DOI: 10.1111/cgf.13929
EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)
Volume 39 (2020), Number 2

Interactive Modeling of Cellular Structures on Surfaces with Application to Additive Manufacturing

P. Stadlbauer[1], D. Mlakar[2], H.-P. Seidel[1], M. Steinberger[2], R. Zayer[1]

[1]Max Planck Institute for Informatics, Germany
[2]Graz University of Technology, Austria

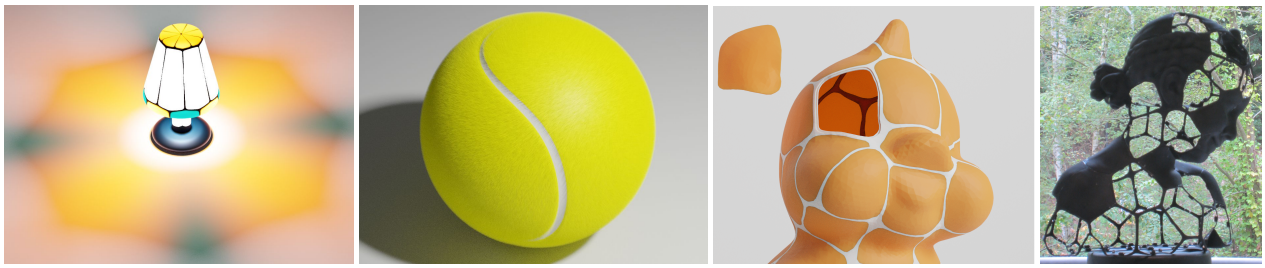


Figure 1: *Our editing primitives allow for elaborate pattern creation on surfaces as illustrated by the lamp (left) and tennis ball (middle left). The decomposition of models into a skeletal structure and inlay shells, as illustrated on the Tweety model(middle-right), channels material saving to shells and stability to the skeleton. Models larger than the 3D printer actual build volume can be conveniently decomposed and packed for printing and then assembled as shown (partially) for the Bimba model (right).*

Abstract

The rich and evocative patterns of natural tessellations endow them with an unmistakable artistic appeal and structural properties which are echoed across design, production, and manufacturing. Unfortunately, interactive control of such patterns-as modeled by Voronoi diagrams, is limited to the simple two dimensional case and does not extend well to freeform surfaces.
We present an approach for direct modeling and editing of such cellular structures on surface meshes. The overall modeling experience is driven by a set of editing primitives which are efficiently implemented on graphics hardware. We feature a novel application for 3D printing on modern support-free additive manufacturing platforms. Our method decomposes the input surface into a cellular skeletal structure which hosts a set of overlay shells. In this way, material saving can be channeled to the shells while structural stability is channeled to the skeleton. To accommodate the available printer build volume, the cellular structure can be further split into moderately sized parts. Together with shells, they can be conveniently packed to save on production time. The assembly of the printed parts is streamlined by a part numbering scheme which respects the geometric layout of the input model.

CCS Concepts
• ***Computing methodologies*** → ***Shape modeling;*** *Parallel computing methodologies;*

1. Introduction

The rich and evocative patterns of natural tessellations endow them with an unmistakable artistic appeal and structural properties which are echoed across design, production and manufacturing. Natural tessellations are commonly modeled using Voronoi diagrams or their centroidal variant[OBS92]. Although Existing algorithmic solutions are efficient enough to allow interactive creation and editing of such patterns in the planar setting, interactive solutions on surface meshes have not been proposed thus far. Voronoi diagrams require computing geodesics between the individual seeds which can be cumbersome and numerically expensive. A workaround is the so called *restricted Voronoi diagrams*, *i.e.*, the intersection of the surface and a volumetric Voronoi diagram. This approach in-

flates the problem size and requires extensive data structure management and elaborate solvers. Subsequently it does not scale well for large meshes, and faces additional challenges arising from complicated geometric and/or topological figures. A Voronoi diagram can be "faked" visually by undersampling the input mesh and using its dual. A more compelling approach is the one based on combining Poisson disk sampling with Voronoi vertex coloring [CCS12, BWWM10] using approximate geodesic distance.

In this work, we capitalize on a different numerical model for natural tessellations. Namely, the model proposed in [ZMSS18] which formulates tessellation formation as a growth process initialized at a set of given seeds (sites), or initial regions, thus guaranteeing a natively parallel formulation. Despite considerable performance gains, the cost is still prohibitive for interactive editing and the method is limited to static seed configurations. We streamline this approach for direct editing and address several challenges arising from the need to perform fast local operations on irregular data structures. For instance, user actions such as the insertion and removal of seeds or the merging of cells introduce dynamic changes to the sparse matrix representation of the field which needs to be updated instantaneously as it is required for the ensuing field computations as well as incoming user actions. Furthermore, we introduce an efficient backward propagation which performs cell boundary offseting on the GPU . This allows smooth contour generation for making local patch cuts. To the best of our knowledge, our work is the first which allows:

- interactive cellular tessellation editing and partitioning
- fully parallel pipeline running on GPU

Our approach naturally extends to surface decomposition. The original input mesh can be decomposed into a hollow cellular structure which acts as a skeleton of the surface and a set of thin shells which fit into the hollow cells and act as the surface skin. We see this decomposition as a practical solution for cutting down on material use in additive manufacturing (AM). In fact, channeling structural stability of the surface to the skeleton allows printing thinner shells and thus reduces the amount of consumed substrate. In order to accommodate printers with smaller printing beds or to print larger models, the cellular structure itself can be further partitioned into smaller parts and packed along with the shells for production. In order to facilitate the assembly of the final product, a meaningful bookkeeping strategy is needed. We propose a numbering strategy for the thin shells and the partitioned cellular skeleton which respects the geometric layout of the original model and reflects the progress of the assembly itself. This leads to the following contributions:

- generic mesh decomposition
- generic mesh assembly
- geometrically meaningful part labeling

While some of the ideas outlined herein may be useful across various printing platforms, the overall pipeline targets processes which do not require artificial supports during printing and therefore allow packing multiple parts more freely within the available build volume.

Hollow skeleton structures such as the ones targeted by our work bear some similarity to the ones resulting from topology or structural optimization methods [BS03]. As such methods can be adapted for different physical or structural requirements, there is a growing interest in the modeling community for using them to steer certain design and/or functionality objectives, e.g., accommodating for thermal comfort in orthopedic casts [ZFD*17]. Nonetheless, their computational cost is prohibitive for interactive use. The topology optimization community have come lately to the realization that designers and architects are not flocking towards computed solutions because "design is mostly driven by aesthetics and is influenced by the designer's inspiration, which is not always easily amenable to a mathematical formulation" [DFM*17]. Our objective is to provide basic and intuitive tools for bringing such inspiration to life without compromising design freedom.

2. Related work

The dominant numerical model for natural tessellations is centroidal Voronoi tessellations (CVT). It can be obtained from a Voronoi diagram by iteratively moving seeds to the centroids of their respective cells [Llo82]. This procedure has been reformulated as an optimization problem in [IMO84] and has steered a fair amount of literature, see e.g. [DE06] and the references therein. The treatment of the arising optimization problem using L-BFGS has been proposed in [LWL*09]. While most of these formulations work well in planar and volumetric settings, evaluations on surfaces have to be performed as a restriction of a volumetric configuration [YLL*09] or in the plane by means of surface parametrization, e.g. [AVDI03]. While restricted Voronoi diagrams (RVD) have been extended to account for lines and graph on the surface [LLW12], any attempt to further boost their performance is hampered by the difficulty of coining fine grained parallelism strategies for numerical solvers such as L-BFGS on irregular grids. Furthermore, preprocessing and postprocessing operations which include, sizing field generation, initial tetrahedral mesh creation, and Delaunay meshing, induce additional computational and memory costs. Recent progress such as the one reported in [MPR19] for Delaunay computations suggest that some of the steps can be improved but the streamlining of the overall RVD procedure remains challenging. Parametrization based approaches have been able to tap into the power of modern graphics hardware, e.g., [RLW*11], but generating proper cuts for surfaces of arbitrary genus as well as parametrization cost pose additional challenges that add up to the unavoidable metric distortion. Existing extensions to the hyperbolic setting offer partial remedies but still suffer from conformal distortion [RJSG11, SGJ13]. As we target direct editing these workarounds are not suitable for our purpose as the interaction needs to take place on the surface itself.

