

ORIGINAL ARTICLE

Open Access



# Development of Texture Mapping Approaches for Additively Manufacturable Surfaces

Bhupesh Verma<sup>\*</sup> , Omid Zarei, Song Zhang and Johannes Henrich Schleifenbaum

## Abstract

Additive manufacturing (AM) technologies have been recognized for their capability to build complex components and hence have offered more freedom to designers for a long time. The ability to directly use a computer-aided design (CAD) model has allowed for fabricating and realizing complicated components, monolithic design, reducing the number of components in an assembly, decreasing time to market, and adding performance or comfort-enhancing functionalities. One of the features that can be introduced for boosting a component functionality using AM is the inclusion of surface texture on a given component. This inclusion is usually a difficult task as creating a CAD model resolving fine details of a given texture is difficult even using commercial software packages. This paper develops a methodology to include texture directly on the CAD model of a target surface using a patch-based sampling texture synthesis algorithm, which can be manufactured using AM. Input for the texture generation algorithm can be either a physical sample or an image with heightmap information. The heightmap information from a physical sample can be obtained by 3D scanning the sample and using the information from the acquired point cloud. After obtaining the required inputs, the patches are sampled for texture generation according to non-parametric estimation of the local conditional Markov random field (MRF) density function, which helps avoid mismatched features across the patch boundaries. While generating the texture, a design constraint to ensure AM producibility is considered, which is essential when manufacturing a component using, e.g., Fused Deposition Melting (FDM) or Laser Powder Bed Fusion (LPBF). The generated texture is then mapped onto the surface using the developed distance and angle preserving mapping algorithms. The implemented algorithms can be used to map the generated texture onto a mathematically defined surface. This paper maps the textures onto flat, curved, and sinusoidal surfaces for illustration. After the texture mapping, a stereolithography (STL) model is generated with the desired texture on the target surface. The generated STL model is printed using FDM technology as a final step.

**Keywords:** Texture synthesis, Design for additive manufacturing, Image processing, Textured surface generation, Texture mapping

## 1 Introduction

Surface textures are used in various engineering components to bestow a specific functionality to a target surface. Functionality such as binding, attaching, gripping, or heat transfer can be added to a component, among

other possibilities, by tuning the basic texture parameters such as bulk porosity, pore, and feature size [1]. The influence of surface texture, for length scales ranging from molecular levels to centimeters, on various surfaces involved in multiphase processes, e.g., boiling, condensation, freezing, etc., can be found in existing literature [2]. Combining different capabilities of engineering interest into a single component helps decrease the number of components in an assembly and the number of steps

\*Correspondence: [bhupesh.verma@dap.rwth-aachen.de](mailto:bhupesh.verma@dap.rwth-aachen.de)

Chair for Digital Additive Production, RWTH Aachen University, 52074 Aachen, Germany

required for manufacturing [1]. Despite these possible advantages, the generation of texture on a surface and its manufacturing is challenging. Various algorithms for texture synthesis [3] and manufacturing techniques have been proposed in the literature to overcome this issue.

Texture synthesis, or texture generation algorithm, is a very active field of research, especially in computer graphics. Textures can be classified into two categories when considered only as images: structural and statistical textures [4]. A structural texture consists of primitives with relocations based on replacement rules, such as scaling, rotation, translation, and reflection. Statistical textures, on the other hand, are formed based on probabilistic models, such as fractal [5], time series [6], and random field models [7]. For generating 2D textures on surfaces, Turk [8] developed a texture synthesis method using the image pyramids to synthesize the texture directly on the model's surface. Wei et al. [9] introduced another generic algorithm to synthesize textures over arbitrary manifold surfaces defined by dense polygon meshes. Magda et al. [10] developed a fully automatic yet user-controllable algorithm without modifying the original mesh to achieve texture generation on arbitrary meshes efficiently. Furthermore, machine learning (ML) algorithms have also been used in some recent studies to generate texture by image-based learning methods [11–13]. These studies have addressed texture generation for either only graphical applications or have not considered setting up a 3D model for manufacturing using additive manufacturing (AM) technology.

A review article published regarding the classification, advantages, and disadvantages of manufacturing techniques for surface texture shows that the selection of fabrication technology is critical and depends on the application to ensure desired texture dimension and precision [14]. In order to make an informed decision, considering the current state-of-the-art manufacturing technologies is essential. With the rapid development in AM technology, it becomes possible to realize the manufacturing of components with complex designs. American Society for Testing Materials (ASTM) has defined AM as "a process of joining materials to make objects from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing methodologies. Synonyms: additive fabrication, additive processes, additive techniques, additive layer manufacturing, layer manufacturing, and freeform fabrication" [15]. This layer-by-layer technology enables the manufacturing of intricate free form surfaces that are usually problematic to manufacture, if possible, using conventional manufacturing (CM) technologies. Although in addition to exploiting the advantages of AM processes, it is necessary to consider the manufacturing capabilities and

manufacturing constraints [16]. The best practice in design for AM is to have the minimum support structures since it can tremendously decrease both manufacturing and post-processing time. Support structure minimization can be carried out with the adaptation of the design to reduce overhang features considering a specific orientation of the component on the build platform. The topic of design for AM (DfAM) has gained significant interest in the last few years, and several studies have been done on this topic [17–19]. Due to the design freedom provided by AM, it is attractive to manufacture complex textured surfaces. A recently published study by Armillotta et al. [20] extended a 2D solid texturing method to generate 3D textured models and then manufactured them using various AM technologies to test build speed and quality. Some drawbacks of their methodology include the dependence of the generation algorithm on texture and a long processing time for mesh processing operations. Some other studies have also addressed texture usage and AM [21, 22]. However, none of them tackles the issues found in the previously mentioned study [20] or consider design principles for AM.

This paper develops an approach for integrating textures on mathematically defined surfaces and ensuring manufacturability using AM technologies. The main contributions of this work are – texture independent 3D extension of the solid texturing method available in the literature for graphical applications; inclusion of critical overhang angle constraint during texture generation to avoid support structures; modification of the texture generation algorithm for closed surfaces; extendable methodology for mapping textures on different surfaces; and a fast novel method for creation of CAD model including the details of generated textures.

The paper is organized into four sections. Based on the introduction to the topic, Section 2 describes the methodology. It starts with explaining the textures, their possible advantages, and clarification of the challenges of texture generation. Then the algorithm used for texture generation is described, along with the modifications to the algorithm for the inclusion of AM overhang angle constraint. Afterward, the implemented algorithms for mapping the generated textures on a mathematically defined surface are explained. Lastly, Section 2 describes the generation of a CAD model using the generated texture. Section 3 illustrates the results of the considered textures on various surfaces. Furthermore, models manufactured using FDM are demonstrated for verifying the methodology. A possible conceptual application is shown, and the applied methodology is discussed. Section 4 presents the paper's outcome and deliberates

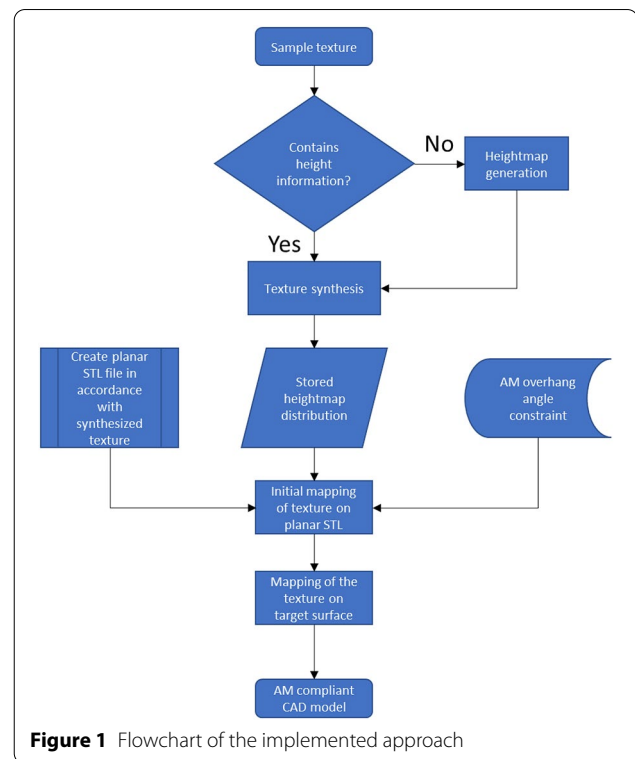
potential future research topics in texture generation for AM.

