A brick in the wall: Staggered orientable infills for additive manufacturing

Thibault Tricard, Jimmy Etienne, Cédric Zanni, Sylvain Lefebvre

A brick in the wall: Staggered orientable infills for additive manufacturing

THIBAULT TRICARD, Université de Lorraine, CNRS, Inria, LORIA, France
JIMMY ETIENNE, Université de Lorraine, CNRS, Inria, LORIA, France
CÉDRIC ZANNI, Université de Lorraine, CNRS, Inria, LORIA, France
SYLVAIN LEFEBVRE, Université de Lorraine, CNRS, Inria, LORIA, France

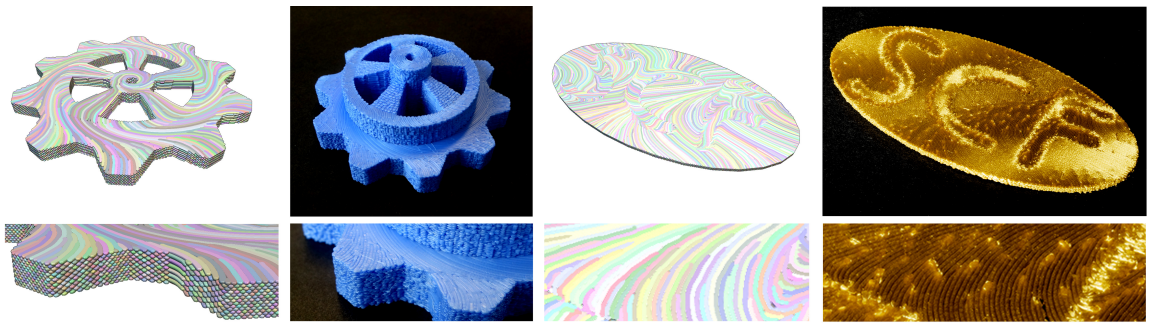

Fig. 1. Our technique generates dense planar infill trajectories, precisely following an input direction field. Through the anisotropy of the deposition process, the direction field controls the appearance and the physical properties of the 3D printed object. The trajectories are arranged in a staggered layout across layers. **Left:** A 3D printed gear where the direction of the trajectories is parameterized to adapt to the functionality of the part. Here, trajectories are mostly circular in the rim and hub while being aligned with the spokes. Note the staggered layout in the side views. **Right:** The anisotropy of the deposition results in anisotropy in the specular reflectance. Here, this is controlled to pattern the appearance of an otherwise flat part, resulting in a brushed metal effect.

Additive manufacturing is typically conducted in a layer-by-layer fashion. A key step of the process is to define, within each planar layer, the trajectories along which material is deposited to form the final shape. The direction of these trajectories triggers an anisotropy in the fabricated parts, which directly affects their properties, from their mechanical behavior to their appearance. Controlling this anisotropy paves the way to novel applications, from stronger parts to controlled deformations and surface patterning.

This work introduces a method to generate trajectories that precisely follow an input direction field while simultaneously avoiding intra- and inter-layer defects. Our method results in spatially coherent trajectories - all follow the specified direction field throughout the layers - while providing precise control over their inter-layer arrangement. This allows us to generate a staggered layout of trajectories across layers, preventing unavoidable tiny gaps from forming tunnel-shaped voids throughout a part volume.

Our approach is simple, robust, easy to implement, and scales linearly with the input volume. It builds upon recent results in procedural generation of oscillating patterns, generating a signal in the 3D domain that oscillates with a frequency matching the deposition beads width while following the input direction field. Trajectories are extracted with a process akin to a marching square.

CCS Concepts: • **Computing methodologies** → *Mesh models*; Texturing.

Additional Key Words and Phrases: additive manufacturing, FDM, infill, procedural generation

ACM Reference Format:
Thibault Tricard, Jimmy Etienne, Cédric Zanni, and Sylvain Lefebvre. 2021. A brick in the wall: Staggered orientable infills for additive manufacturing. In *Symposium on Computational Fabrication (SCF '21), October 28–29, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3485114.3485117

Manuscript submitted to ACM

1 INTRODUCTION

Additive manufacturing produces a part layer by layer, accumulating material progressively. As a consequence, the trajectories along which material is deposited or solidified have a direct influence on the final part properties. On the one hand, this provides unprecedented control over the way the volume is formed, enabling novel and unique possibilities to perfectly adjust a part to its final functionality. On the other hand, the trajectories play a crucial role in the part structural soundness, and care must be taken to minimize potential defects.

Often, the deposition strategies have to enforce competing objectives. For instance, the process may be expected to produce dense parts with minimal porosity. However, it is also desirable for the trajectories to follow specific directions within layers, either due to manufacturing constraints or to maximize part functionality, e.g., stiffness under some external loads. In particular, additively manufactured parts tend to be stronger in the direction of deposition, while in the orthogonal direction, the bond between neighboring beads is less strong [Rybachuk et al. 2017]. In addition, some processes are capable of depositing fibers alongside the trajectories, resulting in even higher ratios of anisotropic mechanical response. Being able to conform the deposition trajectories with specific directions, therefore, paves the way to several applications: stronger parts (e.g. [Steuben et al. 2016]), controlled deformations (e.g. [An et al. 2018]), as well as the ability to pattern the surface roughness (light reflection, grip).

In this work, we propose a novel approach to generate trajectories following an input direction field. The field may be user–created, or may be the result of an optimization process (e.g., topology optimization [Pantz and Trabelsi 2008]). Our objective is to produce dense parts, conciliating the contradictory objectives of fully covering an area with evenly spaced material beads while having them follow an arbitrary direction. We seek a result where defects such as gaps have a minimal impact within a layer and across layers.

Even in the absence of a direction field to follow, avoiding defects in infills is a problem that remains a focus of research (we provide details in Section 2). Adding the direction objective makes this problem significantly harder, as following the direction requires paths to move away from one another or to regroup in specific locations, creating branching patterns where tiny gaps are difficult to avoid. Due to spatial coherence of trajectories geometry across layers, such small infill defects within a layer tend to accumulate *across layers* and degenerate into critical defects: internal tunnel-shaped voids that endanger the structural integrity of the part. This problem is worsened when following a given direction field that is similar from one layer to another.

Our technique generates trajectories for planar layers. However, to address the aforementioned challenges, we generate infill trajectories that not only follow the given directions but also form a coherent structure *across* layers. In particular, the deposited beads form a staggered structure akin to the layout of a brick wall. As with bricks and mortar [Nicholson 1823], the intent is to prevent the weaker lateral bond between beads from forming vertical sheets prone to failure. This also prevents tiny gaps in split/merge locations from forming coherent structures across layers within the volume. Please note however that in this paper we focus solely on the modeling aspects of a staggered infill. Measuring to what extent this truly benefits mechanical properties will require an in-depth mechanical study left as future work.

We achieve a staggered layout by building upon the *regularized phasor noise* technique introduced by Tricard et al. for microstructure design [Tricard et al. 2020]. This efficient, scalable approach generates oscillating fields within volumes, whose direction can be controlled. We define and extract our trajectories from crossed phasor fields (Section 4.1, 4.2 and 4.3), and exploit the exposed argument of the oscillations to obtain a staggered layout (Section 4.5). By controlling the oscillations frequency, we can further adjust the deposition width (Section 4.2).

