EUROGRAPHICS 2016 / J. Jorge and M. Lin
(Guest Editors)

Volume 35 (2016), Number 2

Improved Surface Quality in 3D Printing by Optimizing the Printing Direction

W. Wang[†1,2] and C. Zanni[1] and L. Kobbelt[1]

[1]RWTH Aachen University, Germany
[2]Dalian University of Technology, China

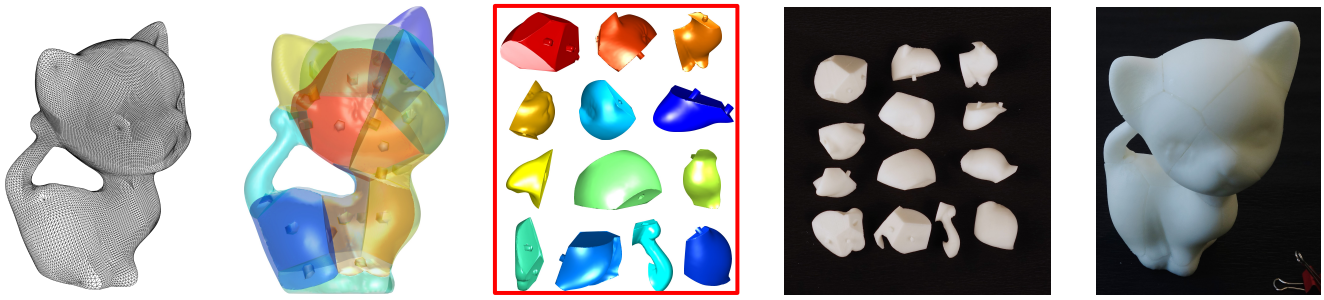

Figure 1: *For a given polygon model, we compute a decomposition into parts which is optimized such that the outer normals vectors can be aligned mostly perpendicularly to the printing direction. We leverage the observation that "vertical" surfaces can be printed without staircase artifacts and without support structures. Along with the part decomposition, we also compute an assembly order for the pieces and add connectors to stick them together.*

Abstract

We present a pipeline of algorithms that decomposes a given polygon model into parts such that each part can be 3D printed with high (outer) surface quality. For this we exploit the fact that most 3D printing technologies have an anisotropic resolution and hence the surface smoothness varies significantly with the orientation of the surface. Our pipeline starts by segmenting the input surface into patches such that their normals can be aligned perpendicularly to the printing direction. A 3D Voronoi diagram is computed such that the intersections of the Voronoi cells with the surface approximate these surface patches. The intersections of the Voronoi cells with the input model's volume then provide an initial decomposition. We further present an algorithm to compute an assembly order for the parts and generate connectors between them. A post processing step further optimizes the seams between segments to improve the visual quality. We run our pipeline on a wide range of 3D models and experimentally evaluate the obtained improvements in terms of numerical, visual, and haptic quality.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—Geometric Algorithms

1. Introduction

Most of the established 3D printing technologies are based on layered manufacturing, e.g., fused deposition modeling or stereo lithography. An immediate consequence of this slice-by-slice approach is that the effective printing resolution and consequently the resulting surface smoothness is strongly anisotropic. This implies a considerable variance in (visual and haptic) surface quality depending on the spatial orientation. An effect which becomes even more pronounced when a larger layer thickness is chosen to save printing time.

For most 3D printers, the optimal surface quality is achieved if the surface normal is perpendicular to the printing direction since in this

† wwmdlut@gmail.com

case the relatively high x/y-resolution plus the physical smoothing effect emerging, e.g., from the melted plastic in fused deposition can be exploited. The more the surface normal aligns with the (positive or negative) printing direction the more staircase artifacts appear between layers. Moreover, as the surface normal tilts towards the negative printing direction ("overhang"), auxiliary support structures have to be generated so that the printing quality in surface region touching the support structures will be poor and the removal of support structures may cause additional damage to the object's surface. Furthermore, many complex models, especially the ones with many branches (see the Octopus model in Figure 8), can not be printed with high quality in one global orientation and need excessive support structures - the removal of which can damage the surface or even break some parts of the model.

In order to reduce the need for support structures and improve the surface quality in 3D printing, our goal is to decompose the input model into a set of parts such that each individual component can be oriented in a way that reduces the deviation of the surface normals from the horizontal direction (assuming that the printing direction is vertical). This decomposition, however, has to take additional constraints into account, which guarantee that the shape of the individual pieces remains reasonably compact and that there exists a practical assembly order in which the parts can be put together. We achieve this by (initially) defining the parts by the intersection of convex spatial cells with the input model's volume.

Obviously, the improvement of the surface smoothness through a decomposition approach is compromised by the appearance of visible gaps between the parts in the final assembly. These gaps cannot be avoided completely due to manufacturing tolerances. However, in practice, complex 3D shapes have to be decomposed for 3D printing anyway (to avoid excessive and hard-to-remove support structures) and we present ideas to design the interface between neighboring parts that reduce the visible width of seams. We evaluate the perceived quality improvement by collecting user feedback which confirms the effectiveness and usefulness of the proposed algorithm.

1.1. Contribution

In this paper we develop a complete pipeline of algorithms which generates, for a given input model, a set of parts such that each part can be printed in an orientation where the normals of the (outer) surface are safely separated from the (positive or negative) printing direction. By this we avoid staircase artifacts as well as support structures being attached to the (outer) surface.