## 2 Methodology

The surface of an object often has different properties such as color, smoothness, orientation, and texture. Textures usually convey two different concepts. Either it can simply refer to a combination of colors applied on a surface that is expressible using two parameters. Or it can be a 3D feature created by changing the surface geometry of an object which may or may not contain a color distinction. The goal behind adding texture to an object is usually to either decorate external surfaces or add extra functionalities to an object. The latter is a demanded characteristic for industrial applications aiming at different purposes. For example, adding textures to decrease the grip force required for objects [23] or high-enough textures, which depend on the object size, may be used to enhance the heat transfer behavior of a surface since texture addition increases the surface area. Sahiti et al. [24] found that a remarkable heat transfer improvement can be achieved by using small cylindrical pins on the surfaces of heat exchangers. Fabrication of such textures is a challenging task as it depends on various factors such as dimensional accuracy, repeatability, material properties, texture dimension, and more [14]. Since AM technologies can theoretically fabricate complex shapes and detailed structures, they motivate the designing of 3D textures. However, the main design challenge is to generate structures with a fine texture in which the geometrical features are close to each other. It is sometimes difficult to obtain a CAD model containing texture as it can become computationally expensive. This paper proposes a possible solution to this problem by combining texture synthesis, image processing, consideration of AM overhang angle constraint, and a novel methodology for 2.5D surfaces to create an AM manufacturable CAD model, as summarized in Figure 1.

### 2.1 Texture Generation

Erfos et al. [25] defined texture generation or texture synthesis problem as the process in which a finite digital sample texture, available as an image, is used to construct samples with larger sizes of the same texture. Liang et al. [26] applied MRF-based methods successfully for texture synthesis for graphics applications. These methods model the texture as a local and stationary random process realization, characterizing each pixel on the texture based on its neighboring pixels. The MRF-based texture synthesis method can be represented as using a small texture size to synthesize the output texture, with similar characteristics as the input [27].

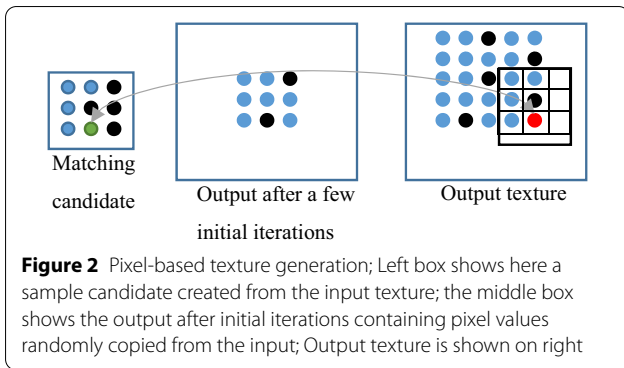


However, even though the MRF approach has many merits, it also exhibits certain disadvantages, such as heavy computation or complex algorithms, making this method difficult to use and understand. Considering the synthesis unit, the texture synthesis algorithms can be classified into two groups: pixel- and patch-based synthesis algorithms. These methods are discussed briefly in the following sections, followed by the description of this paper's contribution to texture synthesis.

#### 2.1.1 Pixel-based Texture Synthesis

The output texture is initialized in the pixel-based synthesis algorithm by copying a small, randomly selected seed region (a pixel) from the input. Then the best suitable pixel on the input is copied to the output after comparing its user-defined spatial neighbors to all neighbors in the input. After this, the synthesized region is grown pixel by pixel until attaining the required texture size. Figure 2 illustrates the pixel-based algorithm.

The algorithm collects values of already synthesized pixels around it in a user-defined search space ( $3 \times 3$  in this case as in Ref. [25]) to find the value of the red pixel in the output texture. After that, the algorithm goes through the set of candidates created using the input sample and identifies the best matching candidate, using this partial neighborhood collected from the output



sample, to assign a value to the red pixel in the output. The algorithm continues until it reaches the required size of output texture.

**2.1.2 Patch-based Texture Synthesis**

Patch-based synthesis can be considered an extension of pixel-based synthesis, taking the units as patches instead of pixels for texture generation. In the case of a patch-based texture synthesis algorithm (Figure 3), a larger seed region, namely a 'patch' containing multiple pixels, is used. The synthesized region grows by selecting the best-matched patch from a set of possible candidates created using the input sample and copying it to the output to accelerate the texture synthesis. In this case, a search space containing various sample candidate patches is also created from the input. The value of each patch in the output is determined using these sample patches, ensuring its consistency with the already synthesized neighbor patches.

The significant difference between the pixel-based and patch-based algorithms is copying units from the sample space to the output texture. The pixel values are copied directly into the output texture in the pixel-based

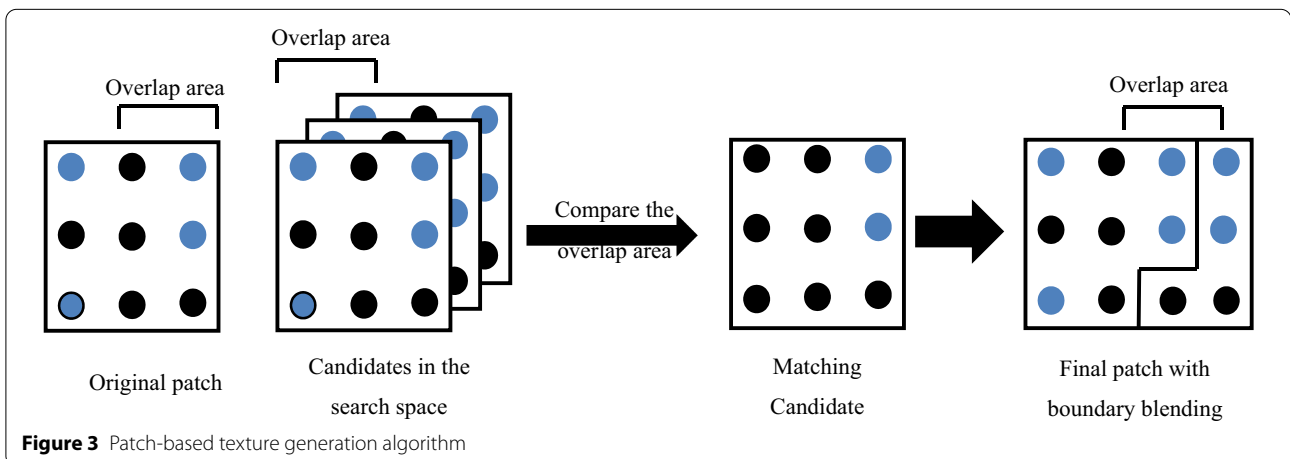
algorithm. However, copying the patch instead of the pixel is more complicated since the copied patch usually overlaps with the existing patches. The overlapping area should be processed to avoid or minimize conflict within the texture.