This results in a simple, robust, and efficient method for generating dense, oriented trajectories. Thanks to the stochastic nature of the underlying synthesis process, the unavoidable tiny gaps in branching locations are evenly distributed within the volume and are further prevented to align and form long tunnel-shaped void by the staggered layout of the beads. We analyze results and discuss applications in Section 5.

2 PREVIOUS WORK

The processing pipeline for additive manufacturing is responsible for generating the instructions used by the machine to fabricate the final part, layer by layer [Livesu et al. 2017]. Our work focuses on the later steps of the pipeline, where the geometry of layers has already been determined, but surface-covering trajectories remain to be computed. In particular, our technique is orthogonal to and compatible with slicing approaches such as adaptive slicing [Alexa et al. 2017; Kulkarni and Dutta 1996; Pandey et al. 2003; Sabourin et al. 1996].

Techniques computing trajectories for inner parts of a layer are often referred to as *infill* strategies. For *dense* infills, the two most common techniques are contour-parallel infills [Held et al. 1994; Yang et al. 2002] and zigzag (raster) infills [McMains et al. 2000], and their typical combination. Note that we do not review here techniques for *sparse* infill unless technically related, as our focus is to produce fully filled parts.

The contour parallel and zigzag infills leave gaps within a layer, as it is generally not possible to fully and exactly cover an arbitrarily shaped polygon with fixed-width strokes. These defects are understood as *underfill* (gaps) and *overfill* (too much material deposited in a same location). Therefore, several methods proposed to reduce these artifacts, in particular by locally varying the deposition radius [Ding et al. 2016; Hornus et al. 2020; Jin et al. 2017; Kao and Prinz 1998; Kuipers et al. 2020; Xiong et al. 2019]. However, in all these approaches, the direction of the infill patterns is driven by the contour of the shape (contour parallel) or is fixed in space (zigzag), possibly chosen differently in different zones [Ding et al. 2014].

A recent trend is, however, to consider the direction of the trajectories with respect to the desired functionality – most often the part structural rigidity, but also to, e.g., trigger controlled deformations under heat [An et al. 2018]. This exploits the natural anisotropy of the mechanical resistance of 3D printed parts [Fang et al. 2020; Rybachuk et al. 2017] which can be further reinforced by the inclusion of fibers. In this work, we do not presume a specific application and assume the input direction field given.

In order to obtain oriented infills, Steuben et al. [2016] optimize a continuous scalar field in which gradient aligns with a principal stress direction. Trajectories are extracted as iso-values in this field. However, there is no specific treatment for maintaining a constant spacing between trajectories. Ezair et al. [2018] propose a curved slicing approach that has the potential to address this issue, as the trajectories are extracted from a given tri-variate field while maintaining their spacing by hierarchical subdivision [Elber and Cohen 1996; Etienne and Lefebvre 2020]. Fang et al. [2020] extract curved trajectories as iso-values of a field that is specially optimized to enforce spacing and direction constraints. The main drawback of these approaches is that computing a tri-variate field under strict direction constraints is a computationally and numerically challenging task. In addition, as we will later show, adjusting spacing through iso-value extraction leads to gaps in coverage and to an alignment of the split/merge locations detrimental to the resulting quality. Boddeti et al. [2020] rely on the optimization of 2D stripe patterns in individual layers to produce trajectories. This is based on the global optimization of a periodic parameterization [Knöppel et al. 2015]. This allows a more natural adaptation to changes in direction. Similar methodologies have been applied to generate conforming lattices [Wu et al. 2021] and flexible microstructures [Tricard et al. 2020], using techniques respectively from hex-meshing [Gao et al. 2017] and

procedural texturing [Tricard et al. 2019]. Our technique is rooted in similar methodologies: we generate periodic 3D signals to extract oriented trajectories across all 2D layers, arranged in a globally coherent staggered layout.

3 BACKGROUND ON PHASOR NOISE

Our technique relies on the phasor noise introduced by Tricard et al. [Tricard et al. 2019]. A phasor noise ϕ is a procedural noise allowing the definition of fiber-like patterns. It is defined as the argument of a complex Gabor noise G [Lagae et al. 2009]:

$$\phi(x) = \arg\left(G(x)\right) \tag{1}$$

with :

$$G(x) = \sum_j e^{-b\|x-x_j\|^2} e^{2if_j d_j.(x-x_j)+i\varphi_j} \tag{2}$$

It is an unstructured representation, defined from the convolution of complex Gabor kernels with random seed positions x_j. The kernel properties locally define the properties of the noise: its bandwidth b, frequency f_j, the direction of anisotropy d_j, and phase shift φ_j. This generates a scalar field in 3D space: an oscillating signal of precisely controlled direction and scale. In our work, we use the oscillations to define trajectories for material deposition.

Without special treatment, a phasor noise exhibits local defects located around singularities of the phase field. These result in a less coherent pattern, and in our case, would result in less continuous deposition trajectories. In [Tricard et al. 2020] two methods are combined to significantly decrease the number of defects in the generated patterns. First, the phase φ_j of neighboring kernels is iteratively adjusted in a simple and efficient optimization process. Second, a closed-form regularization filter is applied when sampling values from the phasor noise. Through these combined methods, highly regular anisotropic patterns are obtained. The high quality and regularity of the fields enable the definition of local parametric patches for microstructure design; for more details, please refer to [Tricard et al. 2020]. In our work, we extend this methodology to define deposition trajectories for dense infill patterns.

The phasor noise presents several advantages for our application. First, it is a meshless process. In other words, it means that it can be computed in any Euclidean space while following arbitrary input fields. Secondly, it scales to large volumes as to the complexity of the iterative phase alignment, in both space and time, is only proportional to the number of kernels – not the resolution of the generated patterns. Finally, the entire framework maps well onto GPUs, enabling fast, parallel optimization and evaluation of regularized phasor fields. For the remainder of the paper, it is important to remember that the phasor noise is defined in 3D.

4 OUR APPROACH

Our technique takes as an input a 3D model for which to generate 3D trajectories as well as a direction field d^U that is everywhere orthogonal to z (the manufacturing direction). The algorithm starts by defining two crossed phasor fields used to generate oscillating patterns everywhere throughout the object volume. The oscillation frequency f is chosen to match the target trajectory spacing – typically the nozzle width w (with $f = 1/w$). However, by varying the frequency, we naturally support varying deposition widths, see Section 4.2. The direction of the phasor fields closely follows d^U.

Once the fields are optimized, we proceed layer by layer to efficiently generate actual trajectories. This step extracts a set of discrete curved trajectories from the continuous definition of two phasor fields. We describe it in Section 4.1. Note that this is performed in a streaming fashion (we only process a fixed number of slices at a time) and that slices anywhere in space may be extracted in parallel.

