

# Machines as Craftsmen: Localized Parameter Setting Optimization for Fused Filament Fabrication 3D Printing

John M. Gardner,\* Kevin A. Hunt, Austin B. Ebel, Evan S. Rose, Sean C. Zyllich, Benjamin D. Jensen, Kristopher E. Wise, Emilie J. Siochi, and Godfrey Sauti\*

Quality control and repeatability of 3D printing must be enhanced to fully unlock its utility beyond prototyping and noncritical applications. Machine learning is a potential solution to improving 3D printing performance and is explored for areas including flaw identification and property prediction. However, critical problems must be resolved before machine learning can truly enable 3D printing to reach its potential, including the very large data sets required for training and the inherently local nature of 3D printing where the optimum parameter settings vary throughout the part. This work outlines an end-to-end tool for integrating machine learning into the 3D printing process. The tool selects the ideal parameter settings at each location, taking into consideration factors such as geometry, hardware and material response times, and operator priorities. The tool demonstrates its usefulness by correcting for visual flaws common in fused filament fabrication parts. An image recognition neural network classifies local flaws in parts to create training data. A gradient boosting classifier then predicts the local flaws in future parts, based on location, geometry, and parameter settings. The tool selects optimum parameter settings based on the aforementioned factors. The resulting prints show increased quality over prints that use global parameters only.

FFF is known to have issues with part quality and consistency, which has limited its use to prototyping and noncritical applications where functional reliability does not affect safety.<sup>[2]</sup> The occurrence of issues such as poor surface finish, layer delamination, and poor dimensional stability depends on a number of parameter settings, including nozzle temperature, print speed, environmental conditions, geometry, and location (Figure 1a,b).<sup>[3]</sup> Experienced operators must set these parameters according to the material, part geometry, and 3D printer. It can be difficult for even an expert to select optimum parameter settings (Figure 1c,d).<sup>[4]</sup> An operator can specify settings for over 100 different printing options using a slicing software such as Cura. Attempting all combinations is not practical, so an operator must rely on their knowledge of 3D printing to adjust parameter settings. This leads to operator-dependent part-to-part variations and uncertainty

in the performance of the final part. Additionally, an operator will typically select a set of global parameter settings that are used throughout the part, or at most make some layer-to-layer adjustments. A more objective solution for selecting parameter settings is needed to ensure optimal use of the printer/material performance space and consistency in 3D-printed parts, regardless of part geometry, material, system, or operator.


Machine learning offers a potential approach to addressing this problem. Machine learning models are trained on thousands to millions of data points to recognize patterns that are too difficult to identify using deterministic algorithms.<sup>[5]</sup> Machine learning has been successfully applied in applications such as image processing, text classification, and speech recognition.<sup>[5–7]</sup> Potential uses for machine learning in 3D printing have also been studied in a limited capacity.<sup>[8]</sup> Examples of their use in both monitoring/feedback applications and predictive models include predicting property outcomes based on parameter settings, predicting global parameter settings for specific outcomes, identifying failures during printing, predicting bead geometry, adjusting geometry to prevent failures, and assessing part manufacturability.<sup>[9–27]</sup> An example of the utility of machine learning in the established quality control method of visual inspection is demonstrated by the use of a neural network to identify flaws in laser powder bed fusion 3D printing.<sup>[28,29]</sup> An

## 1. Introduction

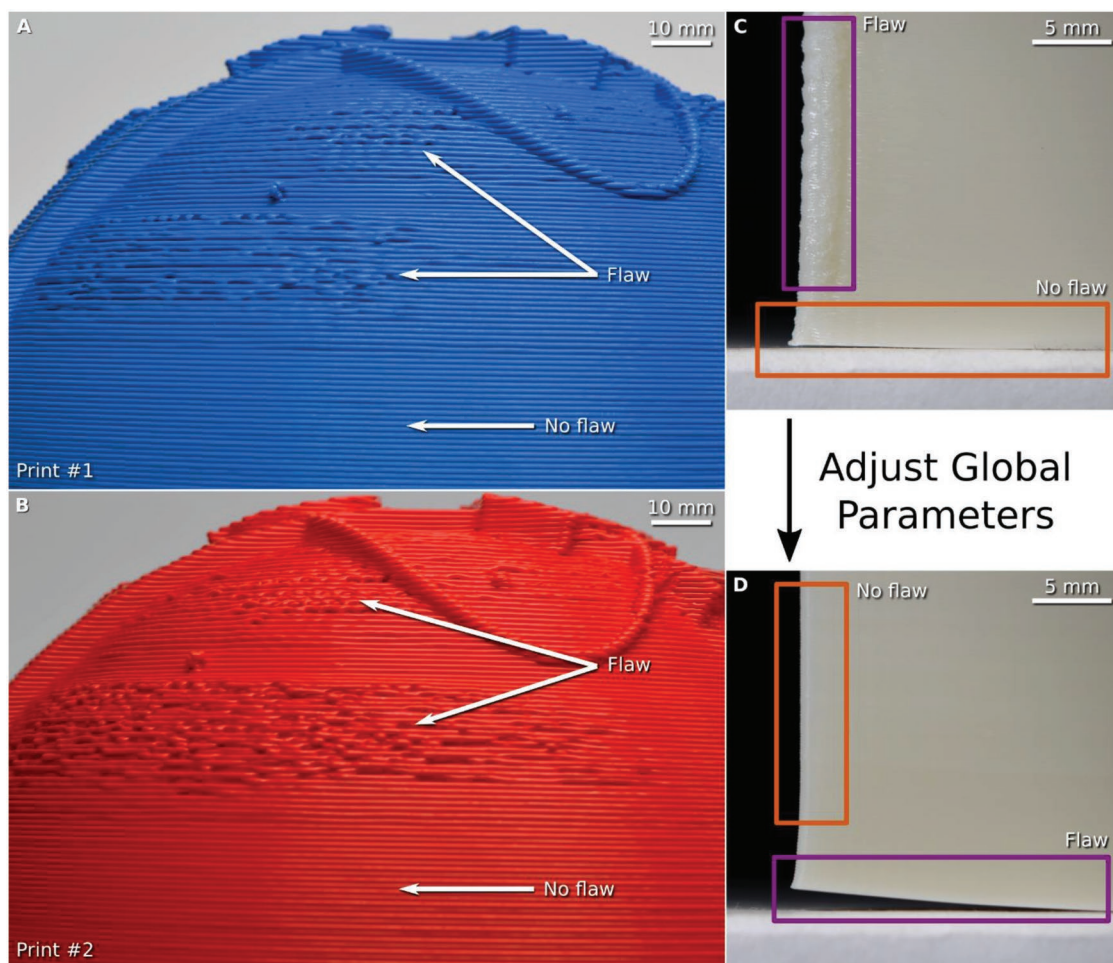
3D printing, also known as additive manufacturing, is a process that constructs objects by depositing material, typically in a layer-by-layer fashion, to yield geometries not achievable by conventional manufacturing techniques. This expanded geometric range allows for parts tailored for properties such as weight, stiffness, strength, or combinations thereof. This capability is of special interest to industries such as aerospace, where weight is a critical design factor. Fused filament fabrication (FFF), also known as fused deposition modeling (FDM), is a common type of 3D printing in these applications because of its scalability and ability to print a range of materials.<sup>[1]</sup> In FFF, a thermoplastic material is extruded from a hot nozzle in a predetermined pattern to build a part from a semicontinuous bead of material.