This work chooses a real-time texture synthesis by patch-based sampling algorithm [26] for texture generation. The optimized k-dimensional tree (k-d tree) algorithm [28] reduces the search time for the best matching patch. Additionally, to accelerate the generation of the search space for the patches, the quadtree pyramid (QTP) and principal component analysis (PCA) is applied to reduce the data amount and the dimension of the searching space. Feathering is used in the blending step to provide a smooth transition between different patches [29]. The authors have added two modifications to the patch-based texture generation approach, specific to AM and engineering design, as described in the following sections.

**2.1.3 Edge Modification (M-I)**

In some AM technologies such as Fused Deposition Melting (FDM) or Laser Powder Bed Fusion (LPBF), the fabrication of inclined features at an angle below a critical value relative to the build plate requires support structures. This angle depends on the process as well as manufacturing material and is called the critical overhang angle ( $\alpha_{crit}$ ). In this work, a commonly used typical value of  $45^\circ$  is assumed [31] for the critical angle. The samples are fabricated in this work using the FDM process, and this should be considered to decide where support structures are necessary. The texture edge is adjusted to minimize or avoid the support structures with the first modification.

If a textured planar surface is oriented perpendicular to the substrate plate of the FDM or LPBF process, the



bottom textures should be supported and then be post-processed. These critical texture boundaries are modified to generate inclined edges for the textures at these boundaries to avoid them. This modification is achieved by adapting the heightmap for the given texture. Figure 4 illustrates the concept. The red edges marked on the texture are not manufacturable without a support structure. The edge is parallel to the build plate, resulting in an angle of  $0^\circ$  between them, and is lower than the critical overhang angle. The heightmap of the texture is adapted by changing the texture’s pixel values to obtain the manufacturable edges (shown in green).

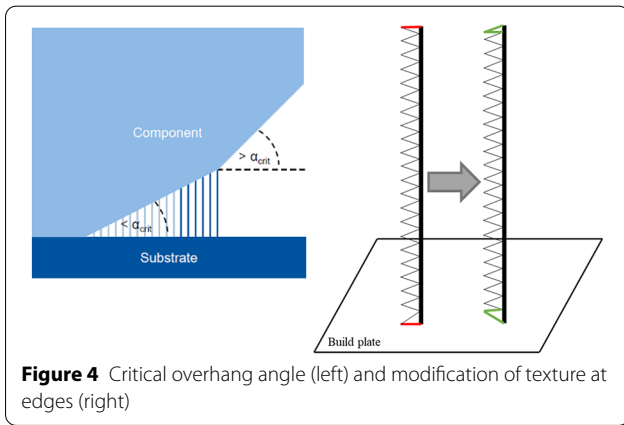
**2.1.4 Closing Patch Modification (M-II)**

The second modification is the application of a matching algorithm for creating a matching boundary in the case of a closed geometry like a cylinder. This algorithm is from the literature, which Erfos et al. [30] used to merge patches. In this work, this algorithm merges the edges of generated texture. Since a 2D image is used for texture synthesis, when mapping the textures on a closed geometry, there will be non-matching textures on the edges of the image, shown in red in Figure 5. An optimized path algorithm is applied for

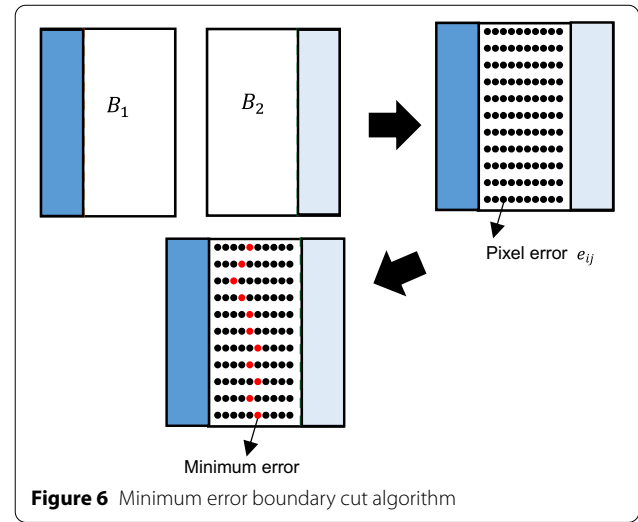
better consistency and matching texture at the edges. As the first step, the synthesized texture has a higher than required resolution (taken to be 100px in this paper). These extra pixels are divided equally into both sides of the image (50px each for light blue and orange region and used as the overlap region when matching the edges. The minimum error boundary cut algorithm computes the path with the best matching texture between the overlap areas, and Figure 6 illustrates this. Assuming the blocks  $B_1, B_2$  are overlap areas along the vertical edges (Figure 6), the pixel error on each overlapped position can be computed as  $e_{ij} = (B_{1,ij} - B_{2,ij})^2$ . The optimized path is calculated from the minimum of the cumulative pixel error  $E$  (Eq. (1)).

$$E_{ij} = \begin{cases} e_{ij}, & i = 1, \\ e_{ij} + \min(E_{i-1,j-1}, E_{i-1,j}, E_{i-1,j+1}), & i > 1. \end{cases} \quad (1)$$

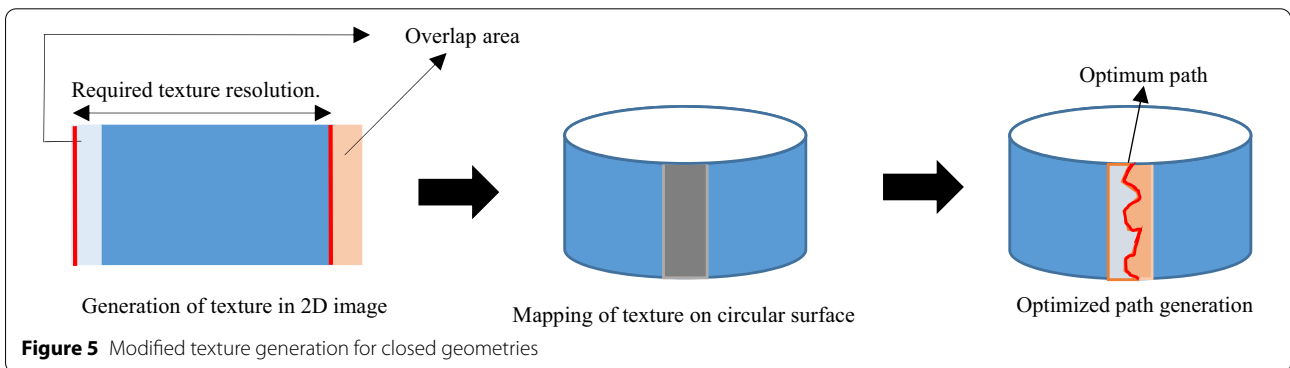
For better consistency of the path, only three neighbors of the current node are considered possible candidates for the next steps. The optimized path is the



**Figure 4** Critical overhang angle (left) and modification of texture at edges (right)



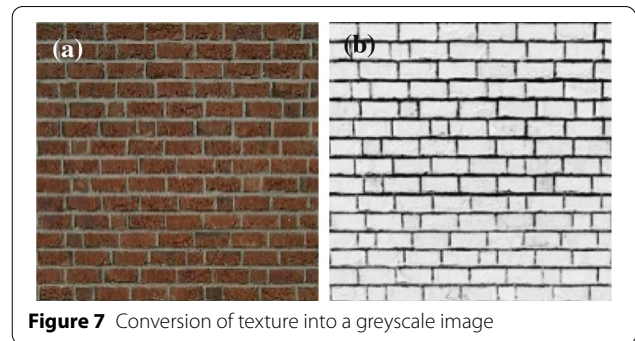
**Figure 6** Minimum error boundary cut algorithm



**Figure 5** Modified texture generation for closed geometries

collection of all nodes, which leads to the minimum cumulative pixel error. The image is cut along this path to avoid edge conflicts.