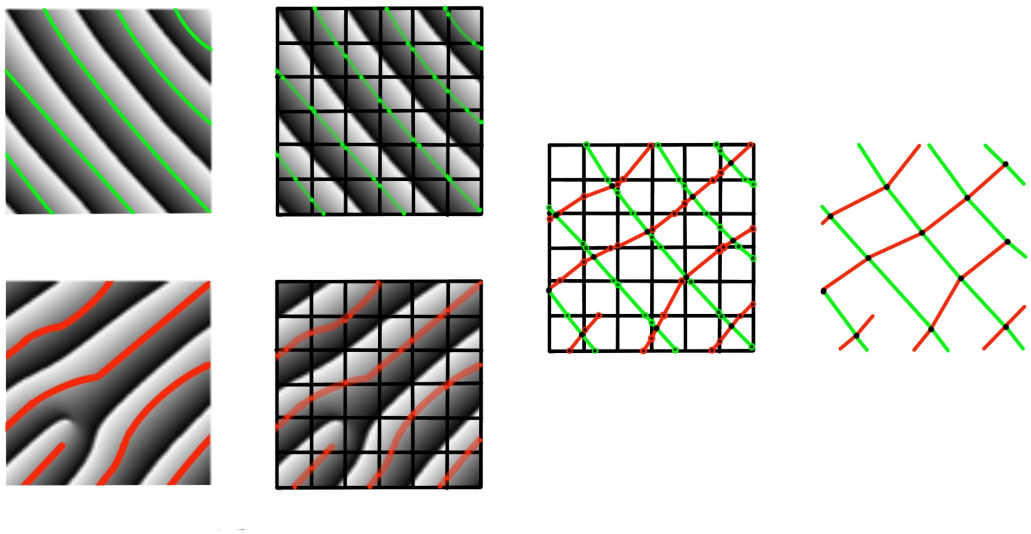

Fig. 2. Representation of the graph extraction process. First: both phasor fields are sampled; here, their arguments are remapped in [0,1] for visualization. The set of curves C_π^U is highlighted in green while C_π^V is in red. Then intersections of C_π^U and C_π^V with edges of the grid are computed separately, creating multiple polylines. The intersection between polylines of both sets is computed to define the vertices of the graph (in black here). Finally, we collapse edges until all vertices either belong to the two edge sets (valence equal four) or are curve end-points (valence equal one).

To prevent trajectories from creating inter-layer artifacts by always being exactly aligned, we introduce a *phase shift* every other layer. This produces staggered trajectories across layers while still closely following the direction field.

4.1 Defining trajectories

In this section, we describe our method to define dense infills - and extract the associated trajectories - following the arbitrary direction field d^U given by the user, whose directions must be constrained to the slicing plane. Let us define d^V a second field orthogonal to d^U, also constrained to the slicing plane (i.e. $d^V = d^U \times z$). Note that d^U and d^V are defined as 3D fields, and their values can evolve along the Z-axis. As a result, the trajectories' direction can evolve across layers. The trajectory extraction is based on two orthogonal periodic variable fields, U and V, obtained from two phasor fields. To any given point x in space we associate a value pair $UV(x) \in [0, 2\pi]^2$. We note $U(x)$ and $V(x)$ the two components of $UV(x)$ (see Figure 2, left). In practice, $U(x)$ (respectively $V(x)$) defines a locally oscillating signal taking the shape of a sawtooth. This defines a periodic variable field following the direction d^U (respectively d^V). From this periodic variable field we can construct a set of curves defined by $U(x) = \pi$ (respectively $V(x) = \pi$). The generated curves C_π^U (respectively C_π^V) are orthogonal to d^U (respectively d^V). Therefore, we define the trajectories as the set of curves C_π^V (see Figure 2, left). The orthogonal set of curves C_π^U is used during the computation of inter-trajectories distances and defines the sampling rate of the trajectories (see Section 4.3).

In practice, the extraction of the trajectories is performed in three steps. First, we define and optimize our periodic variable fields (see Section 4.2). Then, for a given printing slice, we extract a planar graph in whose vertices are defined as the intersection between C_π^V and C_π^U (see Figure 2, right). Edges of the graph are subdivided into two sets, E^U and E^V.

The edges of E^U (resp. E^V) form a tessellation of the C_π^V curves (respectively C_π^U). The graph extraction is described in Section 4.3. Finally, we use E^U to define the trajectories and use E^V to measure the local spacing between them. This information is later used to adjust the deposition flow (see Section 4.4).

4.2 Periodic variable fields

We define the two local periodic variable fields U and V by relying on two orthogonal 3D phasor noises (see Equation 1). The kernel directions d_i are computed by sampling the input direction field d^U (resp. d^V) at the kernel position x_i. The frequencies f_i are chosen as $1/w$, with w the target deposition bead width. Note that w can be made to spatially vary by sampling an additional user input field. Finally, parameter φ_i is computed by iterative phase alignment [Tricard et al. 2020].

4.3 Graph extraction

As previously stated, we seek to extract two sets of edges E^U and E^V from the periodic variable field UV. Recall that while the fields are defined in 3D, the trajectory extraction is performed in 2D, layer by layer (possibly in parallel). We base our approach on a modified version of the marching square algorithm [An et al. 2018; Wyvill et al. 1986].

First, we sample our periodic variable field on a regular 2D grid which resolution is chosen to sample at four times the maximum field frequency. This sampling frequency is chosen to help to filter the 2π jump in the periodic variable fields. Then, for each square cell of the grid, we extract C_π^U and C_π^V separately. We extract vertices for the intersection between C_π^U (respectively C_π^V) along each edge of the cell. More precisely, given i and j the grid points defining a processed edge, we extract a vertex if $U(i) - \pi$ and $U(j) - \pi$ have opposite signs. We filter out vertices that do not verify $|U(i) - U(j)| \leq \pi$ as they are generated by the 2π jumps in the field. This condition ensures robustness with respect to the periodicity of the field. The positions of the vertices are chosen based on the linear interpolation of the U (resp. V) parameter along the edges.

For a given field, we only extract an edge in a given cell if there are exactly two vertices created. Edges extracted from fields U (resp. V) are registered in the set E^V (resp. E^U). As this is done for both U and V, a cell contains from 0 to 2 edges. If the cell contains two intersecting edges (defined from two distinct fields), the latter are subdivided, and a new vertex is created at the intersection point, resulting in four new connected edges.

When each square cell of the grid is processed, we stitch all the partial results together and collapse edges until all vertices either belongs to the two edge sets (vertices of valence 4 verifying $UV \approx \{\pi, \pi\}$) or are curve end-points (vertices of valence one close to singularity or on the border of the domain). This reduces the sampling frequency of C_π^U (respectively C_π^U) to the frequency of the V (respectively U). Thus the frequency of U controls the polygonization error in curved trajectories. In practice, we use the same frequency for both fields as they generate a sampling equal to or close to the nozzle diameter, which is enough for our application. Note that we keep track of the edge origin during the decimation, thus updating the edge sets E^U (edge marked as V) and E^V (for edge marked as U).

4.4 Clipping and ordering trajectories

To extract the final trajectories from the graph, we clip the set of edge E^U by the contour of the slice; then, we use a greedy approach to extract the longest uninterrupted paths. For each point, we compute its average distance to all its direct neighbors on the E^V set. This average distance – which represents the local bead width – is then clamped in order to remain between half and twice the nozzle size. It is then used to proportionally adjust the deposition volume. Finally,

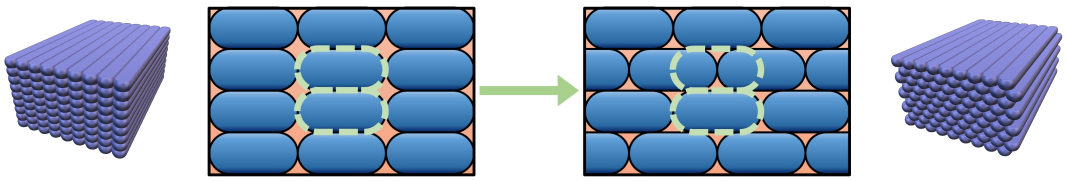

Fig. 3. Trajectories can be offset every other layers by shifting the phases of the kernels. This allows to create staggered beam layouts. Handmade 3D example and sectional view of deposition paths. **Left:** aligned. **Right:** staggered.

we transform the trajectories into GCode and adjust the material deposition at each point by taking into account the inter-spacing previously computed.