The pipeline starts with a surface segmentation that is driven by the co-compatibility of the normals of the input triangles with (random) printing directions (Sect. 4). In order to simplify the shape of the individual patches, the segmentation is post-processed by finding optimal cutting planes to separate neighboring segments using support vector machines (Sect. 5). This set of cutting planes is consolidated into a consistent spatial cell structure by computing a 3D Voronoi diagram which best fits the cutting planes (Sect. 5.1).

The parts into which the input model is decomposed, result from intersecting the Voronoi cells with the input model's volume. We refine the interfaces between neighboring parts (Sect. 6) and compute a collision free assembly order (Sect. 7). The manual assembly is further supported by adding connectors to the part interfaces (Sect. 8) which indicate assembly order and direction.

The motivation and justification for the various design decisions in our pipeline emerge from going backwards through the different stages: In order to always find a proper assembly order with sufficient flexibility for connector placement, it is advisable to decompose the input object into pieces that are convex in the interior of the object. Such a decomposition emerges naturally from an intersection of the input object with the cells of a 3D Voronoi diagram. A good part segmentation results if the faces of the Voronoi cells align well with the patch boundaries coming from a normal-driven surface segmentation of the input mesh. The SVM planes are introduced in order to "translate" this patch boundary information into a form that is better compatible to the Voronoi setting.

2. Related Work

Recently, there have been a large number of publications dealing with shape decomposition and segmentation [Sha08]. However, in this paper, we are mainly focusing on the decomposition and segmentation methods specialized for manufacturing.

3D Printing 3D printing has been an active topic in Compute Graphics and a large number of methods has been proposed to fabricate object with various appearances [VWRKM13, LDPT13], mechanical toy and automata [ZXS*12, CLM*13], and articulated models [CCA*12, BBJP12]. Some methods have been proposed to analyze the structure [SVB*12, ZPZ13, XXY*15], reduce material cost [WWY*13, LSZ*14], reduce time consumption [WCT*15], design and reduce support structures [SU14, DHL14, VGB14a, HLZCO14], print huge model [HFW11, LBRM12, ACP*14, SFLF15], 3D model decomposition and packing [VGB*14b, CZL*15, YCL*15, Att15], printing direction optimization [ZLP*15], and optimize self-balance: static [PWLSH13, YSO14] and dynamic [BWBSH14, PTM*15]. And some other algorithms are presented to deal with the color of the printing [RCM*14, HL14]. In contrast, our work is proposed to reduce the staircase-error caused by layer based manufacturing and the impact of the support structures on the printed object.

Volume Decomposition for manufacturing Recently, a set of methods has been proposed to decompose a 3D model into several pieces for manufacturing, such as 3D printing. Since the printing volume of the printers is limited, they can not print models whose dimensions are larger than the dimensions of the printers. [HFW11, LBRM12, SFLF15] propose algorithms to partition the models into small pieces so that the volume of each piece is smaller than the volume of the printer. For most models, supporting material is needed which is expensive and hard to remove in a clean way. [HLZCO14] decompose the 3D models into approximate pyramidal shapes which can be printed with very little support structures. [VGB*14b, CZL*15, YCL*15, Att15] present algorithms to decompose a 3D model into small parts which are packed into the volume of the printer. These allow to reduce both the printing time and the supporting material required. [HMA15] proposed a method to segment a 3D model into several height fields patches. These patches are then manufactured with classical manufacturing methods, such as 3-axis milling. Finally, the manufactured parts are assembled to be used as a mold for copying the object. Although their method use face normals to segment the model, the printing quality of the object is not considered in their method and the patches obtained cannot be printed directly with a 3D printer. Furthermore, in order to assemble the manufactured parts, the patches should be deformed which changes the geometry of the initial model.

The methods mentioned above do not deal with the problem of

printing quality. Most closely related to our work is [HBA13]. They calculate three orthogonal directions for a model by analyzing the normal of the faces. Then, the model is voxelized into voxels which are clustered into boxes. The printing direction of each box is selected from these three global directions. Constrained by the printing directions, their method limit the gain in printing quality for a large number of models (the model in Figure 10 needs five directions). Furthermore, while some models can be printed with high quality in three directions, these three directions are not necessary orthogonal. In addition, they do not calculate the assembling order nor design the connectors which will make the assembling of parts more difficult for users. In this paper, we propose a printing direction based segmentation algorithm to partition the 3D models into several pieces, each of them having its own unconstrained optimal printing direction.

Connector Design Connectors between adjacent parts are used to ensure the object does not fall apart. They also guarantee that the boundaries of adjacent parts match well during assembling. While [HFW11, LBRM12] design the connectors between adjacent parts, they only handle the case of a single contact plane between the part to assemble and the already assembled ones. Therefore, they do not need to consider the direction of the connectors. On the contrary, in our work, when designing the connectors, more than one contact surface should be considered in most situations. Therefore, their method can not be used directly in our paper. To solve this problem, we propose an algorithm to optimize the directions of the connectors which is also the assembling directions of the parts. Furthermore, their method does not consider the assembly order which is very important to guide the users.

Assembling Sequences The assembling sequences of the parts have been investigated for 3D puzzles [LFL09, XLF*11, SFCO12, SFLF15], 3D assembly instructions [APH*03], design with planar interlocking parts [HBA12, SP13, CPMS14, DPW*14], and interconnected mechanical parts [MYY*10]. Some of them design the parts so that they can interlock each other. Others take into account stability or equilibrium constraints to simplify the assembly process. In contrast, our method uses assembly directions of the parts to guide the generation of the assembling sequence.