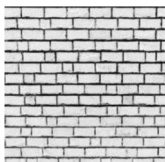
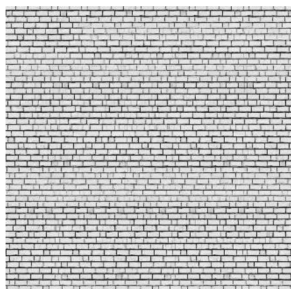
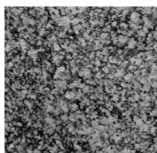
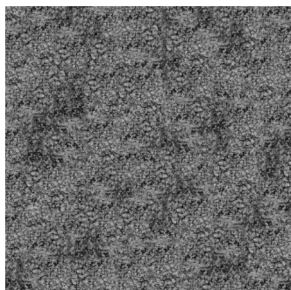
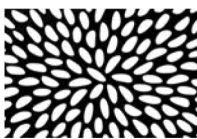
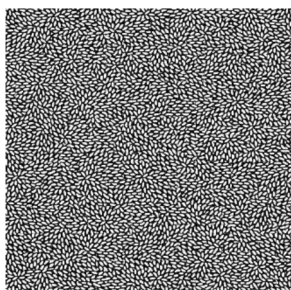
Table 1 shows the considered texture in this paper. These textures are commonly used for texture generation and represent structured and unstructured textures. Since there is no heightmap information available for these images, they are converted to black and white images (Figure 7 (a) & (b)). A heightmap is generated such that white corresponds to the maximum height and black corresponds to the part of the texture lying directly on the surface. Linear interpolation is used for generating height for the grey values in between. Table 1 also shows the generated textures and parameters for the generation algorithm.



### 2.2 Mapping Algorithms

Following the texture synthesis, mapping the texture onto an STL surface is done in two steps. The first step is preparing an STL file for a planar surface with the value of the z-coordinate explicitly set to zero for all vertices.

**Table 1** Texture inputs and the generated results

Patch size = 25 px Overlap size = 10 px	 (150x 100 px) <b>Texture<sub>1</sub></b>	 (800 x 800 px)
Patch size = 45 px Overlap size = 25 px	 (200x200 px) <b>Texture<sub>2</sub></b>	 (800 x 800 px)
Patch size = 45 px Overlap size = 25 px	 (200 x 200 px) <b>Texture<sub>3</sub></b>	 (800 x 800 px)

<sup>1</sup> <https://www.pinterest.cl/pin/242842604885001851/>

<sup>2</sup> <https://www.citiwood.co.za/product/postform-top-moss-granite/>

<sup>3</sup> <https://www.worthpoint.com/worthopedia/ikea-sweden-ab-fabric-anna-salander-1852816571>

During this step, using the image's resolution and the part's physical dimension, we calculate the resolution of the planar STL surface. For example, the dimension of the samples manufactured in this paper is 100 mm in both  $x$  and  $y$ -direction, and the number of pixels in the generated texture is  $800 \times 800$  px. The resolution of the planar STL surface is  $100/800 = 0.125$  mm considering the part and image dimensions considered in this work. Now, using the heightmap information, the texture height can be mapped as a one-to-one function onto each vertex of the STL file since the number of pixels in the image and resolution of the planar surface is equal.

It has already been shown how to generate texture from a given digital image in Section 2.1 and add height information of the texture to a planar surface in this Section. However, most of the objects used in engineering applications have complex and non-planar surfaces. This section describes two mapping schemes for mapping a texture on a planar surface onto a 2.5D surface. This category of surface, usually created by drawing a 2D curve and converting this curve to a surface by the extrude functionality in CAD software, constitutes a substantial portion of surfaces in engineering components. This category of surfaces can be characterized by the following equation within the input design domain:

$$f(x, z) = 0, a \leq x \leq b, c \leq y \leq d. \quad (2)$$

Eq. (2) can also be formulated in other forms, for example, by swapping  $x$  and  $y$  variables. However, this form is mathematically achievable by changing variables or rotation of the part in CAD software. A specific category of the curves described by Eq. (2) can be written in the explicit form as:

$$z = f(x), a \leq x \leq b, c \leq y \leq d. \quad (3)$$

It is assumed that  $f$  is a function, implying that there exists a maximum of one output  $z$  for each unique input  $x$ , and  $f$  is continuously differentiable [31]. Note that the authors have assumed that  $f$  is a function to simplify the mapping algorithm in this paper, but the procedure is applicable even when  $f$  is a non-function. This case when  $f$  is a non-function will be explained later in this section.

### 2.2.1 Distance Preserving Mapping

This first approach is based on a mathematical formulation that aims at preserving the distance between every

two consecutive points of a texture. In other words, the line segments in the planar surface are mapped onto arc divisions on the target surface. This algorithm approximates the curve segments with their corresponding lines, which connect the start and endpoints of the curve segments. If the curve segments are smaller in length, the mapping has a higher accuracy of preserving the distance between the detailed features of textures. Considering small curve segments also implies that the arc length of the curve in the  $xz$ -plane of the destination surface is almost equal to the length of the planar surface. The local arc lengths of the curve in the  $xz$ -plane of the destination surface can be computed as:

$$s = \int_a^b \sqrt{1 + (dz/dx)^2} dx. \quad (4)$$

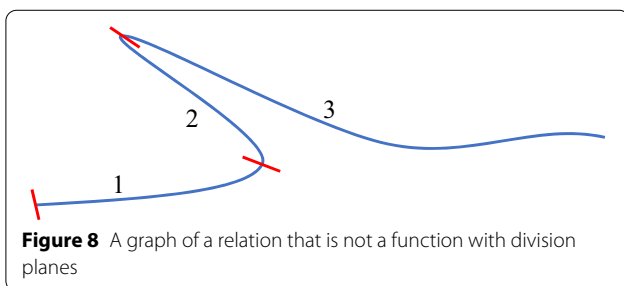
This approach is theoretically not limited to a specific resolution and can be used for high-resolution applications. However, the size or complexity of the target surface may restrict the application of this method. The approach starts with reading the texture data of the planar surface and separating these data into two categories: textural and non-textural data. Textural data include the height information of the planar surface to represent the texture geometrically, and non-textural data are coordinate information of the planar surface in the  $xy$ -plane. Then, the non-textural data are mapped onto the geometrical information in the target surface by preserving the distances between consecutive points. After this, the texture data are added or subtracted to the  $z$ -coordinate of the mapped points for upward and downward-looking curves, respectively. The textures are not oriented to be perpendicular to the target surface because this does not preserve the distance between detailed texture geometries, increasing the complexity of this approach by requiring more calculations. For example, the purpose of textures can be to increase gripping force or provide a decorative appearance. Here, the height values of the texture are usually low, and therefore this approach works well for practical applications. When the height values of the texture would be high, this method is advantageous from a computational point of view. This method ensures the producibility of the texture using the FDM and LPBF processes if the building direction is along the  $z$ -axis since it does not produce textural overhang features. After this mapping, the resultant 3D texture

data can be exported in the form of a point cloud or an STL file. This procedure is depicted briefly in Algorithm 1. This approach can deal with high-resolution textures and provide real-time solutions for most engineering components.