J. M. Gardner, K. A. Hunt, A. B. Ebel, E. S. Rose, S. C. Zyllich, Dr. B. D. Jensen, Dr. K. E. Wise, Dr. E. J. Siochi, Dr. G. Sauti  
NASA Langley Research Center  
Hampton, VA 23681, USA

E-mail: john.m.gardner@nasa.gov; godfrey.sauti-1@nasa.gov

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/admt.201800653>.

DOI: 10.1002/admt.201800653



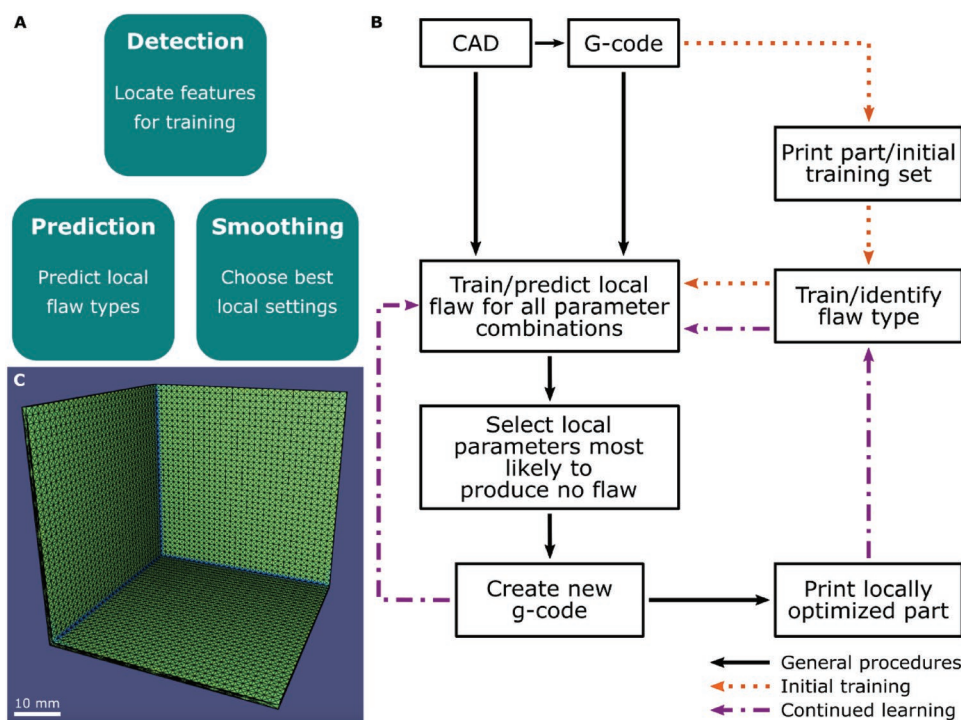
**Figure 1.** Global parameter setting flaws. A,B) Systematic flaws recur in multiple prints of a part and C,D) cannot be readily fixed by a change in the “global” parameter settings without affecting another area of the print.

image of the powder bed was captured after each layer, segmented, and put through a machine learning model to identify flaws at specific locations. A visual representation of the part could then be constructed to show the location of flaws. In another study, a part was thermomechanically modeled using finite element analysis (FEA) to determine part warping during printing.<sup>[30]</sup> The geometry of the part was then adjusted using machine learning to mitigate warping. While the work cited above demonstrates the potential of machine learning in 3D printing, collecting and processing sufficient quantities of high-quality data remains a challenge. Studies using data from printed, physical parts typically have fewer than 100 data points in the entire data set, while machine learning requires tens of thousands of data points for models to be considered valid.<sup>[5,8]</sup> This is a practical constraint given the cost and tedium of printing enough samples to provide sufficient data sets for the identification and classification of printing flaws as well as for determining fixes. Additionally, the work cited focuses on aspects of 3D printing and does not address the full process.

The goal of this work is to develop the foundations for an end-to-end tool that can determine near optimum location specific parameter settings for a given local geometry, material,

and machine without expert intervention. This, in effect, gives the machine qualities similar to those of a human craftsman by anticipating areas where flaws may occur and proactively adjusting parameter settings to maximize the chance of a flaw-free part. We have picked a subset of the 3D printing parameter space in order to develop the key elements of the tool. We focus on systematic visual flaws that occur in repeated prints of the same geometry (Figure 1a,b) whose mitigation often requires an operator to adjust some parameter settings that may be contradictory (Figure 1c,d). We do not directly address random flaws, such as those resulting from defects in the print filament or a fault in the machine. These are best addressed with the addition of sensors and real-time feedback mechanisms. Simplified data capture and processing in the tool open up a path to generate large test data sets for the learning.

We have devised a tool that selects 3D printing parameter settings based on local part geometry and visual flaws. Parts are segmented into discrete components and machine learning models are used to identify flaws in existing part segments. The segmentation process results in thousands of data points per print, increasing the size of the data set used for identifying print defects, even with a limited number of prints. A second



**Figure 2.** Tool overview. A) Elements of the tool showing the three distinct modules. B) Data flow between the CAD model, G-code, and the print. C) The part geometry chosen to develop and demonstrate the functionality of the tool.

set of machine learning models predicts flaws in future part segments based on parameter settings and geometry. From these predictions, the tool selects local parameter settings that give the highest probability of a flaw-free part. Throughout the process, printed optimized parts are used to retrain the models using semisupervised learning, which results in more accurate prints with each iteration. This paper discusses the development, training, and operation of the tool. Evidence of its functionality is demonstrated by predicting flaws in future prints, optimizing a print for minimal flaws, and simultaneously optimizing for minimal flaws and print time.