Printing Direction Optimization Printing direction is very important in 3D printing. It affects the printing time, amount of supporting material, printing quality, shrinkage, curling, distortion, resin flow, material cost and trapped volume etc [PRD07]. In [FF95, XWL*97, AAD98], they optimize the printing direction by taking into consideration building time, accuracy and stability of the part. Surface finish and roughness, lower cost, and volumetric error are considered in [MRI00, TPR04]. However, all of these methods just optimize a single global direction for the entire model, so the improvement of the printing quality by these methods is limited. Moreover,some models have thin branches with changing orientation (such as Octopus and Tree). Such models require additional support structures to hold these thin branches during printing and their removal may damage the surface quality or break some thin branches. In this paper, we partition the 3D input models into several parts and the printing direction of each part is computed separately, support structures tend to be added on the inner cross-sections and therefore do not impair the outer surface quality.

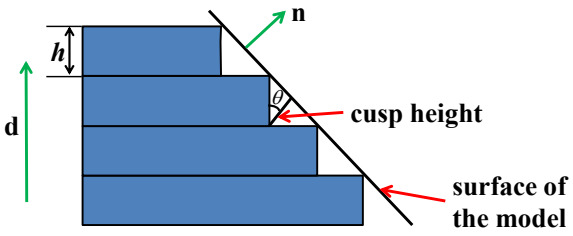

Figure 2: *Cusps in the surface of a 3D printed object with layer thickness h and surface normal* ***n****. The height of the cusps determines the visual as well as the haptic surface quality.*

3. Surface Quality in 3D Printing

The local surface roughness or "cusp height" which results from the layered fabrication of 3D printed objects can be measured by

$$e_\theta = |\cos(\theta)| * h = |\mathbf{n}^T \mathbf{d}| * h$$

where h is the layer thickness and θ is the angle between the surface normal $\mathbf{n}$ and the printing direction $\mathbf{d}$ (see Fig. 2).

To guarantee acceptable surface quality, we require that for each triangle the angle between surface normal $\mathbf{n}$ and printing direction $\mathbf{d}$ lies in some interval $\underline{\theta} < \theta < \bar{\theta}$ with $\underline{\theta} < 90$ large enough so that the cusp height remains small and $\bar{\theta} > 90$ small enough such that no support structures are necessary. Some (empirical) visual evaluation of beams printed in different angles with FDM printers [Dim14, Ult15], motivate a default choice of $\underline{\theta} = 60$ and $\bar{\theta} = 120$ which appears to be feasible for most of today's 3D printers.

Our global surface quality (roughness) measure which is similar to the one used in [AAD98] is defined by integrating the cusp height over the entire surface. For each triangle f we obtain

$$E(f) = e_\theta * \text{area}(f)$$

and consequently for the mesh M:

$$E(M) = \frac{\sum_{f \in M} E(f)}{\sum_{f \in M} \text{area}(f)} \quad (1)$$

where the normalization with the total surface area makes the quality measure scale independent.

We optimize the surface quality by minimizing E_M. The parameters for the minimization emerge from a partitioning of the surface into a number of patches which allows us to chose a different printing direction $\mathbf{d}$ for each part. One trivial solution is to assign each triangle of the input mesh to its own segment. Another solution is to use pairs of adjacent triangles as elementary segments and assign to them the cross product of their respective normal vectors (i.e. the direction of their common edge) as the printing direction. In both cases E_M vanishes but we have to deal with an excessive number of parts. Our goal is to find a moderate number of parts while keeping E_M small.

4. Surface Segmentation

We start by generating a large number of randomly distributed candidate printing directions $\mathbf{d}_i$ (see Fig. 3(b)). For each direction $\mathbf{d}_i$ we collect all triangles $f \in M$ for which the angle θ lies in the prescribed interval $[\underline{\theta}, \bar{\theta}]$. The sum of the areas of these triangles is used as a "coverage score" $S(\mathbf{d}_i)$. We pick a set of initial candidate directions

