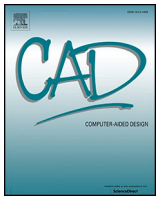
Computer-Aided Design 173 (2024) 103718

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad



Generation of continuous and sparse space filling toolpath with tailored density for additive manufacturing of biomimetics

Sadaival Singh [1], Ambrish Singh [1], Sajan Kapil [*], Manas Das

Department of Mechanical Engineering, Indian Institute of Technology Guwahati, India

ARTICLE INFO

Keywords:
Gradient infill
Continuous toolpath
Porous printing
Additive manufacturing
Travelling salesman problem

ABSTRACT

A method of generating a continuous toolpath that can be biased in a user-specified direction of travel is proposed for the fabrication of density-based functionally graded parts through *Additive Manufacturing (AM)*. The methodology utilizes *Lin Kernighan's (LK) Travelling Salesman Problem (TSP)* solver over a digitized grid within the contour domain to generate a toolpath with minimal lifts and a common start and end point. Three force-based methods of digitization, namely rectangular, circular, and contour adaptive, are proposed in this work. Each of these methods initialize from a structured or an unstructured grid, where the grid points are assumed to be connected with either linear (rectangular digitization) or a combination of linear and torsional springs (circular and contour adaptive digitization). Enforcing an equilibrium amongst the spring forces and appropriately selecting the ideal spring length, the necessary configuration of grid points can be generated for a desired toolpath.

The density of grid points (consequently, part density) can be varied through the user-defined input function or an image-based density map imposed on the ideal spring length over the contour domain. The proposed toolpath, as a case study, was implemented for printing a bone with density prescribed through a CT scan image stack. The CT scan of the printed part qualitatively establishes the conformity of the toolpath to the user-specified density gradient.

1. Introduction

Functionally Graded Materials (FGMs) can be broadly categorized into two varieties, material and density-based FGM. An FGM can also be a combination of both. A material-based FGMs uses multiple print materials, each of which is integral to the fabricated part; however, the percentage of each species varies as dictated by the user [1–3]. A density-based FGM primarily focuses on achieving tailored part density through selective material deposition over the input space. This spatial variation of density is a consequence of path planning and not the property of the material (as would have been the case for the material-based FGM). Spatial variation in density can be achieved by considering either the entire 3D space or a layer at a time. Since *Additive Manufacturing (AM)* uses layerwise deposition, an algorithm for density gradient in each layer (2D) can generate spatial variation in 3D space. In the work of Prevost et al. [4], the spatial variation of density within the part considers the part volume (3D), which is then redistributed to achieve the desired gradient. This variation in density causes a shift in the *Center of Gravity (CG)* of the part, consequently balancing the object (models, figurines, etc.) in a desired pose, schematically shown in Fig. 1 (a). The algorithm uses a voxel-based approach jointly modifying (carving and/or deforming) the interior and the surface of the model to achieve the spatial gradient. Another implementation of a voxel-based approach was proposed by Telea et al. [5], which detected the local thickness of the part being printed and compared it to the printer's print resolution for printability assessment. An interesting application of spatial density variation can be found in the works of Stava et al. [6], wherein the proposed algorithm first identifies the localized regions in part with high-stress values, which is subsequently reinforced with either thickening the part or by means of strut insertion. Open-cell foam structures typically consist of interconnected struts dividing the 3D space into regions. These structures exhibit high irregularity or an aperiodic lattice. The work of Martinez et al. [7] exploits this property using Voronoi open-cell foam to generate objects with gradient elasticity. Fig. 1(c) schematically shows the density variation, using Voronoi-based regions(cells), for a planar part.

* Correspondence author at: Department of Mechanical Engineering, Indian Institute of Technology Guwahati, Guwahati, Assam-781039, India.
E-mail address: sajan.kapil@iitg.ac.in (S. Kapil).
[1] The first and second authors have contributed equally to this work.

https://doi.org/10.1016/j.cad.2024.103718
Received 17 May 2023; Received in revised form 22 March 2024; Accepted 24 April 2024
Available online 26 April 2024

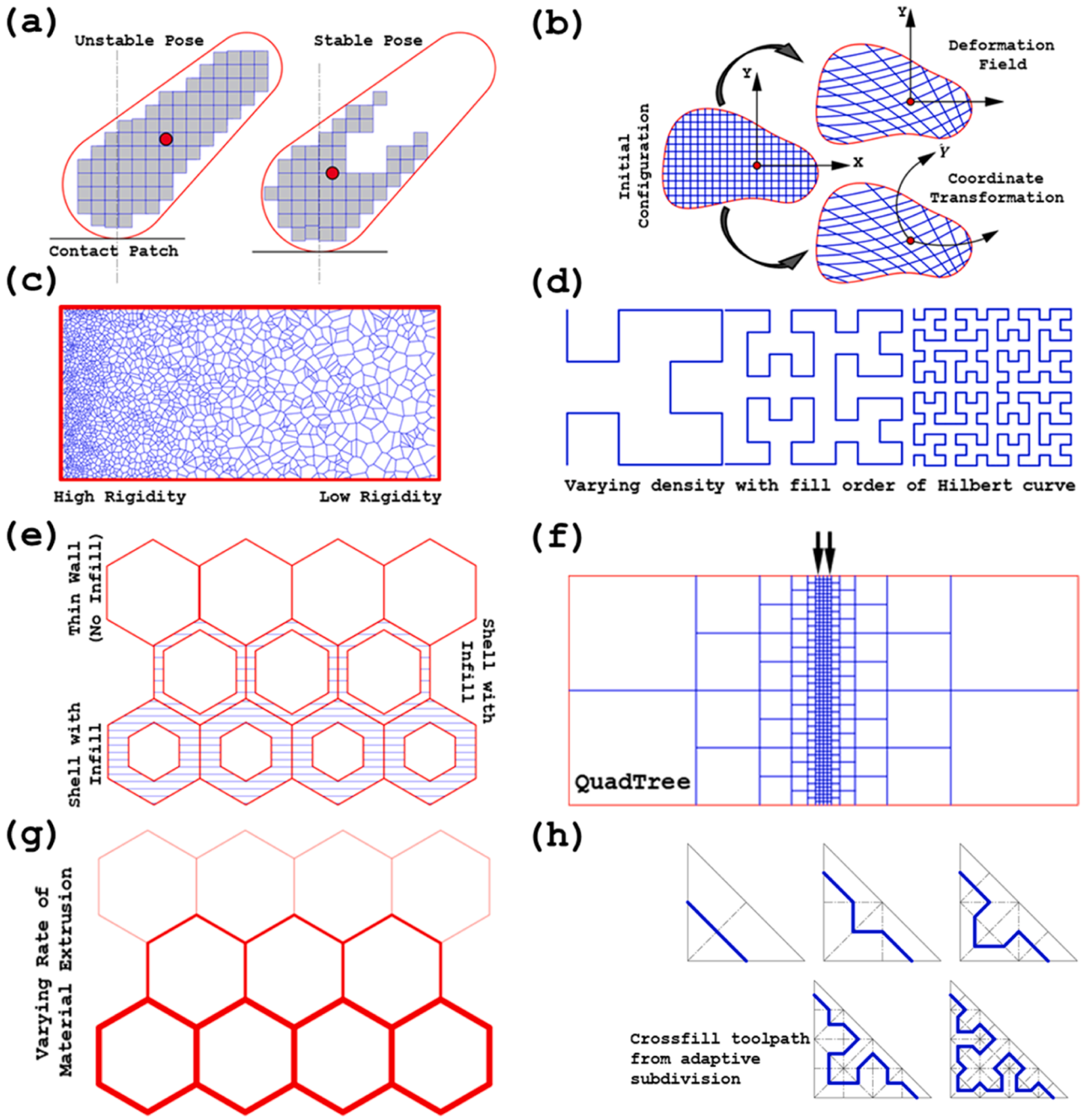


Fig. 1. Schematic representation of user-defined gradient toolpath achieved through (a) modifying print surface and interior [4], (b) deformation-based coordinate transformation matrix [12], (c) Voronoi open-cell foam [7], (d) varying order fractal-based [14] (e) shell with infill [8] (f) adaptive quadtree [11] (g) varying extrusion rate [9] (h) crossfill [17].

Bates et al. [8], showed the variation in density of a honeycomb structure significantly affects its energy-absorbing properties. The gradation in honeycombs, as shown in Fig. 1(e), was achieved by printing infill between the two lines for honeycomb for thicker regions and printing singles lines (bead width equal to nozzle diameter) for thinner regions. A similar study by Choy et al. [9] showed higher specific energy absorption of graded samples of cubic and honeycomb lattice structures as compared to uniform ones. However, gradation of density, as illustrated in Fig. 1(g), was achieved by varying the amount of material deposited, thereby altering the thickness of the struts along the graded direction. The spatial density variation has been extensively studied within the context of part topology optimization [10]. In the works of Wu [11], a quadtree-based toolpath was suggested, which was refined (further discretized) in accordance with the structural information derived from topology optimization. The resulting infill structure, for a given mechanical load, provided optimal stiffness when compared with uniform density, schematically shown in Fig. 1(f). Liu et al. [12] analyzed the shell-graded infill from a geometric perspective describing both the shell and infill in a parameter-based explicit way. The authors consider a uniform lattice that, when subjected to a deformation gradient field, acquires a non-uniform structure. Thus, a coordinate transformation matrix can be constructed in accordance with the deformation gradient field that, when imposed upon the uniform parent lattice, generates an optimized graded infill (Fig. 1(b)). Several fractal-based algorithms also exist in the literature [13–16]; one methodology to achieve gradient with fractal structures is through variation

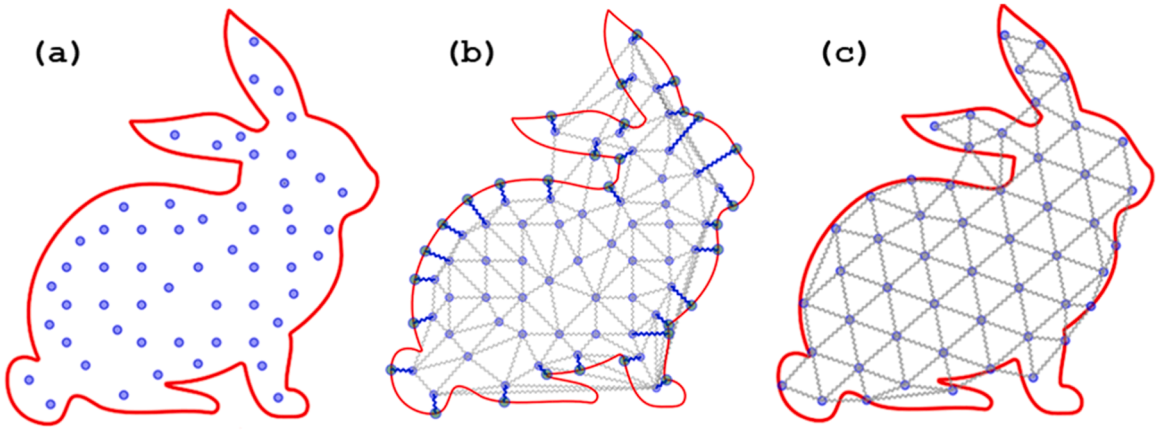


Fig. 2. Force-based algorithm for digitization (a) set of points randomly distributed within the contour, (b) triangulated points (points adjacent to the boundary connected to the nearest point on the contour are represented using blue springs), and (c) sides of triangles represented as spring in equilateral configuration.

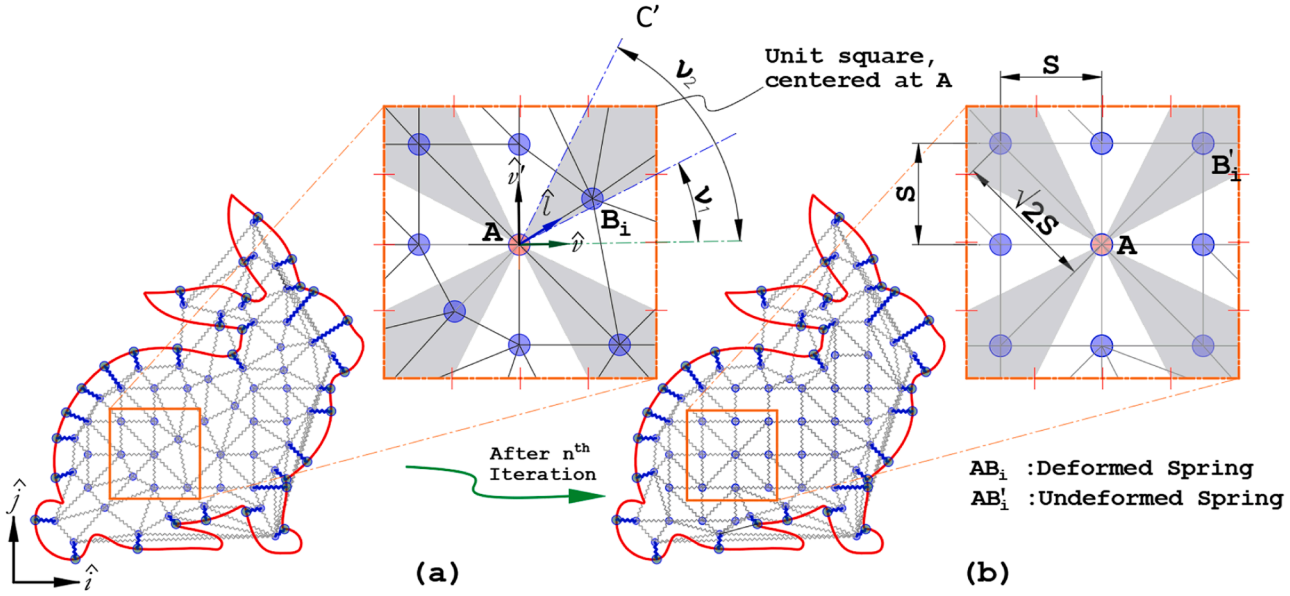


Fig. 3. Points inside the contour after (a) Delaunay triangulation, here, ν_1 and ν_2 are chosen as $\tan^{-1}(1/2)$ and $\tan^{-1}(2)$, respectively, and (b) implementation of the square grid algorithm.

of the order of the curve, as illustrated in Fig. 1(d). An effective algorithm to generate a graded infill with a continuous toolpath, termed 'Cross-fill,' was proposed by Kuiper et al. [17]. The algorithm is based on the adaptive subdivision of a cube encapsulating the input model. This cube is divided into prism-shaped cells, which can be further subdivided as defined by the density map. A data structure constructed from a combination of a graph and a tree representing connectivity and hierarchy, respectively, is used to generate a continuous toolpath. Fig. 1(h) schematically shows the graded infill using 'Crossfill' for a planar layer.

