Proceedings of the ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC/CIE2019
August 18-21, 2019, Anaheim, CA, USA

IDETC2019-98103

VOXEL-BASED CAD FRAMEWORK FOR PLANNING FUNCTIONALLY GRADED AND MULTI-STEP RAPID FABRICATION PROCESSES

Cole Brauer
Student
Ira A. Fulton Schools of Engineering
Arizona State University
Tempe, Arizona
Email: cbrauer@asu.edu

Daniel M. Aukes
Assistant Professor
Ira A. Fulton Schools of Engineering
Arizona State University
Tempe, Arizona
Email: danaukes@asu.edu

ABSTRACT

In this paper we describe a new framework for planning functionally graded and multi-step fabrication processes for use in rapid prototyping applications. This framework is contributing to software tools that will simplify planning multi-material manufacturing processes and thereby make this type of manufacturing more accessible. We introduce the material description itself, low-level operations which can be used to combine one or more geometries together, and algorithms which assist the designer in computing manufacturing-compatible sequences. We then apply these tools to several example scenarios. First, we demonstrate the use of a Gaussian blur to add graded material transitions to a model which can then be produced using a multi-material 3D printing process. Our second example highlights our solution to the problem of inserting a discrete, off-the-shelf part into a 3D printed model during the printing sequence. Finally, we implement this second example and manufacture two example components. The results show that the framework can be used to effectively generate the files needed to produce specific classes of parts.

1 Introduction

Multi-material manufacturing combines multiple fabrication processes to produce individual parts that can be made up of several different materials. These processes can include both additive and subtractive manufacturing methods as well as embedding other components during manufacturing. This yields opportunities for creating single parts that can take the place of an assembly of parts produced using conventional techniques. Some example applications of multi-material manufacturing include parts that are produced using one process then machined to tolerance using another, parts with integrated flexible joints, or parts that contain discrete embedded components such as reinforcing materials or electronics.

Multi-material manufacturing has applications in robotics because, with it, mechanisms can be built into a design without adding additional moving parts. This allows for robot designs that are both robust and low cost, making it a particularly attractive method for education or research. 3D printing is of particular interest in this area because it is low cost, readily available, and capable of easily producing complicated part geometries. Some machines are also capable of depositing multiple materials during a single process.

Current 3D printers are typically used in conjunction with CAM software that takes a CAD-generated file and "slices" it into a layer-based manufacturing solution based on a combination of user preferences and machine capabilities. These programs then output machine code which can be read by the printer. Due to the work-flow of this process, however, these programs typically only take into account the surface geometry of CAD designs. Furthermore, most of the widely-used parametric solid modeling tools used in engineering disciplines to design robots or other devices do not permit users to specify graded or multi-

material parts natively. This makes it difficult to assign material properties during the design process and have those properties become directly available for use in manufacturing planning software or with tools like FEA to understand the stiffness of a design.

Thus, up to this point, planning the process to create a part using multi-material manufacturing has been done manually, requiring specialized knowledge of the tools used. The difficulty of this planning procedure can prevent many students and researchers from using multi-material manufacturing.

We envision that due to the capabilities afforded by current and anticipated manufacturing technology, future design workflows will need to consider both the shape and material composition of parts. We anticipate this approach would permit a designer to specify the mass or strength distribution of 3D printed parts, as well as the location and size of material transitions. Such geometries could be used to compute how to split a design into manufacturing sequences occurring on multiple machines, where each sequence considers the capabilities and limitations of each sub-process. Finally, we envision that knowledge of how something is made could lead to automatic generation of support geometry for orienting and fixturing devices in each machine. These concepts apply to a number of manufacturing processes, including additive processes such as single material 3D printing, multi-material 3D printing, and resin casting, and subtractive processes such as laser cutting and CNC milling.

In this paper we present first steps toward a computational framework for processing 3D models and automatically generating viable manufacturing processes. Using a voxel-based descriptor, we then illustrate how this framework could be used in the development of a multi-material component and a component with embedded electronics. Section 2 describes the history and state of the art of multi-material CAD and manufacturing. Section 3 describes the methods by which we represent and compute multi-material geometries. Section 4 describes the algorithms we have defined for producing general manufacturing features and two exemplar algorithms that use these features to plan for specific manufacturing processes. Section 5 describes our implementation of one of these examples to create a multi-material component.

2 Background

The versatility of multi-material manufacturing is illustrated by a number of examples, including the addition of embedded reinforcement [1], integrated joints [2], embedded sensors [3], and embedded actuators [4]. The robotic hand presented by Dollar and Ma demonstrates the use of several of these techniques to produce a robust and low cost device that uses significantly less components that a similar device produced using conventional techniques [5, 6]. The hexapedal robot presented by Cham also demonstrates the use of these techniques to achieve similar benefits [7].

Kumar has presented a mathematical approach for representing and performing solid geometry operations on a multi-material part [8]. This work describes several possible methods for representing material data in a three-dimensional space and discusses basic material-aware solid geometry operations.

Various structures have been analyzed for computer storage and processing of geometry data combined with material data, including voxels, surfaces, and finite element meshes [9]. Within the area of voxel-based structures, the computational benefits and limitations have been considered and it is noted that although voxel-based models can be computationally intensive, they allow for conceptually simple geometric operations and correspond well to layer based methods for capturing and producing 3D objects [10, 11].

A number of methods for representing graded material properties in a multi-material part have been proposed [9, 12], and methods for generating material gradients based on functional requirements have been discussed [13–17]. Implementations of parts with graded properties have been approached through methods including voxel geometries with multiple discrete materials [4], adjustment of metallurgical properties in metal powder based additive processes [18] and the use of additive processes capable of adjusting the amount of deposited binder material [19].

Work towards the creation of manufacturing process planning algorithms has been done in the areas of SDM manufacturing [20–22], stereolithography [23], assembly sequences [24], and laminate-based robot construction [25]. Validation of generated subtractive manufacturing toolpaths using voxel-based simulation has also been considered [26].