We order the trajectories following a simple nearest-first heuristic (the first trajectory is picked at random, then the next trajectory is the one with closest end-point). This provides reasonably good results at a low cost. An ordering can be visualized in Figure 5, right.

4.5 Staggered bead layout

To obtain a staggered layout, we offset the periodic variable fields for every odd layer. This is easily achieved by shifting the phases of the kernels φ_i when sampling odd layers, adding an offset $\delta = \pi$. Doing so produces a pattern where the beads of one layer are shifted by half the width of the beads from the layer below, forming a coherent staggered layout. Note that this works naturally across directions and frequency (width) controls, as we directly manipulate the phase of the coherent 3D oscillations. This simple approach produces remarkably regular and aligned patterns, as can be seen in Figure 3.

5 RESULTS

In this section, we discuss the specific properties of our method and demonstrate the range of possibilities it offers.

All our tests are done with fused-filament 3D printing, the most widespread additive manufacturing technique. Parts were printed with a variety of printers. Blue CuteOcto was printed with a Prusa MK2S, parts in grey with a micro delta, parts in multicolor with a modified Prusa printer equipped with a diamond mixing extruder, other parts were printed using a Creality CR10S pro. Note that all parts shown here are solely composed of infill to illustrate the method. In practice our infills would likely be combined with a contouring technique for the outer object shell, replacing the internal typical zigzag solid fill. The list of all model sources is provided in Table 1. Our method was implemented in C++ using the Vulkan API, and shaders were programmed with glsl. Trajectories are always extracted on the full bounding box of the object before being clipped to the surface. Therefore we measure runtimes for cubes of increasing volumes and only measure timings related to our method (e.g., alignment of kernel phases and graph extractions, leaving out the actual clipping of trajectories, scheduling as well as GCode generation). Runtimes were measured on an i7-6700HQ CPU (@2.6GHz only one thread used) and GTX1070 graphic card and are given in Table 2. The first thing to notice is that phase alignment runtime is bounded to 28s. This is due to our strategy to compute kernel density and bandwidth such that the phase alignment of kernels can be done in a single pass on GPU while avoiding memory overflow (the maximal number of kernels used is approximately eight millions). If needed, a streaming strategy could be adopted to increase the number of degrees of freedom in larger parts. As expected, the graph extraction (including the sampling of the phasor noises) is linear in terms of processed volume. The grid size for the marching square is based on the target

Table 1. Table of all models used in the article, Figures they can be seen in, and their sources

Model	Figure	Sources
Gear	1	custom made
SCF	1	custom made
Badge	10	thingiverse.com/thing:1670621
Cute octo says hello	5, 8, 7	thingiverse.com/thing:27053
Screwdriver handle	9	thingiverse.com/thing:34852
Gecko	11	thingiverse.com/thing:1363148
Fandisc	12	github.com/dengwirda/jigsaw-models

Table 2. Computation time in function of the volume for a 0.4mm nozzle and a layer height of 0.2mm.

Volume (cm^3)	Total (s)	Phase optimisation (s)	Graph extraction (s)	Time over volume (s/cm^3)
1	2.81	1.64	0.79	2.81
8	17.11	9.27	6.54	2.14
64	82.05	28.14	48.97	1.28
512	355.32	28.20	323.12	0.5

bead width (the length of an edge is one-quarter of the minimal target bead width), and 64 slices are computed at once. For all tested models, the GPU memory used remains bounded to $1.7GB$. Should the memory consumption become problematic, a smaller set of slices and XY tilling could be used.

5.1 Comparison to curve hierarchies

When a bi-variate parametric field is available, curve hierarchies may be used to produce trajectories that follow the field while being roughly equally spaced [Elber and Cohen 1996; Ezair et al. 2018]. We compare our approach to a curve hierarchy in Figure 4. As can be seen, the curve extraction leaves many gaps. This typically happens where the field exhibits a high divergence, as is the case with the radial field shown in Figure 4 (left). There, steep transitions of local density from 1 to 0.5 occur: the change of hierarchy level introduces a sudden jump. In addition, the gaps occur along specific transition curves of the field, which in turn leads to coherent defects within and across layers, weakening the overall part structure. Note that other curves extraction techniques could alleviate this problem [Etienne and Lefebvre 2020], but this has not been demonstrated for dense infills.

In comparison, our method exhibits fewer transitions with tiny gaps well distributed throughout a slice (Figure 4 (right)) – keeping in mind that the paths are additionally laid out staggered from one layer to the next. Finally, recall that this is achieved directly from the direction field *without* the need to optimize for a bi-variate parametric field in the first place.

5.2 Singularities, staggered layout

Figure 5 shows the CuteOcto model sliced with a radial direction field and constant bead width (nozzle diameter). As can be seen, the branching pattern and split/merge locations are evenly distributed within each slice.

Figure 7 reveals the effect of the staggered layout across layers. It is the same model as Figure 5 with a sectional view of the beads. On the left disabling the phase shifts, on the right our result with the staggered layout. This cut reveals that singularities have a limited extent in the Z direction. It also shows how our staggered layout (right) reduces the influence of singularities by breaking the gap coherence, increasing the bond between beads in these areas. It is

important to notice that the staggered layout visibility is reduced on printed parts (see Figure 1(left), 8) due to travel artifacts between trajectories at the boundary of the object (visually similar to larger beads).

Another slicing of CuteOcto presents the combination of direction field and deposition bead width field (see Figure 8 6). Direction field is defined as the gradient of the signed distance field to the surface, while bead width field is radially defined.

Control over the direction field can help for a wide range of applications, from controlling the appearance to the physical properties of the printed object. In the screwdriver handle (see Figure 9), the direction field is used to control the outer texture of the object. A grip-like pattern is created in the locally convex area (by using trajectories locally orthogonal to the surface), while in the concave area, smooth surfaces are used (trajectories parallel to the surface). The direction field can also be used to control the appearance of horizontal parts of an object where filament directions are clearly visible (see Figure 1(right),10). The direction field can also be used to align paths along branches in branch-like structures (see Figure 11). In this case the direction field was created to favor long trajectories across branching regions (see junctions between legs and torso). Finally, the direction field can also be used to adapt fiber direction in mechanical parts (see Figure 1(left),12).