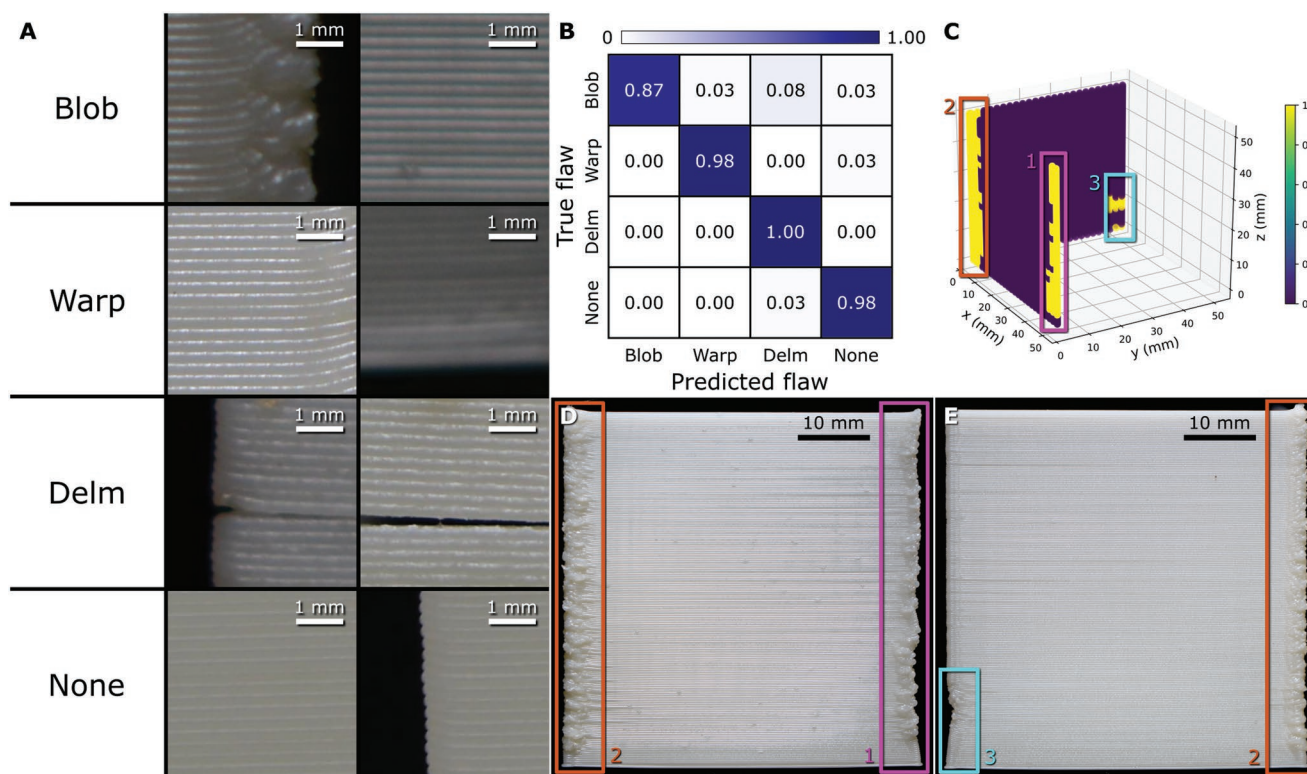
## 2. Results

### 2.1. Tool Overview

The tool uses localized data, obtained from an image recognition algorithm that identifies and classifies flaws and part geometry, to train a model that predicts future flaws in a part. The flaw prediction results are then used to adjust and optimize parameter settings. Only visual flaws are considered because they are easy to identify and appear in unique locations, which helps demonstrate the effectiveness of localized optimization. The tool consists of three independent modules: the detection module, the prediction module, and the smoothing module (Figure 2a). Figure 2b shows a schematic diagram of the tool implemented in this work with the solid (black) lines indicating general use procedures, dotted (orange) lines for training procedures, and dashed (purple) lines for continued learning. The detection module provides data on the location and types of

flaws in a part for training and testing. These data, combined with geometry, parameter settings, and toolpath data, train the prediction module. The prediction module outputs the probability of each flaw classification occurring at every local point for all possible parameter combinations. From these probabilities, the smoothing module generates a new set of machine readable instructions, known as G-code, based on a set of “best” local parameter settings. These parameter settings are chosen to minimize the probability of flaws occurring while meeting hardware limitations and any added operator priorities. Locally optimized parts are printed, analyzed by the detection module, and fed back into the prediction module for continued learning.

Factors including the machine, part geometry, material, and range of parameter settings are set and held constant throughout the experiments to demonstrate the workflow through the tool. This restriction reduces the complexity and variability of the data in the development process but does not limit the demonstration of the essential elements of the workflow. The test part is built with acrylonitrile butadiene styrene (ABS) and consists of two perpendicular thin walls connected by a flat bottom (Figure 2c). This material and geometry combination was chosen for the frequency and type of flaws common in the part and ease of data collection. Digital photographs are used to capture data for the detection module. The flaw classifications used are: blobs, warps, delaminations, and “none” (Figure 3a). Blobs are the appearance of excess material on the surface of the print. Warps are areas of the print that are curved upward and appear in the sample images as layer lines with a slight curvature. Delaminations are defined as two layers that have debonded and separated over part of the interface. “None” are the absence of any of the aforementioned flaws. Both the



**Figure 3.** Detection module. A) Sample images of the blob, warp, delamination (delm), and “none” flaw classifications used to train the detection module and develop the workflow through the tool. B) Confusion matrix showing validation accuracy of the detection module. C) Heat map showing the probability of blobs in the part identified by the detection module. D, E) Printed part with the flaws identified in (C) correlated with the true flaws.

detection module and prediction module are trained on flaws found on the outward facing sides of the walls. The five printing parameters and discrete setting ranges are as follows: nozzle temperature (225, 235, and 245 °C), bed temperature (90, 100, and 110 °C), print speed (20, 40, 60, and 80 mm s<sup>-1</sup>), extrusion multiplier (90, 100, and 110%), and fan speed (0, 30, 45, and 80% of the maximum speed). These parameters and ranges, which are not intended to be exhaustive, are fundamental to the process, easy to program through G-code, and cover a range where the effects of changes are readily observable. There are 432 possible combinations of global parameter settings of which 144 combinations were printed in random order and used to train the models in both modules. The random ordering ensures the training data sample the full parameter space while more data are gathered and enable exploration of the end-to-end functionality of the tool. Each part took between 13 and 45 min to print.