The work on planning for laminate-based construction has been used in the development of a CAD tool, PopupCAD, that allows users to model laminate linkages and generate the cut files needed to produce them. Another software tool, Foundry, has been recently created to aid users in designing multi-material parts that can be produced using multi-material 3D printing [27–29]. This software provides a user-friendly interface that allows for the generation of various features and the modeling of a variety of material types.

This project seeks to combine elements of the prior research described above to create a new planning framework that is capable of automatically identifying the manufacturing steps and generating the required files to produce a user specified part using a series of different materials and processes. This framework will include a model representation supporting graded materials, a mathematical computation framework, and algorithms for process planning and validation. It is intended to apply to a variety of general processes, rather than being limited to a single process type. Finally, it is our intent to develop a user-friendly tool around this framework to make it accessible to a wide range of users.

3 Mathematical Framework

To enable process planning algorithms to be created, a mathematical framework for representing models and performing basic geometric and material operations on model data must first be defined. In addition, steps must be defined to check that the results of operations meet the established constraints of the representation to ensure that they have a valid physical meaning.

3.1 Model Representation

To process multi-material parts, a data structure is needed that can store both the 3D structure of an object as well as its material composition. For this, a voxel-based approach has been selected because it allows for simple definitions of solid geometry operations and can easily represent different materials within a part. A voxel model, represented here by $\boldsymbol{V}$, has been defined as a four-dimensional array where the first three dimensions represent the 3D space occupied by the model. The fourth dimension contains a vector representing the materials present at that voxel. Each material vector can be defined by

$$\boldsymbol{V}_{\langle i,j,k\rangle} = \langle a, m_0, ..., m_n \rangle, \tag{1}$$

where a is a boolean value indicating the presence or absence of material at the corresponding voxel, and $m_{0...n}$ are an arbitrary number of material channels that contain floating point values representing the percentages of materials present. The m_0 value represents a null material and is used to express voxels containing a material density of less than 100 percent. For example, a voxel with $a = 1$, $m_0 = 0.5$, $m_1 = 0.5$ will contain material 1 at 50 percent density. For an object containing n materials, these values are constrained by:

$$a \in \{0,1\} \tag{2}$$

$$m_{i\in\{0..n\}} \in [0,1] \tag{3}$$

$$\sum_{i\in\{0..n\}} m_i = a \tag{4}$$

Additionally, for the remainder of this paper the following assumptions are made:

1. All voxel models are of size $x \times y \times z \times (n+2)$
2. $\boldsymbol{V}^a$ refers to the $x \times y \times z$ boolean array containing the a value for each voxel in $\boldsymbol{V}$
3. $\boldsymbol{V}^m$ refers to the $x \times y \times z \times (n+1)$ array containing the $m_{0...n}$ values for each voxel in $\boldsymbol{V}$
4. $\boldsymbol{V}^m_i$ refers to the $x \times y \times z$ array containing the m_i value for each voxel in $\boldsymbol{V}$

3.2 Constructive Solid Geometry Emulating Operations

To process model data, basic solid geometry operations for voxels must first be defined. These operations are based on the work described in [8] with the addition of material-specific and morphological operations.

As defined below, these operations are restricted to acting on model arrays of the same dimension and size, represented here by $\boldsymbol{A}$ and $\boldsymbol{B}$, and must account for both the geometry and material of voxels. If two models with different physical sizes are used, empty voxels must be added around the models as needed to make their array sizes match. Material property vectors are required to be the same length.

The difference of two models is found by first using logical AND ($\wedge$) and NOT ($\neg$) operations on $\boldsymbol{A}^a$ and $\boldsymbol{B}^a$ to obtain a boolean array of the voxels which will contain material in the output. Element-wise multiplication, represented by $\circ$, is then performed between this array and each material channel to determine the values of the material channels in the output model. The resulting model is then found by augmenting these two matrices. This operation can be defined as follows:

$$\boldsymbol{A} \setminus \boldsymbol{B} := \left[\boldsymbol{A}^a \wedge \neg \boldsymbol{B}^a \middle| \left(\boldsymbol{A}^m_i \circ (\boldsymbol{A}^a \wedge \neg \boldsymbol{B}^a) \right)_{i\in\{0..n\}} \right] \tag{5}$$

The intersection operation can be defined in a similar manner. In this case, $\cap^L$ and $\cap^R$ denote whether the material of the output is taken from the first or second input model and are defined by:

$$\boldsymbol{A} \cap^L \boldsymbol{B} := \left[\boldsymbol{A}^a \wedge \boldsymbol{B}^a \middle| \left(\boldsymbol{A}^m_i \circ (\boldsymbol{A}^a \wedge \boldsymbol{B}^a) \right)_{i\in\{0..n\}} \right] \tag{6}$$

$$\boldsymbol{A} \cap^R \boldsymbol{B} := \left[\boldsymbol{A}^a \wedge \boldsymbol{B}^a \middle| \left(\boldsymbol{B}^m_i \circ (\boldsymbol{A}^a \wedge \boldsymbol{B}^a) \right)_{i\in\{0..n\}} \right] \tag{7}$$

The union operation can be defined using the symmetric difference combined with an intersection operation. The material used in overlapping regions is determined by the use of $\cap^L$ or $\cap^R$. The definition using $\cap^L$ is as follows:

$$\boldsymbol{A} \cup^L \boldsymbol{B} := (\boldsymbol{A} \setminus \boldsymbol{B}) + (\boldsymbol{B} \setminus \boldsymbol{A}) + (\boldsymbol{A} \cap^L \boldsymbol{B}) \tag{8}$$