In this work, an algorithm using a TSP-based solver is proposed that generates a continuous toolpath in accordance with a user-defined gradient. The *Travelling Salesman* problem can be formulated as follows. Given a graph $G = (V,\ A)$ where V and A represent a collection of vertices and connecting edges, respectively and $C = C_{ij}$ representing a cost (distance) matrix containing the cost of going from city 'i' to 'j', then; find a permutation (or order of visit) that minimizes C. The constraint of the problem, as implemented here, is its Hamiltonian nature, implying the tour, having started from a given city, should visit every other city once and terminate at the start city. The resulting tour is a closed-loop path with a common start-end point. The proposed toolpath, analogous to cities, has grid points within the contour domain for a given layer. The arrangement of the grid points, termed digitization, could either be rectangular, circular, or adaptive. The resulting toolpath exhibits a Hamiltonian behavior, thus facilitating minimal retractions. Furthermore, by varying the distance between two consecutive grid points within the domain, the toolpath can be engineered to favor one direction over the other. This reduces the number of turns with minimal deviation from the user-defined density gradient. The direction-favoring of the toolpath lowers the number of turns and is advantageous in several respects, as outlined by Singh et al. [18]. A Hamiltonian toolpath allows for a continuous deposition of material. It must be noted that, during start and stop due to material deposition instabilities, defects in the parts can be introduced. A closed loop path allows for minimal retractions that reduces these defects. Also, have a common start stop point localizes this defect. The user will be able to select the location of these start and stop point such that they are easily accessible for post processing after part printing. This advantage is unique to Hamiltonian path. This is further described in the works of Singh et al. [18]. While in the works of Bedel et al. [19] and Cheng et al. [20], authors have explored the used of Hamiltonian for sparse closed loop generation continuous toolpath generation. This work extends beyond a Hamiltonian path with additional functionality. This additional functionality

Table 1
Algorithm for generation of a rectangular grid using conditions of force balance

`while` (**True**)
  `If` convergence criteria == True:
    `Break`
  `Else`
    `for` **all $'P'$ within contour**
    1. Consider a grid point (say) $A_i(x_{A_i},\ y_{A_i}):\ A_i \in P$, where $P = [x \quad y]_{n\times 2}$
    2. Define vector $\overrightarrow{F_i}$ such that $\overrightarrow{F_i} \in \overrightarrow{F}$, where $\overrightarrow{F_i}$ is the net force on a point A_i due to all springs attached to it
    3. Get ξ_{Ai} the closest point on boundary (Γ) to point A_i
      `If` $l = distance(A_i, \xi_{Ai}) < d_{min}$
        Compute $\overrightarrow{f}_{Boundary}(l, l_0)_{A_i\xi_{Ai}}$ which is the force on A_i due to boundary point ξ_{Ai} using equation (5)
        Update $F_i = F_i + \overrightarrow{f}_{Boundary}(l, l_0)_{A_i\xi_{Ai}}$
    `for` **all $'B_i'$ within B**
    4. Identify a vector $(\overrightarrow{A_iB_i})$ between A_i and the neighboring grid point (say)$B_i(x_{B_i},\ y_{B_i}) : B_i \in B$ and $B \subset P$ where $B = [x_{B_i} \quad y_{B_i}]_{m\times 2}$. '$m$' is the number of points connected to A_i, as a result of Delaunay Triangulation. The direction of $\overrightarrow{A_iB_i}$ is denoted by a unit vector (say) $\hat{l}$.
    5. Define $\gamma = \cos^{-1}(\hat{v}\cdot\hat{l})$
      `if` $\nu_1 < |\gamma| < \nu_2$
        $l_0 = \sqrt{2}S$
      `else`
        $l_0 = S$
      Compute $\overrightarrow{f}_{Linear}(l, l_0)_{A_iB_i}$ which is the force on A_i due to neighbor point B_i using equation (5)
      Update $\overrightarrow{F_i} = \overrightarrow{F_i} + \overrightarrow{f}_{Linear}(l, l_0)_{A_iB_i}$
    6. Update point coordinates of P to $P' = [x' \quad y']_{m\times 2}$ such that $P' = P + \Delta t\overrightarrow{F}$.
7. Run TSP solver on P'

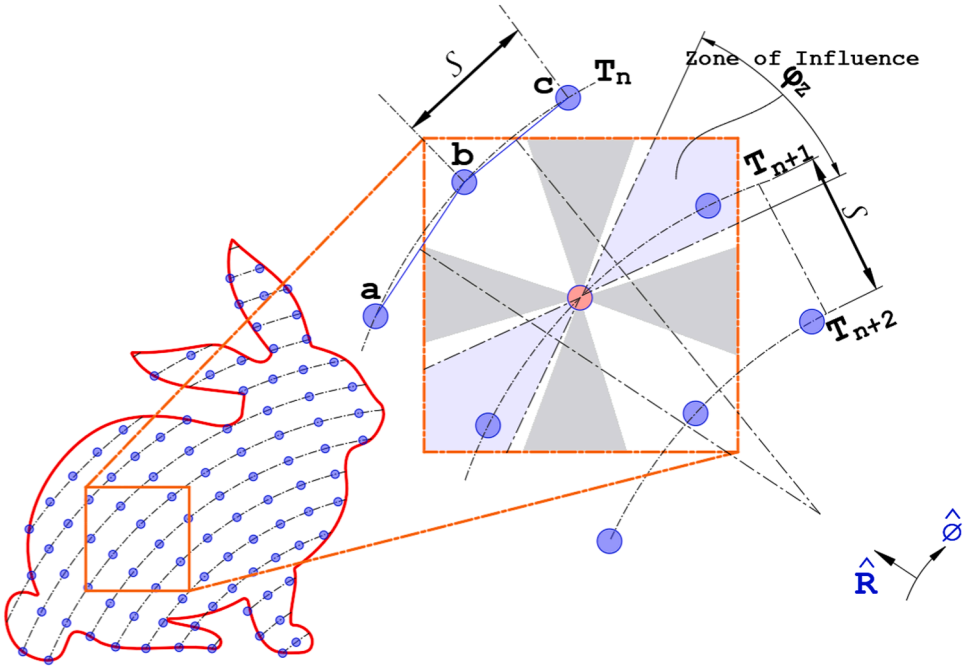


Fig. 4. Generation of circular grids; the domain within the contour is marked with circular concentric tracks ($T_1,\ T_2,\ T_3 \ldots T_n$) with a user-defined center. These tracks are subsequently discretized with grid points.

includes ability to create gradient toolpath with density specified by the user though a grayscale density map. Furthermore, this toolpath can be biased in more than one specific direction of travel while maintaining the specified density and it's Hamiltonian nature. The use of density map, as shown in this study, allows for a unique advantage where the presented method can be used for fabrication of natural structures with relative ease.

2. Methodology

Two methods, one exact and the other heuristics, are widely explored in the literature for solving the traveling salesman problem. While the heuristic methods are fast compared to the exact solutions, they do not guarantee an optimal tour. The existing methods for obtaining exact solutions are computationally expensive and are practical for only a small number of cities. In this work, *Lin-Kernighan (LK)* algorithm, a heuristic-based opensource TSP solver, is used to obtain the toolpath [21].

Fig. 2 outlines the general algorithm of the proposed toolpath. This algorithm, originally developed for the generation of unstructured simplex mesh, was adopted and appropriately modified from the works of Persson et al. [22] for the implementation in toolpath planning for

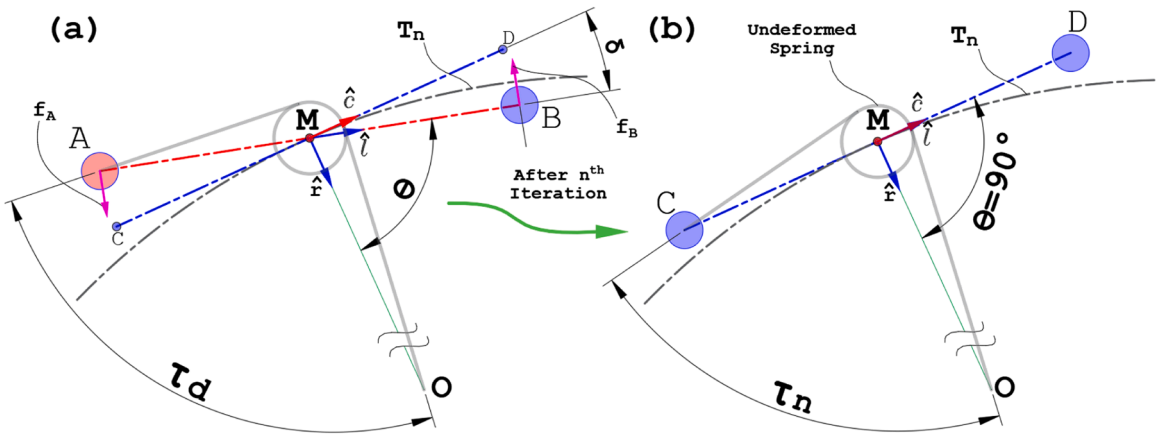


Fig. 5. The presence of a torsional spring, represented by a solid gray line, orients the line segment *AB* perpendicular to the track radius.

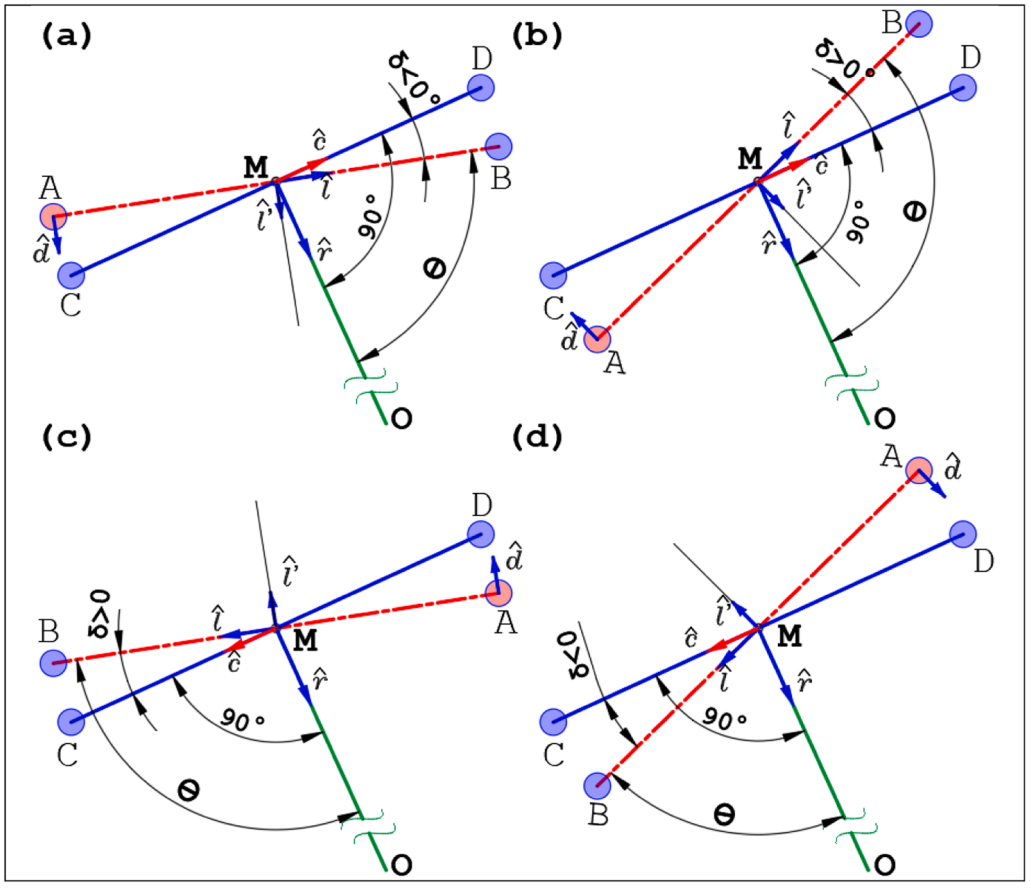


Fig. 6. Method of sorting possible configurations of current grid points relative to their desired configuration for determining the direction of application of force on the 'point in consideration' (a) $\delta < 0$ & $\hat{r}\cdot\hat{l} > 0$ (b) $\delta > 0$ & $\hat{r}\cdot\hat{l} > 0$ (c) $\delta > 0$ & $\hat{r}\cdot\hat{l} < 0$ and (d) $\delta < 0$ & $\hat{r}\cdot\hat{l} < 0$.

FGMs. Essentially, the algorithm consists of a set of points distributed (randomly or in the form of a structured grid) within a given contour (as illustrated in Fig. 2(a)), which can be triangulated using the Delaunay Triangulation algorithm, as shown in Fig. 2(b). The edges of these triangulated points can be considered as springs that push the nodes apart, such that the resulting triangle formed by any given set of points is equilateral, as illustrated in Fig. 2(c). The grid points that are immediately adjacent to the contour are connected with similar springs with the nearest point on the contour, as illustrated in Fig. 2(b), this is done to ensure the points maintain a distance from the boundary.

Linear and circular interpolation form the basis of connecting points for CNC based machines, thus having points spaced in rectangular or circular grid fashion would be of interest. In subsequent sections additional constraints and forces are imposed on this mesh for making rectangular and circular grids. Connecting rectangular and circular grids with a TSP solver, result in a continuous toolpath with constant and defined stepover throughout the grid. This characteristic is notably absent in a hexagonal grid, as illustrated in Fig. 2(c). The rectangular and circular grids establish the foundation of an adaptive grid capable of locally adjusting to conform to the contour boundary.

Table 2
Algorithm for generation of a circular grid using conditions of force balance

`while` (**True**):
  `If` convergence criteria == True:
    `Break`
  `Else`
    `for` all 'P' within contour
    1. Consider a grid point (say) $A_i(x_{A_i},\ y_{A_i}) :\ A_i \in P$, where $P = [x \quad y]_{n\times 2}$
    2. Define vector $\overrightarrow{F_i}$ such that $\overrightarrow{F_i} \in \overrightarrow{F}$, where $\overrightarrow{F_i}$ is the net force on a point A_i due to all springs attached to it
    3. Get ξ_{Ai} the closest point on boundary(Γ) to point A_i
      `If` $l = distance(A_i, \xi_{Ai}) < d_{min}$:
        Compute $\overrightarrow{f}_{Boundary}(l, l_0)_{A_i\xi_{Ai}}$ which is the force on A_i due to boundary point ξ_{Ai} using equation (5)
        Update $\overrightarrow{F_i} = \overrightarrow{F_i} + \overrightarrow{f}_{Boundary}(l, l_0)_{A_i\xi_{Ai}}$
    `for` all 'B_i' within B
    4. Identify a vector ($\overrightarrow{A_iB_i}$) between A_i and the neighboring grid point (say)$B_i(x_{B_i},\ y_{B_i}) : B_i \in B$ and $B \subset P$ where $B = [x_{B_i} \quad y_{B_i}]_{m\times 2}$. '$m$' is the number of points connected to A_i, as a result of Delaunay Triangulation. The direction of $\overrightarrow{A_iB_i}$ is denoted by a unit vector (say) $\hat{l}$.
    5. Repeat step 5 of Table 1 for calculating and storing $\overrightarrow{f}_{Linear}(l, l_0)_{A_iB_i}$ on A_i due to B_i as a result of the rectangular grid into the vector $\overrightarrow{F}$. Take $\hat{v}$ as $\overrightarrow{A_iO}$
    6. Define $\hat{r} = \dfrac{\overrightarrow{OM}}{|\overrightarrow{OM}|}$, where $M = \left[\dfrac{(x_{A_i}+x_{B_i})}{2}, \dfrac{(y_{A_i}+y_{B_i})}{2}\right]$ and θ as $\cos^{-1}(\hat{r}\cdot\hat{l})$
    7. Define δ, such that $\delta = \theta - 90^{\circ}$
      `if` $|\delta| < \delta_c$
        Define $\hat{l} : \hat{l}\cdot\hat{l} = 0$ *and* $\hat{z} : \hat{z} = \hat{l} \times \hat{l}$ *and* $\hat{z} > 0$
        Define $C_1 = sign(\delta)$ and $C_2 = sign\ (\hat{r}\cdot\hat{l})$
        ($sign(a)$ function outputs 1 if $(a) > 0$ and outputs -1 if $(a) < 0$)
        $N =$ **input** $[C_1 \quad C_2]$
          `switch` N
            `case` $[-1 \quad 1]$
              $\hat{d} = \hat{l}$ // Force on A towards O, shown in Fig.6(a)
            `case` $[1 \quad 1]$
              $\hat{d} = -\hat{l}$ // Force on A away from O, shown in Fig.6(b)
            `case` $[1 \quad -1]$
              $\hat{d} = \hat{l}$ // Force on A away from O, shown in Fig.6(c)
            `case` $[-1 \quad -1]$
              $\hat{d} = -\hat{l}$ // Force on A towards O, shown in Fig.6(d)
        Compute $\overrightarrow{f}_{Torsional}(l, l_0)_{A_iB_i}$ on A_i due to neighbor point B_i as a result of torsional spring using equation (10) with direction $\hat{d}$
        Update $\overrightarrow{F_i} = \overrightarrow{F_i} + \overrightarrow{f}_{Torsional}(l, l_0)_{A_iB_i}$
    8. Update point coordinates of P to $P' = [x' \quad y']_{m\times 2}$ such that $P' = P + \Delta t\overrightarrow{F}$.
9. Run TSP solver on $P' = [x' \quad y']_{m\times 2}$

