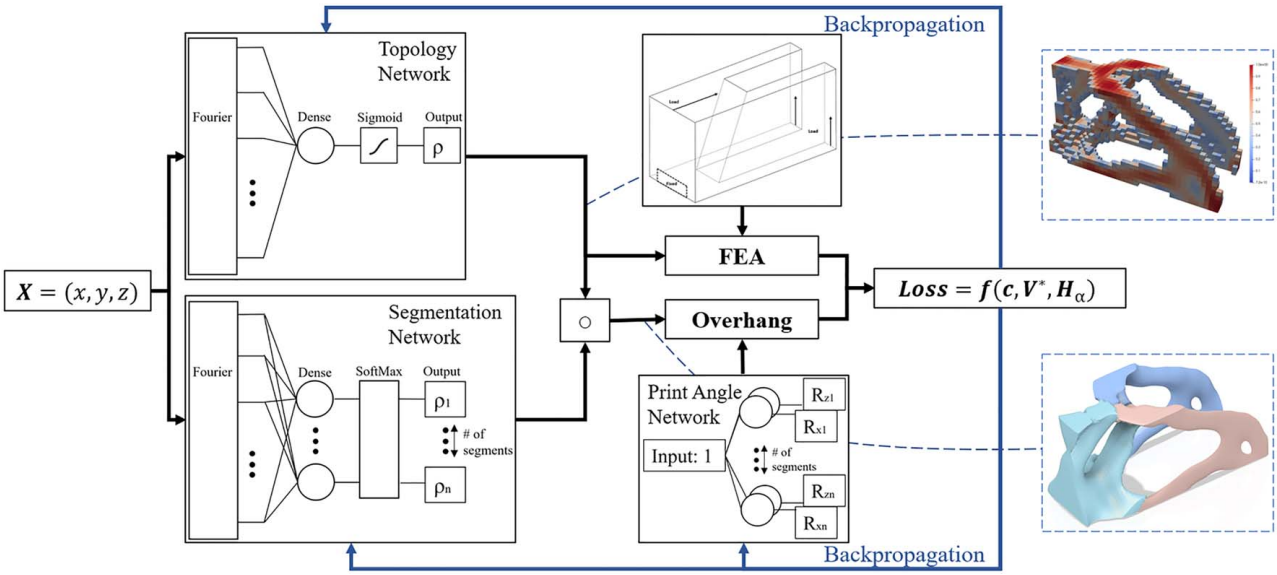


Fig. 1 Our topology, segmentation, and print angle neural network. The topology and segmentation networks use a radial basis layer to transform input coordinates into the Fourier space. The print angle network consists of a single layer of neurons that encode the print angle parameters, which is compatible with the overall neural network-based architecture. On the top right is the voxelized output of the topology network. On the bottom right is an upsampled and smoothed output with a unique color assigned to each segment.

3 Proposed Method

The goal of this work is to reduce the support structure for topology optimization through segmentation and print angle optimization. We decompose the functionality of generating topology, segmentation, and print angle for each segment into three modular neural networks (Fig. 1). These three modular networks enable us to selectively turn on and off different modules depending on the complexity of the problem. Using only topology neural networks, we can run standalone topology optimization similar to SIMP. While print angle optimization can be added for simultaneous print direction and topology optimization similar to Wang and Qian's approach [4]. With segmentation added, concurrent optimization can be achieved. These three modules are linked together by a combined loss function. Backpropagation of the loss function into each module is handled by the machine learning package TensorFlow [34].

3.1 Neural Network. The topology network $T(\mathbf{X})$ learns a density field which is different compared to typical topology optimization which represents the density field as a finite element mesh. The topology neural network takes in 3D coordinates $\mathbf{X} = (x, y, z)$ as inputs and outputs the density value ρ at each coordinate point. The 3D coordinates represent the center of each element in the design domain. For standalone topology optimization, a batch of 3D coordinates that correspond to a 3D mesh grid is fed into the topology network. The output is then sent to the finite element analysis (FEA) solver. The solver outputs the compliance which is combined with volume fraction as a loss. The loss is then backpropagated to adjust the weight of each neuron of the topology network.

For the topology network design, we employed a modified RBNN. The first layer of the neural network consists of a radial basis function layer. This layer applies a nonlinear transformation of the 3D coordinates input. We use TensorFlow's "RandomFourierFeatures" with uniformly sampled kernel to implement this layer [34]. For repeatability, we initialize the kernel location in a linear 3D grid. In contrast to the neural network architecture from Chandrasekhar and Suresh [32], the second layer is a fully connected layer with only one neuron. The final layer consists of a sigmoid activation layer. The sigmoid activation function guarantees the output is between 0 and 1. The main feature of this network design is the removal of a batch normalization layer. Together with the shift-invariant Gaussian kernel, we can choose a flexible batch size as input without worrying about the statistical distribution of the input. The flexible batch size input is useful when enforcing nondesign regions. We can upsample the 3D coordinate input or only sample specific regions of the density field to manipulate the resolution of the discretized visualization. The topology network can be formulated as follows:

$$T(\mathbf{X}) = \sigma(\cos(\mathbf{X} \times \mathbf{K} + \mathbf{b}_{kernel}) \times \mathbf{W} + \mathbf{b}_{dense}) \tag{1}$$

where $\mathbf{X}$ is the 3D coordinate input, $\mathbf{X} = (x, y, z)$. σ is the Sigmoid activation function. $\mathbf{b}_{kernel}$ is the bias of the RBNN layer. $\mathbf{K}$ are kernels for the RBNN layer. $\mathbf{W}$ are weights for the dense layer. $\mathbf{b}_{dense}$ is the bias for the dense layer.

The initial step for conducting overhang analysis is to obtain the density gradient $\nabla\rho$. Since the topology neural network learns a continuous density field with 3D coordinates as input, obtaining the $\nabla\rho$ is straightforward in this case. We simply use the automatic differentiation from TensorFlow to track the coordinate input and the density output to obtain $\delta\rho/\delta\mathbf{X}$ [34]. $\delta\rho/\delta\mathbf{X}$ is the analytical density gradient at each 3D coordinate input. Mathematically, $\nabla\rho$ can be derived from using chain rule by taking the derivative w.r.t. input $\mathbf{X}$, $T'(\mathbf{X}) = \delta\rho/\delta\mathbf{X}$:

$$T'(\mathbf{X}) = (-\sin(\mathbf{X} \times \mathbf{K} + \mathbf{b}_{kernel})(\sigma'(\mathbf{M}) \times \mathbf{W}^T)) \odot \mathbf{K}^T \tag{2}$$

where $\mathbf{M} = \cos(\mathbf{X} \times \mathbf{K} + \mathbf{b}_{kernel}) \times \mathbf{W} + \mathbf{b}_{dense}$, $\sigma'(\mathbf{M}) = \sigma(\mathbf{M})(1 - \sigma(\mathbf{M}))$. $\odot$ represents broadcasting multiplication.