Direct evaluations on surfaces are limited to small meshes as the cost of geodesics on surfaces is relatively high [WYL*15, XLC*16]. The approach of [HHA17] uses the heat diffusion approach in [CWW13] to approximating geodesic distances but numerical evaluations are still far too prohibitive for use in modeling. The approach of [PC06] uses fast marching [KS98, Set96], but comes with all the limitations of level-sets. From a simulation standpoint, early methods for producing patterns on surfaces, *e.g.*, [Tur91] can yield pleasant Voronoi-like patterns, however their lack of controllability and the difficulties in

extracting relevant cells inhibits the adoption of these methods as geometric partitioning tools.

The need for surface partitioning arises across different problems in additive manufacturing. Pertinent issues such as material saving, structural stability, scalability, packing, motivated a steady research effort aimed at one or multiple objectives. In this brief overview, we will restrict ourselves to the most closely related efforts. The reader is referred to, e.g.,[LEM*17] for a recent overview of computer graphics literature and to [NKI*18] for a broader scope on existing processes, their limitations, and prospects.

For certain geometric models, the partition is dictated by semantics. For instance, articulations have been explored in the works of [BBJP12] which deals pre-skinned models with kinematic constraints. In the work of [CCA*12], the user assists and interacts with the rigging, joint placement, and rotational constraints. Both methods rely on the ball joint as a design primitive for articulations. The work of [Att15] focuses saving packaging space in view of cutting down delivery costs of printed parts.

More closely related to our current work are the efforts in [LBRM12, VGB*14, YCL*15]. In [LBRM12] models larger than the available printer volume are broken into smaller pieces. On the other hand, the optimization proposed in [VGB*14] targets reducing the amount of support material and the bounding box volume of the packed segments while keeping the number of segments minimal. The method does not operate directly on the original input but on its tetrahedralization which can lead to large problem sizes when the input mesh is highly detailed. The method proposed in [YCL*15] iteratively interleaves a partitioning optimization step targeted at some relevant quality metrics with a secondary step aimed at tightening the packing quality of the partition within the printer tray. It takes advantage of the distance field associated with the underlying level set representation for handling collision detection. For printing large models, the authors in [SDW*16] propose customized 2D laser cuts which support an external set of coarse outer surface cuts obtained by optimization. The approach presume the shape has enough hollow inner space for hosting the approximating convex polyhedrons and therefore is limited to blobby shapes. Similarly the approach of [CLF*18] uses cubic building blocks to fill-in the inner core of shapes and approximates the residual volume using small pyramidal pieces. The limited scalability of the blocks hampers the overall scalability of the solution.

Avoiding partitioning in favor of microstructure, the authors in [MHSL18] save material and control the properties of the printed shape by use of hollow microstructure thus extending the capabilities Fused Filament Fabrication.

Many of these solutions assume low cost Fused Deposition Modeling (FDM) is used, and their material consumption is still relatively high. As such they are not economic enough for our particular setting as the material cost in technologies such as Selective Laser Sintering (SLS) and Multi Jet Fusion (MJF) is much higher and therefore greedier strategies are needed. Furthermore, the use of expensive numerical optimization methods slows down the modeling-to-printing pipeline and locks the designer out of the decision loop as the focus is on generating a set of cuts fulfilling a number of prescribed constraints automatically. Consequently, there is a lack of modeling tools inlined with the traditional CAD machinery, that allow the user to interactively design the partitions and interact with them in a meaningful manner.

Aiming at more classical manufacturing technologies such as 3-axis milling, or casting, several authors decompose existing freeform shapes into manufacturable patches, i.e., admitting a height field representation [HMA15, MLS*18]. For fabrication from flat materials, the authors in [SC18] explored the generation of surface cuts which account for the induced flattening distortion based on an appropriate curve evolution formulation. The problem of decorating surfaces with predefined stencils while satisfying certain stability constraints has been explored in [STG16].

3. Natural tessellation model

In this work, we adopt the recent natural tessellations generation approach proposed in [ZMSS18] which models tessellation formation as a growth process initialized at a set of given seeds (sites), or initial regions, and driven by the evolution of the growing regions boundaries (interface). Starting with a set of n initial seeds over a given surface, we associate to them n evolving regions (cells). Each region is defined by a function φ_i which takes value 1 inside the cell and 0 outside the cell. Most importantly the boundary of each region is not defined as a line but as a narrow band, where $0 < \varphi_i < 1$, to avoid the numerical problems commonly encountered when dealing with sharp boundaries such as derivative discontinuities. The growth process is similar to a diffusion from the input seeds or regions at similar rates, intertwined with a set of rules: (i) Partition of unity (the different fields φ_i sum up to 1 at any given location on the surface). (ii) Integrity and locality of the individual cells in terms of the weighted product $w_{ij}\varphi_i\,\varphi_j$. (iii) Distinguishability of the individual narrow bands in terms of the weighted gradients $-\frac{1}{2}a_{ij}\nabla\varphi_i\,\nabla\varphi_j$. (iv) Behavior of narrow bands under contact which we model by the function $\sqrt{\varphi_i\varphi_j}$ weighted by a scalar e_{ij}. The minimization of the Lagrangian associated with the energy arising from the above considerations yields the evolution of the fields as a time dependent equation

$$\dot{\varphi}_i = -\sum_{j=1}^{n}\frac{\mu_{ij}}{n}\left(\sum_{k=1}^{n}\left[(w_{ik}-w_{jk})\varphi_k + \frac{1}{2}(a_{ik}^2-a_{jk}^2)\nabla^2\varphi_k\right] - e_{ij}\sqrt{\varphi_i\varphi_j}\right). \tag{1}$$

In this equation, the growth rate is controlled by the mobility term μ_{ij}. The different fields can be thought of as individual layers on the surface and can be numerically encoded as a sparse matrix Φ where each column represents a vertex of the original input mesh and each row represents a cell. Our discretization on surface meshes uses the standard linear finite element approximation.

To steer the field evolution, an additional base layer is initialized to 1 at all vertices. When seeds are initialized, the field Φ is normalized by dividing each column by the sum of its components. This normalization is then performed after each time step. Furthermore, instead of a costly explicit tracking of cross-cell interactions at each iteration, we use the base layer for carrying this task implicity. As two boundary bands move towards each other (or multiple bands at a junction), the base layer between them erodes and its effect on the boundary band is replaced by the one from the incoming cell. With this in mind, both e_{ij} and the mobility term μ_{ij} are set to zero except when either i or j index the base layer. In this case, we used $\frac{1}{30}$ and $\frac{1}{4}$ resp. in our current implementation. The values of the gradient energy coefficients $a_{i,j}$ are set to $\frac{1}{25}$, and the penalty term $w_{i,j}$ to

$\frac{a_{i,j}}{5}$ for $i \neq j$, and to 0 otherwise. This formulation offers a clear advantage in view of efficient parallel implementation as conditional branchings and costly cell-cell interaction tracking are avoided.

After convergence, the cell-cell adjacency matrix can be obtained starting from the initial estimate $A_t = (\bar{\Phi}\bar{M})(\bar{\Phi}\bar{M})^\top$, where $\bar{M}$ is the binary form of the so called mesh matrix of size $n_v \times n_f$ and encodes face-vertex incidence with n_v and n_f being the number of vertices and faces respectively [ZSS17], and $\bar{\Phi}$ is the binary sparse matrix obtained by setting values of Φ that pass a given threshold to 1 (in our experiments the threshold was set to .25). This estimate is then curated for manifoldness and can be used to construct the dual mesh of the cellular tessellation [ZMSS18].

4. Editing requirements

During editing, our approach allows cell visualization with visible narrow band between the cells or sharp cell boundaries. We will use both indistinguishably in our illustrations.

4.1. Initial seed placement

Our cellular tessellation generation can be initiated by letting the user sample points on the surface manually. This can take a long time if the desired seed count is high, thus spoiling the editing experience. To avoid that we start the editing process from a initial seed distribution. Multiple options are available. The simplest is to randomly sample a certain number of seeds from the original mesh vertices. This requires the user to have some broad sense of how many seeds are desirable and does not guarantee a geometrically or esthetically meaningful sampling.

Visual cues on the initial number of seeds can be inferred from subdivisions of the bounding box [O'R85] of the input model. This results in a coarse voxelization of the model and seeds can be placed on the patches within occupied voxels. More seeds can be then inserted by the user or automatically by forcing subdivision of resulting parts.