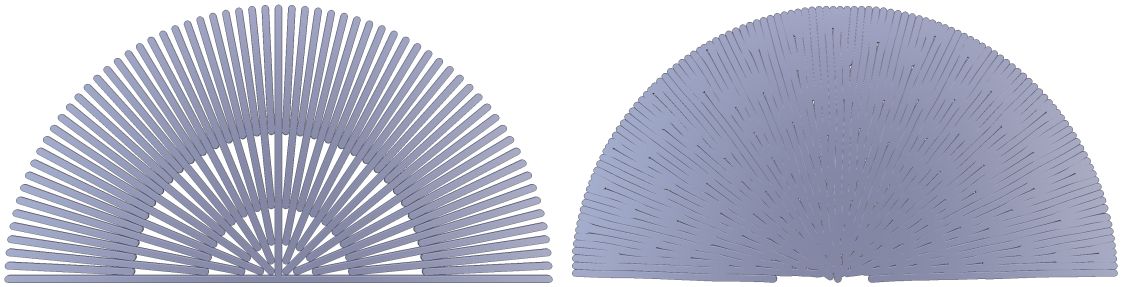

Fig. 4. Comparison to [Ezair et al. 2018] on a radial direction field. **Left :** method of [Ezair et al. 2018], steep transition of local density can be observed. **Right:** our methods, note that unavoidable tiny gaps are well distributed throughout the slice.

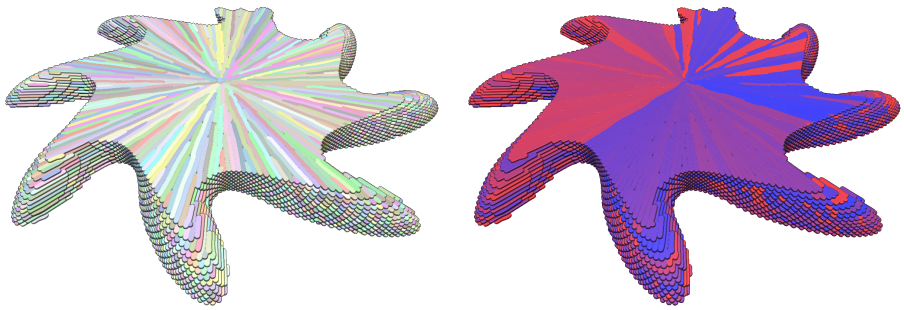

Fig. 5. CuteOcto model sliced with a radial direction field, constant target beam width, and staggered beam layout. **Left:** one layer with colored beads. **Right:** beads colored by order of printing.

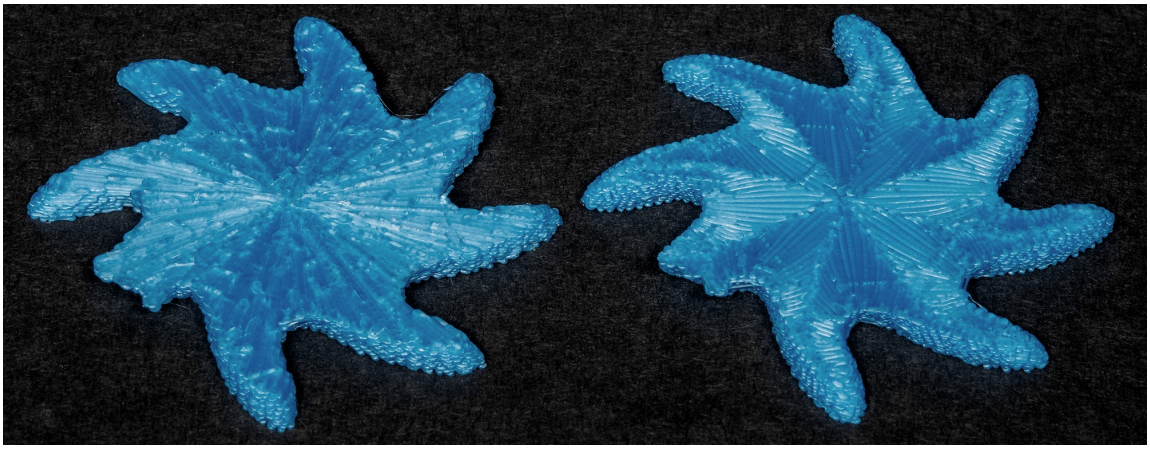

Fig. 6. First slices of the CuteOcto model with constant (left) and varying (right) target bead width.

6 DISCUSSION

We presented a method to stagger the beads forming a slice while orienting them. The staggered layout is achieved by adding a phase shift of π on the result of the phasor noise every odd slices. We chose this specific phase shift to offset the beads by half their size at each layer. In practice it is possible to achieve different staggered layouts by varying the phase shift and the sets of layers onto which it is applied.

In our implementation we input a 3D voxel field to define the orientation of the beads. This field is hand painted by the user in a dedicated tool, and then sampled to orient each phasor kernel. In practice a field of $64 \times 64 \times 64$ voxels was used in all our examples. Other definitions of orientation fields could be used — e.g. procedural — provided they can be sampled at the center of each phasor kernel.

When printing with FDM, using dense and strongly oriented patterns can create deformations (shrinkage / warping) while printing. This effect stems from the tendency of the polymer to retract when cooling [Yaman 2018]. This did not create specific difficulties when printing our example parts with PLA material.

Before being printed the paths are ordered using a greedy approach, where the next bead to be printed is the one with an extremity closest to the last printed position. This heuristic encourages nearby paths to be printed successively while being fast to compute. Naturally, other ordering methods could be devised depending on the material and process requirements.

7 CONCLUSION

We introduced a new approach to extract dense planar trajectories following a direction field. The trajectories are obtained from two periodic variable fields defined in 3D from phasor noises. Interestingly, phasor fields have links with global parametrization techniques for quad- and hex-meshing, as discussed in [Tricard et al. 2020]. Therefore, such approaches could likely be modified for a purpose similar to ours. Our approach however inherits the low setup cost, scalability and simplicity of phasor noises, enabling the efficient extraction of trajectories.

The 3D nature of the field allows to distribute imperfections evenly throughout the volume. It also introduces a way to control spatial coherency across layers, leading to the staggered bead layout. While in-depth mechanical studies are