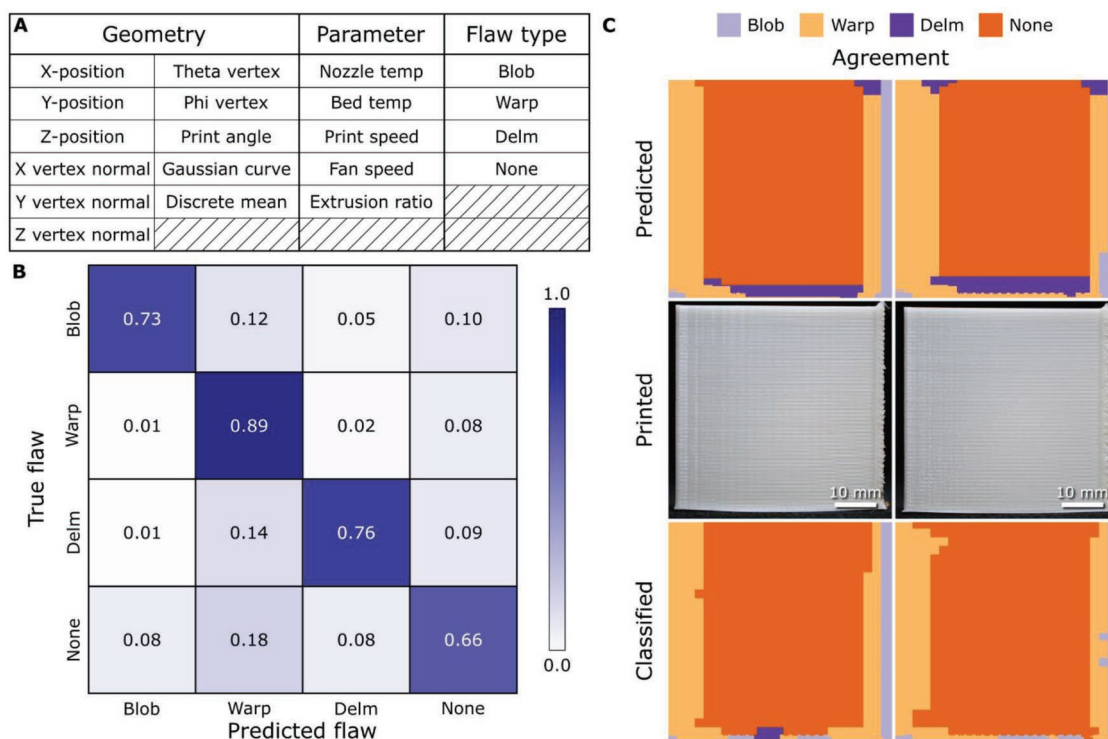
## 2.2. Detection Module

The detection module provides flaw classification and location data, initially from parts printed with global parameter settings, to train and test the prediction module. In this work, we use transfer learning with AlexNet to create an image classification model. The two walls of the printed part are photographed, segmented into smaller images showing 4 mm × 4 mm overlapping sections, and manually labeled (Figure 3a).<sup>[6]</sup> AlexNet is

trained on these images in a pseudo semisupervised learning process where each newly classified image is checked by a human, reclassified if necessary, and the model retrained. The model has an accuracy of 98.26% with a low rate of confusion between flaws (Figure 3b). The detection module classified flaws in parts not included in the training data to further demonstrate its capability. The graphical representation of the detection module output (Figure 3c) shows the probability of blobs detected in a part. Colored boxes in the images directly correlate flaws classified by the detection module with these regions in the part (Figure 3d,e).

## 2.3. Prediction Module

The prediction module calculates the probability of various types of flaws occurring at each location in the print based on parameter settings and geometry. We use a gradient boosting classifier in Scikit-learn to calculate the flaw probability.<sup>[31]</sup> The prediction module outputs a table of these probabilities or the most likely flaw classification (including “none”) at each segment. To calculate the probabilities, local geometries (Figure 4a) harvested from the STL file, toolpath locations, and parameter settings are fed to the gradient boosting classifier. The trained model has an accuracy of 73.69%. The confusion matrix (Figure 4b) shows that the model performs well at predicting the delamination flaws with 89% accuracy but has a high rate of false positives for that category. There is some confusion between blob and warp



**Figure 4.** Prediction module. A) The prediction module processes local geometry and parameter data to predict flaw classifications B) with a confidence shown in the confusion matrix. C) The agreement between the prediction module and the detection module classifications of the flaws in the printed parts.

flaws, both of which commonly occur near the edge of the part. The rate of false positives in these two classifications is lower than that for delaminations. The model is conservative when predicting “none” flaws with a 24% false negative rate.

Several prints not in the training data were selected at random, run through the prediction module, printed, and analyzed using the detection module. The results are compared to the predictions to test the validity of the algorithm. Correlation between the predicted flaws and classified flaws is 95.2 and 82.6% for the test cases shown, respectively (Figure 4c). Overall agreement between the prediction module and detection module models for all test cases is 79.83%.

## 2.4. Smoothing Module

The prediction module does not explicitly include material and hardware response limitations in arriving at the “best” parameter settings. These limitations must be addressed more directly to yield printable G-code. The material and hardware response to parameter setting changes will dictate the minimum step size and the time scale for these changes in the final G-code.

The mechanical responses of the 3D printing hardware and material limit the range of parameter setting changes achievable. For example, nozzle temperature response time is a hardware limited characteristic. Under worst case conditions (highest print speed, fan speed, and extrusion multiplier), the nozzle temperature increases at a rate of around  $14\text{ }^{\circ}\text{C min}^{-1}$ . The cooling rate is a much faster  $60\text{ }^{\circ}\text{C min}^{-1}$ , but the controller

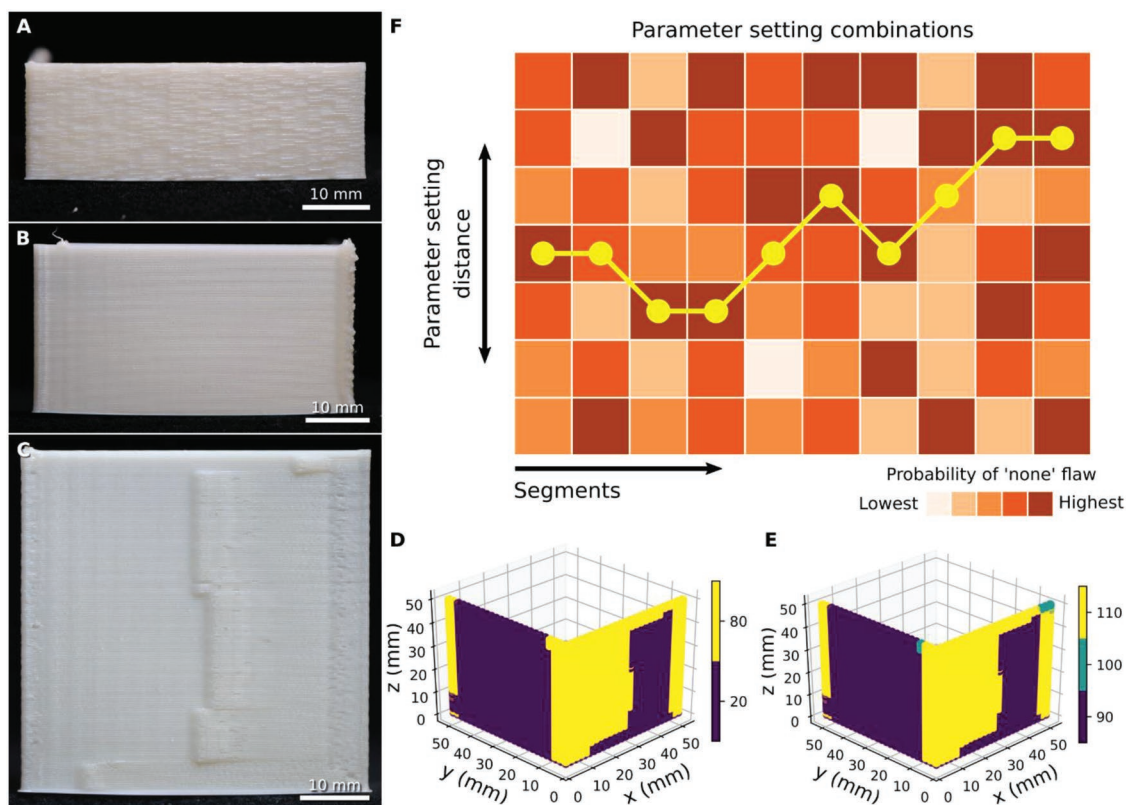
overshoots and takes nearly a minute to settle after reaching the target temperature (Figure S1a, Supporting Information).