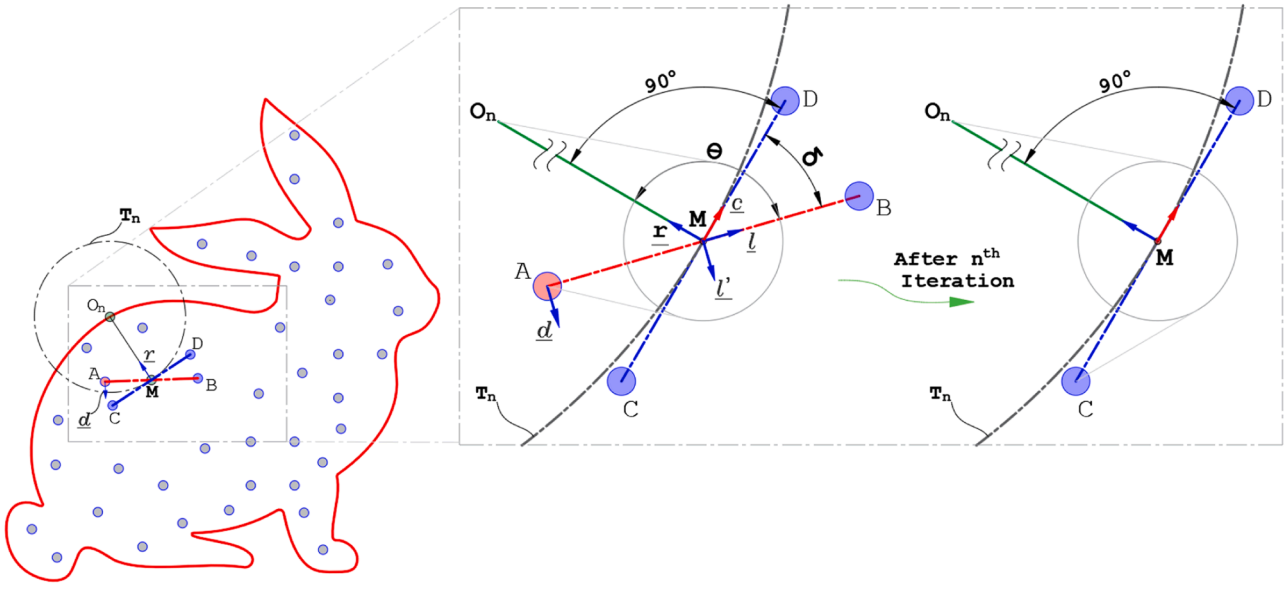


Fig. 7. Generation of contour adaptive grid through the selection of grid-point-specific track center.

Table 3
Algorithm for generation of a contour-adaptive grid using conditions of force balance

```
while (True):
  If convergence criteria == True:
    Break
  Else:
  for all 'P' within contour
```

1. Consider a grid point (say) $A_i(x_{A_i}, y_{A_i}) : A_i \in P$, where $P = [x \quad y]_{n\times 2}$

2. Define vector $\overrightarrow{F_i}$ such that $\overrightarrow{F_i} \in \overrightarrow{F}$, where $\overrightarrow{F_i}$ is the net force on a point A_i due to all springs attached to it

3. Get ξ_{Ai} the closest point on boundary(Γ) to point A_i

```
  If l = distance(A_i, ξ_Ai ) < d_min:
```

Compute $\overrightarrow{f}_{Boundary}(l, l_0)_{A_i}$ on A_i due to ξ_{Ai}

Update $F_i = F_i + \overrightarrow{f}_{Boundary}(l, l_0)_{A_i\xi_{Ai}}$

```
  for all 'B_i' within B
```

4. Identify a vector $(\overrightarrow{A_iB_i})$ between A_i and the neighboring grid point (say)$B_i(x_{B_i}, y_{B_i}) : B_i \in B$ and $B \subset P$ where $B = [x_{B_i} \quad y_{B_i}]_{m\times 2}$. '$m$' is the number of neighboring points connected to A_i, as a result of DT. The direction of $\overrightarrow{A_iB_i}$ is denoted by a unit vector (say) $\hat{l}$.

5. Take ξ_{Ai} as the center point O

6. Repeat step 5 of Table 1 for calculating and storing $\overrightarrow{f}_{Linear}(l, l_0)_{A_iB_i}$ on A_i due to B_i as a result of the rectangular grid into the vector $\overrightarrow{F}$. Take $\hat{v}$ as $\overrightarrow{A_iO}$

7. Repeat steps 7 of Table 2 for calculating and storing $\overrightarrow{f}_{Torsional}(l, l_0)_{A_iB_i}$ on A_i due to B_i as a result of the circular grid into vector $\overrightarrow{F}$.

8. Update point coordinates of P to $P' = [x' \quad y']_{m\times 2}$ such that $P' = P + \Delta t\overrightarrow{F}$.

9. Run TSP solver on $P' = [x' \quad y']_{m\times 2}$

2.1. Generation of a square grid

Consider a case where a square grid, which features equal stepover, is to be generated using the force-based algorithm as outlined above. For the generation of such a grid, as illustrated in Fig. 3(b), the points have to be so positioned such that some points assume a position corresponding to the corner points (on the opposite ends of the two diagonals) while others are the side points of a square.

Let $\hat{l}$ be a unit vector in the direction of spring force for a given point in consideration, say 'A' (marked as red in Fig. 3(a)). Similarly, unit vector $\hat{v}$ points to one of the two orthogonal directions (horizontal or vertical) of the square grid. In order to achieve a square grid, the points which are immediate neighbors of 'A' must adjust themselves such that either they assume the position of a 'diagonal point' (domain marked grey) or a 'side point' (domain marked white), as illustrated in Fig. 3. For sorting the points as 'diagonal' or 'side' points, the following argument can be considered.

$$\overrightarrow{l} = (x_B - x_A)\cdot\hat{i} + (y_B - y_A)\cdot\hat{j} \tag{1}$$

$$\hat{l} = \frac{\overrightarrow{l}}{\left|\overrightarrow{l}\right|} \tag{2}$$

$$\gamma = \cos^{-1}(\hat{v}\cdot\hat{l}) \tag{3}$$

where, x_B and y_B are the coordinates of one of the neighbor points (say) 'B' to the point of consideration 'A', x_A and y_A are the coordinates of the point in consideration. The unit vector $(\hat{v})$ and $(\hat{v}')$, defined at 'A', represents the horizontal and vertical direction of a rectangular grid. For a grid point to be considered a diagonal point it must lie in a region where its angle to the horizontal is near to 45° for a square grid. Therefore, following criteria must be satisfied.

$$\nu_1 < |\gamma| < \nu_2 \tag{4}$$

$\nu_1$and ν_2, defined from the horizontal $(\hat{v})$, represents the lower and upper bound of the angle, the domain within which marks the region considered for a grid point to be a diagonal point. For a square grid ν_1 <45° and ν_2 >45°. It is easier to depict, ν_1 and ν_2 using unit cell side length. By defining the diagonal points region by quarter of unit cell side length we can define, ν_1 and ν_2 to be $\tan^{-1}(1/2)$ and $\tan^{-1}(2)$, respectively as shown in Fig. 3. The above-mentioned test in Eq. (4) is carried out for all the neighboring points to the point of consideration. Having categorized the position of a given point into a diagonal (a grey region point) and a side point (a white region), the following criteria are used to enforce a rectangular grid.

1. The optimal spring length (l_0), to be achieved through force balance, corresponding to the diagonal point with A is considered $\sqrt{2}S$, where 'S' is the stepover
2. The optimal spring length (l_0), to be achieved through force balance, corresponding to the side point with A is considered as S
3. The objective of the algorithm is to update the locations of grid points within the contour such that the resulting spring forces, as illustrated in Eq. (5), achieve equilibrium

$$\overrightarrow{f_l}(l,\ l_0) = \begin{cases} k(l_0 - l) & \text{if} \quad (l < l_0) \\ 0 & \text{if} \quad (l > l_0) \end{cases} \tag{5}$$

4. The force calculation is carried out iteratively using the forward Euler method, expressed in Eq. (6), where the new point coordinates $(P_{\kappa+1})$ is derived from its previous state (P_κ) and the force vector $\overrightarrow{F}$, where $\overrightarrow{F}$ represent summation of all forces action in the point in consideration (i.e., $\overrightarrow{F} = \sum\overrightarrow{f_l}$). The system of connected springs is advanced one step at a time (Δt) by first considering a single point (schematically represented in red in Fig 3) and looping through all its neighbouring points as obtained through Delaunay Triangulation schematically represented in blue in Fig 3). Table 1 briefly summarizes the algorithm.

$$P_{\kappa+1} = P'_\kappa + \Delta t\overrightarrow{F} \tag{6}$$

Enforcing the above spring length criteria and force balance ensures a rectangular gird within a specified tolerance.

2.2. Generation of a circular grid

A circular grid can be achieved through an additional consideration of a torsional spring acting alongside linear springs. A grid is circular if it exhibits the following properties.

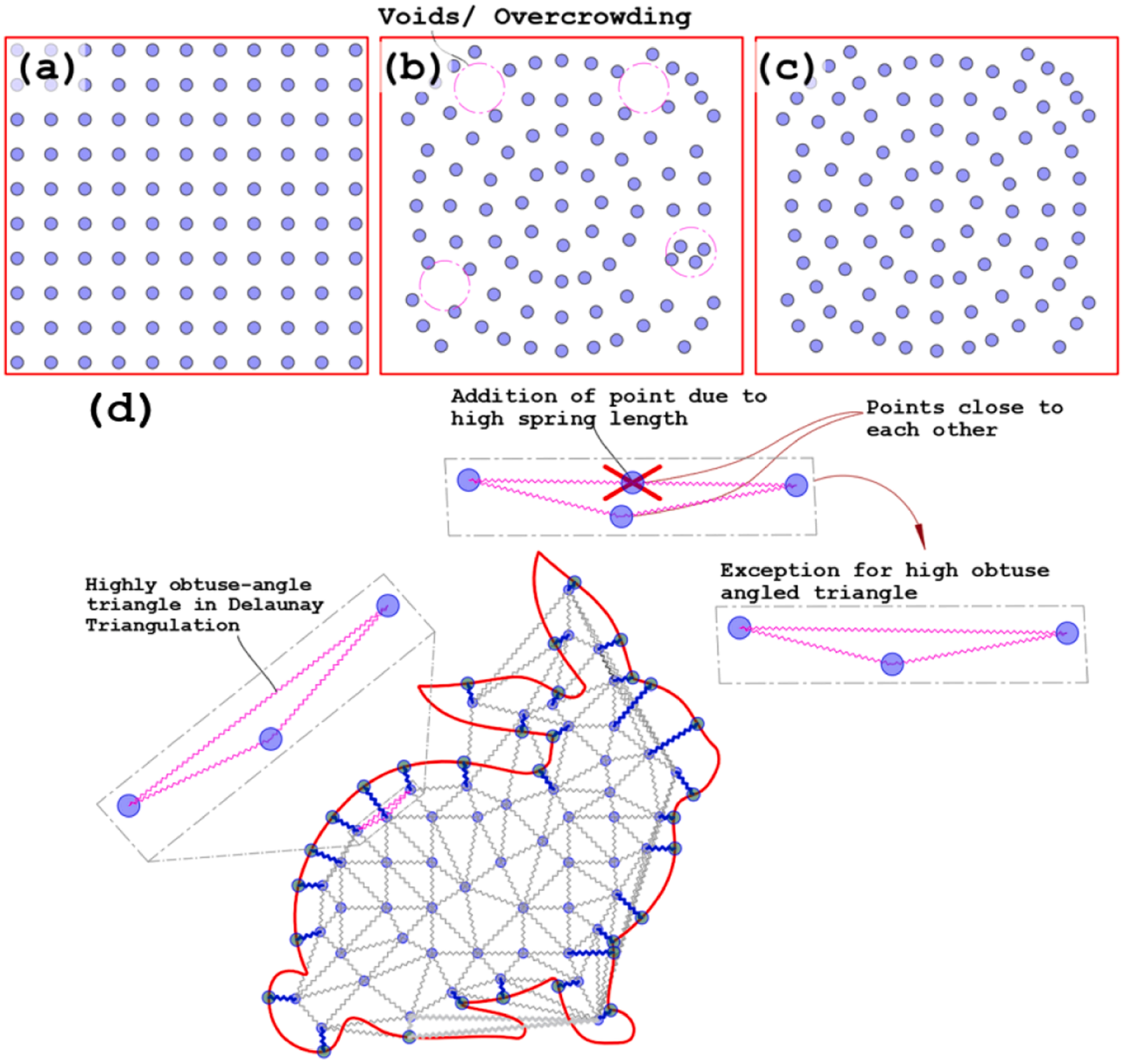


Fig. 8. (a)Rectangular input grid transforming into a circular grid may lead to (b) voids or overcrowding, (c) uniformity of grid points achieved through add/ delete point criteria, and (d) exception for the boundary case for the add/ delete criteria.

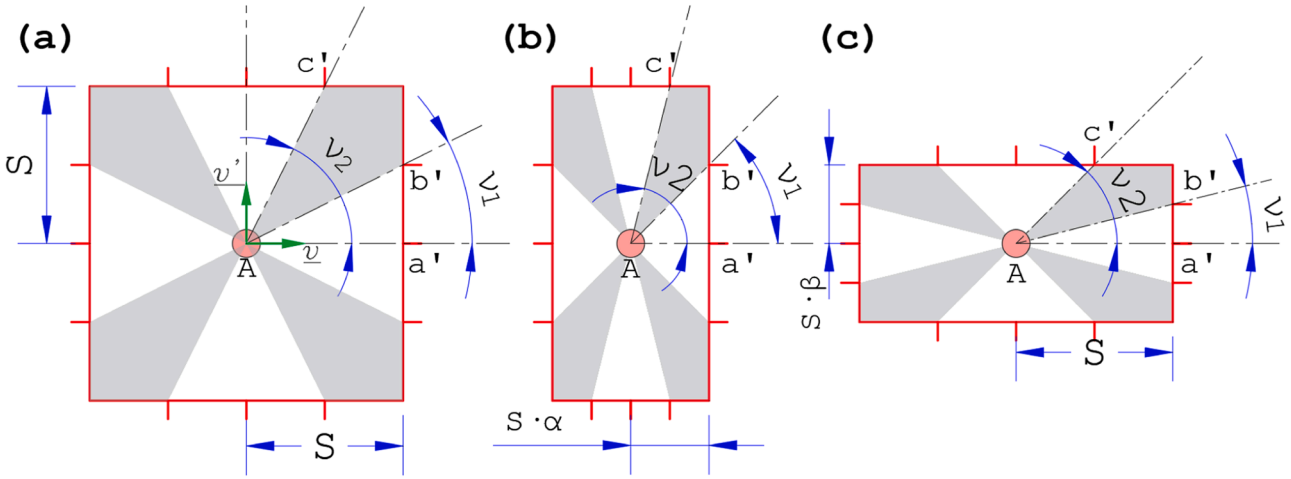


Fig. 9. (a) Equal stepover in all directions for a non-direction-favoring toolpath, (b) reduced stepover along (b) cell length for horizontal, and (c) cell width for vertical direction favoring.

1. The distance between two consecutive circular tracks (T_n and T_{n+1}), as shown in Fig. 4, upon which grid points are placed, is equal to the stepover (S).
2. The distance between two adjacent grid points on a given circular track is equal to stepover.
3. The line segment joining two adjacent points on the same contour should be perpendicular to the radial direction ($\hat{r}$).

The directions ($\hat{v}$) and ($\hat{v}'$) of the rectangular grid now correspond to ($\hat{R}$) and ($\hat{\phi}$) respectively, which are defined from the input center position of the circular grid. The 'zone of influence' for the torsional spring is schematically shown in Fig. 4. The 'zone of influence' is primarily based on the proximity of a given grid point to its respective circular track (T_1, T_2, $T_3 \ldots T_n$).