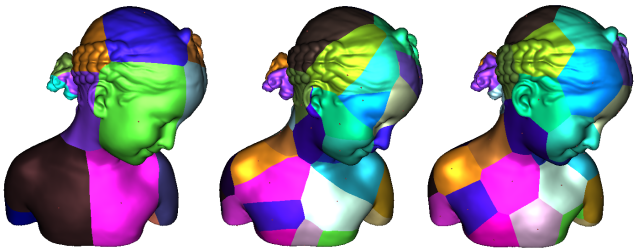


Figure 2: *Coarse voxelization based seed placement(left), random placement(center), and CVT-like placement(right).*

Once the initial seed placement is decided, we run the layered field growth starting from the seeds and we get the initial partitions. For both types, the results are shown in Figure 2-left and middle and they clearly lack in regularity. If a highly regular partition is sought, the user can run the layered field in a Lloyd-like fashion to improve part sizing similar to centroidal Voronoi diagrams. This yields regularly shaped regions and a visually more compelling partitions, see Figure 2-right.

4.2. Interactivity requirements

Even well generated seed placements lack adaption to mesh features and more importantly do not reflect any esthetic aspects as understood by artists and designers. Therefore, there is need to introduce direct editing operations.

A naive approach to interactive editing is to recompute the overall tessellation after each operation by re-running the whole seed growth algorithm with the newer configuration. This approach would lead to run times exceeding the short time response necessary for a fluid modeling experience.

The computational cost in the layered field calculation is dominated by the sparse matrix-matrix multiplication for the field Laplacian evaluation, see Equation 1. In order to cut costs down, each editing operation in our approach extracts the affected regions which we call active area.

4.3. Active area based editing

Once the user selects a seed or a vertex, we restrict the propagation to its n-th neighborhood (in terms of seeds within the dual mesh of the cellular tessellation). In this respect, the layered fields matrix Φ holds all the necessary information. Different seeds/cells are considered neighbors/adjacent if they contribute nonzero values at a given vertex. This can be easily obtained by looking at their corresponding rows in matrix Φ and picking the overlapping nonzero values.

Technically, for a specific seed, we add the direct neighboring seeds into a global hashmap. Performing the same operation again for the content of the hash map yields the 2-neighborhood. Another iteration the 3-neighborhood and so on. The n-th neighborhood used in practice depends on the size of the underlying mesh and the number of seeds. As a single cell can contain a few thousand faces, the default value of n is based on the mesh resolution. It is set to 1 for dense meshes and to 2 for coarser ones. If desired, the user can increase the range to resolve any artefacts.

The active area of a given vertex is extracted in the same way, starting with cells which contribute nonzero value to that vertex. Please note, that the vertex can lay within a single cell or on the narrow band across regions. The 1-neighborhood of vertex is then the regions that contribute nonzero values to it and their immediate neighbors.

To limit the interaction to the active area, we extract all vertices of the active area from the mesh matrix M, creating a smaller matrix. This step essentially corresponds to copying those columns associated with the respective seed contributions, which can be done efficiently using a prefix scan over the involved column pointers. The Laplacian for all vertices belonging to the active area is also extracted. Thanks to this reduction of matrix sizes, the execution time of the matrix-matrix multiplication, which is the major factor, is shortened. After completion, the extracted regions are merged back into the layered fields. All these operations are carried on graphics hardware so there are no back and forth trips between the CPU and the GPU and therefore no idle time.

5. Editing primitives

Our editing operations can be classified into simple primitives and more specialized operations targeting multiple seeds or specific design goals. For the sake of high level algorithmic clarity and exposition flow we will skim technical details about code optimization for the GPU.

5.1. Seed-based editing primitives

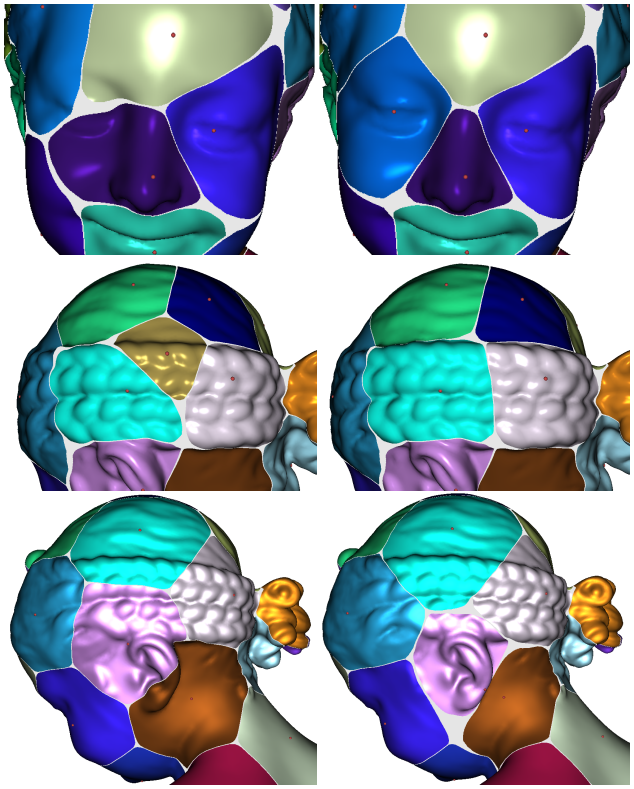


Figure 3: *Illustration of different seed-based editing primitives applied to the original cell layout (left-column) and their outcome (right-column). A Seed is inserted to better capture face details at the left eye (top). A Seed is removed for better enclosing of the braids (middle). A seed is moved to fully enclose the ear (bottom).*

Seed insertion Seed insertion allows additional seeds to be placed on the surface, for instance, for capturing finer geometric details. As the user picks a vertex to be added as a seed, the active area of that vertex is then determined as discussed above. The current field values for vertices within that area are added up to their corresponding entry in the base layer and then set to zero. Seed vertices are an exception as they only have a single field value which is set to one. To guarantee smooth transition on the overall surface, field values at vertices on the outer boundary of the active area are kept unchanged. In this way, the regrowing is limited the 1-neighborhood and performance gets improved. Regrowing on the 2-neighborhood would resolve the smoothness issue, but at a slightly slower rate. Additionally, a reduced Laplacian is extracted for the active area to keep the size of involved matrices as low as possible.

The growth algorithm is then run within the active area, while the borders are kept unchanged. After reaching equilibrium the resulting field values are re-injected back into the layered fields matrix. An illustration of seed insertion is shown in Figure 3-top.

Seed removal Regions where seeds are too densely distributed can be relaxed by seed removal. Similarly to the seed insertion operation, a reduced layered field and Laplacian are used. Since only the selected cell is removed, the active area reduces to the cell area. Neighboring cells then grow into the now void space until equilibrium is reached and then the results are plugged back into the global layered fields matrix. An illustration of seed removal is shown in Figure 3-middle.

Seed movement Adjusting an existing cell layout only by seed removal and insertion operations can be cumbersome as it requires a lot of user interaction. The modeling experience can be made more fluid by allowing seed movement on the surface while giving an instantaneous visual feedback as the user displaces them. In theory, seed movement can be interpreted as the combination of seed removal and insertion operations. In practice, however, the layer re-use and knowledge of the seed location allows for better code optimization.

The active area is extended to include the active area of the movement endpoint. Additionally, before re-initializing the propagation process the coordinates of the selected seed are set to the endpoint location. An illustration of seed movement is shown in Figure 3-bottom.

5.2. Cell-based editing primitives

Cell merging In many editing scenarios, adjacent cells may need to be merged to capture certain surface features more compactly. In this setting, the user specifies the two cells to be merged. The corresponding two layers are then merged into a single layer and all seeds within are marked so as the same cell is re-created should another editing occur. Figure 4-top shows the merging of two cells to enclose the bow on the head of the Bimba model in a single cell.

Cell freeze To avoid interfering with readily established or edited cells, the user can elect to freeze certain cells while performing other editing operations such as seed movement. In this setup, editing operations are made blind to frozen regions. Literally, this translates to removing them from the active area. In Figure 4-bottom, The ear region is frozen to avoid changes when its neighbors are moved. One can see the light blue has been adjusted without interfering with the ear region.

5.3. Design specific primitives

The above-discussed primitives are flexible enough to allow for the creation of more elaborate editing operations. Without being exhaustive, we show some of these operations as proof of concept.