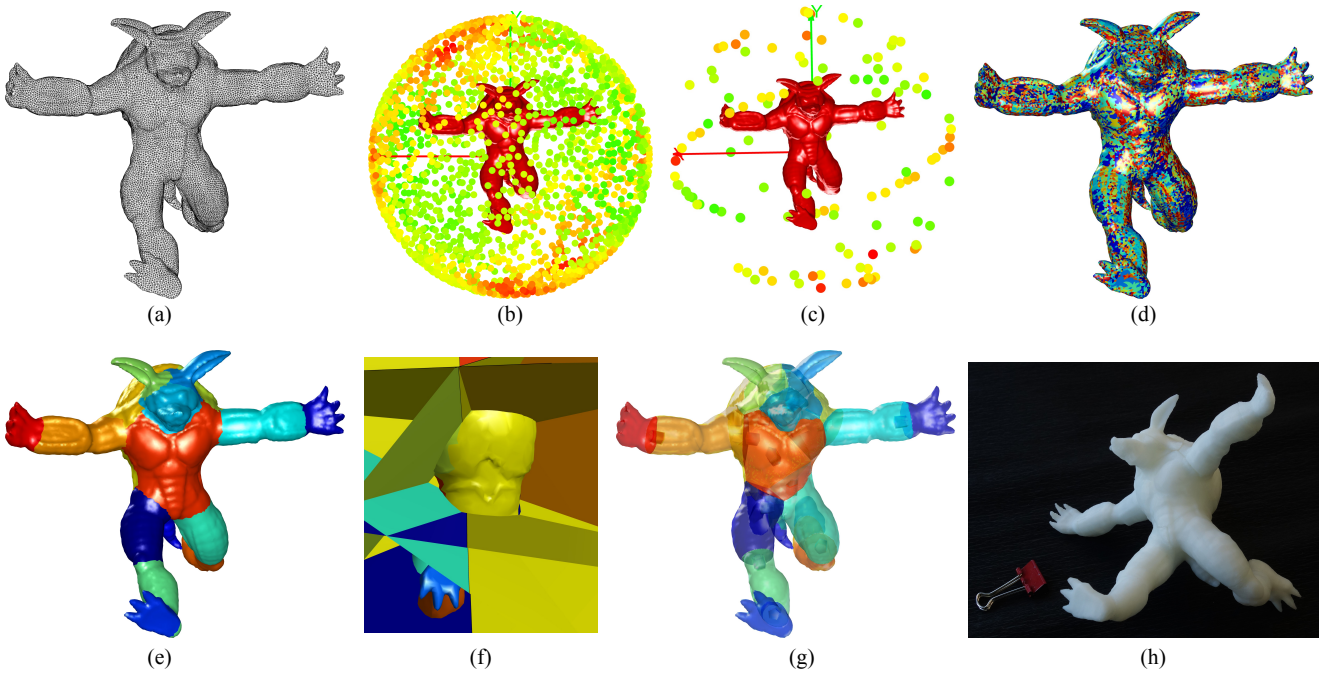

Figure 3: *Illustration of our pipeline. For the input model (a) we generate a large number of candidate printing directions (b). From these we select some major candidate printing directions so that each triangle is covered (c). An initial (over-) segmentation (d) is coarsified by a global clustering scheme (e). The cutting planes of the patches are fitted by SVM (f). Then a volumetric cell complex is computed such that the intersections of the cell with the surface approximate the surface patches. For the resulting parts, we compute an assembly order and, based on the best assembly directions, we add connectors to the inner facets of the parts (g). Figure (h) shows a photo of the physical print.*

D greedily in the order of descending coverage score until each triangle is covered at least once (see Fig. 3(c)).

Computing the minimum set of directions $\mathbf{d}_i$ that together cover the entire surface (which would be an instance of the vertex cover problem) is not sufficient for our purposes. Indeed, in order to facilitate the printing of parts and eventually the manual assembly, we need to promote the compactness of the resulting surface patches.

For efficiency reasons, the generation of the surface patches is performed in two steps. We first compute a set of "pre-patches" by clustering all triangles which are covered by the same set of directions. Then, we set up an affinity matrix for all pairs of neighboring pre-patches and compute the final patches by a more sophisticated global clustering scheme.

Let B be a binary matrix where each row corresponds to a triangle in M and each column corresponds to one of the selected directions. An entry b_{ij} of B is 1 if the triangle f_i is covered by $\mathbf{d}_j$ and 0 otherwise. We generate the pre-patches C_k by merging all triangles with identical rows in B. If a pre-patch happens to fall into several connected components on the surface, we treat each component as a separate pre-patch (see Fig. 3(d)).

For the set of pre-patches we define an affinity matrix A where the affinity between the pre-patches C_i and C_j is given by:

$$A_{ij} = \begin{cases} \lambda * w(C_i,C_j) + b(C_i,C_j) & \text{if} C_i \cap C_j \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Here, $C_i \cap C_j \neq 0$ means that C_i and C_j are adjacent (i.e., have a common boundary) and

$$w(C_i,C_j) = h - min_{\mathbf{d}\in\mathbf{D}} E_{C_i \cup C_j}$$

and

$$b(C_i,C_j) = \frac{max(\text{diag}(C_i),\text{diag}(C_j))}{\text{diag}(C_i \cup C_j)},$$

measure the orientability and compactness of $C_i \cup C_j$, respectively. $\text{diag}(C)$ denotes the diagonal of the bounding box of C.

The rationale for this definition is that the merging of pre-patches into patches should find a good balance between obtaining patches with a high surface quality (small E_C) and promoting the compactness of patches. Compact patches will lead to part decompositions that are easier to assemble.

We apply the Ncut algorithm [SM00] to extract the final patches based on the affinity matrix A (see Fig. 3(e)). By allowing for non-zero affinity weights only between adjacent pre-patches, we make sure that the resulting patches are simply connected since the graph encoded by A has the same manifold topology as the input surface. The tricky part when using Ncut is to determine a good estimate for the number of clusters. We apply self-tuning spectral clustering [ZMP04] to derive a default value and allow the user to adjust it as desired.

5. Initial Part Generation

The boundary curves between patches resulting from the previous stage will, most of the time, be non-planar. Such non-planar boundaries cannot be used directly to define a decomposition into convex

parts, therefore we compute a cutting plane $P_{ij} = (\mathbf{n}_{P_{ij}}, d_{P_{ij}})$ for each pair (i, j) of neighboring patches. To this end, we use Support Vector Machines (SVM) [SVGDB*02] which is much more robust than fitting a least squares plane to the boundary (see Fig. 3(f)). SVMs yield the maximum margin linear classifier for a given training set. We use the vertex locations of the two adjacent patches as SVM's input.

While each SVM plane P_{ij} is optimal for one single boundary between two patches, the collection of all these independently computed cutting planes does not define a consistent spatial cell structure that we can use to define the object's part decomposition. Hence, we consolidate the spatial decomposition by computing a Voronoi diagram such that the planar faces of the Voronoi cells approximate the given SVM planes as well as possible.