These operations use the material from either model $\boldsymbol{A}$ or model $\boldsymbol{B}$. To compute mixtures of materials, operations for addition, subtraction, and multiplication are used. These are defined by applying a logical OR ($\vee$) operation on $\boldsymbol{A}^a$ and $\boldsymbol{B}^a$, and array

operations on $\boldsymbol{A}^m$ and $\boldsymbol{B}^m$:

$$\boldsymbol{A} +^* \boldsymbol{B} := \left[\boldsymbol{A}^a \vee \boldsymbol{B}^a \middle| \boldsymbol{A}^m + \boldsymbol{B}^m\right] \tag{9}$$

$$\boldsymbol{A} -^* \boldsymbol{B} := \left[\boldsymbol{A}^a \vee \boldsymbol{B}^a \middle| \boldsymbol{A}^m - \boldsymbol{B}^m\right] \tag{10}$$

$$c\boldsymbol{A} := \left[\boldsymbol{A}^a \middle| c\boldsymbol{A}^m\right] \tag{11}$$

The final two operations required are dilation and erosion. These are defined by applying the corresponding binary and grayscale morphological operations across each of the material channels in $\boldsymbol{A}$ using a three-dimensional structuring element S, as illustrated below with dilation:

$$\boldsymbol{A} \oplus S := \left[\boldsymbol{A}^a \oplus_{bin} S \middle| (\boldsymbol{A}_i^m \oplus_{gray} S)_{i\in\{0..n\}}\right] \tag{12}$$

Erosion ($\boldsymbol{A} \ominus S$) can be defined in a similar manner.

These operations are used in the creation of manufacturing feature generation algorithms and process generation procedures as discussed in Section 4. It should be noted that some functions may result in voxels that do not meet the constraints established in Eqn. (3) and Eqn. (4). This must be corrected before using the result to generate final manufacturing steps with a cleanup process, as discussed in Section 3.4.

3.3 Material Interface Modification

The basic operations described in the Section 3.2 will maintain a defined boundary between materials. This behavior is consistent with most common manufacturing processes. However, some processes such as 3D printing provide opportunities for creating a precise gradient of material properties within a part via variable infill density or multi-material 3D printing. To permit these capabilities to be leveraged at design time, a blur operation can be defined using the convolution of a three-dimensional Gaussian kernel K and each of the material channels in $\boldsymbol{A}$:

$$Blur(\boldsymbol{A}, K) := \left[\boldsymbol{A}^a \middle| \left((\boldsymbol{A}_i^m * K) \circ \boldsymbol{A}^a\right)_{i\in\{0..n\}}\right] \tag{13}$$

The radius of blurring is determined by the size of K. When applied in this manner, the blur operation will affect all voxels that contain material and that are near a boundary between two materials or between a material and empty space. The blurring effect can be restricted to a region R using

$$BlurRegion(\boldsymbol{A}, K, \boldsymbol{R}) := \left(Blur(\boldsymbol{A}, K) \cap^L \boldsymbol{R}\right) \cup^L \boldsymbol{A}, \tag{14}$$

where $\boldsymbol{R}$ is a voxel model such that $\boldsymbol{R}^a$ indicates which voxels blurring should be applied to. This model must have material vectors present, but the values in them will be ignored. As with the functions in the Section 3.2, the blurring operations may result in voxels that do not meet the constraints established in Eqn. (3) and Eqn. (4). This must be corrected before using the result to generate final manufacturing steps, as described below.

3.4 Cleanup Steps

The geometric output from some solid geometry and blurring operations can include voxels with $m_{0\ldots n}$ elements that do not meet the constraints of being in the interval $[0,1]$ and having a sum equal to a as established in Eqn. (3) and Eqn. (4). To make all voxels follow these constraints, the following steps must be taken for each voxel in the model:

1. Remove negative m values
2. Update a values
3. Scale m values

Negative material values can be removed from a voxel as follows:

$$m_i' = \begin{cases} m_i & m_i > 0 \\ 0 & m_i \leq 0 \end{cases} \tag{15}$$

The new a value for the voxel can then be found using

$$a' = \begin{cases} 1 & \Sigma m \neq 0 \\ 0 & \Sigma m = 0 \end{cases}, \text{ where:} \tag{16}$$

$$\Sigma m = \sum_{i\in\{0..n\}} m_i. \tag{17}$$

Two approaches can be taken to scale the material values in order to satisfy Eqn. (3) and Eqn. (4). The first is to maintain the ratio between all materials including the null material. New $m_{0\ldots n}$ values can be found for a voxel using

$$m_i' = \begin{cases} \frac{m_i}{\Sigma m} & a = 1 \\ 0 & a = 0 \end{cases}. \tag{18}$$

The second option is to adjust the value of the null material without changing any other material values. This can be done using

$$m_0' = \begin{cases} 1 - (\Sigma m - m_0) & a = 1 \\ 0 & a = 0 \end{cases}. \tag{19}$$

Note that the second option only works for voxels with $(\Sigma m - m_0) <= 1$.

4 Manufacturing Algorithms

Using the solid geometry operations discussed in the previous section, algorithms can be created to generate various geometric features needed to plan for manufacturing. These features can then be used in combination with the solid geometry operations to create procedures to generate files for specific materials and manufacturing processes.

4.1 Manufacturing Feature Generation Algorithms

A number of general-purpose features can be defined that are useful in the planning of manufacturing steps, including keep-out, clearance, support, and web. These features are influenced by a number of factors, including the ability of a process to control cutting depth, the tool head size, and the length/kerf/orientation of the cutting tool. As a result, different versions of these operations must be defined for various manufacturing processes depending on constraints and limitations of each process. These features were selected based on the work described in [25] and have been adapted to a voxel-based model representation.

The algorithms presented in this section support four general types of processes: milling, laser machining, FDM 3D printing, and discrete component insertion. In addition, it is assumed that any addition or subtraction of material occurs along the Z axis and that there are no head size or kerf constraints. These algorithms return a region that does not directly correspond to a specific material. As such, occupied voxels in the returned models contain a vector of ones for ease of implementation.

To illustrate the function of the presented algorithms, the sample object shown in Fig. 1 is used. This object includes multiple overhangs and a center cavity. The origin point is considered to be at the front-bottom-left corner of the model and the coordinate axes are aligned as indicated in the figure.