Consider the grid point 'A', marked in red in Fig. 5. The line segment AB, formed by joining the neighboring grid points (lying within the zone of influence for A), makes an angle θ with the radius of the track T_n centered at O, and contains the point M, as illustrated in Fig. 5(a). Under such a grid point configuration, the torsional spring is said to be in a deformed state. The spring force from the deformed torsional spring adjusts the line segment AB (Fig. 5(a)) to CD (Fig. 5(b)) such that the angle θ and δ is 90° and 0° respectively, as shown in Fig. 5(b). The torsional spring can be assumed to be anchored at M in Fig. 5, with its two ends attached to A and B. τ_d and τ_n are the angles between the free (connecting segments) ends of the torsional spring under deformation and its natural rest state, respectively, as illustrated in Fig. 5(a) and Fig. 5(b).

Let $\hat{r}$ be a unit vector pointing towards the center from the midpoint of line segment AB. The angle θ formed between unit vectors $\hat{l}$ and $\hat{r}$, representing directions along the line segment (AB, in Fig. 5) and the radial direction of the circular track T_n, respectively, is expressed by Eq. (7). Consider a unit vector $\hat{c}$, which is perpendicular to $\hat{r}$ and oriented along the general direction as $\hat{l}$. An angle δ is defined between the unit vectors $\hat{c}$ and $\hat{l}$, mathematically expressed by Eq. (7). The torsional spring, ideally, applies a force at point A on the line segment $\overrightarrow{AB}$ such that an angular deflection, about point M, of magnitude δ orients $\overrightarrow{AB}$ to $\overrightarrow{CD}$, as shown in Fig. 5(b). The torque on $\overrightarrow{AB}$ consists of two parallel forces, one acting on A and the other on B; both are equal and opposite to each other but are mutually perpendicular to AB.

$$\delta = \theta - \pi/2 \quad \text{where } \theta = \cos^{-1}(\hat{r}\cdot\hat{l}) \tag{7}$$

If the value of δ is smaller than a user-defined critical value (δ_c), i.e., $|\delta| < \delta_c$ then, the grid points are considered within the zone of influence of the torsional spring. While the value of δ_c is user input, currently, it is arbitrarily compared with a value of $\pi/9$ for quantifying the zone of influence. The torsional spring exerts a torque T_{AB} on line segment AB for which magnitude is given by Eq. (8).

$$|T_{AB}| = k_t \cdot \delta \tag{8}$$

This torque moment is replaced by an equivalent force-couple f_A and f_B acting on points A and B, respectively, as shown in Fig. 5(a). Equating the moment due to force-couple to the moment due to an imaginary torsional spring, expressed in Eq. (9), gives us the value of force acting on point A due to the torsional spring between points A and B.

$$|T_{AB}| = 2 \cdot f_t(\delta) \cdot \frac{l}{2} \tag{9}$$

From Eq. (9) and (10)

$$f_t(\delta) = \frac{k_t \cdot \delta}{l} \tag{10}$$

The magnitude of the force on points A and B is expressed in Eq. (10), where l is the length of line segment AB and k_t is the torsion spring constant. The forward Euler method of integration, as described for linear spring forces, over an incremental time step of Δt is implemented for the calculation of updated point position due to torsional spring force ($\vec{f_t}$). For a line segment formed by joining the grid point in consideration ('A') to its neighboring point ('B') ($\overrightarrow{AB}$ in Fig. 5), represented by the unit vector $\hat{l}$, two possibilities determine the direction of torsional force. First, as shown in Fig. 6(a) and (d), point 'A' must be pulled inwards towards the center 'O'. This tuns $\overrightarrow{AB}$ counter-clockwise for Fig. 6(a) and clockwise for Fig. 6(d), thereby achieving the desired orientation $\overrightarrow{CD}$. Second, as shown in Fig. 6(b) and (c), point 'A' must be pushed outwards away from the center 'O'. This tuns $\overrightarrow{AB}$ clockwise for Fig. 6(b) and counter-clockwise for Fig. 6(c) thereby achieving the desired orientation $\overrightarrow{CD}$. The unit vector $\hat{d}$, defined at point A, represents the direction of pull/ push (toward and away) relative to point O. Therefore, additional force in $\hat{d}$ direction is added to Eq. (5) for aligning the grid towards a center point. The force on a point due to a neighboring point on same contour is given by:

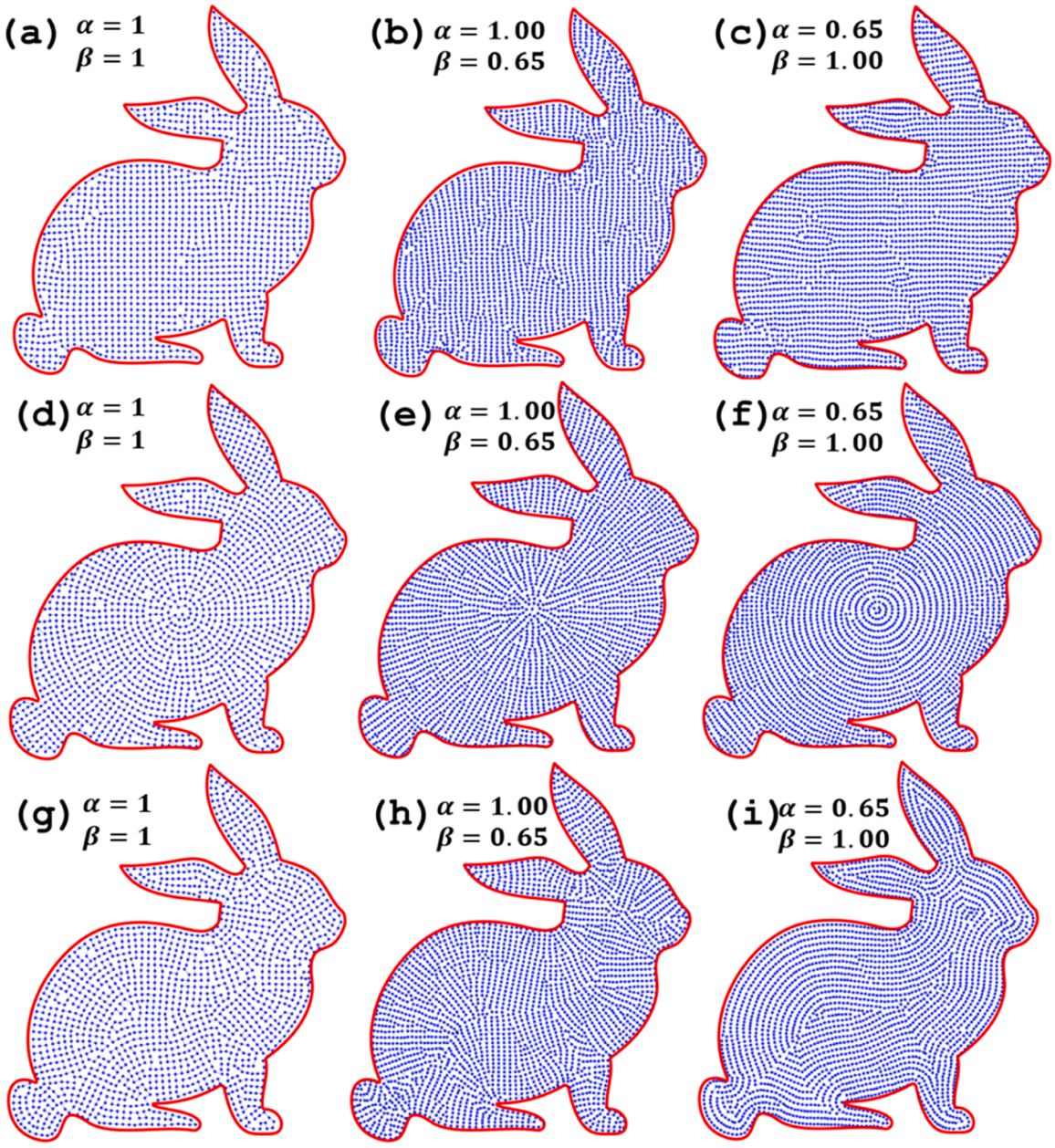


Fig. 10. Digitized space within the contour corresponding to (a) rectangular no favoring, (b) rectangular vertical, (c) rectangular horizontal, (d) circular no favoring, (e) circular-radial, (f) circular-azimuthal, (g) adaptive no favoring, (h) contour-orthogonal, and (f) contour-parallel favoring.

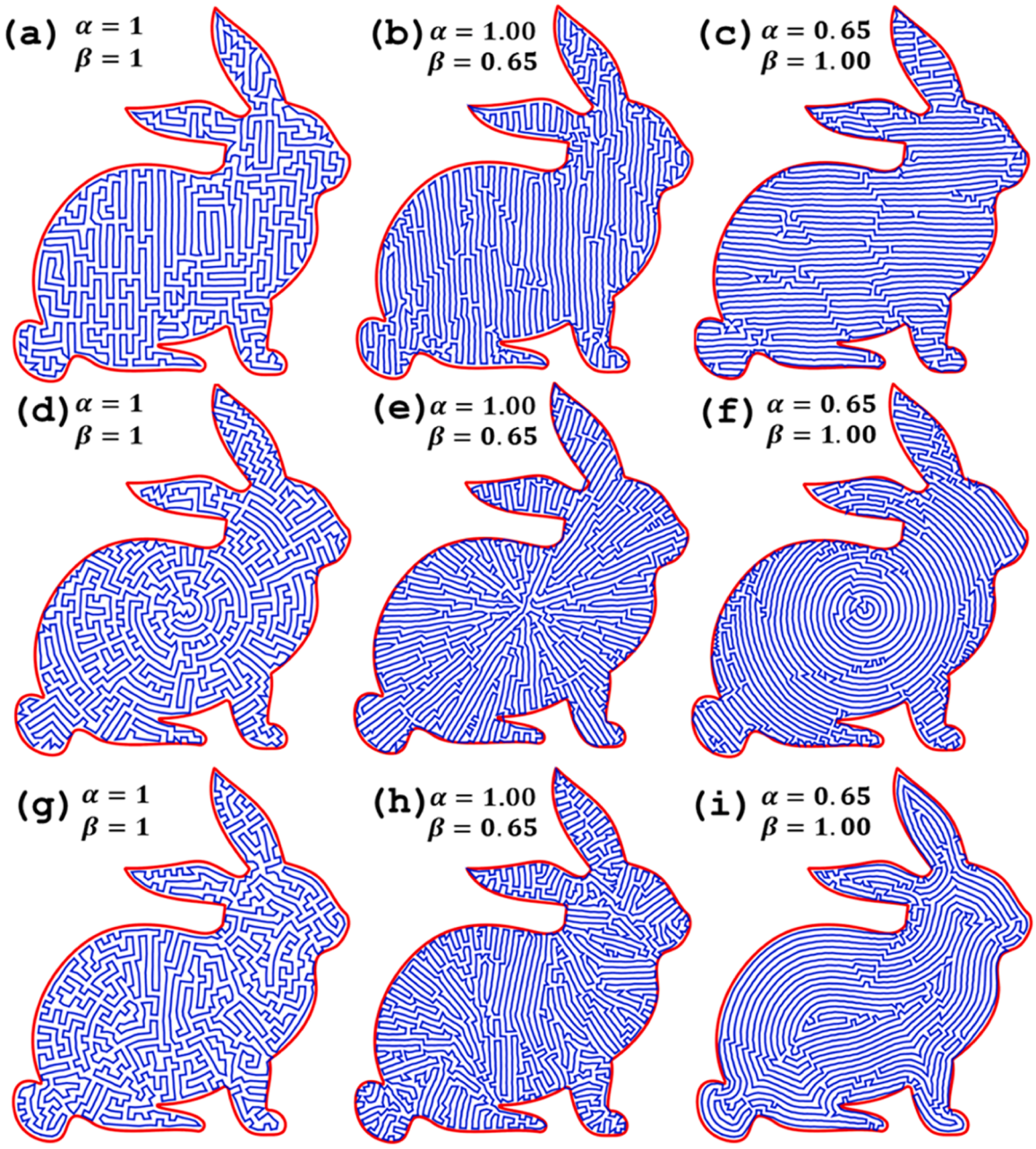


Fig. 11. TSP-based toolpath for (a) rectangular no favoring ($N = 720$), (b) rectangular vertical($N = 532$), (c) rectangular horizontal($N = 601$), (d) circular no favoring($N = 832$), (e) circular-radial($N = 781$), (f) circular-azimuthal($N = 647$), (g) adaptive no favoring($N = 926$), (h) contour-orthogonal($N = 829$), and (f) contour-parallel favoring ($N = 446$).

$$\overrightarrow{f_l}(l,\ l_0, \delta) = \begin{cases} -k(l_0 - l)\ \widehat{l} + \dfrac{k_t \cdot \delta}{l}\ \widehat{d}, & \text{if} \quad (l < l_0) \\ \\ 0\ \widehat{l} + \dfrac{k_t \cdot \delta}{l}\ \widehat{d}, & \text{if} \quad (l > l_0) \end{cases} \tag{11}$$

Table 2 outlines the algorithm for obtaining the magnitude and the direction of the torsional spring force

2.3. Generation of an adaptive grid

Adaptive grid results in an area-filling toolpath that mimics the contour boundary. This toolpath can be considered more or less a continuous equivalent of the common contour-parallel or contour offset infill. Implementing a TSP-based solver to generate such toolpaths results in minimal lifts with a common start-end point, thus being more advantageous over similar area-filling strategies. Furthermore, a force-based digitization algorithm, as proposed here, offers an added ability to fabricate parts with graded density while maintaining a general contour-adaptive property.