Let $S = \{s_i\}$ be the set of centers that defines a 3D Voronoi diagram. We initialize the s_i with the centers of gravity of the corresponding patch and then iteratively update their positions such that the midplane ("Voronoi plane") between s_i and s_j approximates the SVM plane P_{ij}. More precisely, we minimize the objective functional:

$$\sum_{ij} \|V_{ij} - P_{ij}\|^2 = \sum_{ij} \|\mathbf{n}_{V_{ij}} - \mathbf{n}_{P_{ij}}\|^2 + (d_{V_{ij}} - d_{P_{ij}})^2 \qquad (2)$$

where the sum runs over all pairs of adjacent patches ij and V_{ij} denotes the Voronoi plane

$$V_{ij} = (\mathbf{n}_{V_{ij}}, d_{V_{ij}}) = \left(\frac{s_i - s_j}{\|s_i - s_j\|}, -\frac{(s_i - s_j)^T (s_i + s_j)}{2\|s_i - s_j\|} \right)$$

separating the two Voronoi centers s_i and s_j. Since we do not require the Voronoi centers to remain on or close to the input surface, the minimization of (2) is a simple unconstrained optimization problem for the s_i that we solve by using the interior-reflective Newton method [CL96]. Notice that the part topology emerging from the Voronoi diagram is not guaranteed to be identical to the initial patch layout. However this does not impact the segmentation in a critical way.

Note that directly optimizing the Voronoi cells with respect to the patch boundaries and without the SVM planes as guides would correspond to a simple least squares approach.

5.1. Regularization

The unconstrained optimization (2) can occasionally lead to parts with sharp corners as shown in Fig. 4 (middle). This is problematic since the resulting parts are quite thin and thus can easily break. This effect can be prevented by adding a regularizing term that penalizes small angles between adjacent Voronoi planes:

$$\sum_{ij} \|V_{ij} - P_{ij}\|^2 + \mu \sum_{ijk} (\mathbf{n}_{V_{ij}}{}^T \mathbf{n}_{V_{jk}})^2 \qquad (3)$$

Here the second sum runs over all triplets of patches that meet at a common corner. Note that for symmetry reasons, each triplet occurs in all its permutations.

Finally, the intersection between a Voronoi cell and the volume of the input model can result in a set of disjoint components that would cause the creation of un-desired parts (see Fig. 5 (a)). We handle these cases by keeping only the largest component (in terms of volume or surface area) and merge the other components with neighboring parts (see Fig. 5 (b) and (c)). This merge is implemented by locally recomputing the Voronoi diagram *without* the center s_i that belongs to the Voronoi cell which caused the multi component part (see the accompanying video for details).

6. Part Refinement

At this stage, we have decomposed the input model into a set of parts by intersecting the cells of a spatial Voronoi diagram with the model volume. The patch generation and the patch boundary regularization guarantee that the outer surface of the parts have a compact and non-degenerate shape and can be printed in high quality (low cusp height).

What we did not yet consider is the quality of the *inner* surfaces which are simple planar (Voronoi) facets. In general this is a reasonable choice but sometimes the Voronoi planes can intersect the surface of the model in an almost tangential direction which, again, causes thin features that can easily break after 3D printing.

We therefore modify the inner surfaces of those critical parts by computing an approximate thin plate surface for each Voronoi facet that interpolates the boundary of the unaltered facet but in addition, we apply a *clamped* boundary condition on those boundary segments belonging to the outer surface such that the thin plate surface meets the outer surface perpendicularly. To compute the thin plate surfaces, we apply bi-Laplacian smoothing [SKS00] to a moderately refined mesh (see Fig. 6).

Let v_i be the vertex positions in the original planar facet and w_i the vertex positions in the thin plate mesh for the same facet. Then we define a simple blend surface with vertex positions

$$u_i = (1 - \alpha) v_i + \alpha w_i. \qquad (4)$$

A blend coefficient $0 < \alpha < 1$ can be used to prevent potential self-intersections in the rare cases when the thin plate surface intersects the outer surface of the object.

We can further exploit the blending parameter α of (4) to generate a small artificial gap between neighboring parts, e.g., by setting $\alpha < 0.95$. Given the typical manufacturing tolerances of commodity 3D printers, the results of printing the positive and the negative side of the common interface surface between two parts does not guarantee a tight fit. The artificial gap provides some leeway between the parts but it will not lead to a lose and unsteady fit since the connectors that we add in Section 8 make sure that the assembled parts cannot shift. Notice that, since v_i and w_i coincide on the boundary of the facet, the width of the artificial gap shrinks to zero no matter which value we choose for α. Hence, the visible gaps on the surface of the assembled object will remain tight.

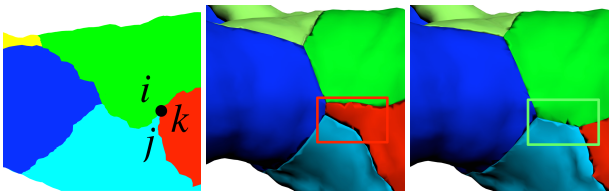

Figure 4: *A triplets of patches (i, j, k) share a common corner in the initial segmentation (left). In the middle, we can see a configuration with a sharp corner. By adding a regularizer term to the objective function (3) for the Voronoi diagram, we can effectively prevent such configurations (right).*