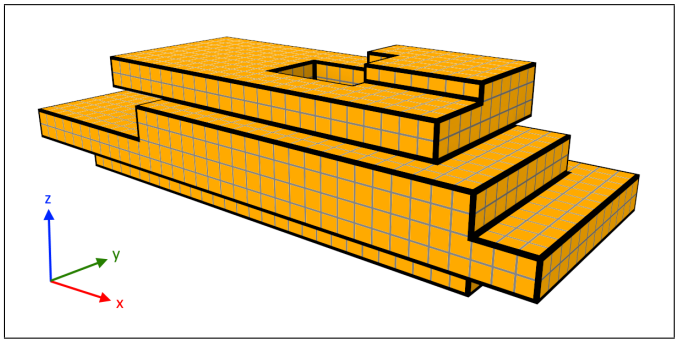


FIGURE 1: Sample object to illustrate the function of various manufacturing feature generation algorithms

4.1.1 Projection

To aid in the definition of various manufacturing features, a projection function is first defined in Algorithm 1. This function returns a model that includes all voxels within the workspace that contain material or that lie in the specified direction with respect to a voxel that contains material. Note that a Z value of 1 corresponds to the bottom layer of the model.

Algorithm 1 Model Projection

function PROJECTZ($\boldsymbol{V}, direction$)
  for all $\langle i,j,k\rangle \in \langle\{1..x\},\{1..y\},\{1..z\}\rangle$ **do**
    $$\Sigma_{col} \leftarrow \begin{cases} \sum_{K\in\{1..k\}} \boldsymbol{V}_{\langle i,j,K,1\rangle} & direction = up \\ \sum_{K\in\{k..z\}} \boldsymbol{V}_{\langle i,j,K,1\rangle} & direction = down \\ \sum_{K\in\{1..z\}} \boldsymbol{V}_{\langle i,j,K,1\rangle} & direction = both \end{cases}$$
    $$\boldsymbol{P}_{\langle i,j,k\rangle} \leftarrow \begin{cases} 1_{(n+2)} & \Sigma_{col} \neq 0 \\ 0_{(n+2)} & \Sigma_{col} = 0 \end{cases}$$
  end for
  return $\boldsymbol{P}$
end function

4.1.2 Keep-out

The keep-out region for a given process and part represents material which the process may not modify while creating the part. This feature primarily applies to subtractive processes. It includes material that will be present in the final part and regions of the workspace that cannot be accessed without affecting this material. In general, additive processes will have no keep-out region because they deposit material from the bottom up. Methods to find keep-out are defined in Algorithm 2. The results from applying this algorithm to the example model are shown in Fig. 2.

Algorithm 2 Keep-out Generation

function KEEPOUT($\boldsymbol{V}, method$)
  if $method = laser$ **then**
    $\boldsymbol{K}^L \leftarrow$ PROJECT($\boldsymbol{V}, both$)
  else if $method = mill$ **then**
    $\boldsymbol{K}^L \leftarrow$ PROJECT($\boldsymbol{V}, down$)
  end if
  return $\boldsymbol{K}$
end function

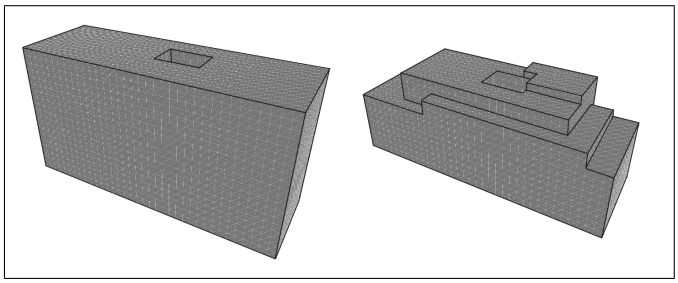


FIGURE 2: Keep-out regions for laser machining (left) and milling (right)

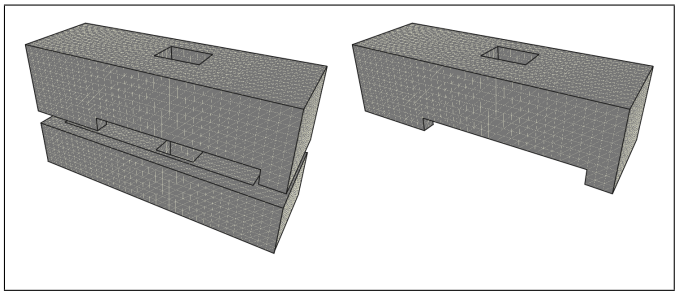


FIGURE 3: Clearance regions for laser machining (left) and milling (right). 3D printing will have a clearance region similar to that for milling, but that also includes regions under overhangs.

4.1.3 Clearance The clearance region for a given process and part represents regions that will be affected by the process acting on the part. Methods to find clearance are defined in Algorithm 3. The results from these functions are shown in Fig. 3.

Clearance can be used to identify regions of a model $\boldsymbol{A}$ that conflict with the manufacturing of another model $\boldsymbol{B}$ using

$$\boldsymbol{A}^{conflict} = \boldsymbol{A} \cap^{L} \textsc{Clearance}(\boldsymbol{B}). \tag{20}$$

If desired, clearance can be used to modify $\boldsymbol{A}$ to prevent the conflict from occurring. The modified model, $\boldsymbol{A}'$, is found using

$$\boldsymbol{A}' = \boldsymbol{A} \setminus \textsc{Clearance}(\boldsymbol{B}). \tag{21}$$