Seed-Lines In certain settings, it is desirable to steer cell growth in a specific manner, as opposed to its the current isotropic form, while enforcing a smooth stable cell border. This can be achieved

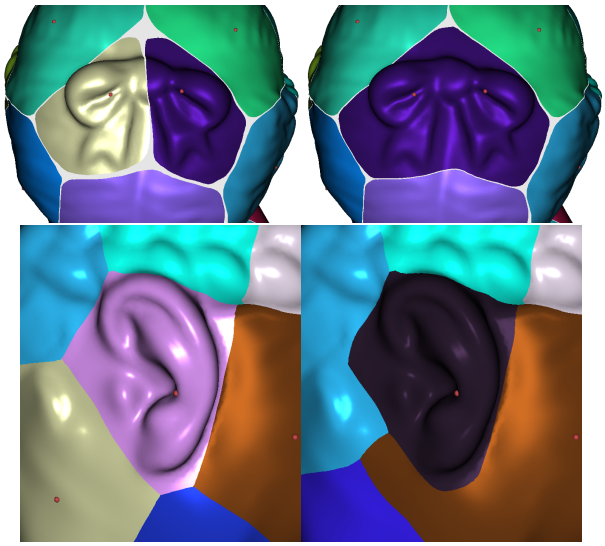


Figure 4: *Merging of the bow cells (top). Freezing of the ear cell during neighboring seed movement.*

by placing seeds along a user defined curve and letting them grow within the same layer, i.e, the same row of the field matrix, this accounts for internal border merging while guaranteeing a smooth stable border. A basic illustration is shown in Figure 5-left. A more elaborate result is shown for a tennis ball in Figure 1. Although the creation of such results might seem trivial using our formalism, it is worthwhile to note that producing similar effects using other frameworks is not as trivial. For instance, the elegant extension of RVD to handle lines and graphes proposed in [LLW12] is sensibly much more involved.

Seed-Rings Specific operations such as placing seeds in a ring shape around a given axis can be performed as well. An illustration of such operation is shown in Figure 5-middle. A more elaborate use case is shown for the decorative lamp model in Figure 1.

Region painting To avoid unnecessary editing of some sophisticated and visually meaningful features, the user can paint them directly as cells. The example in Figure 5-right shows an eye specified as a individual cell in red.

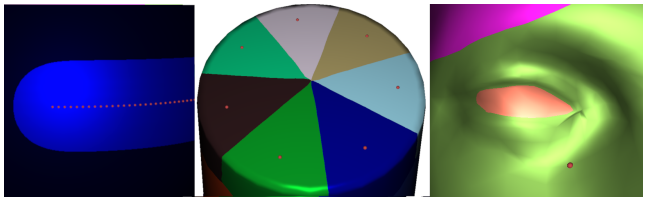


Figure 5: *Seed-Line (left), Seed-Ring (center), Region painting (right).*

Extendability We have restricted this exposition to a few base operations, which do not necessarily require user expertise in differential geometry. In practice, however, one can extend and combine these operations to achieve more elaborate effects. For instance, landmarks such as crest lines, e.g. [YBYS08], can be used to block cells from growing beyond them. We plan to explore this direction extensively in future work.

Figure 6: *Parsimonious material use in nature. The leave skin is supported by a more structurally stable network of natural tessellations. Artistic use of such structures is featured in the "Poly" sculpture by Julian Voss-Andreae. The* 140 *kg cast bronze with patina is installed outside Georgia Tech's Engineered Biosystems Building.*

6. Applications in additive manufacturing

The ability of additive manufacturing (AM) to materialize an object from semi-finished substrates has long made it a viable prototyping tool. Over the last decade or so, rapid progress has swept the underlying technologies allowing adoption of the process in rapid manufacturing. In this respect, powder bed fusion based methods such as the most recent Multi Jet Fusion (MJF) technology [HP18] or the more classical Selective Laser Sintering (SLS) are appealing, as they require no additional support structures during printing. In fact, the unused powder itself provides the part being printed with the necessary support. This makes them particularly suitable for freeform geometries that are difficult to manufacture with other methods. On the other hand, these technologies suffer from a few major limitations, namely, higher printing costs, limited build volume and relatively slow processing. In this section, we attempt to provide a generic method for addressing these issues while adapting the unmistakable artistic appeal of natural tessellations, as elegantly captured by the "Poly" sculpture in Figure 6, to the process.

6.1. Material saving through surface decomposition

Inspired by the parsimonious material usage in natural structures as illustrated in Figure 6, where the lean surface of a leave is sustained by a cellular tessellation structure, we propose decomposing the original input mesh into a hollow cellular structure which acts as a skeleton responsible for carrying the major load and a set of thin shells which can be inlaid into the hollow cells and act as the surface skin, see figure 7. In this way, the structure of the skeleton can be strengthened without inducing substantial material usage while the shells thickness can be thinned to drive down material consumption. Within the layered field formalism, the section between the cells defines the base for a multilayered skeleton, and the shells are enclosed and held by the skeleton on its uppermost layer. Editing the regions not only allows an esthetic improvement

when printing in different colors, it also allows surface details to be captured within single cells to avoid breaking some salient features, as seen in Section 5.

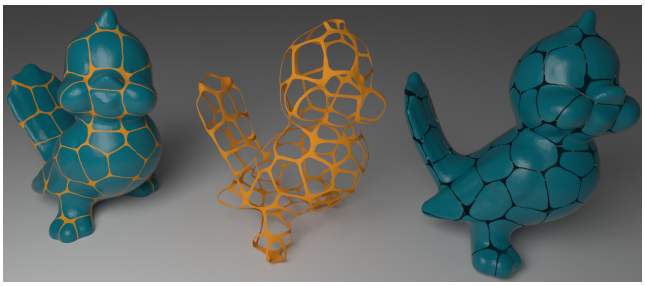


Figure 7: *Decomposition of the Tweety model.*

In practice, we use the isolines of the layered fields within the narrow-band separating the cells to split the original mesh into a skeletal structure and shells. In order to control the skeleton properties, such as the width of the narrow band or the size of the shell inlays, we need be able to adjust the isolines globally or locally as needed. Since we do not want to re-run the whole field propagation with a different setting for the narrow band width because the user might have already edited the fields to her heart's desire, we need a strategy to retract the current narrow band in a consistent manner.

Direct threshold adjustment is not a valid strategy in our setting due to the limited size of the narrow band itself. If we attempt to retract the boundaries naively on the surface, by simple interpolation along narrow band gradients, the boundary looses it original smoothness as shown in Figure 8.

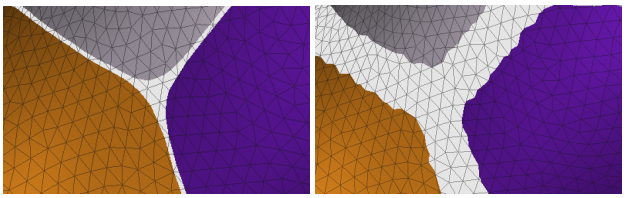


Figure 8: *Naive adjustment of converged cell boundaries (left) induces jaggy lines (right).*

Instead, we propagate the field backwards from the narrow band towards the seeds within the individual cells. This direction reversal is achieved by simply inverting the sign of the coefficient accounting for the boundary condition, namely the e_{ij}'s. Figure 9 shows the smooth evolution of the layered field during reversal.

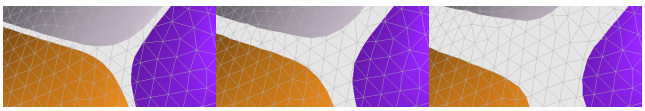


Figure 9: *Smooth border retraction by reversal of field direction.*

6.2. Stability

Besides enhancing the user's interactive design experience and reducing material consumption, our objective is to guarantee the stability of the printed structure. In the current design-to-print pipeline, we use a multilayered skeleton as shown in the inset. The outer layer has an inlay which hosts the shell and matches its geometry, the extent by which the second layer supports the shell is a free parameter that we call the inlay step size. For a given cellular layout, the final structure can be controlled via the following parameters: inlay step size, number of steps (layers), and initial skeleton (border) size.

The inlay size can expanded to provides more support for the shells and enhance the strength of the skeletal edges lengthwise, see Figure10-top two. The field reversal strategy is used for this purpose as discussed above. The depth of the step which host the shells can be also adapted to accommodate thinner shells. This can be set to the specifics of each printing technology and material. Reasonably low values can reach limit wall thickness which is in the range of .6 to .9mm.