Part segmentation refers to breaking up a larger part into smaller segments. Each segment is a monolithic part and will have a unique print angle. The geometry of each segment will be optimized by jointly taking into account overhang minimization. The segmentation network $S(\mathbf{X})$ is different from the topology network in two ways. First, the segmentation network second layer has the same number of neurons as the number of segments that is requested. Second, the final layer is a Softmax layer to guarantee partition of unity. The segmentation network can be formulated as

$$S(\mathbf{X}) = s(\cos(\mathbf{X} \times \mathbf{K} + \mathbf{b}_{kernel}) \times \mathbf{W} + \mathbf{b}_{dense}) \tag{3}$$

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

where $s(x)$ is the Softmax operator $s(z) = e^{z_i}/\sum_{l=1}^{n} e^{z_l}$. Using chain rule, the derivative of $S(\mathbf{X})$ with respect to input $\mathbf{X}$ is

$$S'(\mathbf{X}) = (-\sin(\mathbf{X} \times \mathbf{K} + \mathbf{b}_{kernel})(s_i'(\mathbf{M}) \times \mathbf{W}^T)) \odot \mathbf{K}^T \tag{4}$$

where $\mathbf{M} = \cos(\mathbf{X} \times \mathbf{K} + \mathbf{b}_{kernel}) \times \mathbf{W} + \mathbf{b}_{dense}$, $s_i'(\mathbf{M}) = s_i(\mathbf{M})(\mathbf{1}\{i=j\} - s_j(\mathbf{M}))$; $i, j = 1, 2, \ldots n_{segs}$.

Segmentation is applied by multiplying the topology network output with the segmentation network output. The topology segmentation multiplication is similar to a Hadamard product where we duplicate the output from the topology network number of segments times and multiply with segmentation network output (Fig. 1, "$\circ$" represents the Hadamard product). We also initialize the segmentation with an inverse distance field. This serves as an initial guess for the segmentation while reducing the possibility of disconnected segments

$$\rho = T(\mathbf{X}) \circ S(\mathbf{X}) \tag{5}$$

The print angle for each segment is generated from the print angle network. The print angle network consists of only one layer of neurons which takes in 1 as a constant input. The neurons are directly connected to the output which means the weight of each neuron directly corresponds to the output. The purpose of designing it as a neural network is to allow the print angle to be integrated with the combined framework and maintain consistency during back propagation coding. We would also like to note that free variables can also be used for print angle representation. In addition, by directly manipulating the weights, we can configure the initial condition of print angle for each segment.

3.2 Topology Optimization. During optimization, the topology network outputs the density value at the center for each element. These density values are then sent to the finite element solver to calculate compliance based on the SIMP interpolation

$$E_i(\rho_i) = E_{\min} + \rho_i^p (E_0 - E_{\min}) \tag{6}$$

$$\mathbf{K}\mathbf{u} = \mathbf{F} \tag{7}$$

$$c = \mathbf{F}^T \mathbf{u} \tag{8}$$

We only run 3D examples in this paper with eight node hexahedral finite elements. We consider the material to be isotropic. For material anisotropy, a method similar to that described in [35] can be utilized. The solver is based on the MATLAB code from Liu and Tovar but we modified it to run on PYTHON with CUDA accelerated sparse iterative matrix solver [36]. The finite element solver is treated as a black box within the neural network. It takes in the density of each element and outputs the compliance and the sensitivity for each element with respect to the compliance. We also explore writing the finite element solver using only TensorFlow's tensor representation. The automatic differentiation can correctly calculate the sensitivity but we choose to run the solver outside for computation speed purposes.

Passive elements are nondesigned regions in some problems. It has a density value of 0 to enforce it to be void. Usually, the 0 density values are directly imposed onto the finite element mesh [36]. We employ a similar concept to the application of the continuous field by adding an additional passive loss to the total loss. The coordinates within the passive regions are first defined as $\mathbf{X}_{passive}$. We then use Eq. (9) to compute the L1 loss $\mathcal{L}_{passive}$ to force the density field within the passive region to be 0. Intuitively, $\mathbf{X}_{passive}$ should always be smaller than $\mathbf{X}$ which reduces the memory burden on the GPU

$$\mathcal{L}_{passive} = \sum |T(\mathbf{X}_{passive}) - 0| \tag{9}$$

Symmetry can be enforced by mirroring the input coordinates across the symmetric axis. We then take the average density values from both the original $\mathbf{X}$ and mirrored input $\mathbf{X}_{mirrored}$

$$\rho_{sym} = 0.5(T(\mathbf{X}) + T(\mathbf{X}_{mirrored})) \tag{10}$$

3.3 Optimization for Additive Manufacturing. The first step for overhang detection is to calculate the density gradient $\nabla\rho$. We obtain the analytical solution of the density gradient from the input–output relation of the neural network. We then follow the proposed algorithm for overhang detection from Wang and Qian [4]. We then add an additional height penalization to the overhang regions.

We will briefly review the overhang detection proposed by Wang and Qian with a more detailed explanation of the additional height penalization. The overhang angle is defined as α which can be calculated from the boundary normal $\mathbf{n}$ and the build direction $\mathbf{b}$. The overhang angle α is defined as

$$\cos(\alpha) = \mathbf{b} \cdot \mathbf{n} = \mathbf{b} \cdot \frac{\nabla\rho}{\|\nabla\rho\|_2} \tag{11}$$

The print angle network outputs angles in rotation around x- and z-axes R_x and R_z, respectively. An illustration for coordinate system and rotation definition is shown in Fig. 2(*a*). We use the following to convert it into build direction vector $\mathbf{b}$:

$$\mathbf{b} = (\sin R_z,\ \cos R_x \cos R_z,\ \sin R_x \cos R_z) \tag{12}$$

For the printed structure to self-support, the critical overhang angle $\bar{\alpha}$ is defined (Fig. 2(*b*)). The lower bound of the overhang angle is constrained as $\alpha \geq \bar{\alpha}$. For every boundary region that is self-supporting (Fig. 2(*e*)), it needs to satisfy

$$\mathbf{b} \cdot \mathbf{n} \leqslant \cos(\bar{\alpha}) \tag{13}$$

Wang and Qian applied a Heaviside filter to $\mathbf{b} \cdot \mathbf{n} - \cos(\bar{\alpha})$ such that all boundary regions that do not satisfy the overhang constraint will be filtered positive. A smooth Heaviside function with β set to 10 is used

$$H(\xi) = \frac{1}{1 + e^{-2\beta\xi}} \tag{14}$$

The overhang regions (Fig. 2(*f*)) can be integrated across the design domain to obtain the total overhang area

$$P_{\bar{\alpha}} = \frac{\int_\Omega H_{\bar{\alpha}}(\mathbf{b}, \nabla\rho)\mathbf{b} \cdot \nabla\rho \, \mathrm{d}\Omega}{\bar{A}} \tag{15}$$

$$H_{\bar{\alpha}}(\mathbf{b}, \nabla\rho) = H\left(\mathbf{b} \cdot \frac{\nabla\rho}{\|\nabla\rho\|_2} - \cos(\bar{\alpha})\right) \tag{16}$$

$\bar{A}$ is the characteristic area and we set it to be the largest surface in the design domain. For sensitivity calculation, derivation is provided in Wang and Qian's paper [4]. Since we implemented these calculations using TensorFlow's data structure, automatic differentiation will take care of sensitivity calculation.