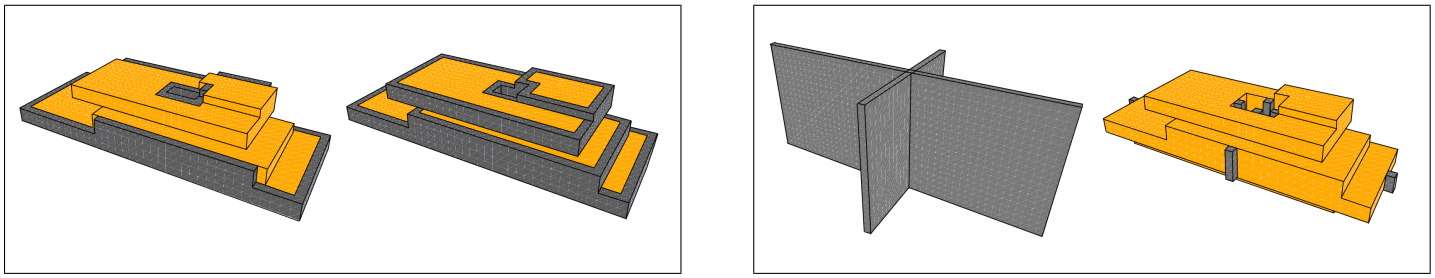
4.1.4 Support For some parts, support material is needed to hold the part in place during manufacturing. The regions where support material may be added to an object are characterized by the process that is used to remove the supports. Once these regions are determined, they can be used to generate valid supports based on a desired support design. Methods to find the supportable regions for planar support structures and to generate support based on a user supplied support model are defined in Algorithm 4. The parameter r_1 is used to determine areas where support is ineffective based on proximity to empty regions that are inaccessible to the removal process. The parameter r_2 represents the desired thickness of the support material. Example results from these functions are shown in Fig. 4.

Algorithm 3 Clearance Generation

function CLEARANCE($\boldsymbol{V}, method$)
  $\boldsymbol{P}^U \leftarrow$ PROJECT($\boldsymbol{V}, up$)
  $\boldsymbol{P}^D \leftarrow$ PROJECT($\boldsymbol{V}, down$)
  $\boldsymbol{P}^B \leftarrow$ PROJECT($\boldsymbol{V}, both$)
  if $method = laser$ **then**
    $\boldsymbol{C} \leftarrow \boldsymbol{P}^B \setminus \boldsymbol{V}$
  else if $method = mill$ **then**
    $\boldsymbol{C} \leftarrow \boldsymbol{P}^B \setminus \boldsymbol{P}^D$
  else if $(method = 3dp) \vee (method = ins)$ **then**
    $\boldsymbol{C} \leftarrow \boldsymbol{P}^U$
  end if
  return $\boldsymbol{C}$
end function

Algorithm 4 Support Generation

function SUPPORT($\boldsymbol{V}, method, r_1, r_2$)
  $d_1 \leftarrow 2r_1 + 1$
  $d_2 \leftarrow 2r_2 + 1$
  $S_1 \leftarrow J_{d_1,d_1,1}$ ▷ Square structuring elements
  $S_2 \leftarrow J_{d_2,d_2,1}$ ▷ for dilation in X and Y
  $\boldsymbol{A} \leftarrow$ KEEPOUT($\boldsymbol{V}, method$)
  $\boldsymbol{B} \leftarrow \boldsymbol{A} \setminus \boldsymbol{V}$
  $\boldsymbol{C} \leftarrow \boldsymbol{B} \oplus S_1$ ▷ Regions where support is ineffective
  $\boldsymbol{D} \leftarrow \boldsymbol{A} \oplus S_2$
  $\boldsymbol{E} \leftarrow \boldsymbol{D} \setminus \boldsymbol{A}$ ▷ Accessible region of thickness r_1
  $\boldsymbol{S} \leftarrow \boldsymbol{E} \setminus \boldsymbol{C}$
  return $\boldsymbol{S}$
end function
function USERSUPPORT($\boldsymbol{V}, \boldsymbol{U}, method, r_1, r_2$)
  $\boldsymbol{S} \leftarrow$ SUPPORT($\boldsymbol{V}, method, r_1, r_2$)
  $\boldsymbol{U}' \leftarrow \boldsymbol{U} \cap^{L} \boldsymbol{S}$
  return $\boldsymbol{U}'$
end function



(a) (b)

FIGURE 4: (a) Supportable regions (shown in dark gray) for removal by laser machining (left) and milling (right). (b) User-designed support region (left) and supports generated from the design (right).

4.1.5 Web Web represents scrap material that surrounds a device. It can be used in the creation of supports or layer alignment fixtures. The web for a given removal process can be found using the process defined in Algorithm 5. The parameters r_1 and r_2 represent the distance from the surface of the part to the inside of the web and the width of the web, respectively. An example result is shown in Fig. 5.

Algorithm 5 Web Generation

function WEB($\boldsymbol{V}, method, r_1, r_2$)
  $d_1 \leftarrow 2r_1 + 1$
  $d_2 \leftarrow 2r_2 + 1$
  $S_1 \leftarrow J_{d_1,d_1,1}$ ▷ Square structuring elements
  $S_2 \leftarrow J_{d_2,d_2,1}$ ▷ for dilation in X and Y
  $\boldsymbol{A} \leftarrow$ KEEPOUT($\boldsymbol{V}, method$)
  $\boldsymbol{B} \leftarrow \boldsymbol{A} \oplus S_1$
  $\boldsymbol{C} \leftarrow \boldsymbol{B} \oplus S_2$
  $\boldsymbol{D} \leftarrow$ (minimum bounding box of $\boldsymbol{C}$)
  $\boldsymbol{W} \leftarrow \boldsymbol{D} \setminus \boldsymbol{B}$
  return $\boldsymbol{W}$
end function

4.2 Exemplar Process Generation Procedures

Using the algorithms defined in the previous sections, procedures can be created to generate the files needed to manufacture a multi-material part. This is demonstrated through manually created procedures for multi-material 3D printing with a blurred region and for 3D printing with a discrete component inserted during the printing process. These procedures are representative of two simple applications and show the utility of the framework. They only make use of a portion of the above operations and algorithms. The remaining operations and algorithms have applications for planning processes for other classes of parts, such as those that make use of subtractive processes or support structures.