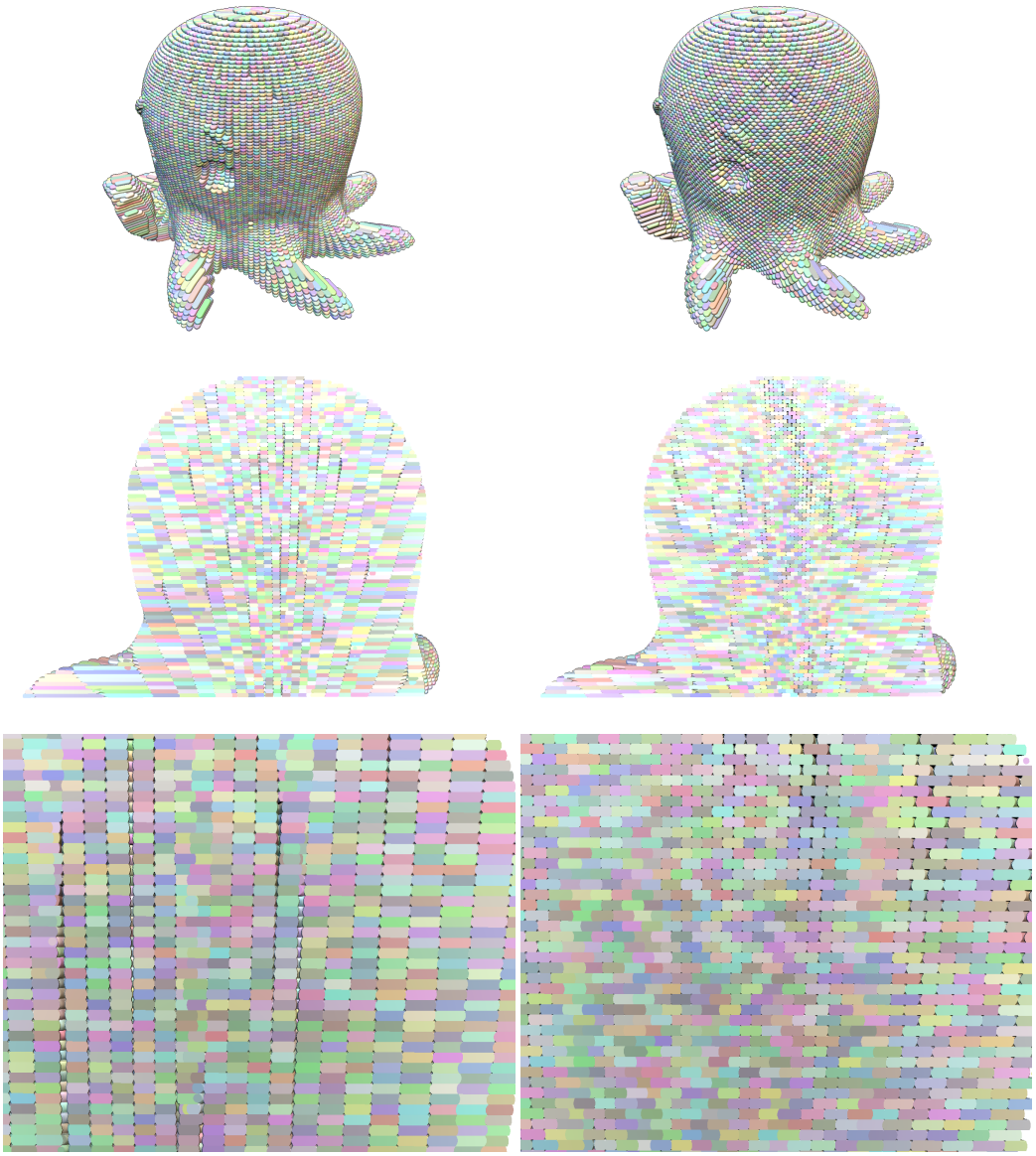

Fig. 7. A sectional view of the beads within the CuteOcto model volume, under a radial direction field. **Left:** phase shifts disabled, note the limited vertical gaps near singularities. **Right:** Our staggered layout breaks the gap coherence, increasing the bond between beads in these areas and throughout the volume. Note that singularities are placed differently as the optimization of phasor kernels is non-deterministic.

required, we believe this inter-layer layout control to be a promising direction of research, as this degree of freedom has not been available before.

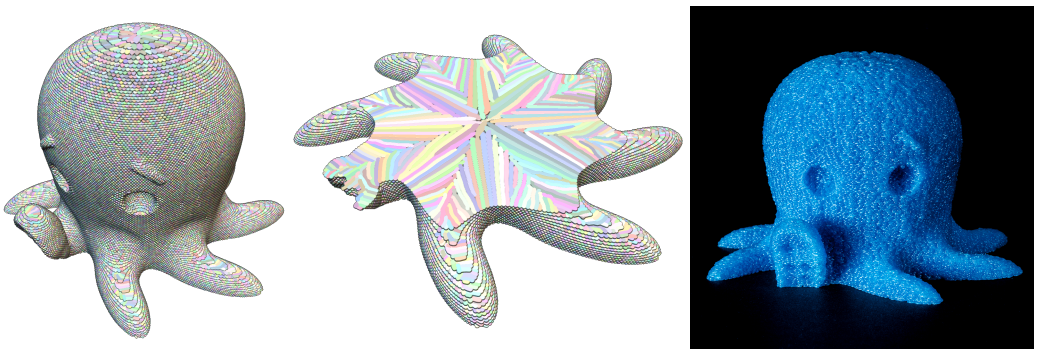

Fig. 8. Bead width can be controlled with an additional user input field. CuteOcto model sliced with a radially defined beam width. Direction field is defined as the gradient of the signed distance field to the surface.

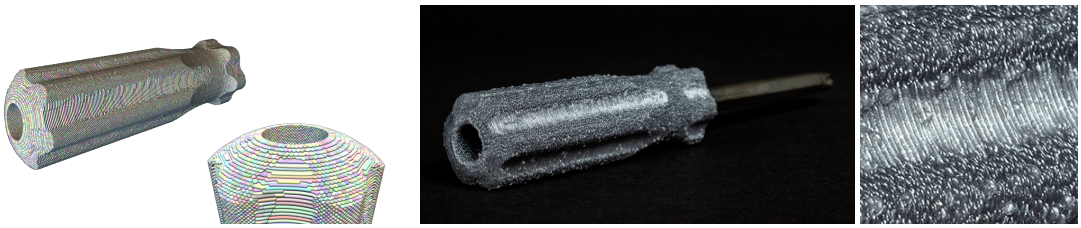

Fig. 9. Direction field near the surface of an object can be used to control surface roughness. Here a screwdriver handle have a grip-like surface in convex area (direction orthogonal to the surface) and a smooth surface in concave area (direction in surface tangent plane).

Our method presents a few limitations. In comparison to traditional infill patterns, it produces heavier GCode due to the complex curved trajectories. This could be addressed by relying on circular arc GCode capabilities. When generating paths that follow the local curvature of the object, our approach does not guarantee a perfect alignment between the object boundary and the deposition paths. This results in some trajectories being cut when encountering the object boundary. As future work, this could be approached by controlling the phase of the Gabor kernels where a precise alignment is required.

Finally, a promising research direction is to generalize our technique to curved layer printing. The key difficulty remains the collision-free guarantees required, both for the direction field definition and the ordering of paths in 3D.

ACKNOWLEDGMENTS

This work was supported partly by the french PIA project « Lorraine Université d'Excellence » (ANR-15-IDEX-04-LUE), and the project IMPRIMA(ANR-18-CE46- 0004). We thank Pierre-Alexandre Hugron for his help in designing and printing the examples shown in this work .

REFERENCES

Marc Alexa, Kristian Hildebrand, and Sylvain Lefebvre. 2017. Optimal discrete slicing. *ACM Trans. Graph.* 36, 1 (2017), 1 – 16.