The material response to changes in extrusion multiplier and print speed is another limiting characteristic. **Figure 5a–c** shows two parts printed with the print speed and extrusion multiplier randomly selected at each segment. Changing print speed results in a part with a rough, wavy texture corresponding to the changes. In contrast, very little effect is seen when the extrusion multiplier alone is adjusted (Figure 5b). Blobs are present in the print (Figure 5c) corresponding to a simultaneous decrease in the print speed and extrusion multiplier (Figure 5d,e).

The smoothing module is used to produce a coherent set of parameter settings to account for the above and other limitations. Various methods can be used to do this, including algorithms that account for the known material and hardware responses. In this work, a simple walk of the “best” parameter settings is implemented to demonstrate the functionality of the module (Figure 5f). Using a table containing the probability of various flaws generated by the prediction module, the smoothing module charts a path between adjacent segments that aims to maintain the highest probability of “none” flaws while minimizing the segment-to-segment change in the parameter settings.

## 2.5. Optimizing for Quality

The tool selects parameter settings to minimize visual flaws. From here on we will define minimizing the visual flaws as



**Figure 5.** Smoothing module. A–C) Artifacts in the print due to the limitations of the hardware and the material response. D,E) Parameter setting changes for print speed and extrusion multiplier prior to smoothing. F) Smoothing of the parameter settings changes by the smoothing module.

increasing the quality of the part. This is done by selecting the parameter setting combinations with the highest probability of “none” flaws. Blindly following this procedure over the entire part results in G-code that cannot be printed in practice (Figure S1b, Supporting Information), so modifications are performed. First, global parameter settings are chosen for nozzle and bed temperature, since hardware response for these two parameters is slow. “Best” parameter settings selection is performed for the remaining three parameters. Then, parameter setting changes over consecutive points are smoothed based on the machine’s capability.

Graphical representations of the G-code for print speed, extrusion multiplier, and fan speed settings for the quality optimized part are shown in Figure 6a–c. The bottom layers and corners show how the module adjusts settings across the three parameters to increase the quality of the part. The print speed is slowed to the lowest setting ( $20 \text{ mm s}^{-1}$ ) in the corners in the bottom layers. Fan speed is increased and extrusion multiplier is decreased for the corners. The part took 24 min and 13 s to print. The resulting part quality (Figure 6d) is 86.41%, which is in the top 0.5% of all parts classified by the detection module.

## 2.6. Optimizing for Print Time and Quality

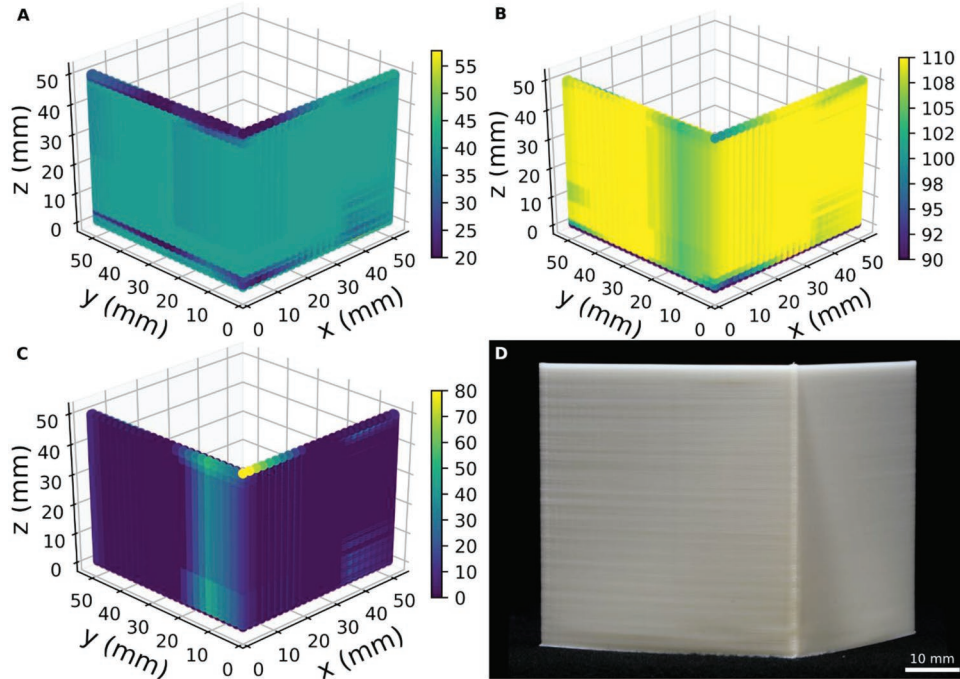
In addition to optimizing purely for print quality, it is often customary for expert operators to also optimize for factors such as print speed at the expense of one of the quality metrics.

Similarly, the tool can account for operator priorities, such as print time, in addition to material and hardware limitations when building G-code. For example, parameter settings can be optimized to yield the lowest print time while maintaining overall part shape and integrity (i.e., no warps and delaminations). The tool achieves this objective by incrementing the print speed to the limit where there are no warps and delaminations at the expense of residual blobs.