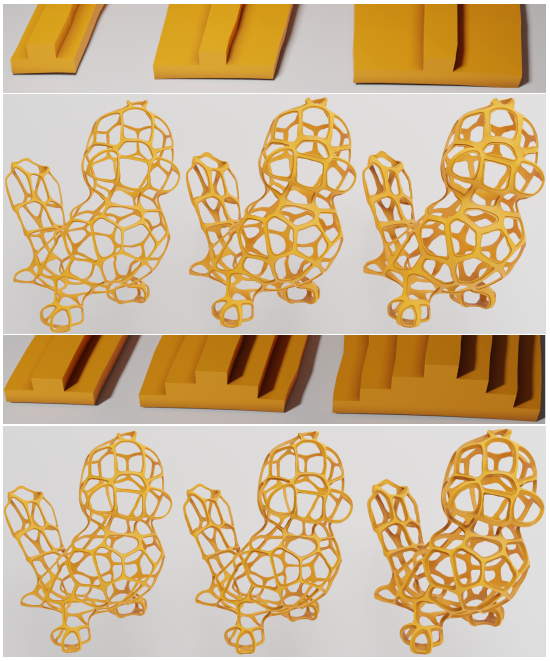


Figure 10: *Varying inlay step size(top), varying step count(bottom). Please zoom in on the electronic version to see the details.*

To provide more transversal support to the skeletal segments, we use a staircase profile, see Figure 10-bottom two. The number of steps, and their thickness can be controlled to account for material usage, and strength. In addition, all the operations discussed above can be performed also locally, giving the possibility to strengthen a certain regions by itself. An example is showing local increase of the inlay size is given in Figure 11.

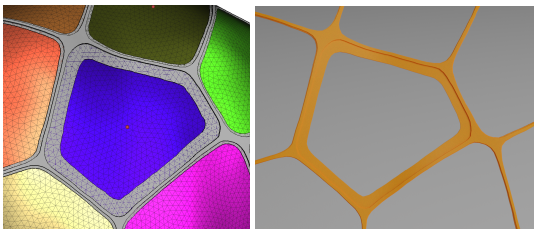


Figure 11: *Local adjustment of the inlay step size of the central cell.*

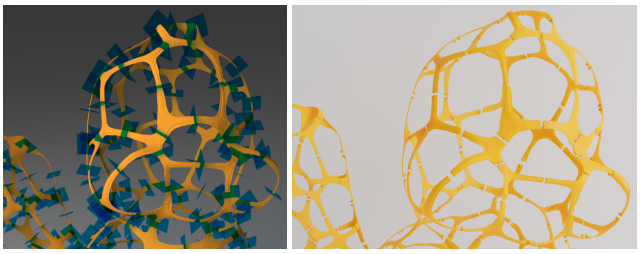


Figure 12: *Cut placement (left) and the corresponding partitioned skeleton in exploded view (right).*

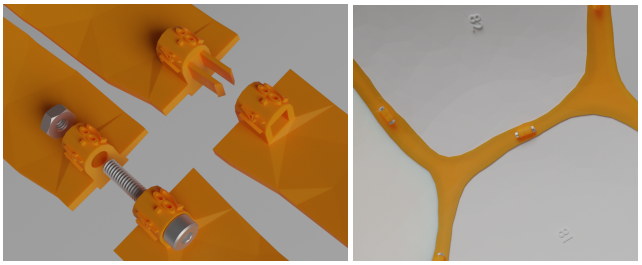
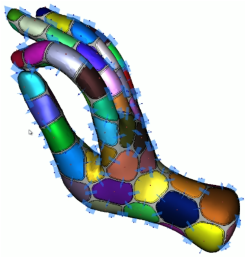


Figure 13: *Different types of connectors (left) and an inner view of a partial assembly of shells and skeletal parts. Please note the numbering on the connectors and the shells.*

6.3. Scalability to build volume

Despite the existence of a few large scale printing solutions, the build volume offered by the majority of 3D printers is limited. It is therefore customary to partition larger volumes into smaller parts that would fit into the available printer bed, but control over the partitioning remains a problem, see Sect. 2. This can be detrimental especially in fields where the cut placements impacts the visual quality of the final print as in arts, fashion, and architecture. Furthermore, the choice of the printing technology largely guides the partitioning decisions and dramatically affects the printing time, and costs. For instance, printing the plastic core for the mold used for the "Poly" sculpture in Figure 6-right took about 10,000 printing hours on several machines using fused deposition modeling technology (FDM) [lul17].



Our surface decomposition readily allows creating small shells suitable for packing into the printer usable volume. We propose to further partition the hollow cellular structure as well. We introduce cuts by placing cutting planes orthogonal to the edges of the dual mesh of the cellular structure and passing through their midpoints as shown in the opposite inset and in Figure 12. As the individual skeleton parts need to be assembled after printing, the positioning of the cutting planes need to leave enough room for the placement of connectors. If the surface curvature in the center exceeds a certain threshold the cut plane is moved along dual edge far enough to a less curved section. Special attention is needed to allow accessibility to connectors during assembly as well. For this reason, the user can still adjust the generated cutting planes interactively to make sure the final print can be assembled properly. Typical cuts are shown in Figure 12.

After the skeleton is split, connectors are placed across the cuts. The connection mechanism can be chosen according to specific requirements for example mesh size, printing material, possibility of assembly and disassembly, or the predicted stress on the model. For instance, when the base printing material is metal powder, connectors such as printable push-snap or snap-fit connectors, Figure 13, might not be a viable option and may also complicate disassembly. In such case, the use of machined bolts might be a much more reliable, see Figure 13.

As a side note, in applications where the goal is to have prints without connectors protruding in the inner side of the model, for instance, the creation of molds (dipping printed plastic material inside a ceramic slurry and burning it away), we do not require a step structure. Snap-fit joints can be placed on the sides of the shells and their opposing counterparts on the cellular skeleton.

7. Part labeling

The number of partitions might vary sensibly depending on the desired number of seeds, the 3D printer build volume, and the desired size of the output. In order to simplify the assembly process, we propose assigning number ids to the shells first and then numbering the junctions of the hollow cellular partitions based on the numbers of the two shells that border them, i.e., on each junction we print the number of its two bordering shells as illustrated in Figure 13. In this way the identification is straightforward. The inscribed numbers ensure that the printed results can be assembled correctly but do not provide any cues about the progress of the assembly itself and leave the end user in front of a large 3D puzzle. A random numbering might require the user to search for part numbers in an unguided manner.

To streamline the assembly process, we seek to minimize the size of the advancing front of the assembly as the user adds more parts. This amounts to minimizing the wavefront of the binary mesh matrix $\bar{M}$, see formal definition in Appendix A. This coincides with the notion of streamability introduced in [IL05] where the authors point out the importance of spectral reordering as it tends to follow the geometry of the mesh (which makes it literally more streamable especially within the out of core setting of their problem).

The spectral sequencing performance reported in [IL05]) is suffi-

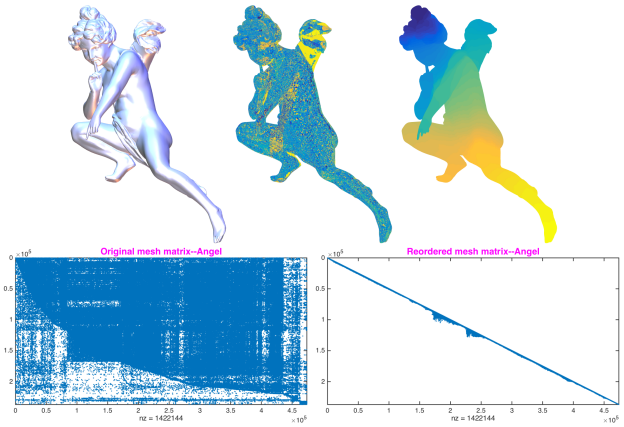


Figure 14: *Visualization of the effect of ordering on mesh traversal of the Angel model (top-left) using its original ordering (top-middle) and spectral ordering (top right), the corresponding mesh matrices are shown in the second row.*

cient within their out of core setting, but too prohibitive in our context. In our view, the bottleneck lies in their initial mesh decimation strategy [ILGS03], which is driven by geometric error. Since we seek only an approximation to the Fiedler vector there is no need to enforce geometric faithfulness as suggested in [KP97]. We adopt an algebraic multi-grid strategy driven by an aggressive simplification of the Laplacian matrix itself [Stu99]. Our current implementation clocks 42s for the 28MΔ Lucy model from the Stanford repository. For all models in this paper, only a few seconds were needed.