3.4 Overhang Formulation With Height Penalization. We also add additional height penalization to Wang and Qian's method [4]. Height penalization requires the detection of the lowest solid region within the design domain. In Wang and Qian's [4] approach, overhang regions are split between the exterior boundary support and the interior boundary support to accommodate the sometimes necessary support on the lower side of the part. With our height penalization, necessary support on the underside is not penalized as much which reduces the need to split the overhang region calculation between the interior and exterior. The input coordinates are normalized between −0.5 and 0.5 for the longest edge of the design domain. We first subtract −1.0 on the y-axis coordinates and then multiply with the Heaviside filtered density at each coordinate (Fig. 2(*h*))

$$\tilde{\rho} = \frac{1}{1 + e^{-2\beta(\rho - 0.5)}} \tag{17}$$

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

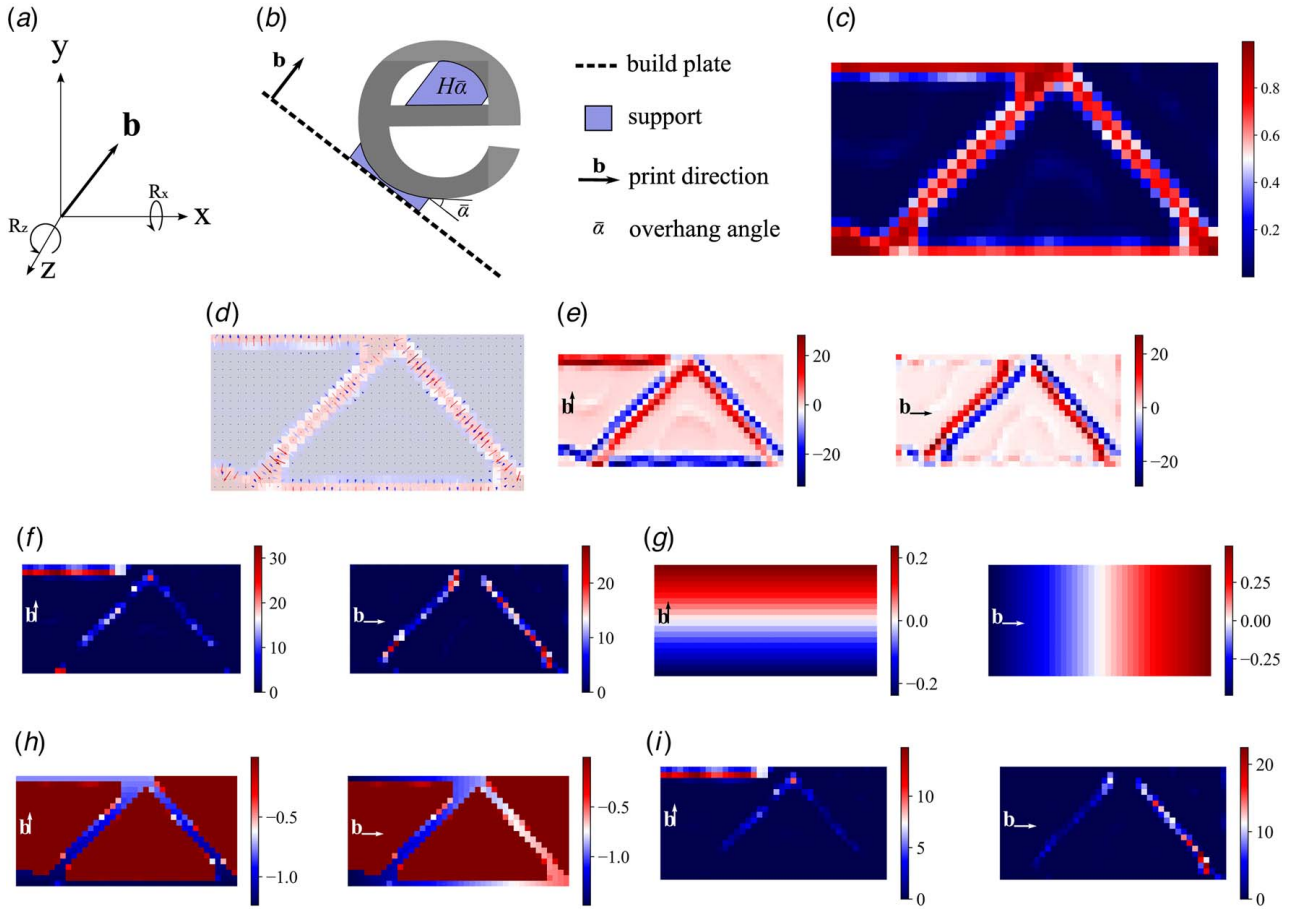


Fig. 2 Overhang regions $P_{\bar{\alpha}}$ can be detected based on Heaviside filtering of the print direction dot product with the density gradient $\boldsymbol{b} \cdot \nabla\rho$. We further augment the overhang regions by multiplying these regions with their distance from the build plate to penalize overhang regions $H_{\bar{\alpha}}$ that are higher above the build plate which require more support material. (*a*) Definition of the coordinate system. (*b*) Overhang and support definition for an "e" shaped structure. (*c*) Density field, ρ. (*d*) Density gradient, $\nabla\rho$, (*e*) $\boldsymbol{b} \cdot \nabla\rho$. (*f*) Overhang region, $P_{\bar{\alpha}}$. (*g*) Rotated *Y*-axis coordinates. (*h*) *Y*-axis coordinates multiplied with filtered density. (*i*) Height penalized overhang region, $H_{\bar{\alpha}}$.

$$\mathbf{X}_{y,lowest} = \min\left((\mathbf{X}_y - 1.0)\tilde{\rho}\right) + 1.0 \tag{18}$$

During this step, any density value close to 0 will result in $(\mathbf{X}_y - 1.0)\tilde{\rho}$ to be zero and filtered out. By taking the minimum of the multiplication, we can obtain the lowest region that has solid elements. We then multiply the $\mathbf{X}_y$ coordinates with the overhang regions to obtain a height penalized overhang constraint (Fig. 2(*i*))

$$H_{\bar{\alpha}} = \frac{\int_\Omega H_{\bar{\alpha}}(\mathbf{b}, \nabla\rho)\mathbf{b} \cdot \nabla\rho(\mathbf{X}_y - \mathbf{X}_{y,lowest})\,\mathrm{d}\Omega}{\bar{A}} \tag{19}$$

3.5 Loss Function. Both topology optimization and optimization for additive manufacturing are integrated into the machine learning framework. Variables that are being optimized are the weights of each neural network. Built-in optimizers such as Adam [34] and SGD are used to train the neural network. The constrained optimization problem needs to be transferred into unconstrained minimization problem for neural network. Chandrasekhar and Suresh [5] formulated the loss function for topology optimization including volume fraction and compliance constraint. We adopt the volume fraction and compliance constraint with the addition of overhang regions. The combined loss function is

$$L = \frac{c}{c_0} + \alpha_1\left(\frac{\bar{\rho}}{V^*} - 1\right)^2 + \alpha_2 H_{\bar{\alpha}} \tag{20}$$

In the optimization, the volume fraction V^* is an equality constraint. When $\alpha_1 \to \infty$, the equality constraint is satisfied. We assign a maximum value of 100 for α_1 with initial value of 0 and gradually increase α_1 every iteration. For the overhang constraint $H_{\bar{\alpha}}$, we do not target complete overhang removal. α_2 gradually increases from 0 to 1 during training. For topology optimization, c_0 is the initial compliance calculated on the design domain with the uniform volume fraction V^*.

4 Results and Discussions

4.1 Validation. We begin with comparing our neural network based optimization with SIMP. For the comparison, we set up SIMP using the same machine learning framework with the same cantilever beam boundary condition. The problem size is $40 \times 20 \times 8$ elements. We only enable the topology network within our modular architecture. The difference is that we configured a one layer neural network with one constant as input and the number of elements within the design domain as output. This configures the neural network to just store the optimization variables similar to the print angle network design. When running the optimization, for the first 100 iterations, we use SGD as the optimizer and afterward we switch to Adam [37]. We run the optimization for 300 iterations.