Byoungkwon An, Ye Tao, Jianzhe Gu, Tingyu Cheng, Xiang 'Anthony' Chen, Xiaoxiao Zhang, Wei Zhao, Youngwook Do, Shigeo Takahashi, Hsiang-Yun Wu, Teng Zhang, and Lining Yao. 2018. Thermorph: Democratizing 4D Printing of Self-Folding Materials and Interfaces. (2018), 1–12. https://doi.org/10.1145/3173574.3173834

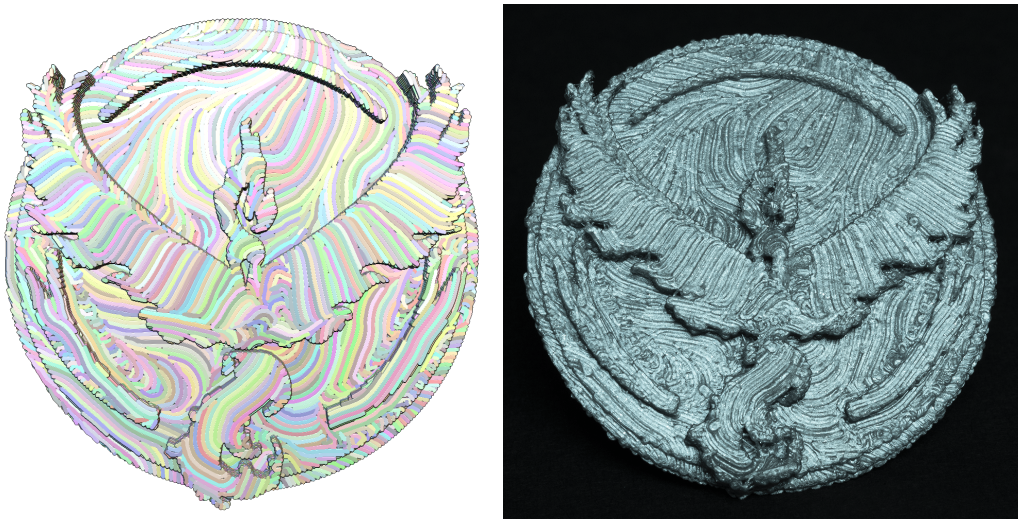

Fig. 10. Direction field can be used to texture horizontal area by using the fact that filament deposition direction is clearly visible on 3D printed models.

Narasimha Boddeti, Yunlong Tang, Kurt Maute, David W. Rosen, and Martin L. Dunn. 2020. Optimal design and manufacture of variable stiffness laminated continuous fiber reinforced composites. *Scientific Reports* 10, 1 (05 Oct 2020), 16507.

Donghong Ding, Zengxi Pan, Dominic Cuiuri, Huijun Li, and Nathan Larkin. 2016. Adaptive path planning for wire-feed additive manufacturing using medial axis transformation. *Journal of Cleaner Production* 133 (2016), 942–952. https://doi.org/10.1016/j.jclepro.2016.06.036

Donghong Ding, Zengxi Stephen Pan, Dominic Cuiuri, and Huijun Li. 2014. A tool-path generation strategy for wire and arc additive manufacturing. *Int. J. Adv. Manuf. Tech.* 73, 1-4 (2014), 173–183.

Gershon Elber and Elaine Cohen. 1996. Adaptive Isocurve-based Rendering for Freeform Surfaces. *ACM Transactions on Graphics* 15, 3 (1996), 249–263.

Jimmy Etienne and Sylvain Lefebvre. 2020. Procedural band patterns. In *Symposium on Interactive 3D Graphics and Games (Symposium on Interactive 3D Graphics and Games)*. Association for Computing Machinery, San Francisco, United States, 1 – 7. https://doi.org/10.1145/3384382.3384522

Ben Ezair, Saul Fuhrmann, and Gershon Elber. 2018. Volumetric covering print-paths for additive manufacturing of 3D models. *Comput. Aided Des.* 100 (2018), 1 – 13.

Guoxin Fang, Tianyu Zhang, Sikai Zhong, Xiangjia Chen, Zichun Zhong, and Charlie C. L. Wang. 2020. Reinforced FDM: Multi-Axis Filament Alignment with Controlled Anisotropic Strength. *ACM Transactions on Graphics* 39, 6, Article 204 (Nov. 2020), 15 pages.

Xifeng Gao, Wenzel Jakob, Marco Tarini, and Daniele Panozzo. 2017. Robust Hex-Dominant Mesh Generation using Field-Guided Polyhedral Agglomeration. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 36, 4 (July 2017). https://doi.org/10.1145/3072959.3073676

Martin Held, Gábor Lukács, and László Andor. 1994. Pocket machining based on contour-parallel tool paths generated by means of proximity maps. *Computer-Aided Design* 26, 3 (1994), 189–203.

Samuel Hornus, Tim Kuipers, Olivier Devillers, Monique Teillaud, Jonàs Martínez, Marc Glisse, Sylvain Lazard, and Sylvain Lefebvre. 2020. Variable-width contouring for additive manufacturing. *ACM Transactions on Graphics* 39, 4 (Proc. SIGGRAPH) (July 2020). https://doi.org/10.1145/3386569.3392448

Yuan Jin, Jianke Du, and Yong He. 2017. Optimization of process planning for reducing material consumption in additive manufacturing. *Journal of Manufacturing Systems* 44 (2017), 65–78. https://doi.org/10.1016/j.jmsy.2017.05.003

Ju-Hsien Kao and Fritz B Prinz. 1998. Optimal motion planning for deposition in layered manufacturing. In *Procs of DETC*, Vol. 98. 13–16.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2015. Stripe Patterns on Surfaces. *ACM Transactions on Graphics* 34 (2015). Issue 4.

Tim Kuipers, Eugeni L. Doubrovski, Jun Wu, and Charlie C.L. Wang. 2020. A Framework for Adaptive Width Control of Dense Contour-Parallel Toolpaths in Fused Deposition Modeling. *Computer-Aided Design* 128 (2020), 102907. https://doi.org/10.1016/j.cad.2020.102907

Prashant Kulkarni and Debasish Dutta. 1996. An accurate slicing procedure for layered manufacturing. *Comput. Aided Des.* 28, 9 (1996), 683 – 697.

Ares Lagae, Sylvain Lefebvre, George Drettakis, and Philip Dutré. 2009. Procedural noise using sparse Gabor convolution. *ACM Transactions on Graphics (TOG)* 28, 3 (2009), 1–10.

Marco Livesu, Stefano Ellero, Jonàs Martínez, Sylvain Lefebvre, and Marco Attene. 2017. From 3D models to 3D prints: an overview of the processing pipeline. *Computer Graphics Forum* 36, 2 (2017).