**Algorithm 1:** Distance preserving mapping algorithm

- 1: Input
  - A texture on a planar surface
  - The function of the surface onto which the mapping is applied provided in the form of equation (3)
- 2: Read the STL file of the textured planar surface
- 3: Separate the texture data into textural (z-components) and non-textural information data (x- and y-components)
- 4: Map the non-textural data to the corresponding geometrical points on the destination surface
  - $(x, y) \rightarrow (X, Y, Z)$
- 5: Add the textural data to the z-component of the mapped coordinates
  - $Z \rightarrow Z \pm z$
- 6: Output  
The destination surface with the same area as the planar surface, with texture

If Eq. (3) is not a function, the destination surface (or curve) can be divided into several piecewise functions, and the approach can be used for each piece. Thus, in non-function curves, a further pre-processing step is required to divide the curve into piecewise functions. Then, the arc length of all the pieces is computed to obtain the total arc length. Afterward, a textured planar curve with an identical length is created. For example, to apply a texture onto the curve demonstrated in Figure 8, the pre-processing step divides the curve into three sub-curves which are describable as mathematical functions. The procedure for this step is semi-automatic, and the division of the curves is carried out by plotting the curve and choosing the division points manually. Then, the arc length of the three curves is computed using Eq. (4), and the corresponding textured planar curve is created using the total arc length. Afterward, the algorithm should be utilized three times for curves 1, 2, and 3 consecutively out of the created textured planar curve. Note that the texture values for curve 2 are subtracted due to the left



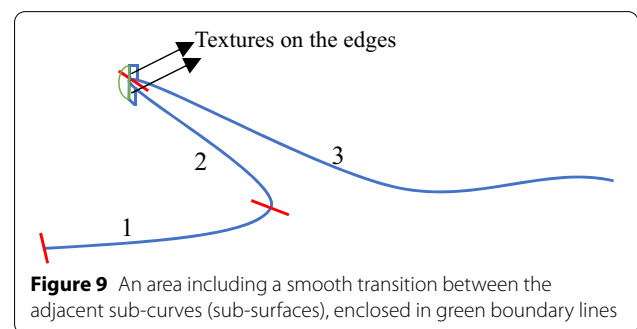
inclination of the curve (downward-looking curve). This modification of texture height should be treated automatically by recognizing downward and upward-looking curves.

Additionally, two further post-processing steps are required in the case of several consecutive curves. First, the intersection between the textures created on the sub-surfaces (sub-curves) should be avoided. For example, there might be an intersection between the textures of curves 1 and 2 in the area close to the division plane in Figure 8. A programming function is developed that uses the input curve data, i.e., curve type as downward or upward looking and curve points, to compute the distances in the area close to the division plane and then removes the intersecting textures. Second, a smooth transition is created between the textures of two consecutive curves. For example, if the algorithm is applied separately to curves 2 and 3 in Figure 8, no smooth transition is created between the two curves.

Figure 9 shows the area that requires smoothening. A simple approach is to fill this area. Another solution is to consider curves 2 and 3 together as one curve and then rotate the curve to achieve an approximate function. Afterward, the algorithm is applied, and the textured curve rotates back to the previous position. In the next step, curve 1 and the combined curve from curves 2 and 3 are assembled, and the intersection analysis and the required smoothening are applied. After removing the intersecting textures close to the division planes, there would be areas close to the planes which are non-textured, and they can be treated by adding smooth transitions. Note that the creation of a smooth transition in the distance-preserving mapping algorithm is a challenging task for complex geometries.

**2.2.2 Angle Preserving Mapping**

The second approach is angle-preserving, implying that the textures are mapped perpendicular to the target surface. It does not preserve the distances between textural features, but it maps the textural points perpendicular to the target surface. The algorithm is similar to the





previous approach except for the textural data mapping, as described in Algorithm 2. This approach has more calculations, making it slightly slower than the previous approach. However, it is more generic than the distance-preserving approach and is advantageous for complex curves since the post-processing step for the smooth transition is not required. For example, the algorithm can be used after dividing the curve in Figure 8 and calculating the total arc length. Note that intersection analysis and treatment are still necessary for this approach. This approach is also appropriate for closed surfaces since it provides a continuous solution.

**Algorithm 2:** Angle preserving mapping algorithm

- 1: Input
  - A texture on a planar surface
  - The function of the surface onto which the mapping is applied [provided in the form of equation (3)]
- 2: Read the STL file of the textured planar surface
- 3: Separate the texture data into textural (z-components) and non-textural information data (x- and y-components)
- 4: Map the non-textural data to the corresponding geometrical points on the destination surface
  - $(x, y) \rightarrow (X, Y, Z)$
- 5: Add the textural data to the x- and z-components of the mapped coordinates
  - $(X, Z) \rightarrow (X, Z) + z \times (n_x, n_z)$
- 6: Output  
The destination surface with the same area as the planar surface **and with the textures normal to the surface**

**2.2.3 Comparison Between Distance- and Angle-preserving Mapping**

This paper uses two mapping approaches to create textured 2.5D surfaces. The distance-preserving approach requires fewer calculations than the angle-preserving approach and is suitable for function curves. Although both the mappings can be used for function curves, the distance-preserving mapping is preferred for the coarse textures with relatively large texture heights, for example, 2-3 mm. The reason is that the distance-preserving type ensures producibility without using support structures with the FDM or LPBF processes as the texture remains perpendicular to the planar surface in this method. The angle-preserving mapping is preferred for non-function curves such as closed surfaces since it does not require the smoothing step. The distance-preserving mapping is dependent on the curve’s orientation, and if the curve is rotated, the resultant texture would be different. However, the angle-preserving mapping is independent of the coordinate system, and it produces the same result as long as the starting point of the mapping is kept the same.

A part of a typical sledgehammer handle is textured and illustrated with both approaches in Section 3.4 to illustrate the differences between two mappings.

**2.2.4 Case Studies for the Mapping Algorithms**

Two case studies are considered to show the capability of the mapping algorithms:

- Circular (or cylindrical) surface

A circular curve in the  $xz$ -plane has the following equation:

$$(x - p)^2 + (z - q)^2 = R^2, \tag{5}$$

where  $R$  is the radius of the curve and  $(p, q)$  is the coordinates of the center point. Re-organizing this equation gives:

$$z = \begin{cases} q + \sqrt{R^2 - (x - p)^2}, & -R + p \leq a \leq x \leq b \leq R + p, \\ q - \sqrt{R^2 - (x - p)^2}, & -R + p \leq a \leq x \leq b \leq R + p. \end{cases} \tag{6}$$

Each part of the relation is a function to which the algorithm can be applied. For simplified calculations, the polar coordinate system is adopted using:

$$\begin{cases} x = p + R\cos(\theta), \\ z = q + R\sin(\theta), \end{cases} \tag{7}$$

where  $0 \leq \theta_0 \leq \theta \leq \theta_1 \leq 2\pi$ . Therefore, Eq. (6) is represented as:

$$\rho = R\theta, \tag{8}$$

where the pole is located at  $(p, q)$ . This coordinate system has the advantage that Eq. (8) can be used instead of Eq. (4) to compute the arc lengths, simplifying the calculations.