The generation of the adaptive grid, in most aspects, follows the same algorithm as that of the circular grid. The deviation in the algorithm from circular to adaptive emerges from the implementation of a grid-point-specific track center (O_n). As discussed in Section 2.2, for a circular grid, the radial direction ($\widehat{r}$) corresponds to the fixed user-defined track center (O). However, for the adaptive grid, the radial direction ($\widehat{r}$) varies in accordance with the 'current' grid point in consideration. The track center (O_n) for the current grid point (A) corresponds to the nearest point on the contour (C) to the midpoint of the line joining circular grid points pair, as illustrated in Fig. 7. Consequently, as the

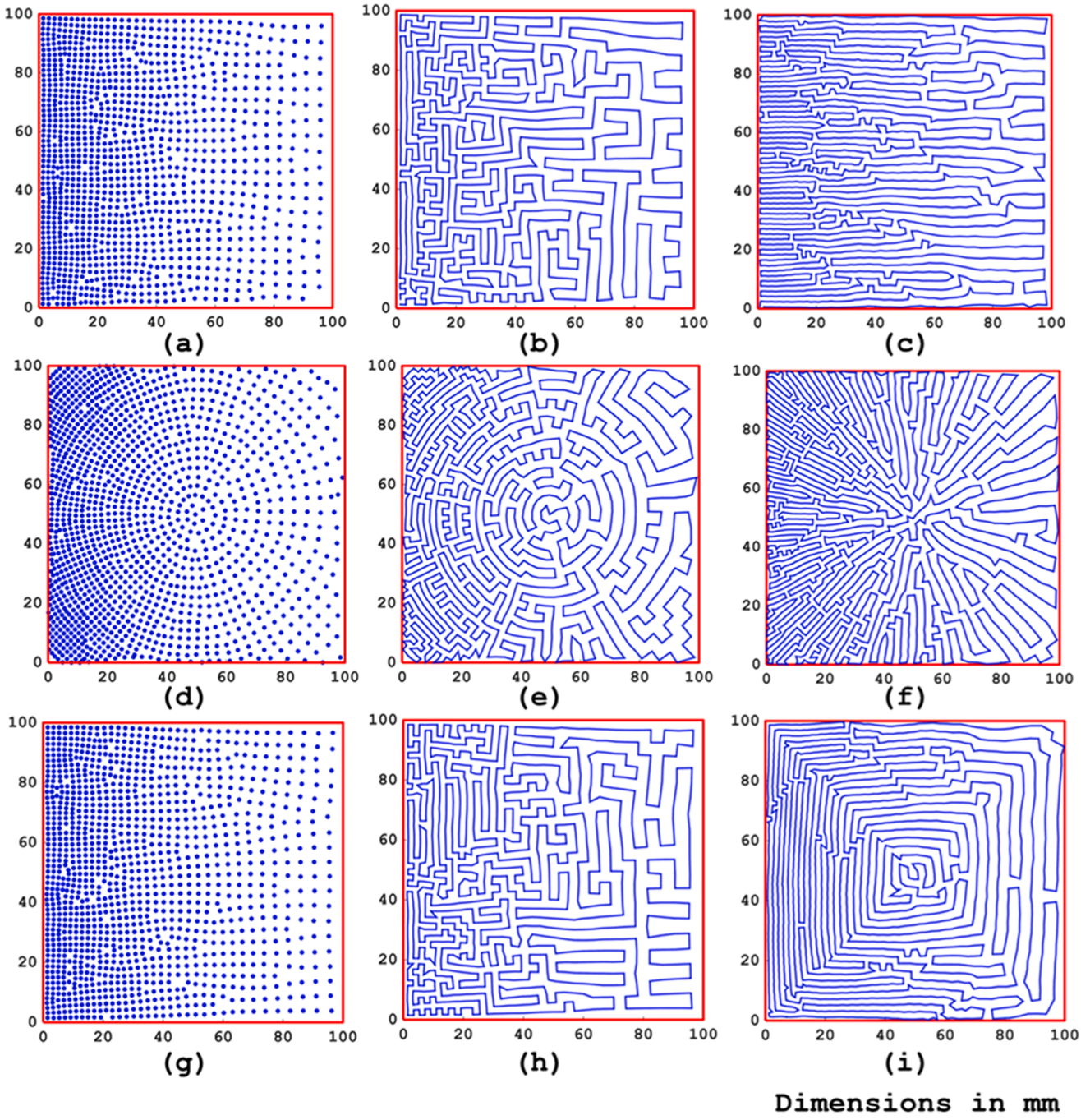


Fig. 12. Digitization with gradient toolpath for (a-b) rectangular grid, (c) rectangular-horizontal biased, (d-e) circular grid, (f) circular-radial biased, (g-h) adaptive, and (i) adaptive contour parallel toolpaths.

algorithm loops through every grid point (updating A), the track center updates, thus updating the radial vector ($\widehat{r}$). This forces the resulting array of grid points to be conformal to the outer contour (C).

It can be observed that as the grid point in consideration (A) changes, the basis vectors ($\widehat{l}$, $\widehat{\varphi}$, and $\widehat{R}$) update themselves, thus resulting in an adaptive grid. The following Table 3 highlights the key aspects of the implemented algorithm.

2.4. Add/ delete points

The array of grid points within a contour can have either an initial user-defined unstructured or a structured configuration. The unstructured configuration implies a random fixed collection of points, whereas a set of grid points following a rectangular, circular, or hexagonal motif is termed a structured configuration. Irrespective of the starting configuration, upon implementation of a force-based algorithm to obtain a structured configuration, the digitization of the space may be non-uniform. The non-uniformity primarily results from a fixed number of grid points transforming from one configuration to the other, leaving either voids or overcrowding of the grid points. Fig. 8 shows a rectangular array of grid points transforming into a circular array of non-uniform 'local' density.

The algorithm implements a check on the spring length (l) after every n^{th} iteration (user-defined). If the spring length exceeds 1.3 times its own value, in the subsequent iteration, a new grid point is added at its center. Conversely, a reduction in spring length below $0.6S$, deletes a point from

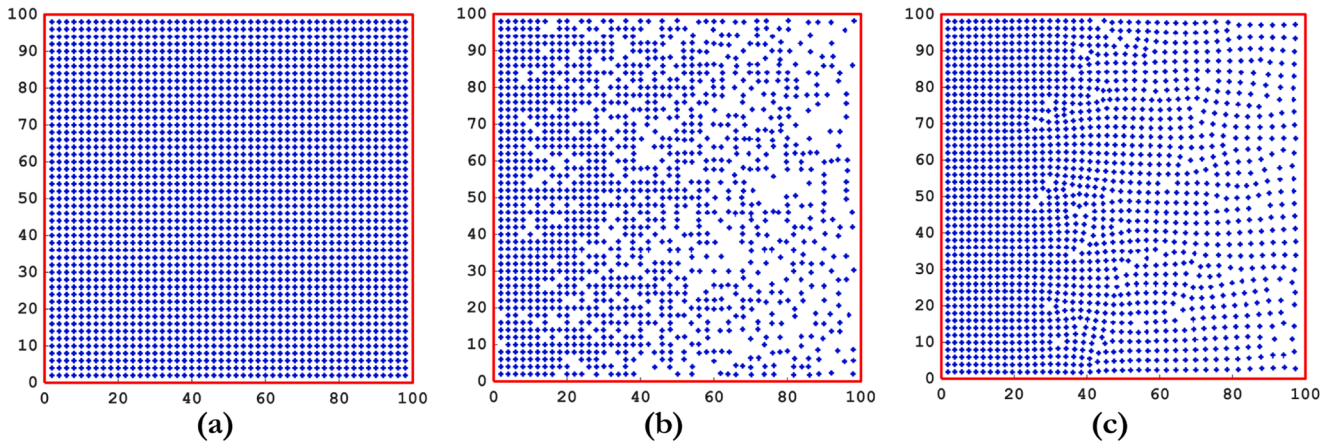


Fig. 13. (a) Rectangular grid with density =1, (b) Points deleted randomly with respect to user defined density function $d(x, y)$, (c) Iteratively solved for final grid. Number of points adjusted after every 5 iterations by Add/Delete point function as described in Section 2.4.

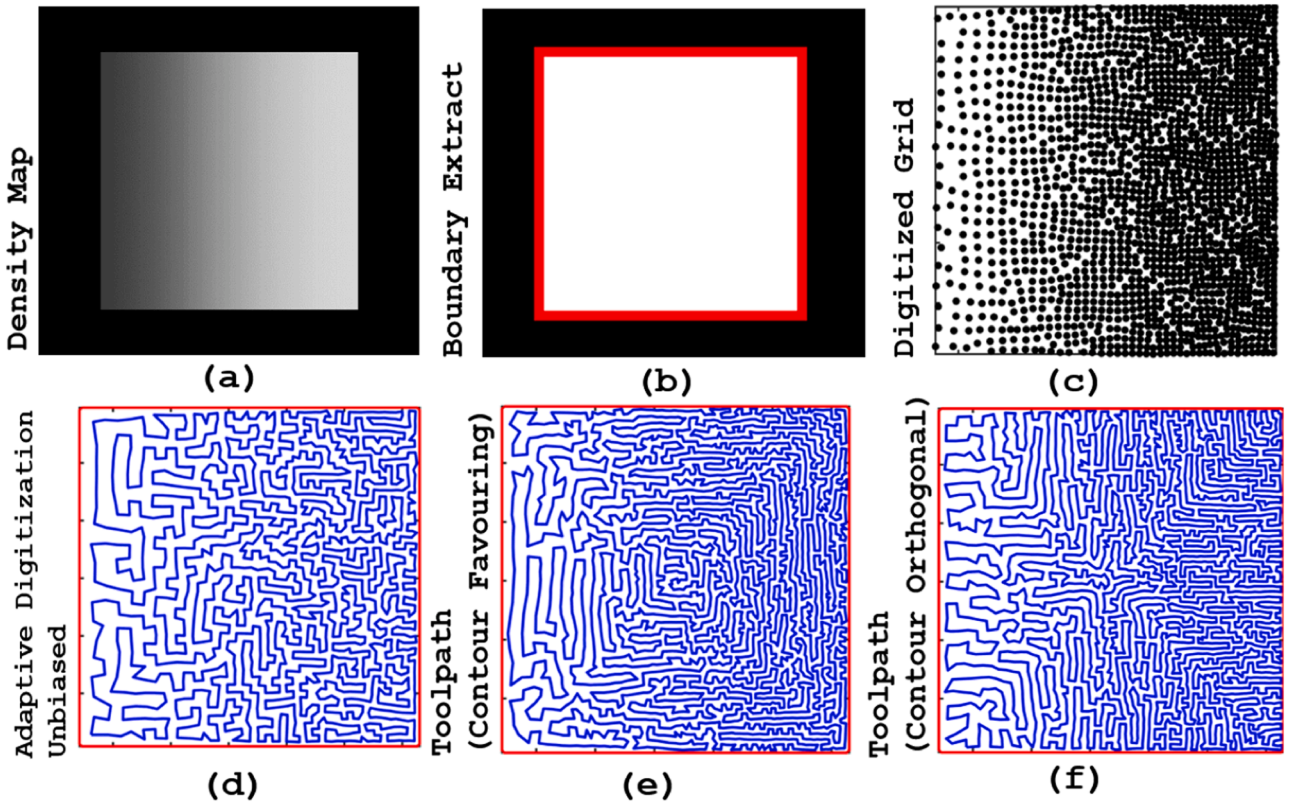


Fig. 14. Density gradation of toolpath prescribed by the user-input density map.

the spring pair in the subsequent iteration.

An exception has to be made for the grid point lying immediately adjacent to the boundary, which, when triangulated, may produce highly obtuse triangles, as shown in Fig. 8(d). In such cases, an additional point on the side opposite to the obtuse angle will result in the newly added point to be very close to the point associated with the obtuse angle. Thus, an additional constraint is enforced on the grid points neighboring the boundary, i.e., for highly obtuse-angled ($> 160^{\circ}$) triangles, no consideration is made for adding the grid points.

2.5. Direction favouring toolpath

Direction favoring is a unique aspect of the proposed toolpath, wherein the deposition direction (or scanning) can be biased by prioritizing one travel direction over another, as shown in Fig. 11. This is achieved through control variables α and β. Each of the three digitization methods, i.e., rectangular, circular, and adaptive, through appropriate selection of control variables, can be designed so that the number of points along a given basis vector is more than its orthogonal counterpart. Consider Fig. 9(a); a unit cell ($S = 1$) is placed at the point of consideration A, and the space within the unit cell is categorized into regions of side and diagonal points, represented with white and grey regions, respectively, as discussed in Section 2.1. These regions are quantified through angles ν_1 and ν_2. The control variables α and β are associated with $\widehat{\nu}$ and $\widehat{\nu^{\perp}}$ direction of travel, respectively. A value of (say) 0.5 for α allows for a direction favoring in $\widehat{\nu}$ by essentially reducing the stepover from S to $s/2$ along $\widehat{\nu}$, as shown in Fig. 9(b). Conversely, a value

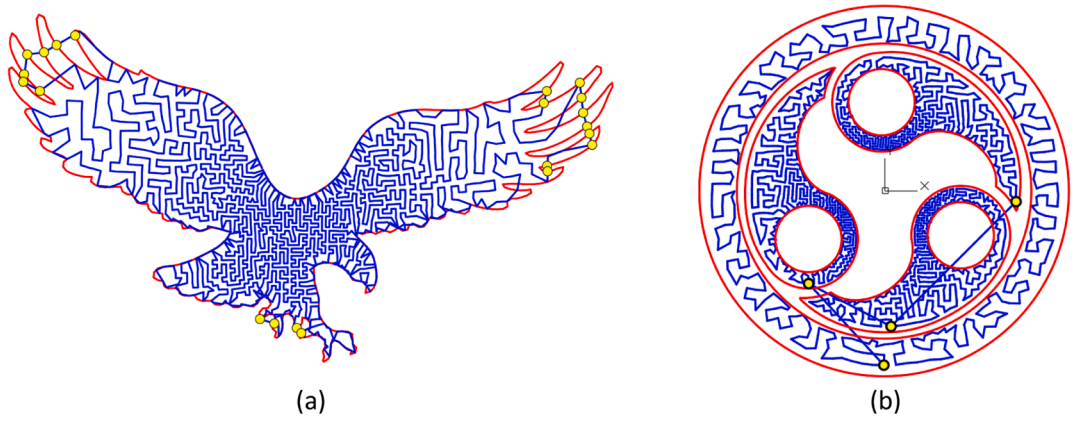


Fig. 15. Retractions added in toolpath due to (a) sharp corners within the contour, (b) multiple contours within a single layer.

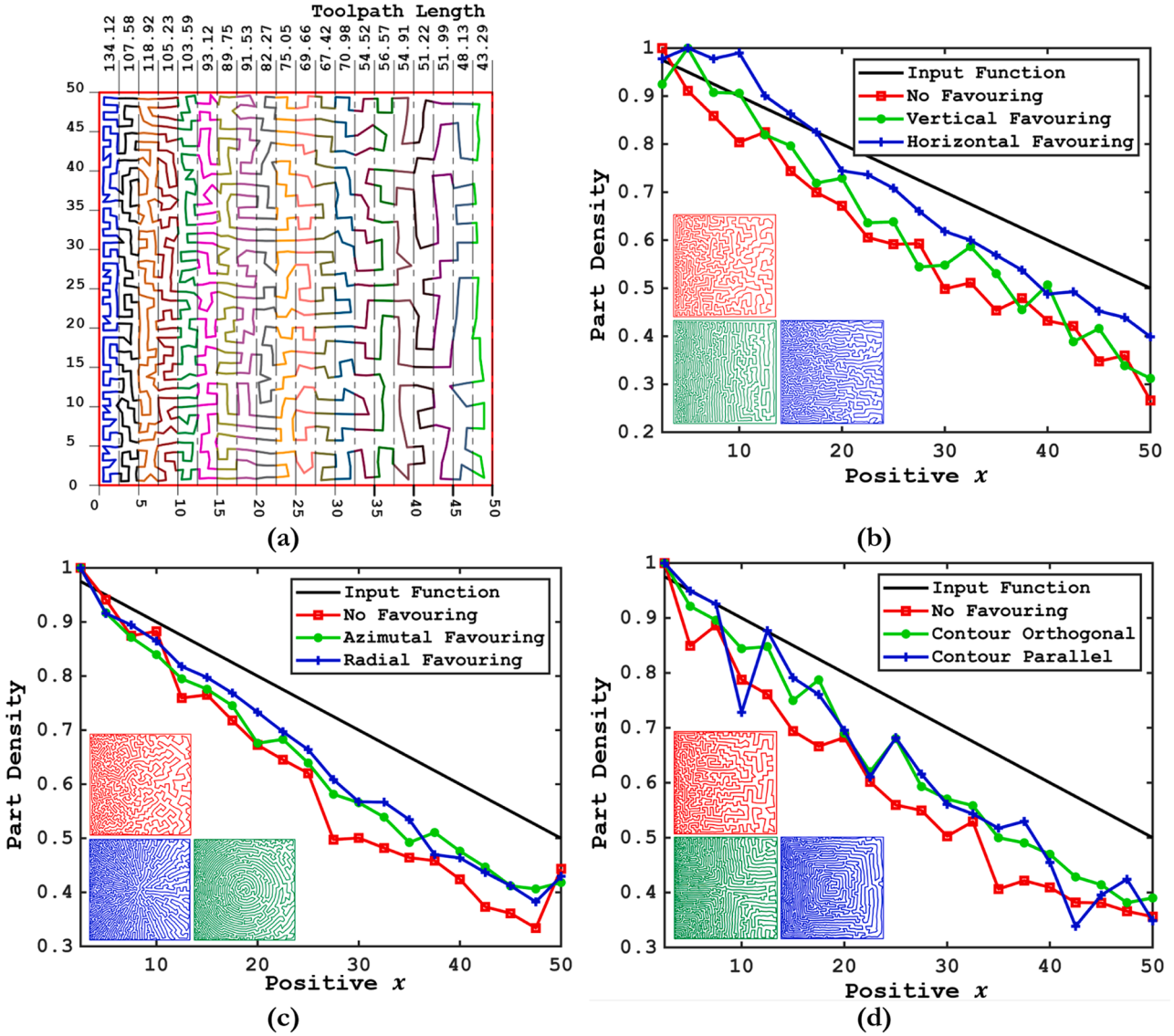


Fig. 16. (a)Process of segmenting toolpaths into blocks and relative part density obtained (after 150 iterations) for the case of (a) rectangular, (b) circular, and (c) adaptive digitized grids.