We observe that compared to SIMP, the topology network converged to a similar compliance. We observe a benefit of using a neural network to learn a continuous density field is that it inherently prevents checkerboard patterns without needing filtering (Fig. 3(*b*)).

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

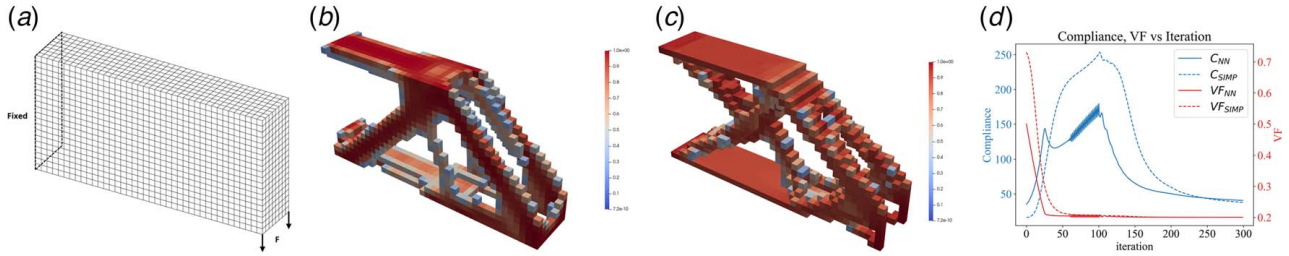


Fig. 3 Comparing the topology and convergence history of neural network (*b*) and SIMP (*c*), the result is similar. (*d*) is the convergence history of neural network and SIMP. The oscillations between 60 and 100 epoch are due to using SGD as the optimizer for the first 100 epoch. (*a*) Boundary condition. (*b*) $c = 40.6$. (*c*) $c = 37.8$.

We also show for a 2D problem (cantilever beam with 0.3 volume fraction) that the KKT conditions are approximately satisfied with the neural network-based topology optimization approach (Fig. 4). The optimization objective in our case consists of compliance and volume fraction constraint violation. There are no other constraints and hence for approximately satisfying the KKT conditions, the gradient of loss with respect to the design variables (the trainable weights of the neural network in our case) needs to be approximately zero. Figures 4(*d*) and 4(*e*) show that the mean and standard deviation of the gradients of loss with respect to these trainable weights in the two layers gradually move towards zero as the iterations increase. We also show this for a fine solution (i.e., we change the scale of the first NN layer) (Fig. 4(*f*)) for the same problem. Hence, it can be said that KKT conditions are approximately satisfied in the neural network-based topology optimization approach.

4.2 Upsampling and Mini-Batch Using the Topology Neural Network. With the continuous density field learned by the topology network, we can increase the resolution of the 3D coordinates $\mathbf{X}_{upsample}$ by sampling at closer intervals within the original coordinates domain and obtain $\rho_{upsample}$ from the topology network

$$\rho_{upsample} = T(\mathbf{X}_{upsample}) \tag{21}$$

We can then obtain a geometry at a higher resolution without the need for interpolation. This upsampled geometry is useful for

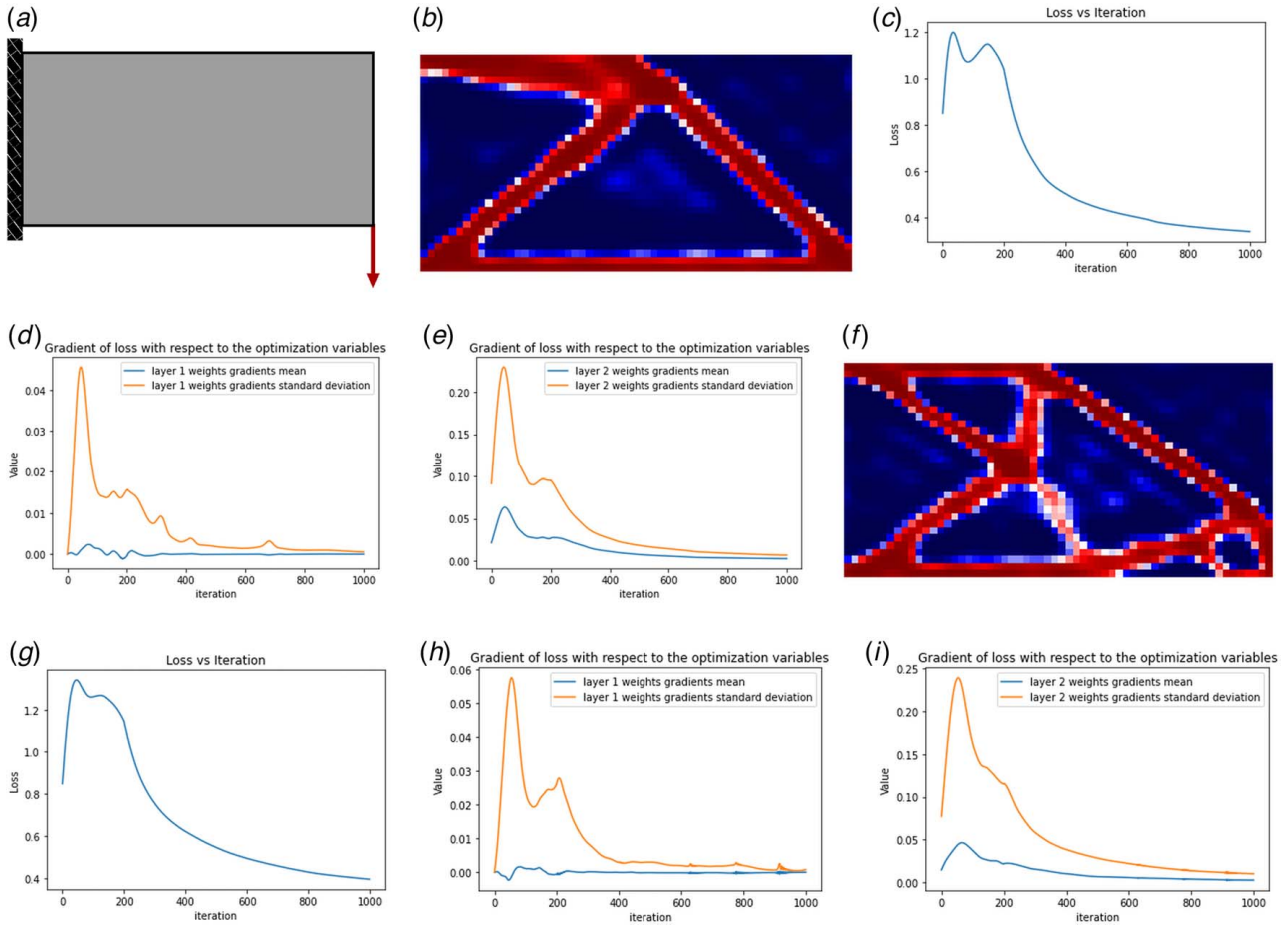


Fig. 4 Verifying KKT conditions are approximately satisfied with the neural network-based topology optimization approach. (*a*) Boundary condition. (*b*) Solution 1 (coarse solution). (*c*) Solution 1 loss convergence history. (*d*) Solution 1 gradients of loss with respect to neural network layer 1 weights. (*e*) Solution 1 gradients of loss with respect to neural network layer 2 weights. (*f*) Solution 2 (fine solution). (*g*) Solution 2 loss convergence history. (*h*) Solution 2 gradients of loss with respect to neural network layer 1 weights. (*i*) Solution 2 gradients of loss with respect to neural network layer 2 weights.