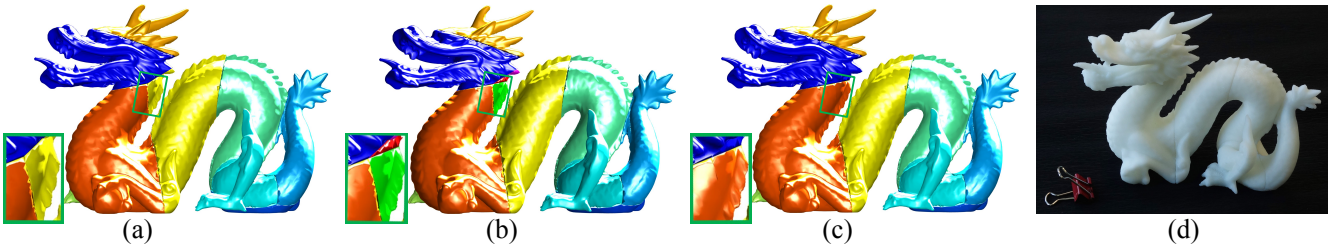

Figure 5: *The dragon model is decomposed into 9 parts. However, when intersecting the surface with the corresponding Voronoi cells, the yellow segment happens to fall into several disjoint components (a). We keep the largest of these components and locally recompute the Voronoi diagram without the center s_i belonging to the "yellow" Voronoi cell for each of the other components (b), and then merge them with their neighboring parts (c). Figure (d) shows the assembled physical print.*

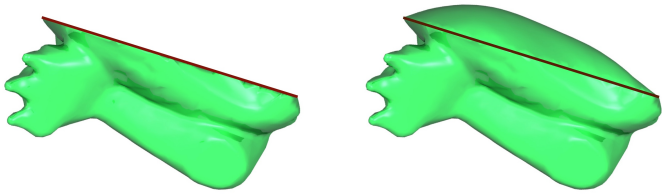

Figure 6: *If a Voronoi plane intersects the surface in near tangential orientation, we can avoid thin parts by replacing the planar inner facets with thin plate surfaces which interpolate the boundary and are clamped along the outer boundary such that they meet the outer surface perpendicularly.*

7. Assembly Order

After we have generated and refined the shape of the individual parts, we have to compute an order in which the parts can be assembled without causing collisions. Since the parts are mostly convex in the interior of the input model (where they are put together) there is a lot of flexibility regarding the assembly order. Even if the refinement of the inner facets (Sect. 6) makes some of the parts non-convex, we observe in practice that the assembly is hardly affected. In any case, we can always use the blending (4) to reduce the degree of non-convexity until the assembly works.

7.1. Assembly Cones

To manually build up the model we pick one part after the other and add it to the partial assembly of previously picked pieces. When we add a piece q, every part p that is adjacent to q and that has been added before, imposes some constraints on the *directions* in which q can be attached. If the common interface between q and p is a planar facet, then the attach-directions of q are constrained to the half-space defined by the common facet's normal vector. If the common interface is a non-planar (e.g. thin plate) surface, we conservatively estimate its cone of normals by an axis $\mathbf{n}$ and an opening angle γ. The axis $\mathbf{n}$ of q is the normalized sum of the normals of all faces on the surface and the opening angle γ of q is derived from the intersection of the half-spaces defined by the faces on the surface. If more than one neighboring part of q has been added before, the attach-direction for q has to lie in the intersection of all the assembly cones imposed by those neighbors (see Figure 7).

Let $\mathbf{n}_i$ and γ_i be the axes and opening angles of the normal cones for the previously added neighbors of q ($\gamma_i = \pi/2$ for planar interfaces). We estimate the axis of the intersection of these cones by a constrained optimization problem:

$$\begin{aligned} &\max_{\mathbf{n}^*} \textstyle\sum_i \mathbf{n}_i^T \mathbf{n}^* \\ &\text{s.t.} \\ &\|\mathbf{n}^*\|^2 = 1, \quad \mathbf{n}_i^T \mathbf{n}^* > \cos(\gamma_i - \varepsilon) \end{aligned} \tag{5}$$

where the added safety margin ε is included to avoid assembly direction that are nearly tangential to one of the interface. The opening angle γ^* of the assembly cone is then

$$\gamma^* = \min_i \gamma_i - \varepsilon - \operatorname{acos}(\mathbf{n}_i^T \mathbf{n}^*).$$

7.2. Part Ordering

We start with an arbitrary part q_0 and progressively add additional parts to the assembly. In order to determine the next part q^* to add to the assembly, we test all the neighbors of previously assembled parts in the order of increasing opening angle γ (i.e. starting from the one with the tightest assembly cone). In order to check if a part is feasible, we compute the new assembly cones of its (not yet added) neighbors as if the part was added to the assembly following the procedure described in the previous section. If this assembly cone computation results in a part's cone to vanish, then we test the next part in order of increasing opening angle, otherwise we effectively add the part to the assembly and update the assembly cones. If no part can be found that does not close any of its neighbor's assembly cones, the assembly fails and we re-run the procedure with a different starting part q_0. In all our experiments this never happened.

8. Connector Design

We add connectors between neighboring parts in order to (1) make sure the parts do not shift in the assembly and (2) help the user to

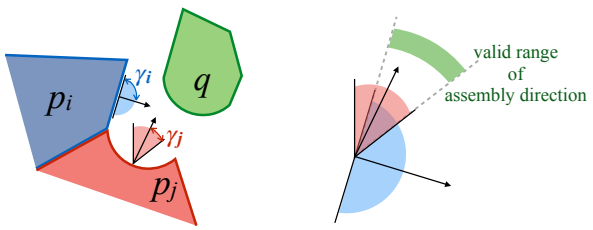

Figure 7: *Two parts p_i and p_j are already added to the assembly order. They impose constraint on the assembly direction for the part q.*

figure out the correct orientation and attach direction for the parts. Connectors are simple pentagonal prisms that we add on one side and subtract on the other side of an interface.

When a part is chosen in the assembly order computation (Sect. 7), we add a connector to each facet which is a boundary to a previously assembled part. The connector orientation is the assembly cone axis. Candidates for connector position are triangle's centers which are at the maximal distance from the interface's boundary. Among them, we choose the one that has the minimal variance of distance to boundary points. The width and height of each connector is chosen such that no collision occurs.