In practice, it would make sense to perform the labeling on the small dual mesh of the cellular tessellation, this however would require computing the Fiedler vector each time a new decomposition or an update to the tessellation is performed. As efficient eigen-analysis on the GPU is still a challenging research topic in its own right, we compute the Fiedler vector on the initial input and then we interpolate the values to the seed locations based on the nearest vertex. In this way, we maintain a fully streamlined pipeline which enhances the designer's experience. In Figure 14, the difference between the original ordering of the Angel model, which is random as there is no reason to expect meshes to be written in any particular order, and the spectral ordering.

8. Packing

As we target printing technologies which do not require building artificial support structures. The resulting partitions obtained by our method can be packed tightly into the printing volume. In this way, we can increase the number of parts per build. There are fairly good performing heuristics for 3D packing both in academia, e.g [ENO07, EW15], and industry, e.g. Netfabb and Fabpilot. In our setting, we have the additional advantage that the decomposition readily yields potato chip like shells which are ideal for packing, whereas the partitioned skeleton parts come generally in the form of triadic parts which are relatively easy to pack. Therefore we do not have the risk of running into problems such as higher genus parts locking into each other (in an olympic rings fashion).

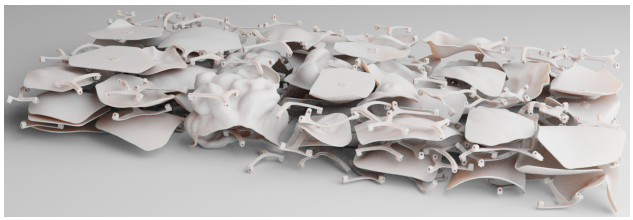


Figure 15: *Packed shells and skeleton of the Bimba model.*

Figure 15 shows a single batch packing of the 1000mm high Bimba model.

9. Results and discussion

Throughout our experiments, we used the same parameters. Our hardware configuration consists of an Intel i7 6800k CPU with 32GB of memory and an NVIDIA Geforce Titan Xp with 3840 compute cores and 12GB of memory.

Typical results of our approach are shown in Figure 1. The lamp model shows that our editing primitives allow fashioning highly elaborate patterns. The tennis ball example, shows that without performing any direct measurement on the surface, our cell boundary curves can be adapted to produce the distinctive seam line of tennis balls. Figure 16 shows edited cell layout for various meshes featuring different genuses and thin parts. In certain examples, such as the hand, the editing allows close control of the seeds such that the resulting cells aline with the finger articulations.

Direct editing offers designers the unique ability to control cell placement in ways that are often hard to achieve otherwise, e.g., using optimization or parametrization. For instance, on the face of the Tweety model, cells can be placed so as to avoid seams crossing distinctive and salient features. The same observation applies to the bunny and horse models. Furthermore, subtle esthetic aspects such as respect of partial symmetry, can be easily enforced in relevant areas as can be seen throughout the models.

Editing performance Performance highly depends on the size of the input mesh, the number of seeds, and their geometric configuration, in a sense, the amount of field growth needed to capture the active area and the size of its underlying triangulation. To get an objective overview, we analyzed the case of seed insertion on different subdivision levels of the Icosahedron.

Starting with 50 randomly distributed seeds on the subdivided sphere, we perform 100 insertion operations at random locations and the we report the average cost of a single insertion in table, 1. In each, case we compare the performance of using the active area against the re-initialization of the full layered field growth. At early subdivision levels, the use of the active area is slower due to the additional processing overhead on the GPU and the negligible cost of the overall computation. As the mesh gets finer (closer to an actual sphere), the benefits of the our selective process become more noticeable. Interactive editing sessions are shown in the accompanying media.

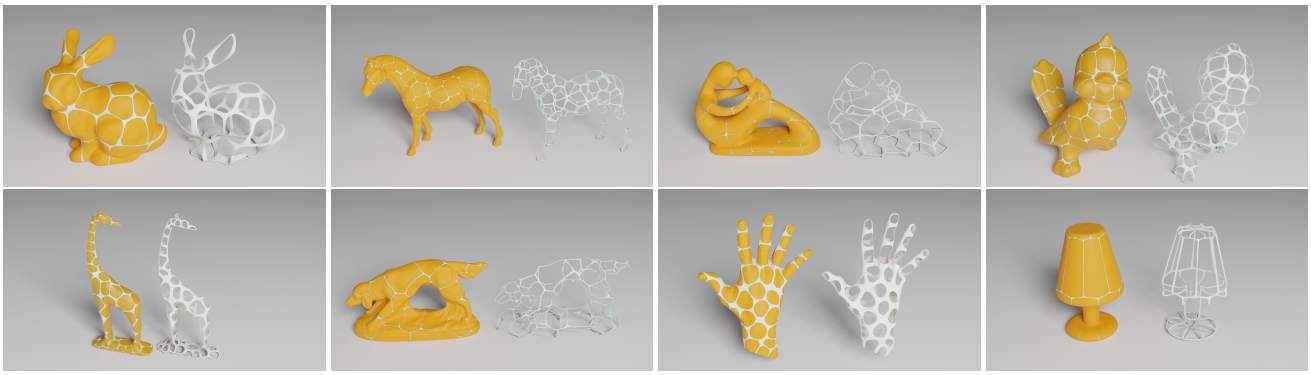


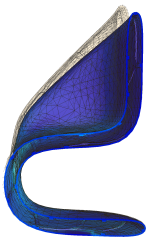
Figure 16: *Typical shape decomposition results obtained using our interactive editing approach. The shells are shown in yellow and the skeletal structure in white.*

#(verts/tris)**K**	(2.5/5)	(10/20)	(41/82)	(164/328)
Active area	50	51	78	148
Full	33	58	158	701

Table 1: *Comparison of seed insertion using the reduced active area and the full layered fields growth. The cost is evaluated at different subdivision levels of the icosahedron. Vertex and triangle counts are in thousands and timings are in ms.*

Stability analysis The impact of the staircase profile, proposed in subsection 6.2, on the stability of the skeletal structure is analyzed using the finite element method.

As a practical use case, we tested on a chair model (a cross section is shown in the inset). We compare a solid chair version without decomposition but at different skin thickness values, to the decomposed model using a fixed thickness for shells (1mm) and different number of step for the skeleton. Although the skeleton carries the major load, the shells are still important for force distribution. This is why we simulated the skeleton combined with the shells (as can be seen in the inset, Please zoom-in on the electronic version).



We fix the base of the chair and apply a load of 10 Newton in the middle of the sitting area. The simulation results summarized in Table 2 show the displacement induced by the different configurations. In all simulation, we used the 10-nodes tetrahedron and the material properties of printed nylon substrate MJF PA12.

It can be observed that the solid configuration at a skin thickness of 1.122mm and the decomposed two step version (with shell thickness at 1mm) have similar volumes, but the stepped version has a much lower displacement. When using a three step-skeleton, the displacement is comparable to doubling the skin thickness of the solid configuration. This suggests that channeling stability to the skeleton effectively helps save material use and subsequently drives the printing costs down. For object like the chairs discussed above, the skeleton structure by itself is esthetically appealing and

Config.	solid			2 steps	3 steps
Thick. (mm)	1	1.122	2	1	1
Vol. (cm^3)	56	63	112	63	75
Displ. (mm)	1.535	1.197	.422	.873	.523

Table 2: *Step mesh displacement*

can fulfill the same function even without the shells. We studied the stability of two chair configurations. A 2-stepped version with 1mm rise per step and a 3-stepped version with similar rise. Figure 17 compares the displacement of both configuration when subjected to the same load.

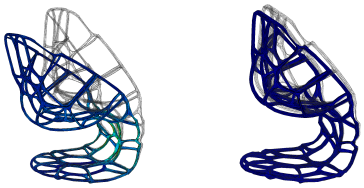


Figure 17: *Simulation of the skeleton structure of a 2 and 3-stepped chair models with 1mm rise per step (left to right resp.).*

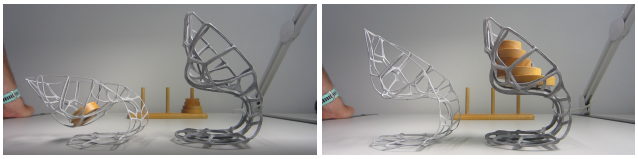


Figure 18: *The effect of a sitting load on the printed skeleton structure of a chair model with two steps at .6mm rise per step (left), and three steps at 1mm rise per step (right).*

The results suggest that the number of steps and thickness can be used in practice to achieve different effects, as for instance bounciness or rigidity. As a proof of concept, we printed two chair config-

urations, the first has two steps at .6mm rise per step, which is the limit wall thickness of the current MJF printer, and the second has three steps at 1mm rise per step. Figure 18 shows the actual prints performing under different load conditions.