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

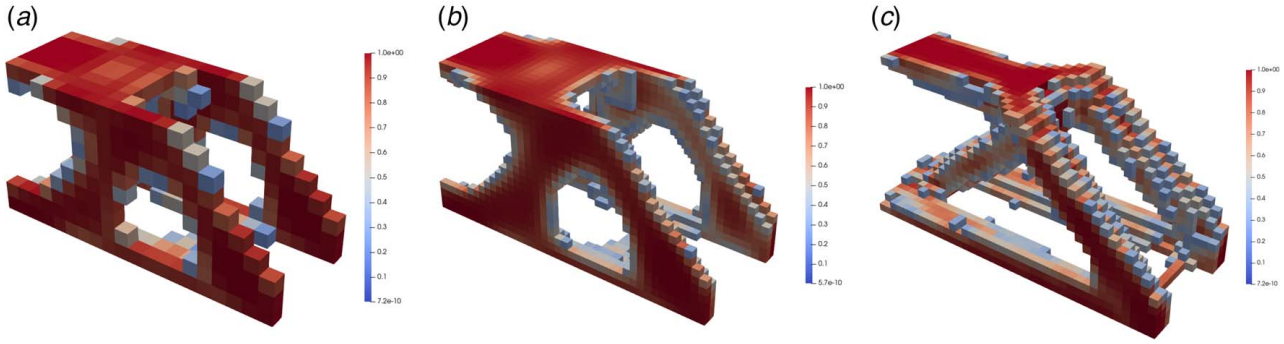


Fig. 5 While running at higher resolution resulted in lower compliance, upsampling requires less computation time, enabling a tradeoff between optimality and speed. (*a*) Beam example at original resolution with neural network. (*b*) Beam example 2× upsampled with neural network, $c = 33.2$. (*c*) Beam example running at 2× the resolution, $c = 20.9$.

exporting to be manufactured and/or running calculations at a higher resolution while still being differentiable.

We compare the compliance and runtime between running at $20 \times 10 \times 8$ with $2\times$ upsampling with running at $40 \times 20 \times 16$ resolution (Fig. 5). When comparing the runtime, running at the lower resolution takes 209 s while the higher resolution example takes 767 s, a 73% reduction in runtime. When upsampling the lower resolution to the same resolution, the compliance is calculated to be 59% higher. Running the optimization from a lower resolution and then upsampling it can be useful when a quick solution or preview of the optimization is desired while still maintaining acceptable structural performance.

There may be cases where topology optimization of high resolution geometries needs to be run. A mini-batch approach can be used here for alleviating the problem of high GPU memory usage. In the mini-batch approach, a small part of the geometry is updated in each epoch instead of the entire geometry, thus reducing GPU memory consumption. An example comparing the mini-batch and full-batch approaches is shown in Fig. 6, with the same boundary conditions as in Fig. 3 example, but with a higher resolution geometry. The mini-batches are formed by splitting up the part evenly in the y-direction. Four parts/batches are made in the given example, with each batch of 5400 voxels. The optimization is run for 300 epochs. The full-batch achieves a compliance of 21.94, while the mini-batch achieves a compliance of 24.77. Though the mini-batch optimization takes more time as FEA calculations for the entire geometry have to be performed for each batch, the maximum GPU memory required is significantly reduced. For the given example run on a Tesla T4 GPU, full-batch optimization took 7 min 8 s and required 5.74 GB GPU memory, while mini-batch optimization took 45 min 28 s and required 2.74 GB GPU memory.

4.3 Overhang Minimization. With the promising result from running standalone topology optimization using only the topology network, we then enable the print direction network for simultaneous build direction and overhang optimization (Fig. 7(*a*)). With height penalization disabled, we can recreate the result from Wang and Qian albeit using a neural network to represent the topology with differentiable density gradients (Fig. 7(*b*)). We run two examples with both overhang region and print direction optimization enabled after 150 iterations to allow the geometry to first roughly converge. The critical overhang angle $\bar{\alpha}$ is set to be 45°. One has height penalization enabled while the other is disabled to compare the effect of height penalization.

When comparing the result of these two examples, the compliance and optimal print direction are all similar. Both managed to significantly reduce the total overhang area. A noticeable distinction is that with the additional height penalty, the cantilever on the top has been removed (Fig. 7(*a*)). Despite being a very small overhang, its location is further away from the build plate which was removed with the addition of height penalization. When comparing the height-adjusted overhang penalization, our method manages a 94% decrease in the total overhang regions.

4.4 Concurrent Build Direction, Part Segmentation, and Topology Optimization. In the previous example, we demonstrated the effect of adding build direction into the optimization routine. We would like to further reduce the support structure for complex designs. We choose a simplified bike frame for this example. The boundary condition is shown in Fig. 8(*a*). Symmetry constraint is enforced on the design. We run three experiments with (1) standalone topology optimization, (2) simultaneous print angle and topology optimization, and (3) three segments with concurrent print angle and topology optimization. For the first 150 iterations, only topology optimization is enabled to obtain a rough geometry. Then for the experiment with segmentation enabled, we run 100 iterations with only print angle optimization enabled. This step is to prevent the topology and segmentation from converging faster

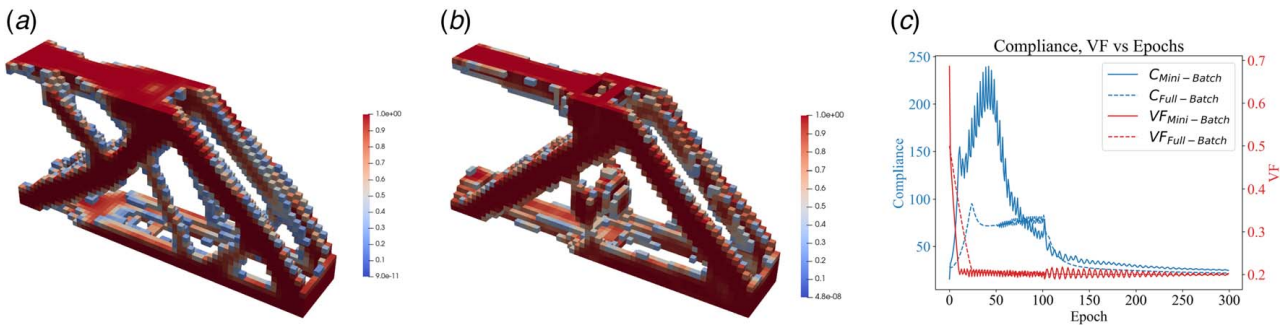


Fig. 6 Comparing the result of (*a*) full-batch optimization, (*b*) mini-batch optimization, the topologies have small differences and compliance is slightly higher in mini-batch optimization. (*c*) is the convergence history comparison. (*a*) $c = 21.9$ and (*b*) $c = 24.8$.