Graphical representations of the G-code for print speed, extrusion multiplier, and fan speed settings are shown in Figure 7a–c. Once again, the tool selects a lower print speed on the first few layers. Print speed and extrusion multiplier are reduced around the corners and fan speed is increased. Fan speed is reduced in the lower sections of the part and increases as it approaches the top. The extrusion multiplier shows a similar trend. The part (Figure 7d) took 16 min and 56 s to complete and displays no flaws in most of the part, although more pronounced blobs are observed around the corner. The part is 76.61% “none” flaw as classified by the detection module.

## 2.7. Overall Results

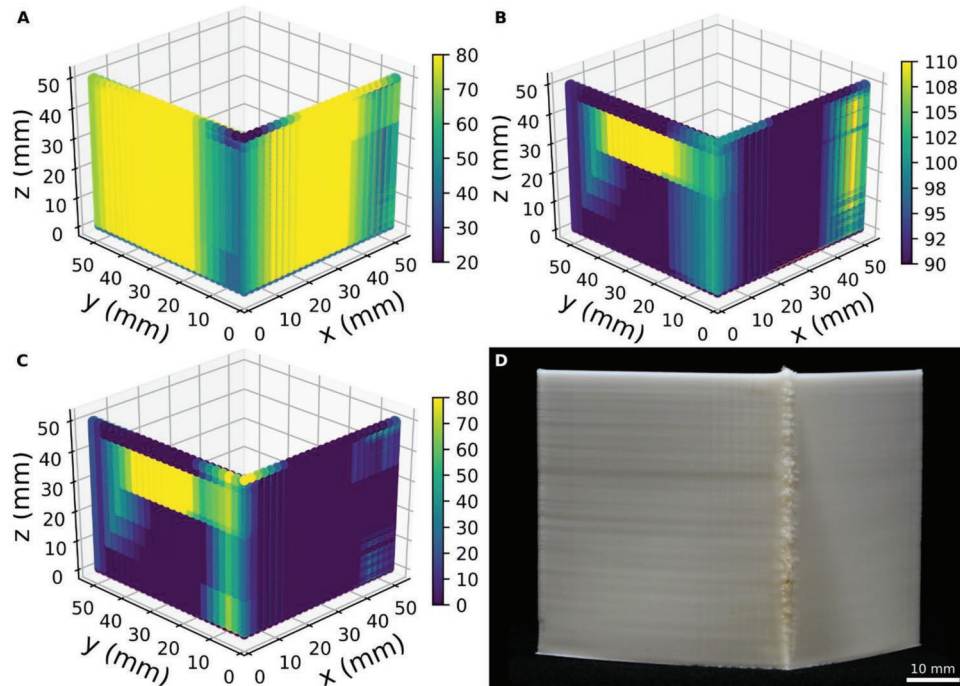
The tool, using locally optimized parameter settings, printed high-quality parts when compared to parts printed with global parameters settings without the need for operator intervention, demonstrating its functionality and versatility. Comparisons of detection module classified quality and print time for the



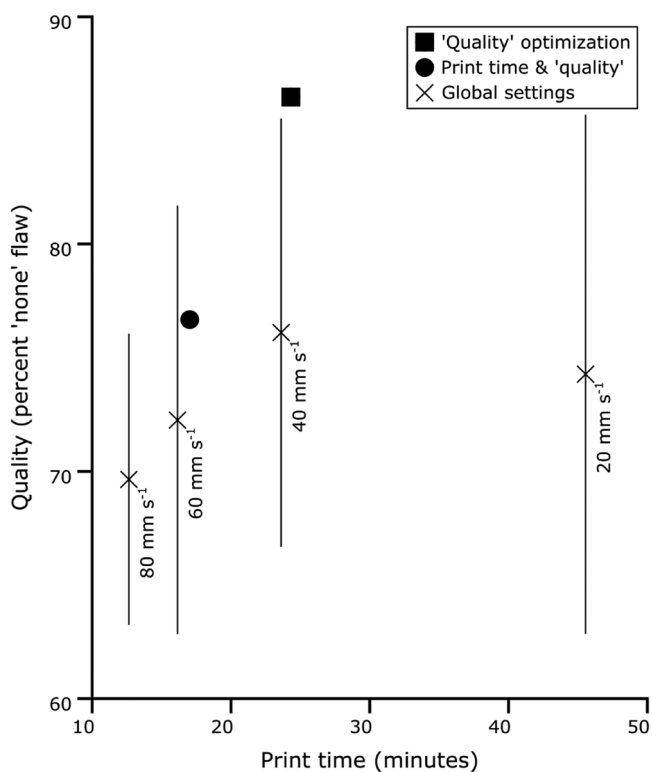
**Figure 6.** Quality optimized part. Smoothed changes in the A) print speed, B) extrusion multiplier, and C) fan speed for a D) quality optimized part.

quality optimized part, print time and quality optimized part, and training (global settings) parts show how the tool performs against global parameter setting combinations (Figure 8). The quality optimized print is 14% higher “none” flaw than the average part with global settings and  $40 \text{ mm s}^{-1}$  print speed. The print time and quality optimized part quality (76.61%) is

similar to the average quality of the  $40 \text{ mm s}^{-1}$  setting combinations (76.04%) but has a print time that is 28% faster. Compared with one another the quality optimized part has a 13% higher quality, while the print time and quality optimized part prints 30% faster. Overall, the tool sacrifices quality for speed, or vice versa, depending on the operator’s priorities.



**Figure 7.** Print time and quality optimized part. Smoothed changes in the A) print speed, B) extrusion multiplier, and C) fan speed for a D) print time and quality optimized part.



**Figure 8.** Overall performance of the tool in optimizing for print quality and optimizing for print time and quality compared against prints using global parameter settings.

### 3. Discussion

#### 3.1. Local Parameter Setting Trends

We observe three types of parameter setting responses/trends output by the tool: operator-like behavior, complex behavior, and emergent behaviors. Operator-like behaviors include those parameter setting changes that are common practice in 3D printing and can be easily programmed by the operator. Nozzle temperature changes are evidence of this behavior (Figure S1b, Supporting Information). The nozzle temperature is set to the highest setting (245 °C), particularly around the corners, in the first few layers at the bottom of the print. A higher nozzle temperature closer to the bottom of the print is a technique that increases the adhesion between the first layers of the part and the print bed. This has the effect of reducing warping at the bottom of the part by preventing the first layers from detaching. Another example of this behavior is found in the print speeds selected for the first few layers of both the quality and print time and quality optimized parts (Figures 6a and 7a). Speed is lowered, which has the effect of reducing warping in the first few layers of the part. Both of these are operator selectable options in the Cura slicing software.