In the few critical cases where the assembly cone's normal is nearly tangential to the interface or if the area of the interface is too small, we simply discard the connector. This does not affect the overall robustness of the assembly.

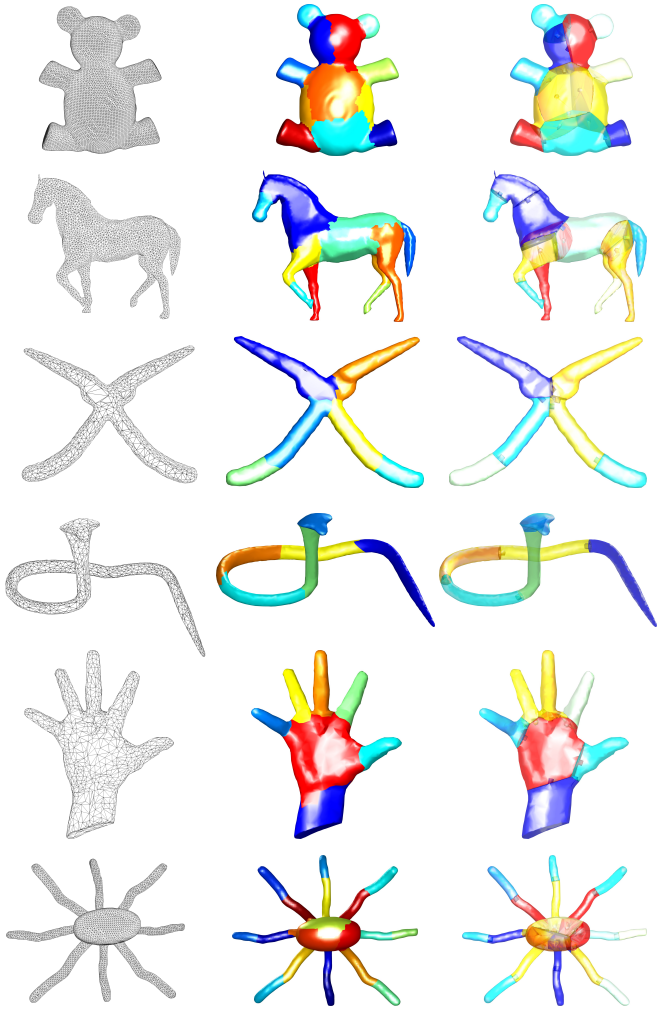

Figure 8: *Decomposition results for a number of model with different complexities. The input models are shown on the left, surface segmentations in the center and the resulting parts on the right.*

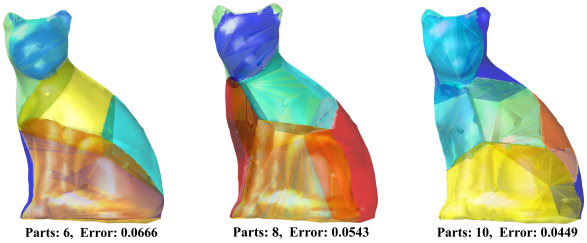

Figure 9: *Increasing the number of parts from 6 (left) to 8 (center) and to 10 (right) moderately reduces the quality measure (cusp measure) E_M from* 0.0666 *to* 0.0543 *and to* 0.0449. *Printing the model in one piece yields* $E_M^{ref} = 0.1410$.

9. Experimental Results

9.1. Implementation Details

Our algorithm was implemented in mixed C++ and Matlab, and it was run on on a PC with Intel(R) Core(TM) i5-4210M CPU @ 2.60GHz and 8GB memory. The results are fabricated by [Dim14] with printing layers of 0.33 *mm*. We used the same set of standard parameters in all our experiments. The number of samples for the printing directions is 100000. In the construction of the affinity matrix, λ is set to 1. In Equation (3), μ in is set to 1, and in Equation (5), ε is set to 10^{-6}.

Running times of our implementation are provided in columns eight to twelve of Table 1. We see that, for most models, running times are below 5 minutes which is negligible compared to the printing time of the models.

9.2. Decomposition Results

We have tested our algorithm on a set of both, simple and complex, 3D models which have been selected from the SHREC database [LGB*11]. All models where scaled such that the longest edge of its bounding-box measures 150 *mm*. Some decompositions are shown in Figure 8. We see that faces with similar optimal printing directions are collected to form patches which can be printed with small cups errors.

For complex models, especially models with many branches (see the Octopus model in Figure 8), a lot of support structures would be required. These structures increase printing time and their removal can impair the surface quality. Furthermore, it is sometimes difficult to remove them without breaking some thin parts. The results in Figure 8 demonstrate the effectiveness and robustness of our segmentation method.

In general, increasing the number of parts will decrease the printing error for a given object (see Figure 9). However, using a larger number of parts will increase the number of seams on the surface and make the assembly more difficult. While an optimal number of components may not be obtained automatically, we use *selftuning spectral clustering* [ZMP04] to obtain an initial number of parts. Since Ncuts only takes 3-4 seconds, users can easily test some numbers around the initial one.

More decomposition results are shown in the accompanying video. Photographs of printed results are shown in Figure 1, 5, and 11.