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

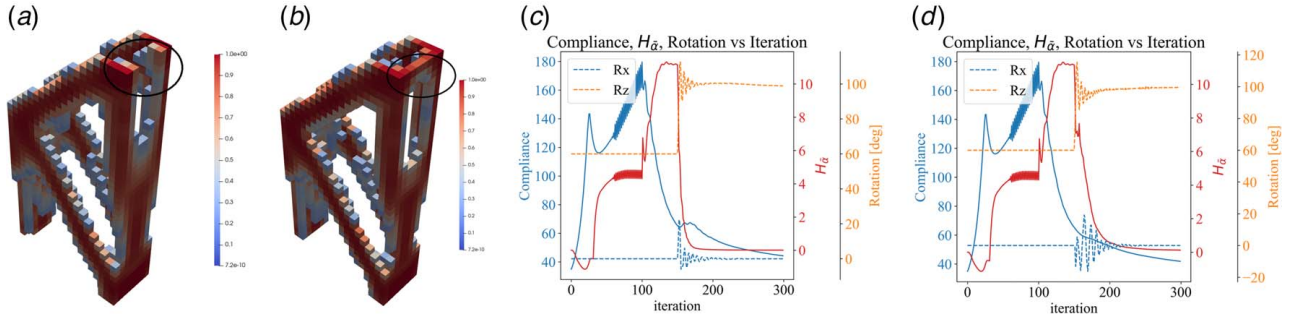


Fig. 7 (*a*) and (*b*) are the voxel plots for with and without height penalization. With only overhang penalization, despite the geometry being optimized, still a noticeable overhang region is present on top (*b*). Adding the height penalization to overhang surfaces further reduced the overhang regions on top (*a*). (*c*) and (*d*) are the convergence history for with and without height penalization. (*a*) $c = 44.3$, $H_{\tilde{\alpha}} = 0.008$. (*b*) $c = 41.9$, $H_{\tilde{\alpha}} = 0.1278$. (*c*) $R_x = 0^\circ$, $R_z = 99^\circ$. (*d*) $R_x = 0^\circ$, $R_z = 100^\circ$.

than the print angle resulting in optimization being stuck at a suboptimal print angle. At 250 iterations, build direction, part segmentation, and topology optimization is enabled.

When comparing the result of the three experiments, we observed that with segmentation (Fig. 8(*d*)), the height penalized overhang is reduced by 30% while compliance is smaller compared to without segmentation (Fig. 8(*c*)). Comparing to standalone topology optimization, our concurrent optimization managed to reduce height penalized overhang by 98.5%. With segmentation, the two segments in the rear can be laid flat on the build plate with fewer

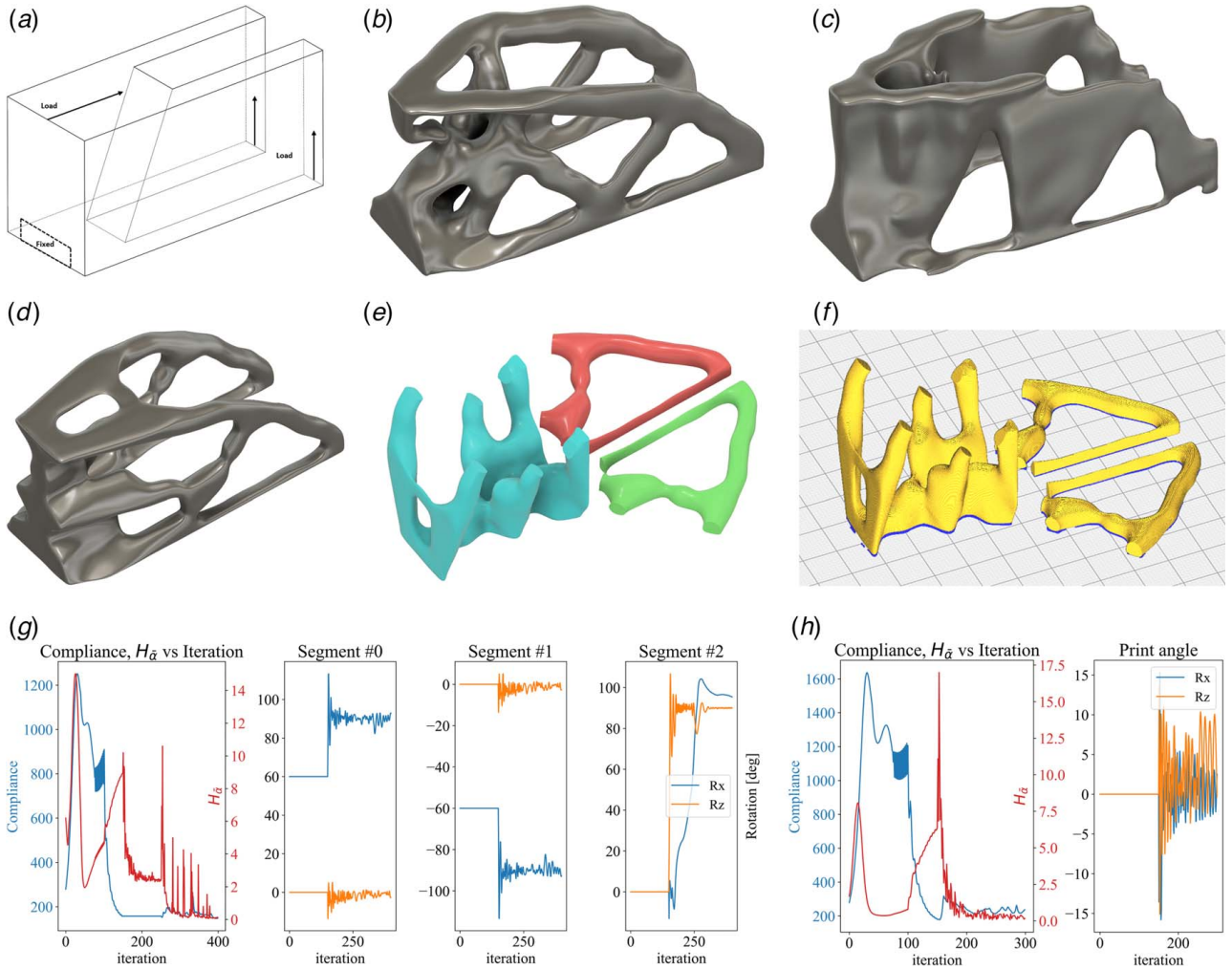


Fig. 8 With segmentation, overhang regions can be further reduced while compliance is also lower since less change to topology is required. The gray geometry is obtained from using Top3dStl [36] to convert voxelized result into stl file. (*a*) Boundary condition for the bike frame. (*b*) Standalone topology optimization result, $c = 116.4$, $H_{\tilde{\alpha}} = 8.229$. (*c*) Optimized topology without segmentation $c = 237$, $H_{\tilde{\alpha}} = 0.168$. (*d*) Optimized topology with segmentation. $c = 148$, $H_{\tilde{\alpha}} = 0.118$. (e) Segmentation visualization for concurrent topology, print angle, and segmentation optimization. (*f*) Toolpath and support structure with segmentation. (*g*) Convergence history with segmentation enabled, segment 1: $R_x = 93^\circ$, $R_z = -3^\circ$, segment 2: $R_x = -93^\circ$, $R_z = -3^\circ$, segment 3: $R_x = -93^\circ$, $R_z = 96^\circ$. (*h*) Convergence history without segmentation, $R_x = 3^\circ$, $R_z = 0^\circ$.