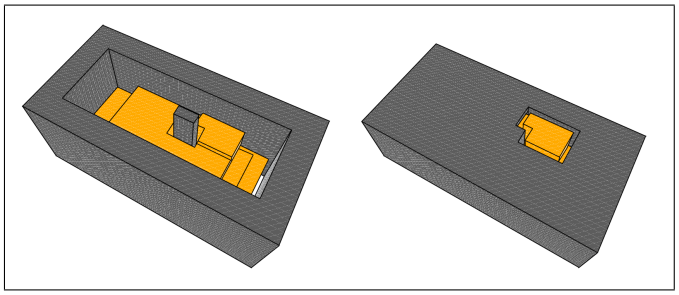


FIGURE 5: Web (shown in dark gray) for removal by laser machining (left) and milling (right)

4.2.1 Multi-material 3D Printing with a Blurred Region Multi-material 3d printing allows for the creation of models that contain a gradient of materials. The operations discussed in Section 3.3 can be used to automatically generate gradient material transitions given an input model containing two distinct materials. The procedure shown in Algorithm 6 will modify the input model with blurring along any material interfaces, reduce the number of generated materials by grouping materials with similar properties, then export a series of stl files for each distinct generated material. An example input model and the corresponding output are shown in Fig. 6. To create this part, the user would execute the following process:

1. Load generated stl files in multi-material 3D printer software
2. Assign correct material properties to each file
3. Print part
4. Remove part from printer

To create this part manually, a user would be required to individually create a separate model for each material step in

the graded area. For a model containing graded areas with a large number of steps or gradients along complex boundaries, this manual workflow would not be practical.

Algorithm 6 Generation of Model and Files for 3D Printing with a Blurred Region

$\boldsymbol{V} \leftarrow$ User created model
$K \leftarrow$ Gaussian kernel of radius r
$\boldsymbol{V}' \leftarrow Blur(\boldsymbol{V}, K)$
$\boldsymbol{V}'' \leftarrow$ Reduce material count
$L \leftarrow$ List of distinct materials in $\boldsymbol{V}''$
for all $i \in \{1..|L|\}$ **do**
    $\boldsymbol{A}_i \leftarrow$ Voxels in $\boldsymbol{V}''$ containing material L_i
    Export($\boldsymbol{A}_i$)
end for

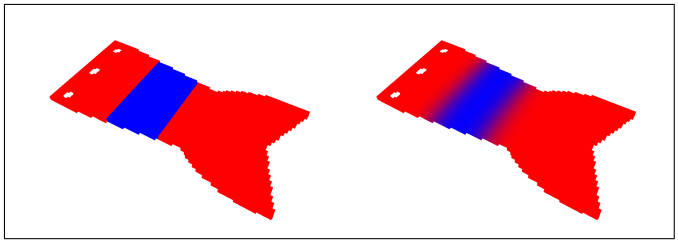


FIGURE 6: Example input model (left) and output model including blurred material transitions (right)

4.2.2 3D Printing with an Inserted Component

One unique capability of multi-material manufacturing is the insertion of a discrete component into a 3D printed object partway though the printing process. Parts added in this way have the advantage of being securely mounted and fully protected. To generate the files to produce this type of part, the procedure shown in Algorithm 7 can be used. This procedure will add any required clearances to the part, generate printer g-code, then add pauses to the g-code at layers where a part must be inserted. To create the part, the user would execute the following process:

1. Load generated g-code file on 3D printer
2. Start printing
3. When printing pauses, insert correct component
4. Resume printing
5. Repeat 3-4 until printing is complete
6. Remove part from printer

The results of this process are discussed further in Section 5.

Algorithm 7 Generation of Model and Files for 3D Printing with an Inserted Component

$\boldsymbol{V} \leftarrow$ User created model
$\boldsymbol{I} \leftarrow$ Inserted components in $\boldsymbol{V}$
$\boldsymbol{P} \leftarrow$ Printed components in $\boldsymbol{V}$
$L \leftarrow \{\}$ ▷ Empty list to store pause layers
$\boldsymbol{I}^S \leftarrow \boldsymbol{I} \oplus J_{3,3,3}$ ▷ Add spacing around $\boldsymbol{I}$
$\boldsymbol{I}^C \leftarrow$ CLEARANCE($\boldsymbol{I}^S, ins$) ▷ Add path to insert $\boldsymbol{I}$
for all $k \in \{1..z\}$ **do**
    if $\Sigma \boldsymbol{I}^S_{\langle:,:,k,:\rangle} = 0$ **then** ▷ If $\boldsymbol{I}^S$ is not in layer k
        $\boldsymbol{I}^C_{\langle:,:,k,:\rangle} =$ Zeros ▷ Remove layer k from $\boldsymbol{I}^C$
        if $\Sigma \boldsymbol{I}^S_{\langle:,:,(k-1),:\rangle} > 0$ **then** ▷ If $\boldsymbol{I}^S$ is in layer $k-1$
            Add $(k-1)$ to L ▷ Save as a top layer of part
        end if
    end if
end for
$\boldsymbol{P}' \leftarrow \boldsymbol{P} \setminus \boldsymbol{I}^C$ ▷ Apply spacing and clearance to $\boldsymbol{P}$
$G \leftarrow$ Slice($\boldsymbol{P}'$) ▷ Generate g-code
for all $l \in L$ **do**
    AddPause(G, l) ▷ Add pauses at top layers of $\boldsymbol{I}$
end for
Export(G) ▷ Export final g-code

5 Implementation

To illustrate the utility of the framework introduced in this paper, the operations and algorithms discussed in the previous sections were implemented using Python. Two sample parts that used discrete components embedded in 3D printed objects were then created based the process presented in Section 4.2.2.

5.1 Workflow

The workflow used to execute Algorithm 7 is shown in Fig. 7. In addition to the Python implementation of the framework presented in this paper, two additional software packages are required by this workflow. The first, represented by P1, is a program to generate models in a voxel file format and the second, represented by P3, is a slicing program for the 3D printer used. MagicaVoxel [30] and Cura [31] were used for these steps. Various Python libraries were used to facilitate the import and export of the data types required.