Printing and assembly Our approach is generic and could be used virtually used for printing any given size. As a proof of concept (and in the interest of budget), we printed a 500*mm* high version of the Bimba model using MJF. This size surpasses the build volume of the current printer. We designed the partition of the model using our editing primitives. Special attention was paid to keep certain important features intact, such as the nose, the mouth, the hair braids, see Figure 1-right. The whole model was decomposed into a two stepped skeleton and shells of 1 mm thickness. All together, we have large 3D puzzle made of 88 shells 134 skeletal parts, see Figure 19-top-right. To speed up the skeleton assembly we put the skeletal part into bins (1-10, 11-20, ...) based on the shells they refer to. For assembly, we used screw connectors to allow for easy assembly and disassembly.

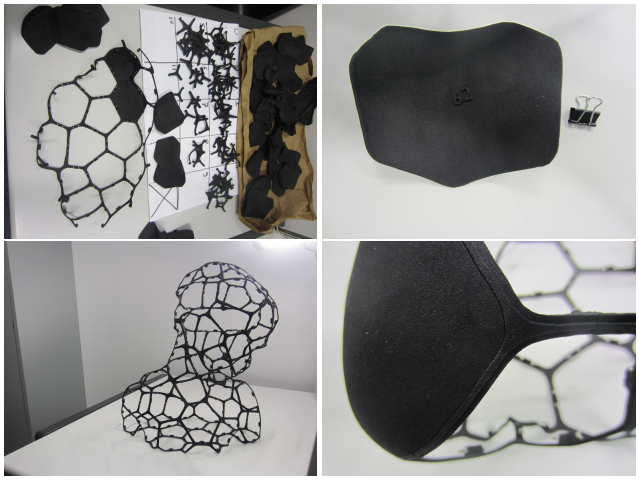


Figure 19: *The assembly of the Bimba model(Figure 1-right) starts by building the skeleton (top-right). The shells can be used to verify the quality of the build. The identification number of each shell is printed on its inner surface (top-right), the actual size can be inferred from the binder clip. The final skeleton (bottom-left) acts as a support for the shells. It can be seen that our approach allows for detailed and precise alignment of the shells within their inlay within the skeleton (bottom-right).*

A partially merged skeleton and shells is shown in Figure 1. Despite the large number of parts, our results demonstrate that a precise alignment of the shells with the skeleton can be achieved on the printed parts and that the method faithfully reproduces the original geometry.

Limitations. Although our approach is suitable for virtually all types of models, some attributes such as watertightness are not guaranteed. Post processing operations such as applying an additional epoxy layer on top might be a possible solution. For models which need to fulfill additional structural stability requirements, thorough FEM analysis and a careful choice of connectors might be necessary. Direct editing of models with highly complex topology might be cumbersome as the user does not have easy access to all locations and the assembly of such models might be challenging.

10. Conclusion

This paper presented a modular approach for creating and editing Voronoi-like cellular structures on surfaces. We focused on interactive modeling aspects and attempted to prioritize modeling over optimization. In this sense, our approach fits into the classical CAD workflow, allowing the user to freely design the partitions and interact with them in a meaningful manner which would be difficult to achieve otherwise. At the heart of our approach is a set of editing primitives which capitalizes on the notion of layered fields and take full advantage of modern graphics hardware.

As printing on high end additive manufacturing platforms is generally expensive, we proposed a model decomposition into a cellular skeletal structure and inlaid shells. In this way, material saving is channeled to the shells and stability to the skeleton. Scalability is achieved by further partitioning the skeleton and model assembly is guided by a spectral based labeling which respect the overall model geometry.

11. Acknowledgments

We thank the sculptor Julian Voss-Andreae for gracefully allowing the use of the “Poly" sculpture Photo in this work (licensed CC BY-SA 4.0 International). We extend our thanks to the anonymous reviewers for their helpful remarks. We also thank Magdalena Polec at the MPII for her assistance with printing services.

Appendix A: Mesh bandwidth and wavefront

For a rectangular sparse matrix $\mathbf{A}$, the bandwidth of a row k, denoted by β_{r_k}, is the maximum width between its nonzeros entries (in terms of column indices). The *row bandwidth* of the matrix $\mathbf{A}$ is then defined as

$$\beta_r = \max_k \{\beta_{r_k}\}. \tag{2}$$

We can define the *column bandwidth* β_c similarly. Then, the bandwidth of a matrix is the maximum of its row and column bandwidth. When the matrix is symmetric, both are equal and we refer to them indifferently as the matrix bandwidth. With respect to the mesh matrix, the row bandwidth defines the vertex bandwidth which we denote β_v. Similarly, the face bandwidth β_f corresponds the column bandwidth. For a symmetric matrix, a column j is said to be *active* in row i if $j \geq i$, and there is a nonzero entry in that column in any row with index $k \leq i$. Let c_i denote the number of active columns in row i. For a general rectangular matrix, this defined as the number of columns that have nonzero entries crossing row i. The matrix row wavefront ω_r is the maximum over all c_i. Similarly, we can define the matrix column wavefront ω_c. With respect to the mesh matrix, these correspond to the vertex wavefront ω_v and face wavefront ω_f. Since a column, or a row cannot be active more than its width, it follows that $\omega_v \leq \beta_v$ and $\omega_v \leq \beta_v$.

Appendix B: Supplementary Materials

Video

See the supplementary materials in the online version

References

[Att15] ATTENE M.: Shapes in a box: Disassembling 3d objects for efficient packing and fabrication. *Comput. Graph. Forum 34*, 8 (Dec. 2015), 64–76. 3

[AVDI03] ALLIEZ P., VERDIÈRE E. C. D., DEVILLERS O., ISENBURG M.: Isotropic surface remeshing. In *Proc. of the Shape Modeling International 2003* (2003), SMI '03, IEEE, pp. 49–58. 2

[BBJP12] BÄCHER M., BICKEL B., JAMES D. L., PFISTER H.: Fabricating articulated characters from skinned meshes. *ACM Trans. Graph. 31*, 4 (July 2012), 47:1–47:9. 3

[BS03] BENDSØE M., SIGMUND O.: *Topology Optimization - Theory, Methods, and Applications*. Springer Verlag, Germany, 2003. 2

[BWWM10] BOWERS J., WANG R., WEI L.-Y., MALETZ D.: Parallel poisson disk sampling with spectrum analysis on surfaces. *ACM Trans. Graph. 29*, 6 (Dec. 2010), 166:1–166:10. 2

[CCA*12] CALÌ J., CALIAN D. A., AMATI C., KLEINBERGER R., STEED A., KAUTZ J., WEYRICH T.: 3d-printing of non-assembly, articulated models. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 130:1–130:8. 3

[CCS12] CORSINI M., CIGNONI P., SCOPIGNO R.: Efficient and flexible sampling with blue noise properties of triangular meshes. *IEEE Transaction on Visualization and Computer Graphics 18*, 6 (2012), 914–924. 2

[CLF*18] CHEN X., LI H., FU C.-W., ZHANG H., COHEN-OR D., CHEN B.: 3d fabrication with universal building blocks and pyramidal shells. *ACM Trans. Graph. 37*, 6 (Dec. 2018), 189:1–189:15. 3

[CWW13] CRANE K., WEISCHEDEL C., WARDETZKY M.: Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*, 5 (Oct. 2013), 152:1–152:11. 2

[DE06] DU Q., EMELIANENKO M.: Acceleration schemes for computing centroidal voronoi tessellations. *Num. Lin. Alg. with App. 13*, 2-3 (2006), 173–192. 2

[DFM*17] DAPOGNY C., FAURE A., MICHAILIDIS G., ALLAIRE G., COUVELAS A., ESTEVEZ R.: Geometric constraints for shape and topology optimization in architectural design. *Computational Mechanics 59*, 6 (Jun 2017), 933–965. 2

[ENO07] EGEBLAD J., NIELSEN B. K., ODGAARD A.: Fast neighborhood search for two- and three-dimensional nesting problems. *European Journal of Operational Research 183*, 3 (2007), 1249 – 1266. 9

[EW15] EDELKAMP S., WICHERN P.: Packing irregular-shaped objects for 3d printing. In *KI 2015: Advances in Artificial Intelligence* (Cham, 2015), Springer International Publishing, pp. 45–58. 9