- Sinusoidal surface

The equation of a sinusoidal curve may be written as:

$$z = A\sin\left(\frac{2\pi x}{L}\right), a \leq x \leq b, \tag{9}$$

where  $A$  and  $L$  are the amplitude and period of the curve. Computing the arc length using Eq. (4) leads to:

$$\begin{cases} s = \int_a^b \sqrt{1 + (dz/dx)^2}, \\ dx = \int_a^b \sqrt{1 + \left(\frac{2\pi A}{L} \cos\left(\frac{2\pi x}{L}\right)\right)^2}, \\ dx = \int_a^b \sqrt{1 + \left(\frac{2\pi A}{L}\right)^2 (1 - \sin^2\left(\frac{2\pi x}{L}\right))} dx. \end{cases} \tag{10}$$

The equation can be simplified by defining a variable as  $u = 2\pi x/L$ . Substituting  $x$  by  $u$  into Eq. (10) gives:

$$\begin{cases} s = \frac{L}{2\pi} \int_e^f \sqrt{1 + \left(\frac{2\pi A}{L}\right)^2 (1 - \sin^2(u))}, \\ du = \frac{L}{2\pi} \sqrt{1 + \left(\frac{2\pi A}{L}\right)^2} \int_e^f \sqrt{1 - \frac{\left(\frac{2\pi A}{L}\right)^2 \sin^2(u)}{1 + \left(\frac{2\pi A}{L}\right)^2}} du. \end{cases} \quad (11)$$

Note that  $e$  and  $f$  should be evaluated based on the limit values of the variable  $x$ . Furthermore, the following variable is introduced:

$$0 \leq m = \frac{\left(\frac{2\pi A}{L}\right)^2}{1 + \left(\frac{2\pi A}{L}\right)^2} < 1. \quad (12)$$

Substituting this gives:

$$s = \frac{L}{2\pi} \sqrt{1 + \left(\frac{2\pi A}{L}\right)^2} \int_e^f \sqrt{1 - m \sin^2(u)} du. \quad (13)$$

The last term represents an elliptic integral of the second kind [32].

These equations are used to apply both distance- and angle-preserving algorithms on the two non-planar surfaces, and the results are provided in section 3.

### 2.3 CAD Model Generation

After the generation and mapping of textures onto the surface, the next step is the creation of 3D geometry ready for additive manufacturing. This geometry creation was established in this work using Rhinoceros® and Grasshopper®, but this method can generate the models with other CAD software. A closed CAD model can be easily prepared for the samples with planar surfaces at the top and bottom by generating the capping surfaces on the sides. For the samples with curved and sinusoidal geometry, two separate approaches were used depending on the geometry of the surface lying below the textured surface. For the samples mapped onto the curved surface, a concentric surface was taken as the bottom surface. The sweep function used two rails to define the start and end profile and a guiding curve along the textured surface to prepare the surfaces that connect the top and bottom surfaces. For the case of a sinusoidal surface with texture, a planar surface was chosen as the bottom surface, as it is also possible to have a different geometry for the two surfaces in real-life samples. The free form curve was extracted from the textured surface and used, along with the other three edges, to create the side surfaces. The following section presents the resulting CAD models.

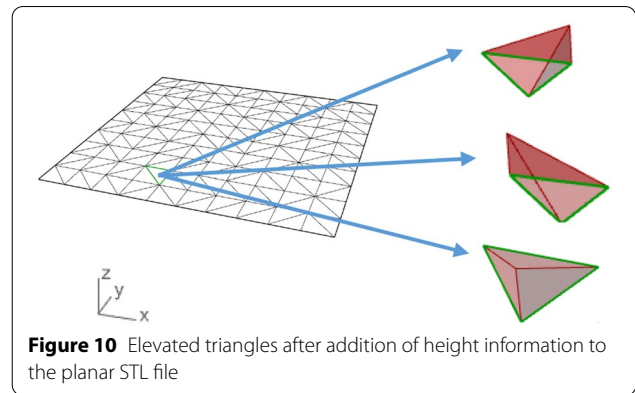


Figure 10 Elevated triangles after addition of height information to the planar STL file

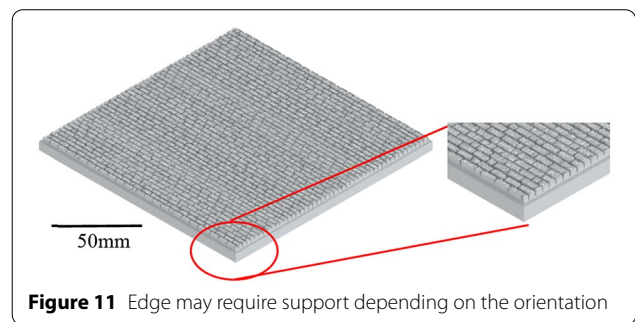


Figure 11 Edge may require support depending on the orientation

## 3 Results and Discussion

This section starts with a discussion about the advantages of the developed methodology specific to AM. It is followed by results from manufacturing the test samples for flat and non-flat textures. Finally, a case study for designing grip-enhancing texture for a sledgehammer is presented to highlight a possible application of the developed methodology.

### 3.1 AM-compliant Texture Generation

One of the significant advantages of using the image-based texture generation method introduced in this work is that this method ensures that the CAD model generated will not have any undercut features on the textured surface. This is due to the discrete nature of the texture and CAD model generation methodology. When a 2D image is used for texture generation, there is no information regarding the texture depth. When such an image is used for creating a textured surface STL model, it results in elevated triangles with an angle of 90° or more relative to the base plane of the used planar STL. After adding the information of texture height to each vertex of the planar STL, linear interpolation is used for connecting the elevated point to other vertices of the triangle. Figure 10 shows the illustration of some

possible cases after the initial mapping step for one triangle (marked in green).

It is a noteworthy result as the detailed structure of textures may lead to undesired overhang features in a component, and at the same time, it ensures that the designed components are AM favorable. It should also be noted that the edges of the textured surface may need to be modified to avoid using support structures when the texture is not manufactured parallel to the build plate, as illustrated in Figure 11.

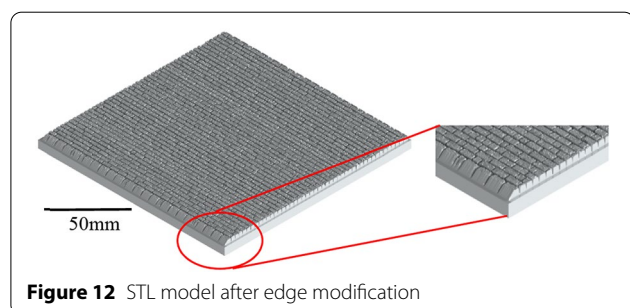
Although it should be considered that this discussion holds for all the flat textures created with this method, it is not always true when the textures are mapped onto non-flat surfaces. This is due to the influence of the new surface's topology. Additionally, the texture's height has an impact and needs to be considered during design depending on the AM technology.

### 3.2 AM Overhang Angle Constraint

As discussed in the previous section, overhanging features in a component design are not manufacturable using AM technology without support structures. This is important because the texture can be included in any part of a component, and the algorithm should be able to adapt the texture to ensure its manufacturability. In order to exclude overhang features at the edges, edge modification is included in this work as a part of texture generation. After including the AM overhang angle constraints on the CAD model's edge (illustrated in Figure 11), Figure 12 shows the final CAD model. This sample can be manufactured in different orientations relative to the build plate. The heightmap for the pixels near the edge of the synthesized textures was modified to generate these textures.

### 3.3 Mapping Texture on Surfaces

The described approach is used for generating the STL models with different surfaces and textures. Three textures are mapped onto different mathematically defined surfaces. For these samples, the maximum height of the texture is taken as 2 mm. The created models are manufactured using FDM. For printing the samples using



**Figure 12** STL model after edge modification

**Table 2** FDM sample manufacturing orientation

Sample	Orientation to build plate (°)
Texture 1 flat	0
Texture 1 curved	30
Texture 1 sinusoidal	15
Texture 2 flat	45
Texture 2 curved	45
Texture 2 sinusoidal	0
Texture 3 flat	15
Texture 3 curved	60
Texture 3 sinusoidal	30

FDM, Ultimaker 3 and PLA are used as the manufacturing machine and material. The orientation of the samples on the build plate is listed in Table 2. The rendered CAD models and the printed test samples are shown in Figure 13 for all the textures. The column on the left shows the rendered CAD models, and on the right, the samples manufactured using FDM.