5.2 Fabrication of Example Components

Figure 8 shows two exemplar parts that were selected for fabrication. These models were created in MagicaVoxel using a voxel size of 1 mm. They were then processed using Algorithm 7 and created using the steps in Section 4.2.2. The manufacturing

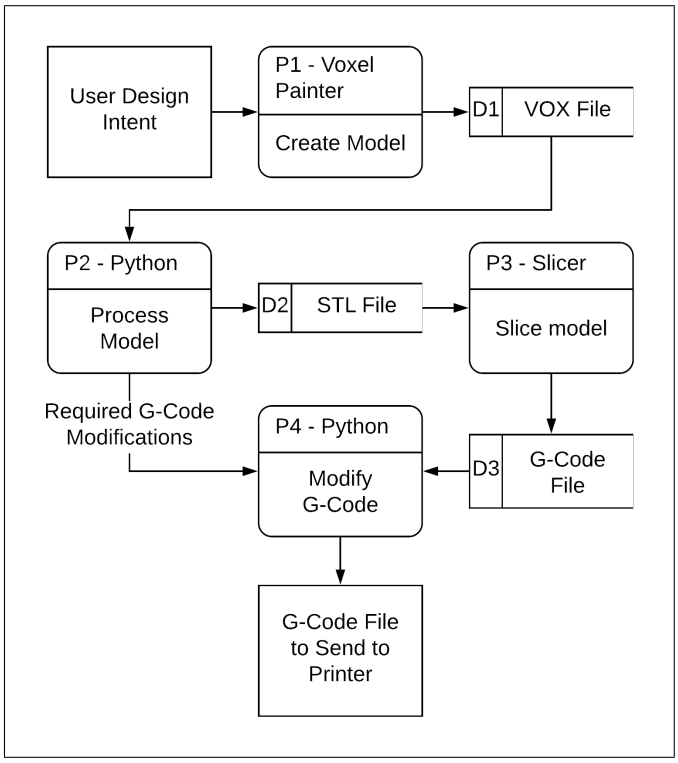
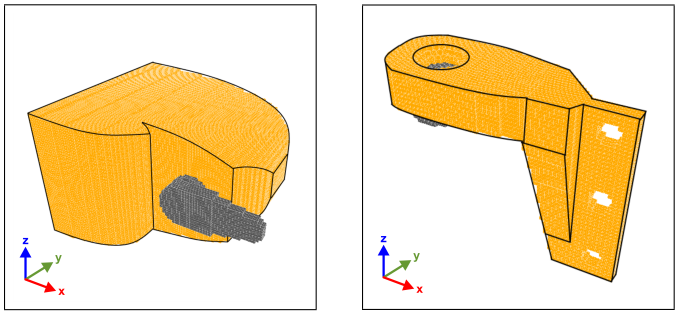


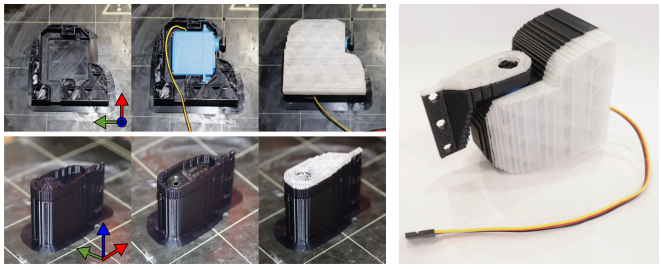
FIGURE 7: Data flow diagram for processing a 3D printed body with an embedded component



(a) (b)

FIGURE 8: Example model designs consisting of (a) a servo motor embedded in a robot body and (b) a servo horn embedded in a mounting arm

process types used by these steps include 3D printing, for which a Prusa i3 MK2S [32] was used, and discrete component insertion. The execution of these steps for the two components is shown in Fig. 9a.



(a) (b)

FIGURE 9: (a) Steps taken in the execution of the fabrication process (from left to right): print until pause, insert component, finish printing. (b) Completed and assembled components. The black and white materials were used to highlight the first and second printing stages and are not representative of a true multi-material computation.

5.3 Implementation Results

The final manufactured parts functioned as intended and are shown assembled in Fig. 9b. Since these parts fall in the same class of components, the same processing algorithm can be used for both with no modifications. To produce these parts manually, the user would have been required to design in all necessary clearances, identify the exact printing layers where the embedded components needed to be inserted, and then modify the g-code by hand to pause at these layers.

A voxel size of 1 mm was identified as providing a good balance between model resolution and processing time. However, due to the precision limitations when rounding dimensions to the nearest millimeter, some cavities in the printed parts were larger than expected. This caused the inserted components to have a loose fit inside of the printed parts. It was also observed that inserted components without a flat top surface, such as the tapered servo horn, could cause the model generation algorithm to produce a model that allowed the component to move or that was missing part of the top layer used to hold the component in place.

Loose fitting parts could be addressed by the use of non-uniform voxel sizes to allow smaller voxels to be placed around inserted components, or by the generation of support features that the user can trim down until a snug fit is achieved. The generation errors for the part without a flat top surface were corrected for this example by manually rotating the part to a more suitable orientation. This could be addressed more generally by the addition of an algorithm that automatically attempts to find the optimal part orientation.

6 Conclusion and Future Work

This project is seeking to make multi-material manufacturing more accessible through the creation of a framework that will allow for the automatic generation of manufacturing steps and files. This paper describes the first steps towards the creation of this framework and illustrates its utility through an example implementation. The results from implementing the framework and using it to create a multi-material component show that the framework can be used effectively to process 3D models and generate the files needed to produce them.

Future work will be towards expanding the framework with additional algorithms and using it in the creation of a software tool that will automate the planning of multi-material manufacturing processes. This tool is intended to automatically determine the best process for producing a part and then provide the user with the process steps in addition to the required files. It should also be able to identify designs that cannot be produced and notify the user of the problem areas. Other areas for future work include adding the ability to import models from standard CAD file formats and optimizing the implementation to permit higher model detail and reduced processing time.

REFERENCES

[1] Belter, J. T., and Dollar, A. M., 2015. "Strengthening of 3d printed fused deposition manufactured parts using the fill compositing technique". *PLOS ONE,* **10**(4), 04, pp. 1–19.