Complex behavior refers to parameter setting changes for which the reasoning is understandable but not easily implemented using basic slicing software. This type of behavior is evident around sharp corners of the part. The tool chooses a lower print speed in these areas for both the quality and print

time and quality optimized parts. This is an understandable reaction of the tool as reducing print speed allows for increased adhesion between layers, which prevents delaminations. Simultaneously, a decreased print speed and decreased substrate temperature result in fewer blobs. Fan power increases as well, which also helps to cool the substrate and further prevent blobs. Currently, the only way for a human operator to adjust parameter settings at specified points such as these is to manually write the changes into G-code.

Emergent behaviors occur when the correlation between parameter setting changes and printing results is not *a priori* predictable. One such emergent behavior is that of fan speed and extrusion multiplier. These two parameters are coupled in both of the optimizations but do not display the same relationship. The fan speed and extrusion multiplier are inversely proportional in the quality optimized part and directly proportional in the print time and quality optimized part. Additionally, fan speed and extrusion multiplier increase at higher layers in the print time and quality optimized part, while no such change is observed in the quality optimized part. Although theories could be offered, they would be largely speculative. We can, however, conclude that the tool does not use a single type of parameter setting adjustment for a specific flaw.

Analysis of the parameters selected reveals that the tool is capable of solving complex problems. The combination of behaviors outlined above shows that the tool not only makes changes to the parameter settings that are consistent with known techniques but also implements solutions beyond the capability of a human operator.

#### 3.2. Hardware Response

The mechanical and thermal response of the printer and material is an important factor to consider when designing the tool. Simply changing parameter settings at each segment will not result in flaw-free locations as predicted by the prediction module because of inherent limitations due to response times. The 3D printer used is unmodified and, therefore, has a fixed nozzle mass, heater, and control system that is not designed to execute sudden changes in temperature. Poor print quality with sudden changes in print speed and extrusion multiplier is likely due to the compressible nature of molten ABS. It is possible that the combination of the machine limitations and physics of the material's response to print conditions also play a role.

The algorithm implemented in the smoothing module is crucial to achieving quality parts since the prediction module was not trained for flaws induced by the mechanical response. Even the simple walk of the “best” parameter settings technique used for these experiments greatly improved the quality of the part over what could have resulted without smoothing. Further work on implementing more sophisticated smoothing algorithms will likely lead to further improvements in print quality. Training the predictive model to account for printing conditions preceding a segment of interest would provide additional smoothing and increase quality, as mechanical response would be represented in the predictions sent to the smoothing module. This would reduce the occurrence of blobs seen in the print time and quality optimized part as those are likely due



to factors that are not accounted for in the current prediction module. Additionally, incorporating data from computational modeling of the printing process would complement learning from the printed results.

Hardware improvements are also important to reduce the response time of the printer and complement the smoothing algorithm. Changes to the control system of the machine, for example, have been shown to reduce flaws frequently encountered at high speeds.<sup>[32]</sup> Despite the major effects of hardware and material limitations, the tool is still able to produce quality parts and demonstrate the utility of machine learning in 3D printing.

### 3.3. Continued Learning

The advantage of using machine learning to optimize 3D printing is that it continues to improve as it encounters new situations and reinforces old ones. More data are necessary to improve the accuracy and generalizability of the machine learning models used in the tool now that its effectiveness has been demonstrated. Examples of additional training data include more varied geometries, inputs from more and different sensors, and optimizing for additional print parameters. The predictive model trained in this work only saw one type of geometry, so the weights are closely tied to location coordinates. A more generalizable training geometry would allow the tool to shift focus to geometric features and increase the likelihood of a correct response when the tool encounters parts it has not seen before but contain similar features. Additional sensors could augment the detection module to ensure that the flaws inferred are real. For example, the detection module had issues correctly classifying warps that occurred at the bottom of the part, often misclassifying them as delaminations. A sensor that can detect when the bottom layers of the part detach from the print bed could be used to improve the reliability of the detection module in identifying warps. It is also known that factors including substrate temperature can affect how and when the flaws form in the part. Having this type of data available would be useful in improving the accuracy of the predictive model.

This work focuses on using the tool to optimize for visible print flaws, but other metrics, such as road width and dimensional stability, could also be addressed assuming the effects can be measured locally. Correlations between local flaws and overall part performance, such as mechanical properties, can be deduced with further characterization and the data made available for training. While modifying the tool to optimize for other metrics would require adaptation for new sensors, data, and outputs, the high-level structure of the tool would remain intact.

The flexibility of the tool is the result of each module being independent of each other. This allows each to be modified, tailored, and improved upon so the tool can be easily modified to handle different scenarios. Only the input and output variables need to be compatible, as each module feeds the next. Also, using separate modules to detect flaws and optimize parameter settings means that the output of the tool is not constrained to what the detection module is trained to identify or the prediction module to predict. The quality and print time and quality optimizations demonstrate this directly. The smoothing module was adjusted to produce unique responses for different

scenarios with the other two modules left unchanged. This modular design removes constraints on geometry, parameters, targeted output, machine, 3D printing technique, or manufacturing process in general.

## 4. Experimental Section

*Printing:* All printing was performed on an unmodified Lulzbot Taz Mini 3D Printer (Aleph Objects, Loveland, CO) with a 0.5 mm nozzle using 3 mm diameter natural ABS filament (Village Plastics, Barberton, OH) in an ambient environment. Slicing and toolpath planning was performed using Cura version 3.2.21 and formed the base G-code. G-code for all remaining parameter combinations were produced by modifying the base G-code using a Python script. The initial parameter settings and ranges were chosen based on the default settings and physical limits of the filament and printer. Layer height was set at 0.25 mm for all prints. Infill was not required due to the thickness of the walls.

*Data Collection:* Data were collected by photographing the two outside walls of the printed part using a Nikon D3200 DSLR camera with an 85 mm macro lens against a black backdrop. The images were used as input to a MATLAB script that segmented them into smaller 4 mm × 4 mm subsections of the part overlapping every 2 mm. Part geometry data were collected by first using Autodesk NetFabb to refine the STL mesh size in the part to a maximum edge length of 0.1 mm and analyzed using libigl<sup>[33]</sup> to extract geometric information from each vertex. The G-code was segmented into roughly 2 mm linear sections for alignment with the geometric data. Finally, the coordinates of the G-code and geometry data were aligned and the data mapped to their respective points. Data for the images, geometry, and G-code were brought together and stored using the Python Pandas DataFrame.

*Model Training and Validation:* AlexNet in MATLAB r2018a was used for the detection module. The complete data set for the detection module contained over 170 000 image segments in total, however, around 95% of these images were “none” flaws. Therefore, the training data were reduced to 459 images per flaw classification (1836 total images). The limiting factor for the detection module model was delaminations. The model was trained on the data set using a Quadro K4000 GPU with parallelization for a total training time of ≈1 h and can classify all 676 image segments of each wall in 35 s.