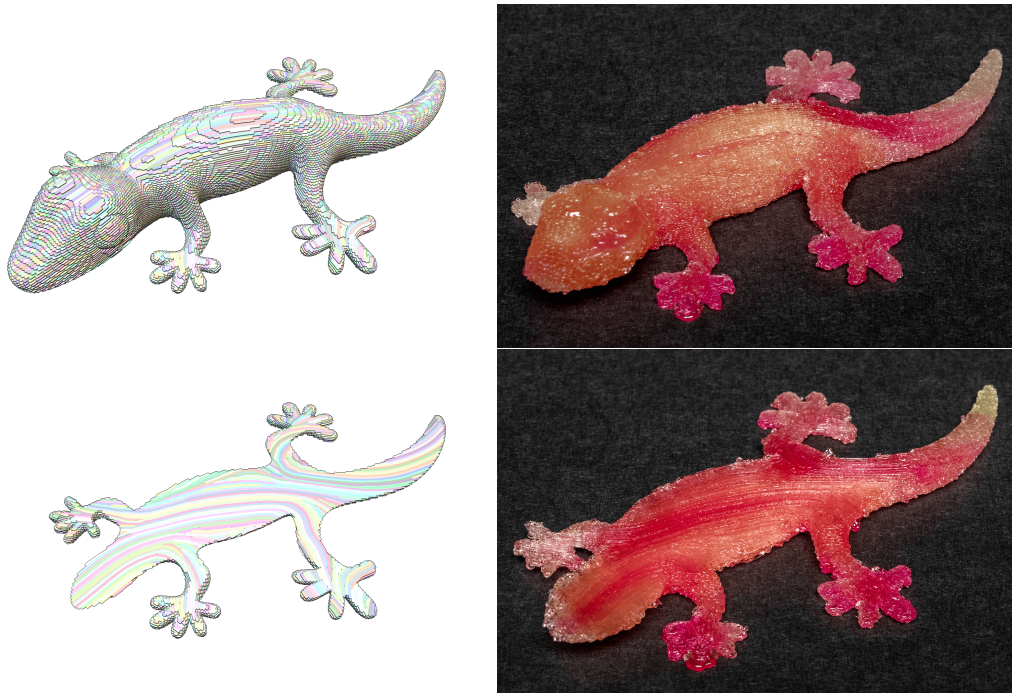

Fig. 11. Gecko object exhibiting a branching structure. **Left:** Paths are generated in the direction of branches while allowing long deposition paths in the neighborhood of the junctions. **Right:** The object was printed with a printer equipped with a diamond mixing extruder. The color mixing ratios were chosen to highlight trajectories direction and continuity.

Sara McMains, Jordan Smith, Jianlin Wang, and Carlo Séquin. 2000. Layered Manufacturing of Thin-Walled Parts. In *ASME Design Engineering Technical Conference*.

Peter Nicholson. 1823. *The New Practical Builder and Workman's Companion, Containing a Full Display and Elucidation of the Most Recent and Skilful Methods Pursued by Architects and Artificers... Including, Also, New Treatises on Geometry..., a Summary of the Art of Building..., an Extensive Glossary of the Technical Terms..., and The Theory and Practice of the Five Orders, as Employed in Decorative Architecture*. Vol. 59. Thomas Kelly.

PM Pandey, N Venkata Reddy, and Sanjay G Dhande. 2003. Slicing procedures in layered manufacturing: a review. *Rapid Prototyp J.* 9, 5 (2003), 274–288.

O. Pantz and K. Trabelsi. 2008. A Post-Treatment of the Homogenization Method for Shape Optimization. *SIAM Journal on Control and Optimization* 47, 3 (2008), 1380–1398. https://doi.org/10.1137/070688900 arXiv:https://doi.org/10.1137/070688900

Maksym Rybachuk, Charlène Alice Mauger, Thomas Fiedler, and Andreas Öchsner. 2017. Anisotropic mechanical properties of fused deposition modeled parts fabricated by using acrylonitrile butadiene styrene polymer. *Journal of Polymer Engineering* 37, 7 (2017), 699–706. https://doi.org/doi:10.1515/polyeng-2016-0263

Emmanuel Sabourin, Scott A Houser, and Jan Helge Bøhn. 1996. Adaptive slicing using stepwise uniform refinement. *Rapid Prototyp J.* 2, 4 (1996), 20–26.

John C. Steuben, Athanasios P. Iliopoulos, and John G. Michopoulos. 2016. Implicit slicing for functionally tailored additive manufacturing. *Computer-Aided Design* 77 (2016), 107 – 119.

Thibault Tricard, Semyon Efremov, Cédric Zanni, Fabrice Neyret, Jonàs Martínez, and Sylvain Lefebvre. 2019. Procedural Phasor Noise. *ACM Transactions on Graphics* 38, 4 (July 2019), Article No. 57:1–13. https://doi.org/10.1145/3306346.3322990

Thibault Tricard, Vincent Tavernier, Cédric Zanni, Jonàs Martínez, Pierre-Alexandre Hugron, Fabrice Neyret, and Sylvain Lefebvre. 2020. Freely orientable microstructures for designing deformable 3D prints. *ACM Transactions on Graphics* (Dec. 2020). https://doi.org/10.1145/3414685.3417790

Jun Wu, Weiming Wang, and Xifeng Gao. 2021. Design and Optimization of Conforming Lattice Structures. *IEEE Transactions on Visualization and Computer Graphics* 27, 1 (January 2021), 43–56. https://doi.org/10.1109/TVCG.2019.2938946 https://arxiv.org/abs/1905.02902.

Geoff Wyvill, Craig McPheeters, and Brian Wyvill. 1986. Data Structure for Soft Objects. *The Visual Computer - VC* 2 (08 1986), 227–234. https://doi.org/10.1007/BF01900346

Yi Xiong, Sang-in Park, Suhasini Padmanathan, Audelia Dharmawan, Shaohui Foong, David Rosen, and Gim Soh. 2019. Process planning for adaptive contour parallel toolpath in additive manufacturing with variable bead width. *International Journal of Advanced Manufacturing Technology* 105 (12

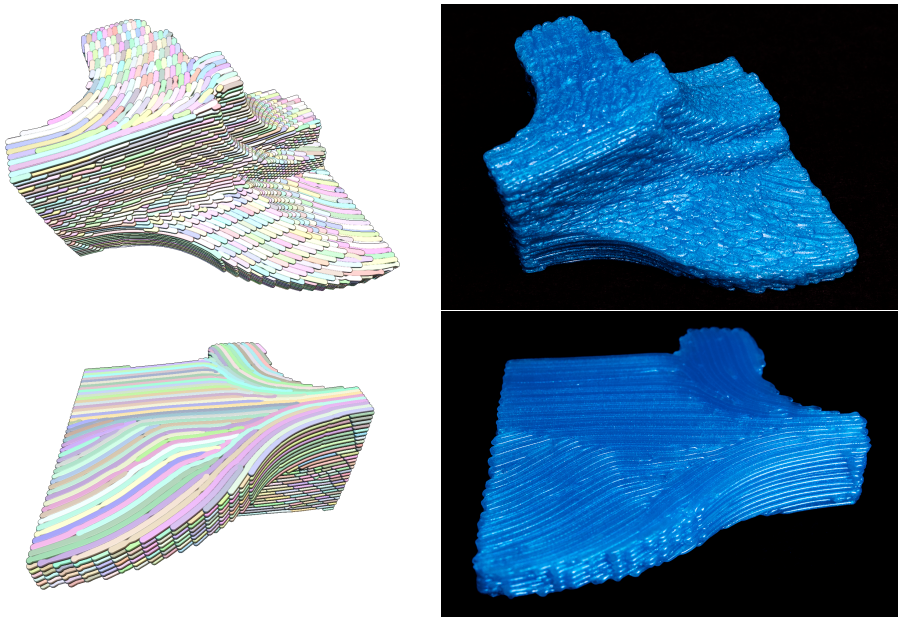

Fig. 12. A mechanical part. A view from below is provided in the second row to highlight the input direction field.

2019). https://doi.org/10.1007/s00170-019-03954-1

Ulas Yaman. 2018. Shrinkage compensation of holes via shrinkage of interior structure in FDM process. *The International Journal of Advanced Manufacturing Technology* 94, 5 (2018), 2187–2197.

Y Yang, HT Loh, JYH Fuh, and YG Wang. 2002. Equidistant path generation for improving scanning efficiency in layered manufacturing. *Rapid Prototyping J.* 8, 1 (2002), 30–37.