[HHA17] HERHOLZ P., HAASE F., ALEXA M.: Diffusion Diagrams: Voronoi Cells and Centroids from Diffusion. *Computer Graphics Forum* (2017). 2

[HMA15] HERHOLZ P., MATUSIK W., ALEXA M.: Approximating free-form geometry with height fields for manufacturing. *Comput. Graph. Forum 34*, 2 (May 2015), 239–251. 3

[HP18] HP: Hp jet fusion. online, 2018. 6

[IL05] ISENBURG M., LINDSTROM P.: Streaming meshes. In *16th IEEE Visualization Conference (VIS 2005)* (2005), p. 30. 8

[ILGS03] ISENBURG M., LINDSTROM P., GUMHOLD S., SNOEYINK J.: Large mesh simplification using processing sequences. In *IEEE Visualization, 2003. VIS 2003.* (2003), pp. 465–472. 9

[IMO84] IRI M., MUROTA K., OHYA T.: *A fast Voronoi-diagram algorithm with applications to geographical optimization problems*. Springer, 1984, pp. 273–288. 2

[KP97] KUMFERT G., POTHEN A.: *Two Improved Algorithms for Envelope and Wavefront Reduction*. Tech. rep., Norfolk, VA, USA, 1997. 9

[KS98] KIMMEL R., SETHIAN J. A.: Computing geodesic paths on manifolds. In *Proc. Natl. Acad. Sci. USA* (1998), pp. 8431–8435. 2

[LBRM12] LUO L., BARAN I., RUSINKIEWICZ S., MATUSIK W.: Chopper: Partitioning models into 3d-printable parts. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 129:1–129:9. 3

[LEM*17] LIVESU M., ELLERO S., MARTÍNEZ J., LEFEBVRE S., ATTENE M.: From 3d models to 3d prints: An overview of the processing pipeline. *Comput. Graph. Forum 36*, 2 (May 2017), 537–564. 3

[Llo82] LLOYD S.: Least squares quantization in pcm. *IEEE Transactions on Information Theory 28*, 2 (Mar 1982), 129–137. 2

[LLW12] LU L., LÉVY B., WANG W.: Centroidal voronoi tessellation of line segments and graphs. *Comput. Graph. Forum 31*, 2pt4 (May 2012), 775–784. 2, 6

[lul17] LULZBOT: 3d printing enables captivating metal sculptures. online, 2017. 8

[LWL*09] LIU Y., WANG W., LÉVY B., SUN F., YAN D.-M., LU L., YANG C.: On centroidal voronoi tessellation - energy smoothness and fast computation. *ACM Trans. Graph. 28*, 4 (Sept. 2009), 101:1–101:17. 2

[MHSL18] MARTÍNEZ J., HORNUS S., SONG H., LEFEBVRE S.: Polyhedral voronoi diagrams for additive manufacturing. *ACM Trans. Graph. 37*, 4 (July 2018), 129:1–129:15. 3

[MLS*18] MUNTONI A., LIVESU M., SCATENI R., SHEFFER A., PANOZZO D.: Axis-aligned height-field block decomposition of 3d shapes. *ACM Transactions on Graphics 37*, 5 (2018). 3

[MPR19] MAROT C., PELLERIN J., REMACLE J.-F.: One machine, one minute, three billion tetrahedra. *International Journal for Numerical Methods in Engineering 117*, 9 (2019), 967–990. 2

[NKI*18] NGO T. D., KASHANI A., IMBALZANO G., NGUYEN K. T., HUI D.: Additive manufacturing (3d printing): A review of materials, methods, applications and challenges. *Composites Part B: Engineering 143* (2018), 172 – 196. 3

[OBS92] OKABE A., BOOTS B., SUGIHARA K.: *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Inc., New York, NY, USA, 1992. 1

[O'R85] O'ROURKE J.: Finding minimal enclosing boxes. *Intern. Journal of Computer & Information Sciences 14*, 3 (Jun 1985), 183–199. 4

[PC06] PEYRÉ G., COHEN L. D.: Geodesic remeshing using front propagation. *Intern. Journal of Computer Vision 69*, 1 (2006), 145. 2

[RJSG11] RONG G., JIN M., SHUAI L., GUO X.: Centroidal voronoi tessellation in universal covering space of manifold surfaces. *Comput. Aided Geom. Des. 28*, 8 (Nov. 2011), 475–496. 2

[RLW*11] RONG G., LIU Y., WANG W., YIN X., GU X., GUO X.: GPU-assisted computation of centroidal Voronoi tessellation. *IEEE Trans. Visualization and Computer Graphics 17*, 3 (2011), 345–356. 2

[SC18] SHARP N., CRANE K.: Variational surface cutting. *ACM Trans. Graph. 37*, 4 (2018). 3

[SDW*16] SONG P., DENG B., WANG Z., DONG Z., LI W., FU C.-W., LIU L.: Cofifab: Coarse-to-fine fabrication of large 3d objects. *ACM Trans. Graph. 35*, 4 (July 2016), 45:1–45:11. 3

[Set96] SETHIAN J. A.: A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences 93*, 4 (1996), 1591–1595. 2

[SGJ13] SHUAI L., GUO X., JIN M.: Gpu-based computation of discrete periodic centroidal voronoi tessellation in hyperbolic space. *Comput. Aided Des. 45*, 2 (Feb. 2013), 463–472. 2

[STG16] SCHUMACHER C., THOMASZEWSKI B., GROSS M.: Stenciling: Designing structurally-sound surfaces with decorative patterns. *Computer Graphics Forum 35*, 5 (2016), 101–110. 3

[Stu99] STUBEN K.: *Algebraic multigrid (AMG): an introduction with applications*. GMD Report 70, Sankt Augustin, Germany, 1999. 9

[Tur91] TURK G.: Generating textures on arbitrary surfaces using reaction-diffusion. In *Proc. SIGGRAPH* (1991), ACM, pp. 289–298. 2

[VGB*14] VANEK J., GALICIA J. A. G., BENES B., MŹCH R., CARR N., STAVA O., MILLER G. S.: Packmerger: A 3d print volume optimizer. *Comput. Graph. Forum 33*, 6 (Sept. 2014), 322–332. 3

[WYL*15] WANG X., YING X., LIU Y.-J., XIN S.-Q., WANG W., GU X., MUELLER-WITTIG W., HE Y.: Intrinsic computation of centroidal voronoi tessellation (cvt) on meshes. *Computer-Aided Design 58* (2015), 51 – 61. Solid and Physical Modeling 2014. 2

[XLC*16] XIN S.-Q., LÉVY B., CHEN Z., CHU L., YU Y., TU C., WANG W.: Centroidal power diagrams with capacity constraints: Computation, applications, and extension. *ACM Trans. Graph. 35*, 6 (Nov. 2016), 244:1–244:12. 2

[YBYS08] YOSHIZAWA S., BELYAEV A., YOKOTA H., SEIDEL H.-P.: Fast, robust, and faithful methods for detecting crest lines on meshes. *Computer Aided Geometric Design 25*, 8 (2008), 545 – 560. 6

[YCL*15] YAO M., CHEN Z., LUO L., WANG R., WANG H.: Level-set-based partitioning and packing optimization of a printable model. *ACM Trans. Graph. 34*, 6 (Oct. 2015), 214:1–214:11. 3

[YLL*09] YAN D.-M., LÉVY B., LIU Y., SUN F., WANG W.: Isotropic remeshing with fast and exact computation of restricted voronoi diagram. In *Proceedings of the Symposium on Geometry Processing* (Goslar, DEU, 2009), SGP'09, Eurographics Association, pp. 1445–1454. 2

[ZFD*17] ZHANG X., FANG G., DAI C., VERLINDEN J., WU J., WHITING E., WANG C. C.: Thermal-comfort design of personalized casts. In *Proc. UIST* (2017), ACM, p. 243–254. 2

[ZMSS18] ZAYER R., MLAKAR D., STEINBERGER M., SEIDEL H.-P.: Layered fields for natural tessellations on surfaces. *ACM Trans. Graph., (Proc. SIGGRAPH Asia) 37*, 6 (nov 2018), 264:1–264:15. 2, 3, 4

[ZSS17] ZAYER R., STEINBERGER M., SEIDEL H.-P.: A gpu-adapted structure for unstructured grids. *Computer Graphics Forum 36*, 2 (2017), 495–507. 4