[2] Ma, R., Belter, J. T., and Dollar, A. M., 2015. "Hybrid deposition manufacturing: Design strategies for multimaterial mechanisms via three-dimensional printing and material deposition". *Journal of Mechanisms and Robotics,* **7**, 05, p. 021002.

[3] Dollar, A. M., Wagner, C. R., and Howe, R. D., 2006. "Embedded sensors for biomimetic robotics via shape deposition manufacturing". In The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006., pp. 763–768.

[4] Hiller, J., and Lipson, H., 2012. "Automatic design and manufacture of soft robots". *IEEE Transactions on Robotics,* **28**(2), pp. 457–466.

[5] Dollar, A. M., and Howe, R. D., 2010. "The Highly Adaptive SDM Hand: Design and Performance Evaluation". *The International Journal of Robotics Research,* **29**(5), feb, pp. 585–597.

[6] Ma, R. R., Odhner, L. U., and Dollar, A. M., 2013. "A Modular, Open-Source 3D Printed Underactuated Hand". In 2013 IEEE International Conference on Robotics and Automation(preprint), IEEE.

[7] Cham, J. G., Bailey, S. A., Clark, J. E., Full, R. J., and Cutkosky, M. R., 2002. "Fast and Robust: Hexapedal Robots via Shape Deposition Manufacturing". *The International Journal of Robotics Research,* **21**(10-11), oct, pp. 869–882.

[8] Kumar, V., and Dutta, D., 1997. "An approach to modeling multi-material objects". *Proceedings of the fourth ACM symposium on Solid modeling and applications - SMA '97*, pp. 336–345.

[9] Jackson, T. D., 2000. "Analysis of FGM Object Representation Methods". PhD thesis, Massachusetts Institute of Technology.

[10] Jense, G., 1989. "Voxel-based methods for cad". *Computer-Aided Design,* **21**, 10, pp. 528–533.

[11] Chandru, V., Manohar, S., and Prakash, C. E., 1995. "Voxel-based modeling for layered manufacturing". *IEEE Computer Graphics and Applications,* **15**(6), Nov, pp. 42–47.

[12] Kou, X. Y., and Tan, S. T., 2007. "Heterogeneous object modeling: A review". *CAD Computer Aided Design,* **39**(4), pp. 284–301.

[13] Qian, X., and Dutta, D., 2004. "Feature-based design for heterogeneous objects". *CAD Computer Aided Design,* **36**(12), pp. 1263–1278.

[14] Samanta, K., and Koc, B., 2005. "Feature-based design and material blending for free-form heterogeneous object modeling". *CAD Computer Aided Design,* **37**(3), pp. 287–305.

[15] Wang, S., Chen, N., Chen, C. S., and Zhu, X., 2009. "Finite element-based approach to modeling heterogeneous objects". *Finite Elements in Analysis and Design,* **45**(8-9), pp. 592–596.

[16] Gupta, V., Kasana, K. S., and Tandon, P., 2010. "Computer Aided Design Modeling for Heterogeneous Objects". *IJCSI International Journal of Computer Science Issues,* **7**(2), pp. 31–38.

[17] Hiller, J. D., and Lipson, H., 2009. "Design Automation for Multi-Material Printing". *20th Annual International Solid Freeform Fabrication Symposium*, pp. 279–287.

[18] Kieback, B., Neubrand, A., and Riedel, H., 2003. "Processing techniques for functionally graded materials". *Materials Science and Engineering A,* **362**(1-2), pp. 81–105.

[19] Chiu, W. K., and Yu, K. M., 2008. "Direct digital manufacturing of three-dimensional functionally graded material objects". *CAD Computer Aided Design,* **40**(12), pp. 1080–1093.

[20] Ramaswami, K., and Prinz, F., 1997. "Process planning for shape deposition manufacturing".

[21] Binnard, M., 1999. *Design by Composition for Rapid Prototyping*. Springer US, Boston, MA.

[22] Hatanaka, M., and Cutkosky, M., 2003. "Process planning for embedding flexible materials in multi-material prototypes".

[23] Kim, H., Choi, J., and Wicker, R., 2010. "Scheduling and

process planning for multiple material stereolithography". *Rapid Prototyping Journal,* **16**(4), pp. 232–240.

[24] Jiménez, P., 2013. "Survey on assembly sequencing: a combinatorial and geometrical perspective". *Journal of Intelligent Manufacturing,* **24**(2), apr, pp. 235–250.

[25] Aukes, D. M., Goldberg, B., Cutkosky, M. R., and Wood, R. J., 2014. "An analytic framework for developing inherently-manufacturable pop-up laminate devices". *Smart Materials and Structures,* **23**(9), aug, p. 094013.

[26] Jang, D., Kim, K., and Jung, J., 2000. "Voxel-based virtual multi-axis machining".

[27] Chen, D., Levin, D. I. W., Didyk, P., Sitthi-amorn, P., and Matusik, W., 2013. "Spec2fab: a reducer-tuner model for translating specifications to 3d prints". *ACM Trans. Graph.,* **32**, pp. 135:1–135:10.

[28] Wang, S.-P., Ragan-Kelley, J., Matusik, W., and Vidimce, K., 2014. "Openfab: A programmable pipeline for multi-material fabrication". *ACM Transactions on Graphics,* **32**, 09.

[29] Vidimce, K., Kaspar, A., Wang, Y., and Matusik, W., 2016. "Foundry : Hierarchical Material Design for Multi-Material Fabrication". *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16*, p. 12.

[30] ephtracy. MagicaVoxel. `https://ephtracy.github.io`. Accessed: 2019-05-22.

[31] Ultimaker BV. Ultimaker Cura software. `https://ultimaker.com/en/products/ultimaker-cura-software`. Accessed: 2019-05-22.

[32] Prusa Research s.r.o. Original Prusa i3 MK2S Kit. `https://shop.prusa3d.com/en/3d-printers/59-original-prusa-i3-mk2-kit.html`. Accessed: 2019-05-23.