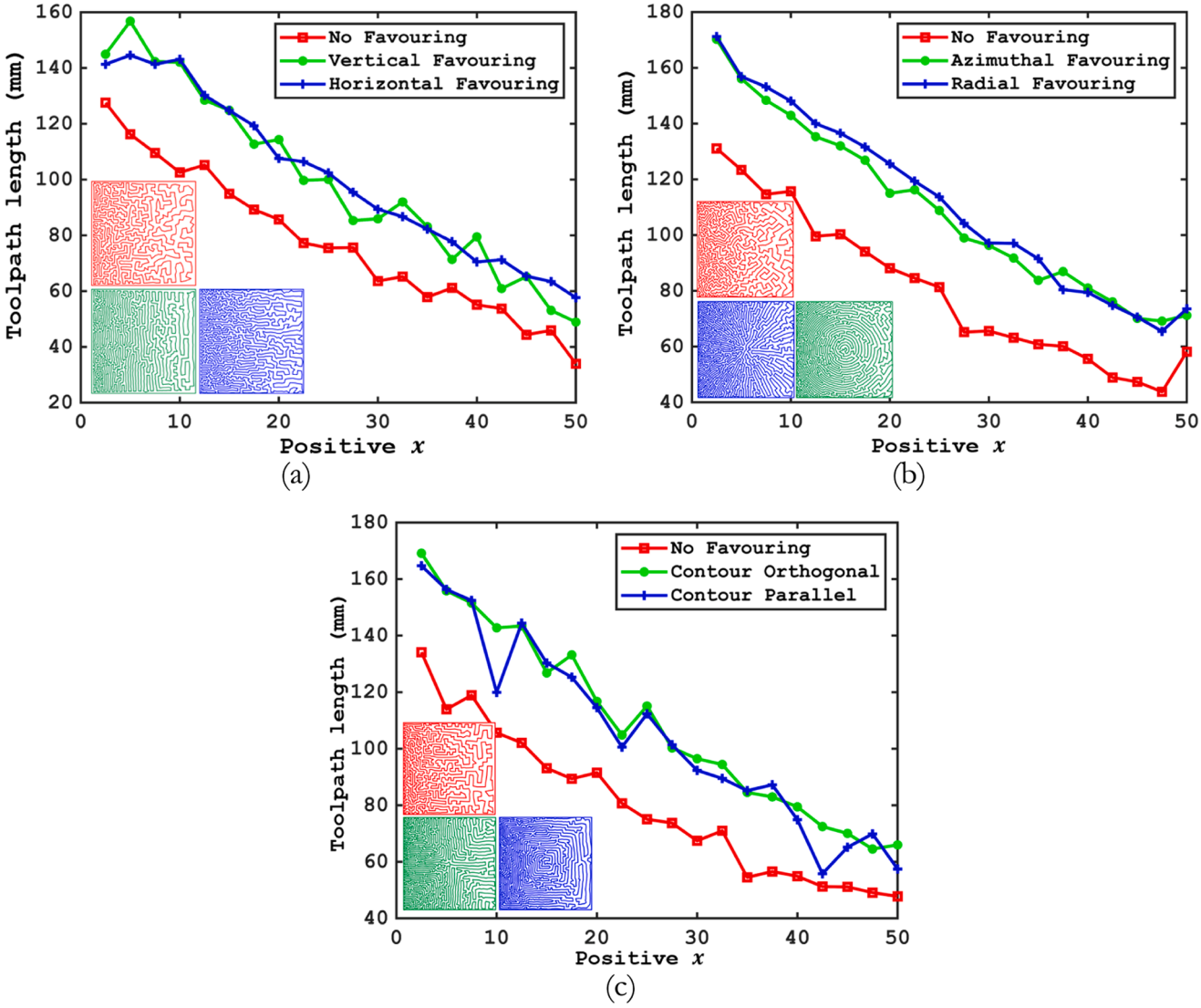


Fig. 17. Toolpath length observed within each segment for **(a)**rectangular, **(b)**circular, and **(c)** adaptive methods of digitization.

of (say) 0.5 for β allows for direction favoring in $\widehat{\nu}$, as shown in Fig. 9(c).

However, introducing these control variables to alter the unit cell changes the span of grey and white regions; thus, the angles ν_1 and ν_2 must be updated accordingly. Eq. (12) expresses the updated angles ν_1 and ν_2 in terms of control variables α and β relative to an altered unit cell with unequal stepover values.

$$\nu_1 \rightarrow \tan^{-1}\left(\frac{\beta}{\alpha}\tan\nu_1\right) \quad \text{and} \quad \nu_2 \rightarrow \tan^{-1}\left(\frac{\beta}{\alpha}\tan\nu_2\right) \tag{12}$$

The conditional statement for a point to be considered a diagonal point in the unit cell, as expressed in Eq. (4), is updated through Eq. (12) to accommodate the control variables for grid favoring. It is to be noted that the spring lengths corresponding to $\widehat{\nu}$, $\widehat{\nu}$, and the diagonal points are multiplied by α, β and $\sqrt{\alpha^2+\beta^2}$, respectively.

A similar approach can be implemented for the case of circular and adaptive grids, wherein the control variables α and β are associated with azimuth ($\widehat{\phi}$) and radial ($\widehat{R}$) directions, respectively. Fig. 10 shows the digitized contour where the grid points are biased in either of the basis directions.

An advantage of the direction-favoring property of the proposed toolpath strategy is its ability to reduce the number of turns (N) in a toolpath, as is evident from Fig. 11. A reduction in the number of turns is often associated with uniformity of material deposition as a consequence of the 'jerk-free' motion of the travel head, especially observed at the corners (or sharp turns) of a toolpath.

2.6. Toolpath for gradient density

In the discussions so far, the optimal length of the spring l_0, as described in Section 2, is constant irrespective of the location of the grid point within the contour. This, upon implementation of the force-based algorithm, results in digitized space with uniform density of the grid points. A gradient in density can be achieved by varying l_0 (consequently stepover S) as a function of 2D space within the contour ($\mathbb{R}^2$). The input arguments for a gradient density are; a governing density function $d(\mathbb{R}^2)$ and an initial stepover value (S). The initial stepover value is specified by the user and corresponds to l_0 at maximum density. The density function $d(\mathbb{R}^2)$ outputs a range of $(0,\ 1]$, with lower and upper limits corresponding to minimum and maximum relative densities, respectively. Thus, the density function $d(\mathbb{R}^2)$ acts as a scaling factor to the initial stepover S to output spring length l_0, expressed through Eq. (13).

$$l_0(\mathbb{R}^2) = \frac{S}{d(\mathbb{R}^2)} \tag{13}$$

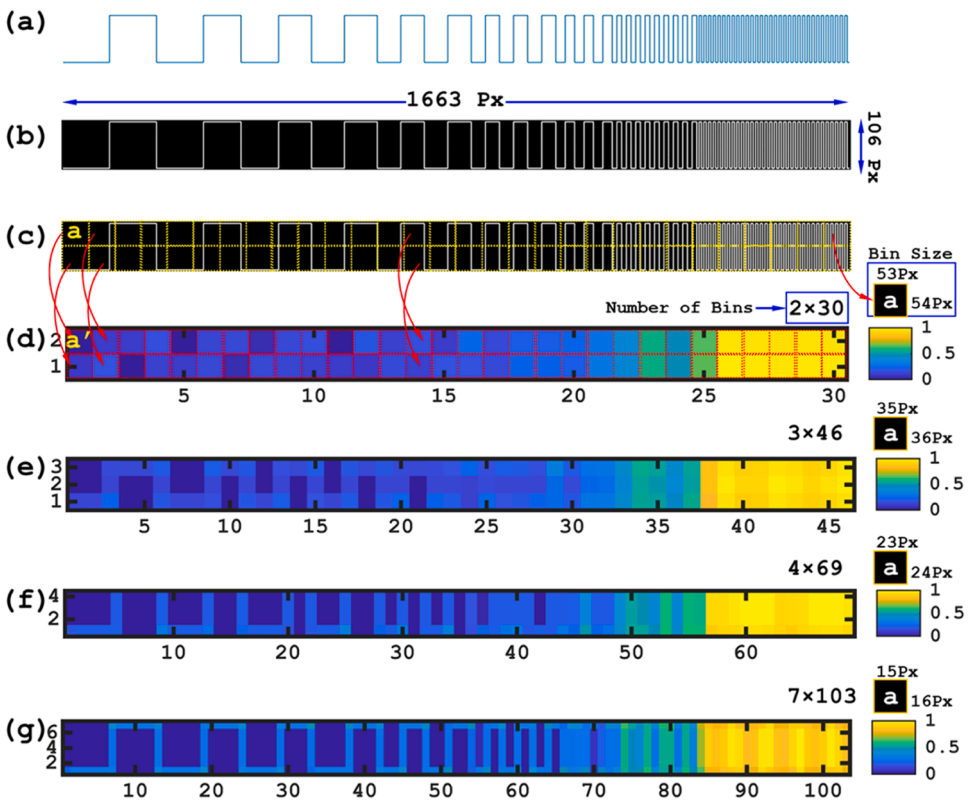


Fig. 18. **(a)** Example toolpath **(b)** grayscale image of the toolpath **(c)** bin size over which averaging is carried out for **(d)**, and representation of variation in density as a collection of **(e)**138, **(f)** 276, **(g)** 721 bins.

Fig. 12 shows illustrative examples of gradient toolpaths for rectangular, circular, and adaptive digitizations with and without direction favoring. The gradient in the toolpath (for $S = 2$) is expressed through Eq. (14)

$$d(x, y) = 1 - (0.5x/100). \tag{14}$$

For achieving the required density function, a starting grid of $d_{initial}(\mathbb{R}^2) = 1$ is assumed. Fig. 13 shows an example of starting grid to be rectangular with $S = 2$. Each point is in this starting grid is passed through a random function which either keeps the point or deletes it. The probability of point deletion ($P_{point\ deletion}$) is based on the used user-defined density function $d(\mathbb{R}^2)$.

$$P_{point\ deletion} = \left(1 - d\left(\mathbb{R}^2\right)\right)$$

Subsequently, the grid is adjusted iteratively by virtual spring forces as mentioned in previous sections.

The minimum and maximum stepover (consequently, the limits of density gradient) that can be achieved depends largely on the AM process that is being used.

2.7. Density-based digitization from image

The digitization of the domain within the contour can also be achieved through an image-based density map. The gradient in the toolpath corresponds to the image intensity of the user-input grayscale image. Consider, for example, Fig. 14; the white and the dark regions of the density map correspond to the maximum and minimum material densities, respectively. In order to establish the gradient of digitization, the input image maps to the part density, where the regions with the smallest and highest intensity values correspond to the minimum and maximum part densities, respectively. The boundaries of the input image can also be extracted through 'thresholding,' where the user sets the upper and lower bounds of the grayscale image. The 'gray level' above and below the upper and the lower bounds is set to one and zero, respectively. The grid points can acquire one of three configurations, rectangular, circular, or adaptive (as outlined in Sections 2.1, 2.2, and 2.3) while preserving the density distribution as dictated by the density map. Furthermore, the direction-favoring method, as outlined in Section 2.5, can be applied to bias the toolpath in one or the other direction.

2.8. Retractions

Solving the Travelling Salesman problem results in a Hamiltonian path which has no retractions. However, the presence of complex geometry imposes limitations on the existence of such closed-loop circuits. Geometries featuring closely positioned sharp corners, as depicted in Fig. 15a, often cause the toolpath to extend beyond the boundary. During digitization, springs were employed to attach points to the boundary contour, thereby ensuring that the points remain within the boundary. Nevertheless, despite the points being confined within the boundary, connecting them using a TSP Solver often causes the toolpath to stray outside of the boundary when local stepover of points is much greater than the complexity of boundary in that region. It can be seen in such regions the toolpath is not able to follow the contour boundary due to lack of number of available points for solving TSP.

Retractions need to be added at such points shown by the yellow points in Fig. 15, to make the toolpath viable. In geometries with multiple separated contours (Fig. 15b) also retraction would need to be added.

2.9. Convergence criteria

A limit on the movement of the grid points corresponding to the spring forces is used to prevent an infinite loop in the algorithm. The stopping criteria can be set by the user, or the program terminates with default values if no input is received. The grid points can be monitored

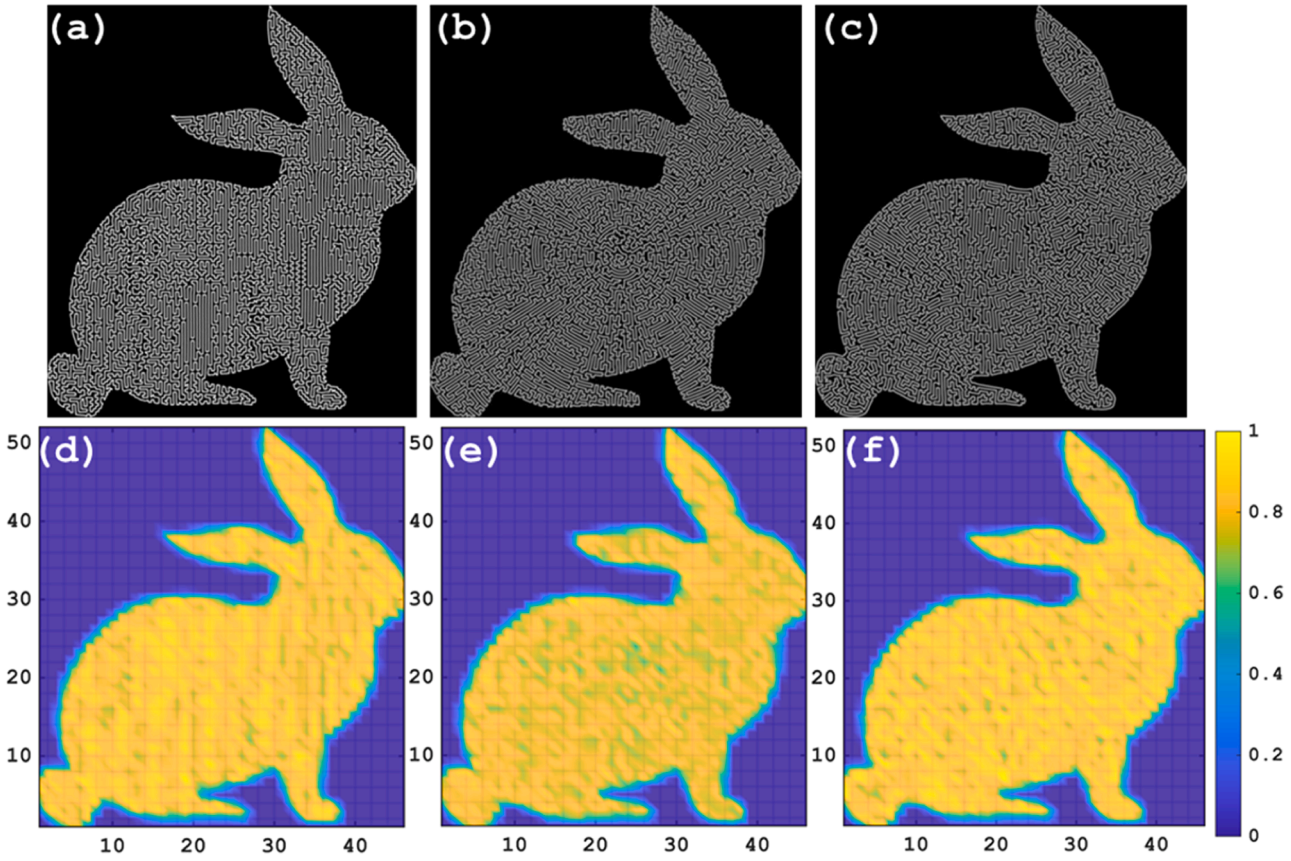


Fig. 19. (a-c)Grayscale image of uniform density toolpaths (obtained after 350 iterations) and their (d-f) corresponding density distribution for (a) and (d) rectangular, (b) and (e) circular, and (c) and (f) adaptive grid.