A gradient boosting classifier in Scikit-learn (Python 3.6.5) was used for the prediction module. The training data set for the prediction module consists of 49 164 data points evenly spread over the flaw classifications. Downsampling was employed to prevent overfitting caused by the high number of “none” flaws and low number of other flaws including delamination. Data points from a single print appeared in either the training data or test data only. After training, the model took around 3 s to predict the  $3.24 \times 10^6$  local parameter setting combinations for a part. All validation experiments were performed using the same equipment and environment as the training data set.

## Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

## Acknowledgements

The authors would like to thank Mr. Ben Malouf of Aleph Objects for assistance with material procurement and Mr. Christopher J. Stelter of NASA Langley for help with data collection. This article is part of the special series on Advanced Intelligent Systems that showcases the outstanding achievements of leading international researchers on intelligent systems.

## Conflict of Interest

The authors declare no conflict of interest.

## Keywords

3D printing, additive manufacturing, artificial intelligence, machine learning, process automation

Received: November 26, 2018

Revised: January 12, 2019

Published online: January 30, 2019

- 
- [1] S. S. Crump, *US Patent 5,121,329*, **1992**.
- [2] L. E. J. Thomas-Seale, J. C. Kirkman-Brown, M. M. Attallah, D. M. Espino, D. E. T. Shepherd, *Int. J. Prod. Econ.* **2018**, *198*, 104.
- [3] H. Kim, L. Yirong, T.-L. B. Tseng, *Rapid Prototyping J.* **2018**, *24*, 645.
- [4] S. Abdollahi, A. Davis, J. Miller, A. Feinberg, *PLoS One* **2018**, *13*, e0194890.
- [5] A. Singh, N. Thakur, A. Sharma, in *Int. Conf. Computing for Sustainable Global Development*, IEEE, NY, NY **2016**, pp. 1310–1315.
- [6] A. Krizhevsky, I. Sutskever, G. Hinton, in *Proc. Advances in Neural Information Processing Systems*, Curran Associates, Inc., Red Hook, NY **2012**, pp. 1090–1098.
- [7] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, B. Kingsbury, *IEEE Signal Process. Mag.* **2012**, *29*, 82.
- [8] I. Baturynska, O. Semeniuta, K. Martinsen, *Proc. CIRP* **2017**, *67*, 227.
- [9] O. A. Mohamed, S. H. Masood, J. L. Bhowmik, *Materials* **2016**, *9*, 895.
- [10] E. Vahabli, S. Rahmati, *Int. J. Precis. Eng. Manuf.* **2016**, *17*, 1589.
- [11] V. Vijayaraghavan, A. Garg, J. S. L. Lam, B. Panda, S. S. Mahapatra, *Int. J. Adv. Manuf. Technol.* **2015**, *78*, 781.
- [12] M. Asadi-Eydivand, M. Solati-Hashjin, A. Farzad, N. A. A. Osman, *Rob. Comput.-Integr. Manuf.* **2016**, *37*, 57.
- [13] M. Khanzadeh, P. Rao, R. Jafari-Marandi, B. K. Smith, M. A. Tschopp, L. Bian, *J. Manuf. Sci. Eng.* **2017**, *140*, 031011.
- [14] W. Yan, S. Lin, O. Kafka, Y. Lian, C. Yu, Z. Liu, J. Yan, S. Wolff, H. Wu, E. Ndip-Agbor, M. Mozaffar, K. Ehmann, J. Cao, G. J. Wagner, W. K. Liu, *Comput. Mech.* **2018**, *61*, 521.
- [15] A. Boschetto, V. Giordano, F. Veniali, *Int. J. Adv. Manuf. Technol.* **2013**, *67*, 2727.
- [16] R. V. Rao, D. P. Rai, *Eng. Sci. Technol., Int. J.* **2016**, *19*, 587.
- [17] G. Casalino, *Opt. Laser Technol.* **2018**, *100*, 165.
- [18] R.-J. Wang, X.-H. Li, Q.-D. Wu, L. Wang, *Int. J. Adv. Manuf. Technol.* **2009**, *42*, 621.
- [19] M. Hirsch, P. Druburgh, S. Catchpole-Smith, R. Patel, L. Parry, S. D. Sharples, I. A. Ashcroft, A. T. Clare, *Addit. Manuf.* **2018**, *19*, 127.
- [20] F. Baumann, D. Roller, *MATEC Web Conf.* **2016**, *59*, 06003.
- [21] N. G. Makagonov, E. M. Blinova, I. I. Bezukladnikov, presented at 2017 IEEE Conf. Russian Young Researchers in Electrical and Electronic Engineering, St. Petersburg, Russia, February **2017**.
- [22] J. Xiong, G. Zhang, J. Hu, Y. Li, *Int. J. Adv. Manuf. Technol.* **2013**, *69*, 743.
- [23] D. Ding, Z. Pan, D. Cuiuri, H. Li, S. van Duin, N. Larkin, *Rob. Comput.-Integr. Manuf.* **2016**, *39*, 32.
- [24] L. Cheng, A. Wang, F. Tsung, *IIEE Trans.* **2018**, *50*, 394.
- [25] A. Noriega, D. Blanco, B. J. Alvarez, A. Garcia, *Int. J. Adv. Manuf. Technol.* **2013**, *69*, 2301.
- [26] Y. Shi, Y. Zhang, S. Baek, W. De Backer, R. Harik, *Comput.-Aided Des. Appl.* **2018**, *15*, 941.
- [27] Y. Wang, R. Blache, P. Zheng, X. Xu, *J. Mech. Des.* **2018**, *140*, 5.
- [28] E. N. Malamas, E. G. M. Petrakis, M. Zervakis, L. Petit, J. D. Legat, *Image Vision Comput.* **2003**, *21*, 171.
- [29] L. Scime, J. Beuth, *Addit. Manuf.* **2018**, *19*, 114.
- [30] S. Chowdhury, K. Mhapsekar, S. Anand, *J. Manuf. Sci. Eng.* **2017**, *140*, 031009.
- [31] J. H. Friedman, *Ann. Stat.* **2001**, *29*, 1189.
- [32] M. Duan, D. Yoon, C. Okwudire, *Mechatronics* **2018**, *56*, 287.
- [33] A. Jacobson, D. Panozzo, C. Schüller, O. Diamanti, Q. Zhou, S. Koch, J. Dumas, A. Vaxman, N. Pietroni, S. Brugger, K. Takayama, W. Jakob, N. De Giorgis, L. Rocca, L. Sacht, K. Walliman, O. Sorkine-Hornung, libigl: A simple C++ geometry processing library, **2018**, <http://libigl.github.io/libigl/>.