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

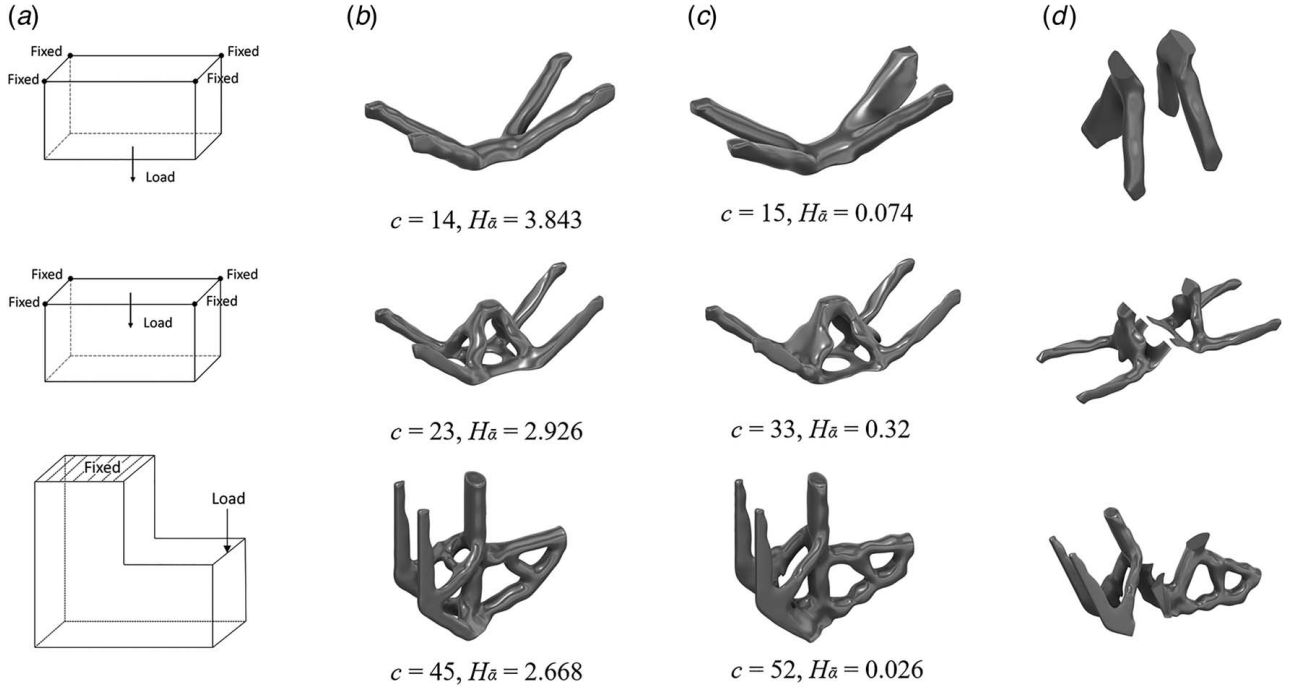


Fig. 9 Additional concurrent optimization examples with segmentation and rotation angle applied. (*a*) Problem boundary condition. (*b*) Topology optimization only. (c) Concurrent topology, segmentation, and print angle optimization. (*d*) Segmentation with rotation applied.

restrictions on its topology to further reduce overhang (Fig. 8(*d*)). While there is some small amount of overhang on the bottom of the front segment, the optimization still rotated the part such that support higher up the build plate is eliminated. Without segmentation, the second experiment also has overhang regions reduced but has higher compliance and more overhang (Fig. 8(*c*)). Due to the geometry of the bike frame, print angle optimization alone has limited effect.

To explore the relationship between overhang and support structure, we exported the geometry as stereolithography files and imported them into Cura with the optimized print direction applied [38]. We set the overhang angle to be 45°. The support structure is on the bottom of the part. We observe that almost all support material has been removed. We also demonstrate additional concurrent optimization examples in Fig. 9.

5 Limitations and Future Work

One concern with segmented parts is that it needs welding or assembly after printing. However, large components require segmentation to fit inside the build chamber already. Cost may also be a consideration to decide if the support structure reduced is worth the additional effort of piecing parts together. Nevertheless, we demonstrated with the addition of segmentation, support structures can be further reduced. We also observe that during the concurrent optimization of topology, segmentation, and print angle, the print angle has small oscillation. This is the result of using "min" operator when determining the lowest point of the part. Switching to p-norm to calculate the lowest point of the part may further improve optimization stability.

Currently one bottleneck for topology optimization using neural networks lies in the memory consumption during back propagation and hence very high resolution structures have not been presented in this paper. We show using mini-batches can potentially alleviate this issue and future work will explore creating a more robust and scalable framework.

The present work explored optimization for additive manufacturing. In the future, we plan to further develop upon the current framework by adding cost models to conduct optimization for manufacturing costs. With the cost model added, we can explore the cost related to the number of segments by accommodating the assembly cost. In the present work, the segmentation field requires an educated guess for initialization. This is due no penalizing disconnected islands within the segmentation network. A recent work by Zhou et al. [23] demonstrated segmentation with anisotropy modeled. We could incorporate material anisotropy into our framework not only to improve simulation accuracy for additive manufactured components but also reduce disconnected islands when performing segmentation. Currently, we treat the mating surface between each segment with the same mechanical performance while welding and assembly may alter the mechanical performance. Furthermore, subtractive machining constraints may be added from leveraging the continuous density field by the neural networks.

6 Conclusions

We present a framework for concurrent build direction, part segmentation, and topology optimization using neural networks that aim to reduce support structures. With only the topology network, we benchmark against SIMP and demonstrate that similar results can be obtained from using a neural network to learn the density field of topology optimization. When the print direction network is enabled, simultaneous print direction and topology optimization can be achieved. Detecting overhang regions and minimization can be done effectively using the analytical density gradient obtained from the input–output relationship of neural networks. Finally, with a bike frame example, we demonstrated the effectiveness of combined build direction, part segmentation, and topology optimization in reducing support structures.

Acknowledgment

This research was funded by Army Research Laboratory 50114.5.9.1130260. We would like to thank Krishnan Suresh and Aaditya Chandrasekhar for their insightful feedback. We also

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024

thank Sangjin Jung for help in problem formulation, data generation, and validation.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The data and information that support the findings of this article are freely available at online.[3]

References

[1] Nie, Z., Jung, S., Kara, L. B., and Whitefoot, K. S., 2020, "Optimization of Part Consolidation for Minimum Production Costs and Time Using Additive Manufacturing," ASME J. Mech. Des., **142**(7), p. 072001.

[2] Qian, X., 2017, "Undercut and Overhang Angle Control in Topology Optimization: A Density Gradient Based Integral Approach," Int. J. Numer. Methods Eng., **111**(3), pp. 247–272.

[3] Mirzendehdel, A. M., and Suresh, K., 2016, "Support Structure Constrained Topology Optimization for Additive Manufacturing," Comput. Aided Des., **81**, pp. 1–13.

[4] Wang, C., and Qian, X., 2020, "Simultaneous Optimization of Build Orientation and Topology for Additive Manufacturing," Addit. Manuf., **34**, p. 101246.

[5] Chandrasekhar, A., and Suresh, K., 2021, "Tounn: Topology Optimization Using Neural Networks," Struct. Multidiscipl. Optim., **63**, pp. 1135–1149.

[6] Thompson, M. K., Moroni, G., Vaneker, T., Fadel, G., Campbell, R. I., Gibson, I., Bernard, A., et al. 2016, "Design for Additive Manufacturing: Trends, Opportunities, Considerations, and Constraints," CIRP Ann. Manuf. Technol., **65**(2), pp. 737–760.