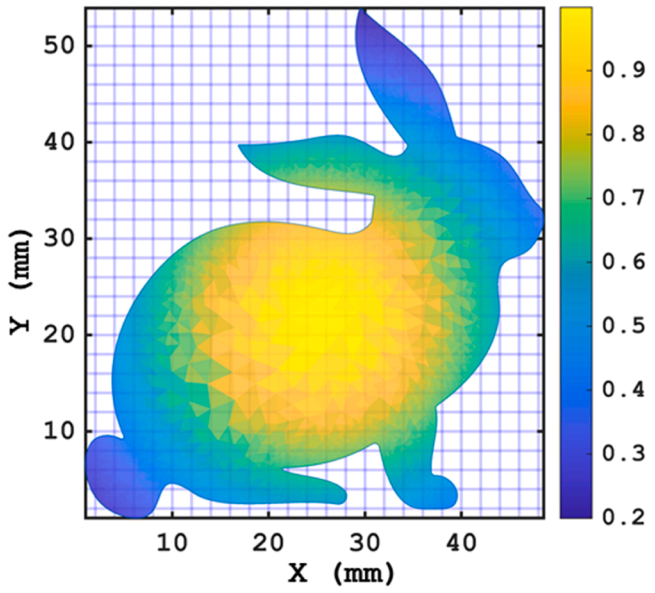


Fig. 20. Ideal density gradient within the contour, corresponding to the input function expressed through Eq. (15).

for the following

1. The maximum value of movement, for a given grid point, in each iteration.
2. The average movement of the collective digitized grid in each iteration.
3. Number of iterations

Another method that produces minimal noise in the convergence graph is to monitor the rolling average of the grid movement for a given iteration. For instance, if the change in the average movement of the grid points in the current iteration is less than a preset value of the previous iteration, then convergence is said to be achieved. The user can also interrupt the iterations upon qualitative inspection of the digitized grid point after a specific number of iterations.

3. Results and discussion

The proposed toolpath was implemented on an FDM-based 3D printer, and each of the digitization methods was explored. The image-based and quantitative analysis of the toolpath was performed to establish the conformance of the toolpath to the input function.

3.1. Toolpath length

Consider a square contour with the gradient toolpath, as shown in Fig. 16. The input function varies linearly in positive 'x' direction and is expressed as: $d(x) = 1 - (0.5x/50)$. The density function implies a fully dense part at the extreme left ($d(0) = 1$) and a 50 percent infill at the extreme right ($d(50) = 0.5$). In order to establish the validity of the variation, the generated toolpath is sectioned into 20 discrete segments

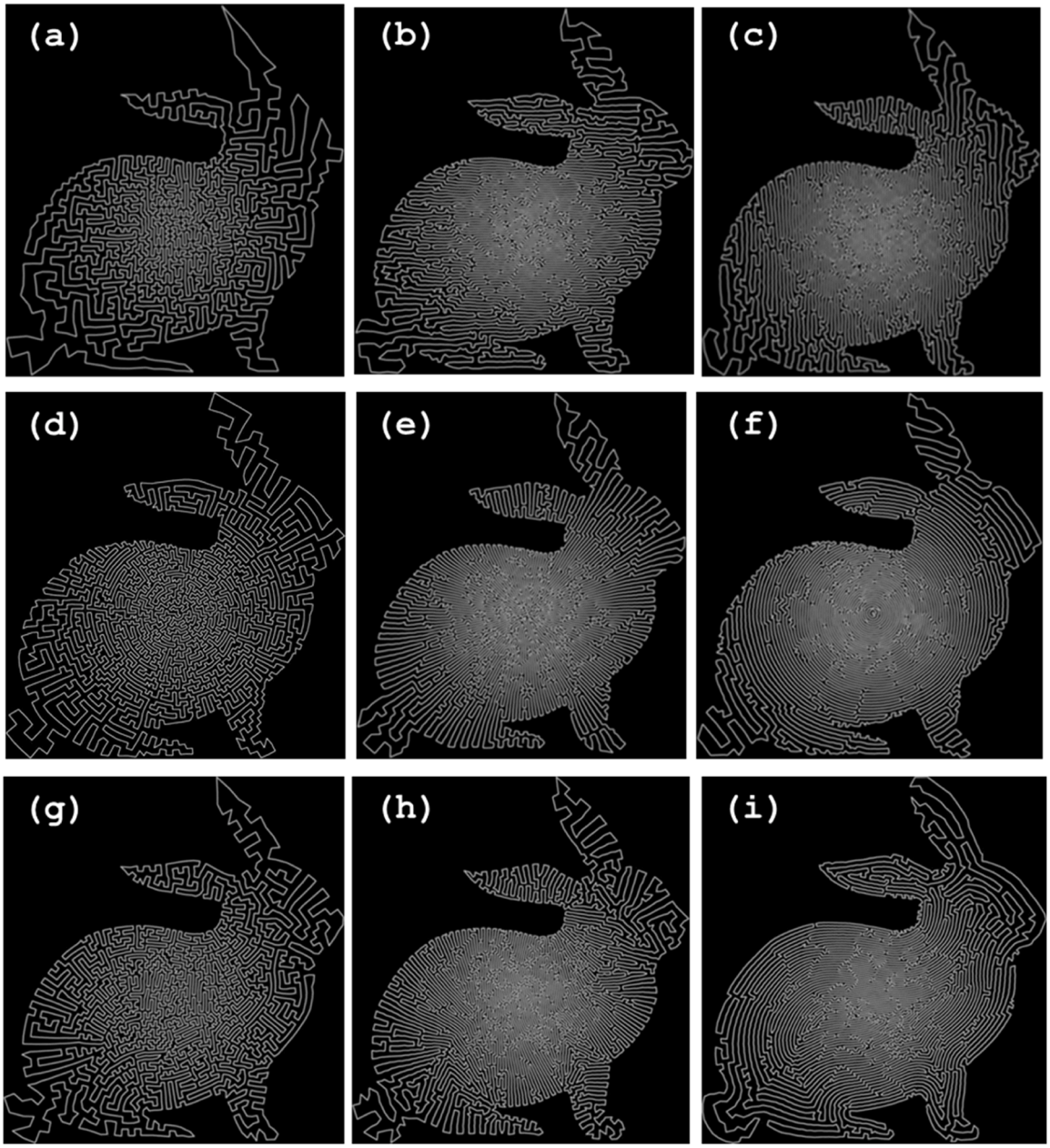


Fig. 21. Grayscale image of the graded-density toolpath following Eq. (11) for rectangular toolpath with (a) no direction favoring, (b) horizontal favoring, (c) vertical favoring; circular toolpath with (d) no direction favoring, (e) radial favoring, (f) azimuthal favoring, and adaptive toolpath with (g) no direction favoring, (h) contour-orthogonal, and (i) contour-parallel direction favoring.

(or blocks) along the positive 'x' direction. The length of the toolpath contained within each segment is measured and is shown in Fig. 16(a). This process is repeated for every category of digitization (rectangular, circular, and adaptive) with and without direction favoring (for the sake of brevity, Fig. 16(a) outlines the processes for rectangular toolpath without direction favoring only). Considering a unit bead width, the toolpath length for each category of digitization is normalized by the maximum value to obtain relative part density. The maximum deviation from the intended part density occurs on the far right of the square contour, as is evident from Fig. 16(b-c). However, globally speaking, the part density follows the same trend as the input function. The toolpath was obtained after 150 iterations.

Another observation made for the case of toolpath length within each segment suggests, irrespective method of digitization, the toolpath length for non-favoring cases is lower than the direction-favoring counterpart, as illustrated in Fig. 17. This can be primarily attributed to the additional grid points introduced during digitization leading to a higher toolpath length.

3.2. Image-based gradation analysis

In this method, the spatially graded toolpath is first saved as a grayscale image with a specified resolution, where the image intensity varies from zero to 255. This image is further processed using a block mean method. The intensity values in the grayscale image contain, say 'r' and 'c' number of rows and columns, respectively, where each element of the matrix holds an intensity value of the pixel at that location. This intensity matrix is sectioned into smaller sub-matrices or bins with a user-specified number of rows (r_{avg}) and columns(c_{avg}). The elements in the intensity matrix are then replaced by the average values of these sub-matrices. The resulting matrix, termed the 'block mean' matrix, represents the variation in global part density as a collection of a

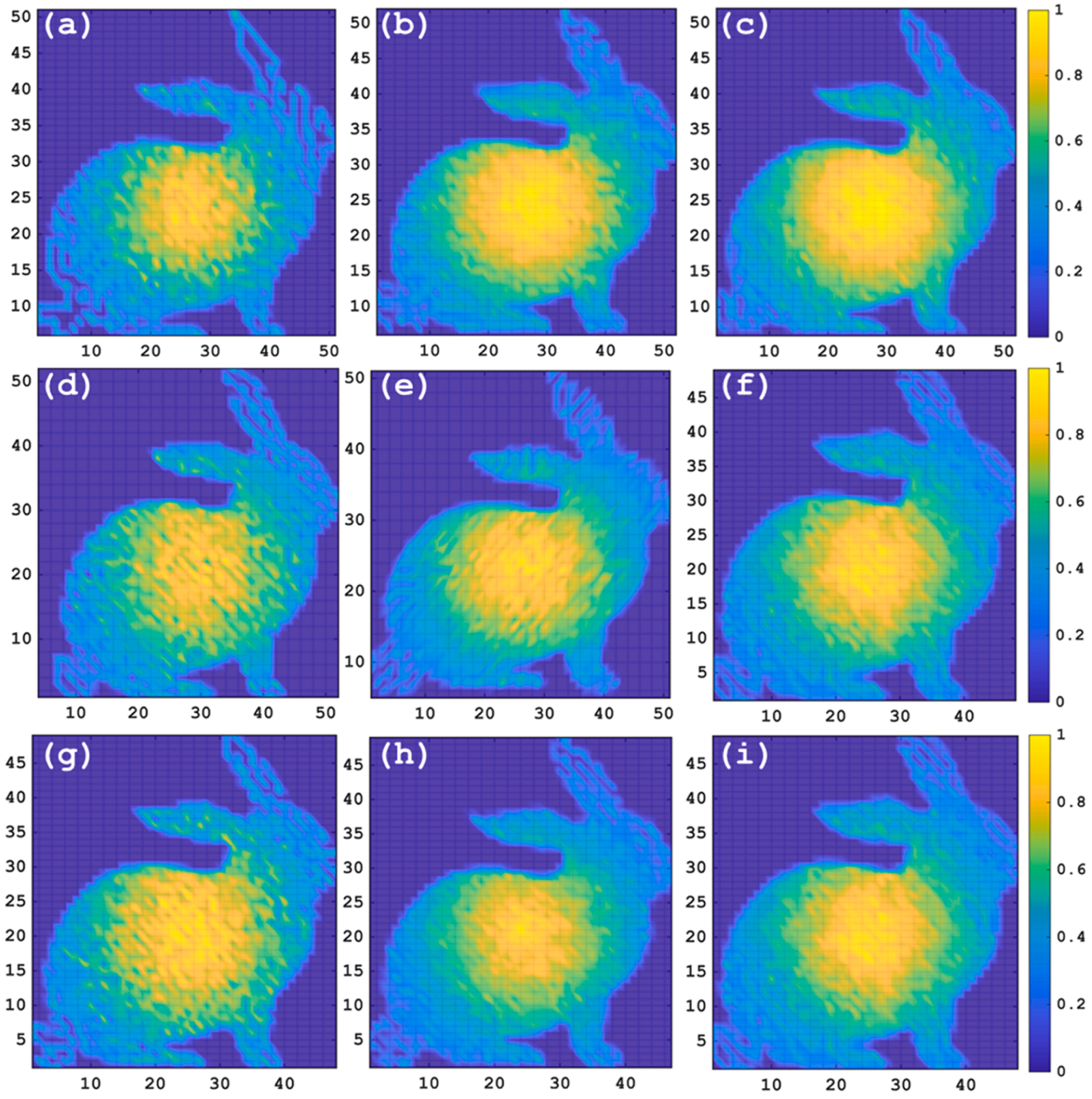


Fig. 22. Density distribution of the graded toolpath following Eq. (15) for rectangular toolpath with (a) no direction favoring, (b) horizontal favoring, (c) vertical favoring; circular toolpath with (d) no direction favoring, (e) radial favoring, (f) azimuthal favoring, and adaptive toolpath with (g) no direction favoring, (h)contour-orthogonal, and (i) contour-parallel direction favoring.

specified number of bins. Furthermore, the elements of the block mean matrix are normalized with maximum value to arrive at the relative part density, varying between zero and one. Fig. 18 shows an illustrative example, where Fig. 18(a) shows the gradient toolpath, and Fig. 18(b) is its grayscale image of size $r = 106$Px and $c = 1663$Px. A bin, marked yellow in Fig. 18(c), labeled as 'a' of size $r_{avg} = 53$Px and $c_{avg} = 54$Px, represents the domain over which the intensity values are averaged to obtain Fig. 18(d). Thus, for Fig. 18(d), a collection of uniform intensity bins, two along the width and thirty along the length represents the global variation in density along the toolpath. Furthermore, a selection of even smaller bin sizes, Fig. 18(e-g), would produce a sharper distinction of the toolpath with the space that it fills.

Fig. 19 shows the case for uniform infill, where each of the three digitization categories produced fully dense toolpaths with a stepover of 0.4 mm.