The textures were manufactured successfully at different orientations without any support structures for the texture-containing surface. The approach developed has been implemented mainly using open-source software packages like python, and according to the experiments, the whole process chain is computationally affordable for relatively small samples.

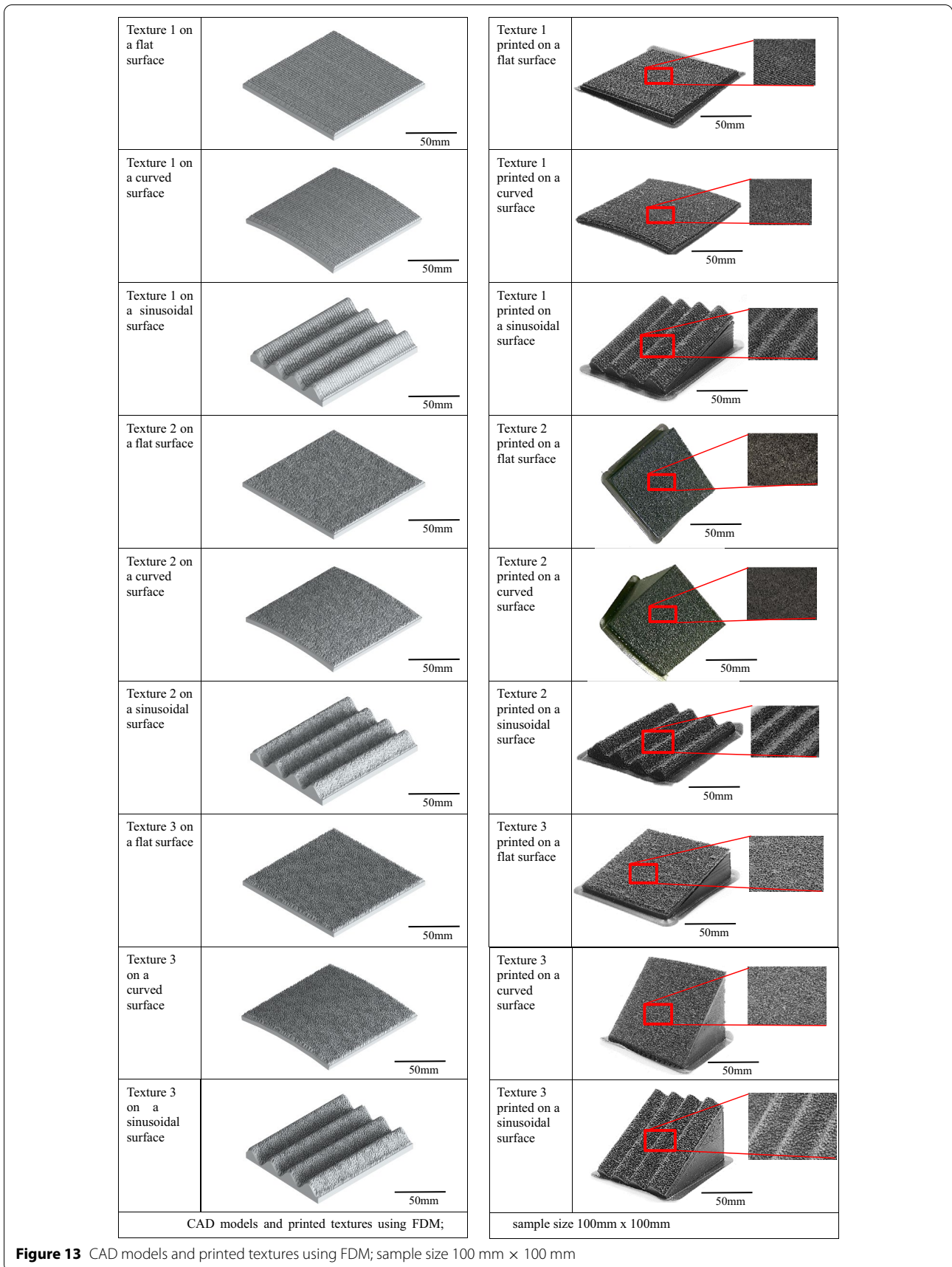
### 3.4 Case Study

As a case study for this approach, the texture is added to a sledgehammer's handle to improve the grip. Here, the boundary matching algorithm is used to generate consistent textures at the boundary. Both the mapping algorithms were used to add texture to the sledgehammer handle for this task.

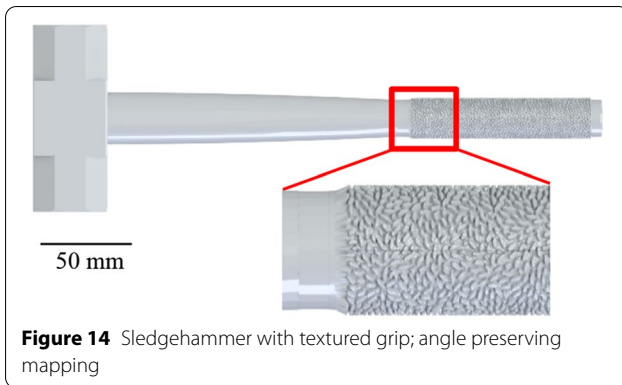
To show the extendibility of the approach to modular components, the cylindrical surface is divided into two halves, the texture is added to each part separately, and is finally assembled. Figures 14 and 15 show the results of the two approaches and demonstrate that the angle-preserving approach provides a continuous solution within a tolerance of 62  $\mu\text{m}$ . The distance preserving approach does not work well for closed geometries (Figure 15). However, it is a quick approach and is helpful for open surfaces with relatively low texture height.

## 4 Conclusion & Outlook

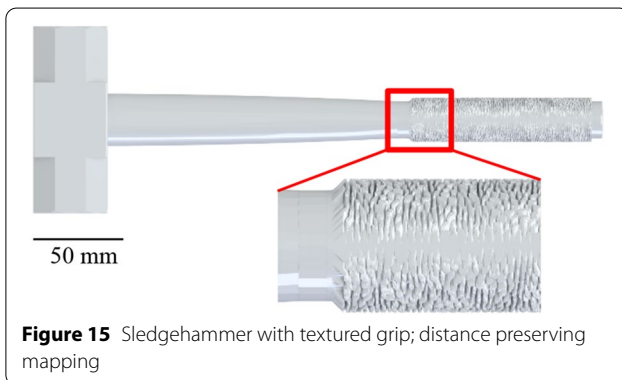
The important findings of this work, as well as future research possibilities, can be summarized as follows:



**Figure 13** CAD models and printed textures using FDM; sample size 100 mm x 100 mm



**Figure 14** Sledgehammer with textured grip; angle preserving mapping



**Figure 15** Sledgehammer with textured grip; distance preserving mapping

- (1) The paper described the AM favorable approach for texture generation, and its mapping on a surface is successfully applied for different textures on multiple 2.5D surfaces.
- (2) The patch-based texture synthesis algorithm was modified to generate textures on open and closed surfaces. The generated texture is AM suitable without any undercuts.
- (3) Support structures were not required during manufacturing after applying edge modification.
- (4) The mapping algorithms are applied on 2.5D surfaces based on the equation of curves, and they provide real-time results for the considered case studies. These methods are extendable to address 3D free form surfaces defined by parametric equations.
- (5) The modifications proposed to the texture generation, and mapping algorithms can be used for including textures on modular designs, commonly used in design for additive manufacturing.
- (6) The described approach is modular, independent of texture, and can produce real-time results for small geometries. Further research is needed to test it for large components as this approach can become

computationally expensive if a high-resolution texture design is required.