[7] Brackett, D., Ashcroft, I., and Hague, R., 2011, "Topology Optimization for Additive Manufacturing," 2011 International Solid Freeform Fabrication Symposium, Austin, TX, Aug. 8–10, pp. 348–362.

[8] Leary, M., Merli, L., Torti, F., Mazur, M., and Brandt, M., 2014, "Optimal Topology for Additive Manufacture: A Method for Enabling Additive Manufacture of Support-Free Optimal Structures," Mater. Des., **63**, pp. 678–690.

[9] Gaynor, A. T., and Guest, J. K., 2014, "Topology Optimization for Additive Manufacturing: Considering Maximum Overhang Constraint," AIAA Aviation 2014–15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Atlanta, GA, June 16–20,

[10] Mhapsekar, K., McConaha, M., and Anand, S., 2018, "Additive Manufacturing Constraints in Topology Optimization for Improved Manufacturability," ASME J. Manuf. Sci. Eng., **140**(5), p. 051017.

[11] Langelaar, M., 2016, "Topology Optimization of 3D Self-supporting Structures for Additive Manufacturing," Addit. Manuf., **12**, pp. 60–70.

[12] van de Ven, E., Maas, R., Ayas, C., Langelaar, M., and van Keulen, F., 2018, "Continuous Front Propagation-Based Overhang Control for Topology Optimization With Additive Manufacturing," Struct. Multidiscipl. Optim., **57**(5), pp. 2075–2091.

[13] Zhang, W., and Zhou, L., 2018, "Topology Optimization of Self-supporting Structures With Polygon Features for Additive Manufacturing," Comput. Methods Appl. Mech. Eng., **334**, pp. 56–78.

[14] Cheng, L., and To, A., 2019, "Part-Scale Build Orientation Optimization for Minimizing Residual Stress and Support Volume for Metal Additive Manufacturing: Theory and Experimental Validation," Comput. Aided Des., **113**, pp. 1–23.

[15] Liu, J., and To, A. C., 2017, "Deposition Path Planning-Integrated Structural Topology Optimization for 3D Additive Manufacturing Subject to Self-support Constraint," Comput. Aided Des., **91**, pp. 27–45.

[16] Zhang, K., Cheng, G., and Xu, L., 2019, "Topology Optimization Considering Overhang Constraint in Additive Manufacturing," Comput. Struct., **212**, pp. 86–100.

[17] Mezzadri, F., Bouriakov, V., and Qian, X., 2018, "Topology Optimization of Self-supporting Support Structures for Additive Manufacturing," Addit. Manuf., **21**, pp. 666–682.

[18] Wang, C., Qian, X., Gerstler, W. D., and Shubrooks, J., 2019, "Boundary Slope Control in Topology Optimization for Additive Manufacturing: For Self-support and Surface Roughness," ASME J. Manuf. Sci. Eng., **141**(9), p. 091001.

[19] Chandrasekhar, A., Kumar, T., and Suresh, K., 2020, "Build Optimization of Fiber-Reinforced Additively Manufactured Components," Struct. Multidiscipl. Optim., **61**, pp. 77–90.

[20] Ulu, E., Korkmaz, E., Yay, K., Ozdoganlar, O. B., and Kara, L. B., 2015, "Enhancing the Structural Performance of Additively Manufactured Objects Through Build Orientation Optimization," ASME J. Mech. Des., **137**(11), p. 111410.

[21] Ulu, E., Huang, R., Kara, L. B., and Whitefoot, K. S., 2019, "Concurrent Structure and Process Optimization for Minimum Cost Metal Additive Manufacturing," ASME J. Mech. Des., **141**(6), p. 111410.

[22] Ulu, E., Gecer Ulu, N., Hsiao, W., and Nelaturi, S., 2019, "Manufacturability Oriented Model Correction and Build Direction Optimization for Additive Manufacturing," ASME J. Mech. Des., **142**(6), p. 062001.

[23] Zhou, Y., Nomura, T., and Saitou, K., 2021, "Anisotropic Multicomponent Topology Optimization for Additive Manufacturing With Build Orientation Design and Stress-constrained Interfaces," ASME J. Comput. Inf. Sci. Eng., **21**(1), p. 011007.

[24] White, D. A., Arrighi, W. J., Kudo, J., and Watts, S. E., 2019, "Multiscale Topology Optimization Using Neural Network Surrogate Models," Comput. Methods Appl. Mech. Eng., **346**, pp. 1118–1135.

[25] Banga, S., Gehani, H., Bhilare, S., Patel, S., and Kara, L., 2018, "3D Topology Optimization Using Convolutional Neural Networks." arXiv preprint arXiv:1808.07440.

[26] Behzadi, M. M., and Ilieş, H. T., 2021, "Real-Time Topology Optimization in 3D Via Deep Transfer Learning," Comput. Aided Des., **135**, p. 103014.

[27] Zheng, S., He, Z., and Liu, H., 2021, "Generating Three-Dimensional Structural Topologies Via a U-net Convolutional Neural Network," Thin-Walled Struct., **159**, p. 107263.

[28] Cang, R., Yao, H., and Ren, Y., 2019, "One-Shot Generation of Near-Optimal Topology Through Theory-Driven Machine Learning," Comput. Aided Des., **109**, pp. 12–21.

[29] Nie, Z., Lin, T., Jiang, H., and Kara, L. B., 2021, "Topologygan: Topology Optimization Using Generative Adversarial Networks Based on Physical Fields Over the Initial Domain," ASME J. Mech. Des., **143**(3), p. 031715.

[30] Iyer, N. S., Mirzendehdel, A. M., Raghavan, S., Jiao, Y., Ulu, E., Behandish, M., Nelaturi, S., and Robinson, D. M., 2021, "PATO: Producibility-Aware Topology Optimization Using Deep Learning for Metal Additive Manufacturing," CoRR, abs/2112.04552

[31] Chi, H., Zhang, Y., Tang, T. L. E., Mirabella, L., Dalloro, L., Song, L., and Paulino, G. H., 2021, "Universal Machine Learning for Topology Optimization," Comput. Methods Appl. Mech. Eng., **375**), p. 112739.

[32] Chandrasekhar, A., and Suresh, K., 2021, "Length Scale Control in Topology Optimization Using Fourier Enhanced Neural Networks," CoRR, abs/2109.01861.

[33] Chandrasekhar, A., and Suresh, K., 2021, "Multi-material Topology Optimization Using Neural Networks," Comput. Aided Des., **136**, p. 103017.

[34] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., et al. 2016, "Tensorflow: Large-Scale Machine Learning on Heterogeneous Distributed Systems."

[35] Orbay, G., Fu, L., and Kara, L. B., 2015, "Deciphering the Influence of Product Shape on Consumer Judgments Through Geometric Abstraction," ASME J. Mech. Des., **137**(8), p. 081103.

[36] Liu, K., and Tovar, A., 2014, "An Efficient 3D Topology Optimization Code Written in Matlab," Struct. Multidiscipl. Optim., **50**, pp. 1175–1196.

[37] Kingma, D. P., and Ba, J. L., 2015, "Adam: A Method for Stochastic Optimization."

[38] Ultimaker, 2020, "Ultimaker Cura," Oct.12

[3] https://github.com/HongRayChen/seg-angle-topopt

Downloaded from http://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/145/9/091702/7020854/md_145_9_091702.pdf by Purdue University at West Lafayette user on 19 April 2024