A graded toolpath, with an input function given by Eq. (15), was also fabricated. Fig. 20 shows the ideal gradation to be generated by Eq. (15) within the contour bounds.

$$d(x,y) = \frac{1}{\sqrt{2\pi}} e^{\frac{1}{2}\left(\left(\frac{(x-25)}{18}\right)^2 - \left(\frac{(y-21)}{18}\right)^2\right)*2.5} \tag{15}$$

Fig. 21 shows the grayscale image of the toolpaths corresponding to Eq. (15) for each of the three proposed toolpath varieties, i.e., rectangular, circular, and adaptive with and without direction favoring. The intensity distribution of the toolpaths, as shown in Fig. 22, further

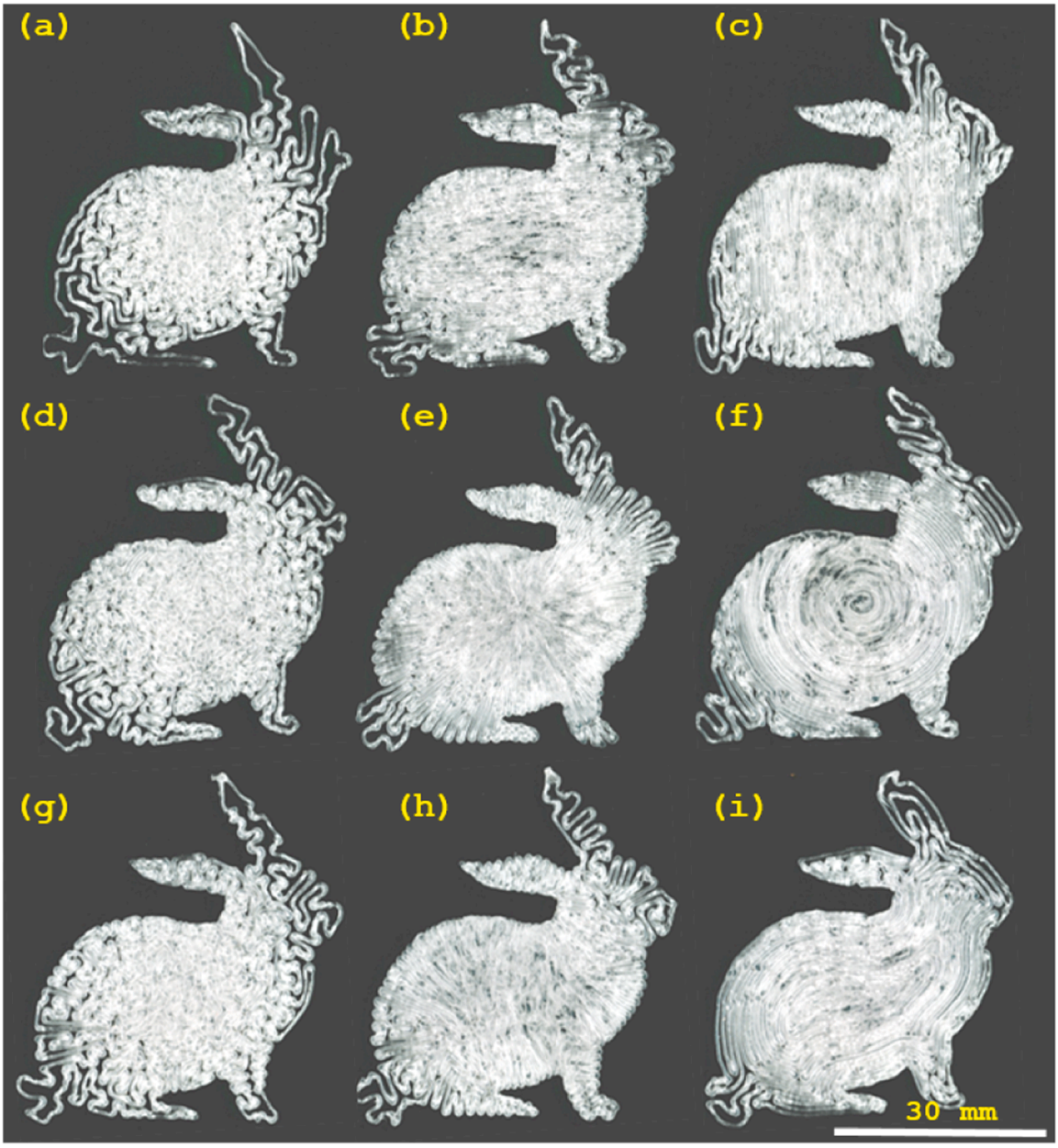


Fig. 23. 3D printed parts using density gradient illustrated in Fig. 20. (a) rectangular no-favoring, (b) rectangular-horizontal, (c) rectangular-vertical, (d) circular no-favoring, (e) circular-radial, (f) circular-azimuthal, (g)adaptive no-favoring, (h) adaptive-contour orthogonal, and (i) adaptive-contour parallel.

Table 4
Weight of printed parts for each category of digitization.

Toopath	Weight (g)
Rectangular (No favoring)	3.95
Rectangular (Horizontal favoring)	5.75
Rectangular (Vertical favoring)	5.59
Circular (No favoring)	4.69
Circular (Radial favoring)	5.96
Circular (Azimuthal favoring)	5.76
Adaptive (No favoring)	4.65
Adaptive (Contour orthogonal)	6.07
Adaptive (Contour parallel)	5.70

suggests that the proposed algorithm closely follows the density gradation as specified by the input function. The direction-favoring aspect of the proposed toolpath, as outlined in Section 2.5, preserves the user-specified density gradation. These toolpaths were also implemented on an FDM-based 3D printer; the results are shown in Fig. 23. In the case of directional toolpaths, within each category, the image shows a more tightly packed infill as compared to their non-direction-favoring counterparts. This observation is also reflected in the observed weights of the printed parts, as shown in Table 4.

3.3. Gradient infill through density map

An example of a gradient infill is the porous structure of a bone. The proposed toolpath, with a density map as the input function, can be used to fabricate such structures. The density map can be obtained from a CT scan or MR imaging, which serves as the governing function for achieving density gradient in the toolpath. Fig. 24 illustrates the

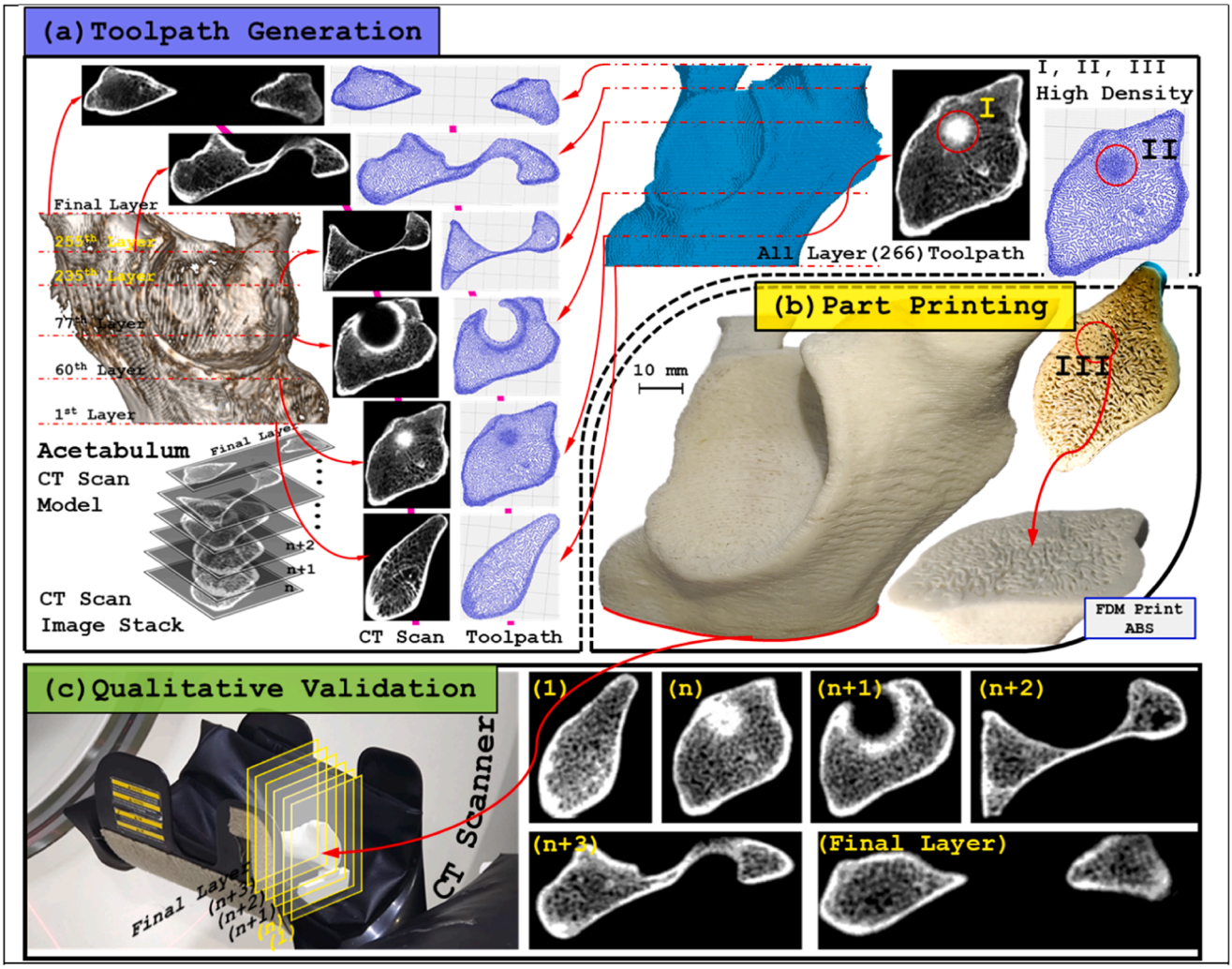


Fig. 24. Process sequence for (a) generating gradient toolpath in accordance with CT scan image stack. Regions marked 'I,' 'II,' and 'III' show an example of a 'local' high-density infill in CT Scan, generated toolpath and printed part, respectively (b) part fabricated using computed toolpath, and (c) gradient validation through CT scan of the printed part and comparison with the density map.

methodology for fabricating such structures. A high-density region around the bone's edges corresponds to the bone's cortical part, and a low-density region (inside) is termed the cancellous (spongy) part. The open-source image datasets used in this experiment were from the Laboratory of Human Anatomy and Embryology, University of Brussels (ULB), Belgium [23].

Consider the Fig. 24; the workflow is segmented into three sections. The density map is extracted from CT scan images of the bone in the first section, illustrated in Fig. 24(a). In the second section, these density maps are used to generate a gradient toolpath, illustrated in Fig. 24(b), which subsequently is used for part printing. The final section of the workflow is validation, as described in Fig. 24(c). During validation, the printed part is subject to CT Scan. This CT scan of the printed part (material is ABS in this instance) is compared with the CT scan image of the actual bone that was previously used as density maps to compute the toolpath. A qualitative comparison between the CT scan of the printed part and the actual bone establishes the conformity of the density gradient in part.

4. Conclusion

Through the utilization of a heuristic-based TSP solver, a methodology for generating gradient-density AM parts is proposed. The algorithm digitizes the domain within a contour into a set of spring-connected points, which can be positioned to obtain either a rectangular, circular, or contour adaptive grid. While the rectangular configuration of the grid points can be achieved through linear springs, additional consideration of torsional spring is required for circular and contour-adaptive digitization. Within each of these three digitization methods, through variation in stepover values along basis vector directions, the toolpath can be biased to favor one direction of travel over the other, thus reducing the number of turns.

The gradient in the toolpath can be accommodated through a user-defined density function or a density map that varies the ideal spring length (stepover) the digitized grid assumes at the equilibrium of the spring forces(or convergence). The algorithm was implemented for each of the three methods of digitization for producing gradient toolpaths with and without direction favoring. Through quantitative analysis of toolpath length and image-based analysis, the conformity of the part density to the user-input function (or density map) was established. While the toolpath length was higher for direction-favoring toolpaths as compared to unbiased ones, both, within the limits of a prescribed number of iterations of spring force calculation, followed the input density function. As a case study, using the density map from a CT scan image stack, a region of bone from the acetabulum of the pelvis was printed on an FFF-based 3D printer.

CRediT authorship contribution statement

Sadaival Singh: Data curation, Formal analysis, Investigation, Methodology, Software, Writing – original draft, Writing – review & editing, Conceptualization. **Ambrish Singh:** Data curation, Investigation, Methodology, Writing – original draft, Writing – review & editing, Formal analysis. **Sajan Kapil:** Conceptualization, Funding acquisition, Investigation, Project administration, Supervision. **Manas Das:** Project administration, Resources, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

The authors acknowledge Indo-German Science & Technology Centre (IGSTC), for their financial support for project No. IGSTC/Call 2020/MAMM-WAAM/50/2021–22/259 entitled "Multi-Axis Multi-Material Wire Arc Addtive Manufacturing (MAMM-WAAM)".

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cad.2024.103718.

References

[1] Mustafa I, Kwok TH. Interlacing Infills for Multi-Material Fused Filament Fabrication Using Layered Depth Material Images. Micromachines (Basel) 2022;13. https://doi.org/10.3390/mi13050773.
[2] Nazir A, Gokcekaya O, Md Masum Billah K, Ertugrul O, Jiang J, Sun J, et al. Multi-material additive manufacturing: a systematic review of design, properties, applications, challenges, and 3D printing of materials and cellular metamaterials. Mater Des 2023;226:111661. https://doi.org/10.1016/j.matdes.2023.111661.
[3] Ghanavati R, Naffakh-Moosavy H. Additive manufacturing of functionally graded metallic materials: a review of experimental and numerical studies. J Mater Res Technol 2021;13:1628–64. https://doi.org/10.1016/j.jmrt.2021.05.022.
[4] Prévost R, Whiting E, Lefebvre S, Sorkine-Hornung O. Make it stand: balancing shapes for 3D fabrication. ACM Trans Graph 2013;32. https://doi.org/10.1145/2461912.2461957.
[5] Telea A, Jalba A. Voxel-based assessment of printability of 3D shapes. Lect Notes Comput Sci (Including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 2011;6671:393–404. https://doi.org/10.1007/978-3-642-21569-8_34. LNCS.
[6] Stava O, Vanek J, Benes B, Carr N, Měch R. Stress relief: improving structural strength of 3D printable objects. ACM Trans Graph 2012;31. https://doi.org/10.1145/2185520.2185544.
[7] Martínez J, Dumas J, Lefebvre S. Procedural voronoi foams for additive manufacturing. ACM Trans Graph 2016;35. https://doi.org/10.1145/2897824.2925922.
[8] Bates SRG, Farrow IR, Trask RS. Compressive behaviour of 3D printed thermoplastic polyurethane honeycombs with graded densities. Mater Des 2019;162:130–42. https://doi.org/10.1016/j.matdes.2018.11.019.
[9] Choy SY, Sun CN, Leong KF, Wei J. Compressive properties of functionally graded lattice structures manufactured by selective laser melting. Mater Des 2017;131:112–20. https://doi.org/10.1016/j.matdes.2017.06.006.
[10] Wu J, Aage N, Westermann R, Sigmund O. Infill Optimization for Additive Manufacturing-Approaching Bone-Like Porous Structures. IEEE Trans Vis Comput Graph 2018;24:1127–40. https://doi.org/10.1109/TVCG.2017.2655523.
[11] Wu J. Continuous optimization of adaptive quadtree structures. CAD Comput Aided Des 2018;102:72–82. https://doi.org/10.1016/j.cad.2018.04.008.
[12] Liu C, Du Z, Zhu Y, Zhang W, Zhang X, Guo X. Optimal design of shell-graded-infill structures by a hybrid MMC-MMV approach. Comput Methods Appl Mech Eng 2020;369:113187. https://doi.org/10.1016/j.cma.2020.113187.
[13] Kapil S, Joshi P, Yagani HV, Rana D, Kulkarni PM, Kumar R, et al. Optimal space filling for additive manufacturing. Rapid Prototyp J 2016;22:660–75. https://doi.org/10.1108/RPJ-03-2015-0034.
[14] Mohan SR, Khaderi SN, Simhambhatla S. 3D printing of components with tailored properties through Hilbert Curve Filling of a discretized domain. 3D Print Addit Manuf 2020;7:288–99. https://doi.org/10.1089/3dp.2020.0048.
[15] Soo SC, Yu KM. Tool-path generation for fractal curve making. Int J Adv Manuf Technol 2002;19:32–48. https://doi.org/10.1007/PL00003966.
[16] Chiu WK, Yeung YC, Yu KM. Toolpath generation for layer manufacturing of fractal objects. Rapid Prototyp J 2006;12:214–21. https://doi.org/10.1108/13552540610682723.
[17] Kuipers T, Wu J, Wang CCL. CrossFill: foam structures with graded density for continuous material extrusion. CAD Comput Aided Des 2019;114:37–50. https://doi.org/10.1016/j.cad.2019.05.003.
[18] Singh S, Singh A, Kapil S, Das M. Utilization of a TSP solver for generating non-retractable, direction favouring toolpath for additive manufacturing. Addit Manuf 2022;59:103126. https://doi.org/10.1016/j.addma.2022.103126.
[19] Bedel A, Coudert-Osmont Y, Martínez J, Nishat RI, Whitesides S, Lefebvre S. Closed space-filling curves with controlled orientation for 3D printing. Comput Graph Forum 2022;41:473–92. https://doi.org/10.1111/cgf.14488.
[20] Cheng P, Liu WK, Ehmann K, Cao J. Enumeration of additive manufacturing toolpaths using Hamiltonian paths. Manuf Lett 2020;26:29–32. https://doi.org/10.1016/j.mfglet.2020.09.008.
[21] Traveling Sal sman Problem n.d. http://www.math.uwaterloo.ca/tsp/index.html (accessed April 3, 2022).
[22] Persson PO, Strang G. A simple mesh generator in MATLAB. SIAM Rev 2004;46:329–45. https://doi.org/10.1137/S0036144503429121.
[23] Laboratory of human anatomy and embryology, University of Brussels (ULB), Belgium n.d. https://isbweb.org/data/vsj/iliac/.