- (7) The methodology developed in this work can be used to integrate surface roughness as textures in the CAD model of the components produced using AM. A heightmap can be acquired from the scan data of the manufactured component. Such a model can be used to study the effect of wall roughness on fluid flow and heat transfer, as the surface roughness of AM-produced components is significantly larger than those produced using CM.

#### Acknowledgments

The authors sincerely thank our colleagues, Mr. Daniel Merget and Mr. Jan Theunissen from RWTH Aachen University, for critical discussion, feedback on the manuscript, and insights.

#### Author contributions

BV oversaw the whole trial and contributed to model generation and manufacturing; BV, OZ, and SZ wrote the manuscript; SZ carried out texture generation tasks; OZ was responsible for mapping algorithms; JHS contributed to the conception of the study and critical revision of the manuscript.

#### Authors' Information

Bhupesh Verma is currently a Ph.D. candidate at *Digital Additive Production (DAP)*, RWTH Aachen University, Germany. He received his master's degree in Simulation Sciences from RWTH Aachen University, Germany, in 2017. His research interests include design for additive manufacturing and fluid dynamics.

Omid Zarei is currently a Ph.D. student at *Digital Additive Production (DAP)*, RWTH Aachen University, Germany. He received his master's degree in Simulation Sciences from RWTH Aachen University, Germany, in 2017. His research interests include lightweight design and structural mechanics.

Song Zhang is currently a Ph.D. student at *Digital Additive Production (DAP)*, RWTH Aachen University, Germany. He received his master's degree in Electrical Engineering from RWTH Aachen University, Germany, in 2018. His research interests include quality assurance for additive manufacturing and machine learning.

Johannes Henrich Schleifenbaum is head of the chair for *Digital Additive Production (DAP)*, RWTH Aachen University, Germany, and *Managing Director of the ACAM – Aachen Center for Additive Manufacturing*.

#### Funding

Supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - EXC 2023 Internet of Production/ 390621612

#### Competing Interests

The authors declare no competing financial interests.

Received: 30 January 2021 Revised: 7 April 2022 Accepted: 20 April 2022  
Published online: 14 July 2022

#### References

- [1] A Curodeau, E Sachs, S Caldarise. Design and fabrication of cast orthopedic implants with freeform surface textures from 3-D printed ceramic shell. *Journal of Biomedical Materials Research: An Official Journal of The Society for Biomaterials, The Japanese Society for Biomaterials, and The Australian Society for Biomaterials and the Korean Society for Biomaterials*, 2000, 53: 525–535.
- [2] D Attinger, C Frankiewicz, A R Betz, et al. Surface engineering for phase change heat transfer: A review. *MRS Energy & Sustainability*, 2014: 4.
- [3] N Pietroni, P Cignoni, M Otaduy, et al. Solid-texture synthesis: a survey. *IEEE Computer Graphics and Applications*, 2010, 30: 74–89.

- [4] C C Chen. Texture synthesis: A review and experiments. *Journal of Information Science and Engineering*, 2003, 19: 371–380.
- [5] Y Fischer. *Fractal image compression: theory and application*. Springer Science & Business media, 2012.
- [6] E J Delp, R L Kashyap, O Robert Mitcheli. Image data compression using autoregressive time series models. *Pattern Recognition*, 1979, 11: 313–323.
- [7] J Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 1974, 36: 192–225.
- [8] G Turk. *Texture synthesis on surfaces*. Los Angeles: California, 2001.
- [9] L Y Wei, M Levoy. Texture synthesis over arbitrary manifold surfaces. L. Pocock (Ed.), *Proceedings of ACM SIGGRAPH 2001*, Association for Computing Machinery (ACM), New York, N.Y., 2001: 355–360.
- [10] S Magda, D Kriegman. Fast texture synthesis on arbitrary meshes. *Proceedings of the 14th Eurographics workshop on Rendering*, 2003: 82–89.
- [11] P Hao, D Liu, K Zhang, et al. Intelligent layout design of curvilinearly stiffened panels via deep learning-based method. *Materials & Design*, 2021, 197: 109180.
- [12] Gang Liu, Y Gousseau, G S Xia. Texture synthesis through convolutional neural networks and spectrum constraints. *23rd International Conference on Pattern Recognition (ICPR)*, IEEE, 2016: 16824420.
- [13] N Jetchev, U Bergmann, R Vollgraf. Texture synthesis with spatial generative adversarial networks. *arXiv*, 2017.
- [14] D G Coblas, A Fatu, A Maoui, et al. Manufacturing textured surfaces: State of art and recent developments. *Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology*, 2015, 229: 3–29.
- [15] J Alcisto, A Enriquez, H Garcia, et al. Tensile properties and microstructures of laser-formed Ti-6Al-4V. *J. of Mater. Eng. and Perform.*, 2011, 20: 203–212.
- [16] B Vayre, F Vignat, F Villeneuve. Designing for additive manufacturing. *Procedia CIRP*, 2012: 632–637.
- [17] A Alafaghani, A Qattawi, M A Ablat. Design consideration for additive manufacturing: Fused deposition modelling. *OJAppS*, 2017, 7: 291–318.
- [18] J Shah, B Snider, T Clarke, et al. Large-scale 3D printers for additive manufacturing: design considerations and challenges. *Int. J. Adv. Manuf. Technol.*, 2019, 104: 3679–3693.
- [19] M K Thompson, G Moroni, T Vaneker, et al. Design for additive manufacturing: Trends, opportunities, considerations, and constraints. *CIRP Annals*, 2016, 65: 737–760.
- [20] A Armillotta. Direct texturing for additive manufacturing: Software support and build tests. *Rapid Prototyping Journal*, 2020, 26: 881–894.
- [21] Y Chen. 3D texture mapping for rapid manufacturing. *Computer-aided Design and Applications*, 2007, 4: 761–771.
- [22] K Xiao, F Zardawi, R van Noort, et al. Developing a 3D colour image reproduction system for additive manufacturing of facial prostheses. *Int. J. Adv. Manuf. Technol.*, 2014, 70: 2043–2049.
- [23] J R Flanagan, A M Wing. Effects of surface texture and grip force on the discrimination of hand-held loads. *Perception & Psychophysics*, 1997, 59: 111–118.
- [24] N Sahiti, F Durst, A Dewan. Heat transfer enhancement by pin elements. *International Journal of Heat and Mass Transfer*, 2005, 48: 4738–4747.
- [25] A A Efros, T K Leung. Texture synthesis by non-parametric sampling. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, IEEE, 1999.
- [26] L Liang, C Liu, Y Q Xu, et al. Real-time texture synthesis by patch-based sampling. *ACM Trans. Graph.*, 2002, 20: 127–150.
- [27] L Y Wei, S Lefebvre, V Kwatra, et al. State of the art in example-based texture synthesis. *Eurographics Association*, 2009: 93–117.
- [28] D M Mount. *ANN programming manual, Technical report*. Maryland: University of Maryland, 1998.
- [29] R Szeliski, H Y Shum. Creating full view panoramic image mosaics and environment maps. G.S. Owen (Ed.), *SIGGRAPH 97: The 24th International Conference on Computer and Interactive Techniques*, Association for Computing Machinery, 1997.
- [30] A A Efros, W T Freeman. Image quilting for texture synthesis and transfer. L. Pocock (Ed.), *Proceedings of ACM SIGGRAPH 2001*, Association for Computing Machinery (ACM), New York, N.Y., 2001.
- [31] I Düntsch, G Gediga. *Sets, relations, functions*. Bangor: Methodos Publishers, 2000.
- [32] NIST Digital Library of Mathematical Functions, <http://dlmf.nist.gov/>.

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

---

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)

---