9.3. Comparisons

In order to demonstrate the efficiency of our method (named *PDBD*), we compare it with both an optimized single printing direction

Model	Mesh #Face	Parts Number	E_M (*mm*)			*RunningTime* (*s*)				
			OOD	OSAM	PDBD	Segmentation	Parts	Order	Connectors	Total
Bunny	27648	12	0.1361	0.1169	0.0912	116.74	194.32	9.31	95.69	416.06
Cat	7062	6	0.1410	0.0984	0.0666	31.52	26.50	6.03	28.08	92.13
Chair	17306	8	0.1241	0.1004	0.0143	81.55	92.37	5.66	19.34	198.92
Desk	27424	5	0.0506	0.0476	0.0170	104.12	92.86	5.59	47.53	250.40
Dragon	20000	9	0.1390	0.1297	0.1043	130.57	106.74	6.31	37.20	280.82
Dove	3996	6	0.0947	0.0773	0.0653	21.95	13.28	1.28	9.80	46.31
Glass	14028	6	0.1063	0.0969	0.0599	42.18	48.02	3.04	17.95	111.19
Hand	3996	7	0.1321	0.1108	0.0674	24.71	16.75	2.35	14.42	58.23
Horse	11072	11	0.1218	0.1050	0.0789	42.22	71.34	3.70	33.08	150.34
Kitten	55502	13	0.1503	0.1196	0.1036	234.74	229.59	10.32	102.80	577.45
Man	3996	9	0.1588	0.1080	0.0695	22.87	19.15	1.67	13.48	57.17
Armadillo	30418	18	0.1519	0.1349	0.088	114.39	148.62	8.99	87.79	359.79
Pig	16818	10	0.1165	0.0859	0.0677	53.60	97.48	6.19	60.73	218.00
Plane	14936	5	0.0705	0.0659	0.0347	34.97	43.67	1.91	18.26	98.81
Pliers	3996	6	0.1146	0.0659	0.0429	21.41	12.69	0.85	11.93	46.88
Snake	3996	6	0.1391	0.1011	0.0625	23.26	13.54	1.05	5.61	43.46
Spider	14498	20	0.1332	0.1222	0.0643	62.24	174.14	5.26	48.09	289.73

Table 1: *Quantitative evaluation of the surface quality obtained with OOD, OSAM, and PDBD (ours) and the running times of our algorithm. Columns two and three show the number of vertex and faces of the models. While OOD always prints the model in one piece. OSAM and PDBD have been tuned to generate the same number of parts (column four). Columns five to seven show the E_M values for the respective model and method. Last five columns show the running time of each step of our system and the total running time where the unit is "second".*

[TPR04] (named *OOD*) and the decomposition of [HBA13] (named *OSAM*). In the remainder of the paper, we will call printing error the roughness measure E from equation (1).

First, the bench model of Figure 10 demonstrates the problem of *OOD* and *OSAM*. For *OOD*, with a single printing direction (in this case the y-axis in the left of Figure 10) large area is affected by the maximal printing error (resulting in a global printing error of 0.0988). The constraints on the segmentation and printing direction enforced by *OSAM* limits the error reduction which remains close to the one of OOD (error of 0.0914 instead of 0.0988). On the contrary, our method (*PDBD*) leads to a more natural decomposition where each part can be printed with its own optimal printing direction resulting in a much lower error of 0.0244.

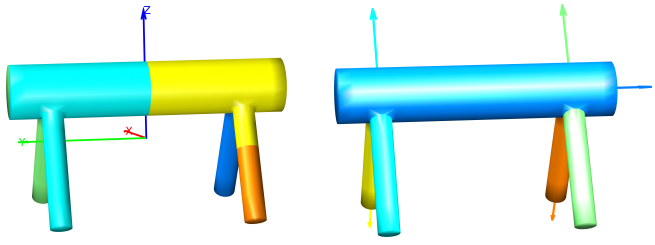

Figure 10: *Comparison of our algorithm (PDBD) (right) with OSAM (left) and OOD (the y-axis of the left). Since OOD only computes one global printing directions it cannot properly handle the legs of the bench. OSAM computes a decomposition but is restricted to a set of three perpendicular printing directions. Our algorithm finds the optimal decomposition and the optimal printing direction for each piece. The quality measures E_M is* 0.0988 *for OOD,* 0.0914 *for OSAM and* 0.0244 *for PDBD.*

Table 1 presents numerical comparison of the printing errors resulting from the three methods. The same numbers of parts are used for both *OSAM* and *PDBD*, it was obtained by testing several value around an initial one estimated with *self-tuning spectral clustering* [ZMP04]. We can use *OOD* as a reference to compute improvement rates of the printing quality resulting from a segmentation :

$$Rate_{algo}(M) = \frac{E_{OOD}(M) - E_{algo}(M)}{E_{OOD}(M)}, \tag{6}$$

where, M is a 3D model and $E_{algo}(M)$ is the printing error of M caused by the method $algo$ and $algo$ is either *OSAM* or *PDBD*. In average, the improvement rate of *OSAM* is 17.33%, while the one of *PDBD* is 47.43%. While *OSAM* ensures improvement over *OOD*, the restriction to orthogonal printing directions and cubical partitioning limits the reduction of the printing error for some models (such as the dragon and desk which improvement rates are below 7%). In comparison, our method ensures a minimal improvement of 24% over *OOD* and up to 88%.

In Figure 11, we show the printed Armadillo model generated by *OOD*, *OSAM*, and *PDBD*, respectively. In the left column, we see the influence of the staircase error which is largely reduced by our method. In the second column, we see the damages inflicted to the model by the support structures: Due to an optimal local printing direction, our method tends to require less and smaller support structures that are most of the time added to the interface between parts. Thanks to both these advantages our method tends to provide smoother surfaces and better preservation of details. For instance, the jagged artefact on the armadillo's ear is removed by our method and the patterns on armadillo's shell are cleaner and better visible. Finally, since *OSAM* does not calculate the assembly order nor creates connectors, it is more difficult in practice to assemble correctly two neighboring parts.

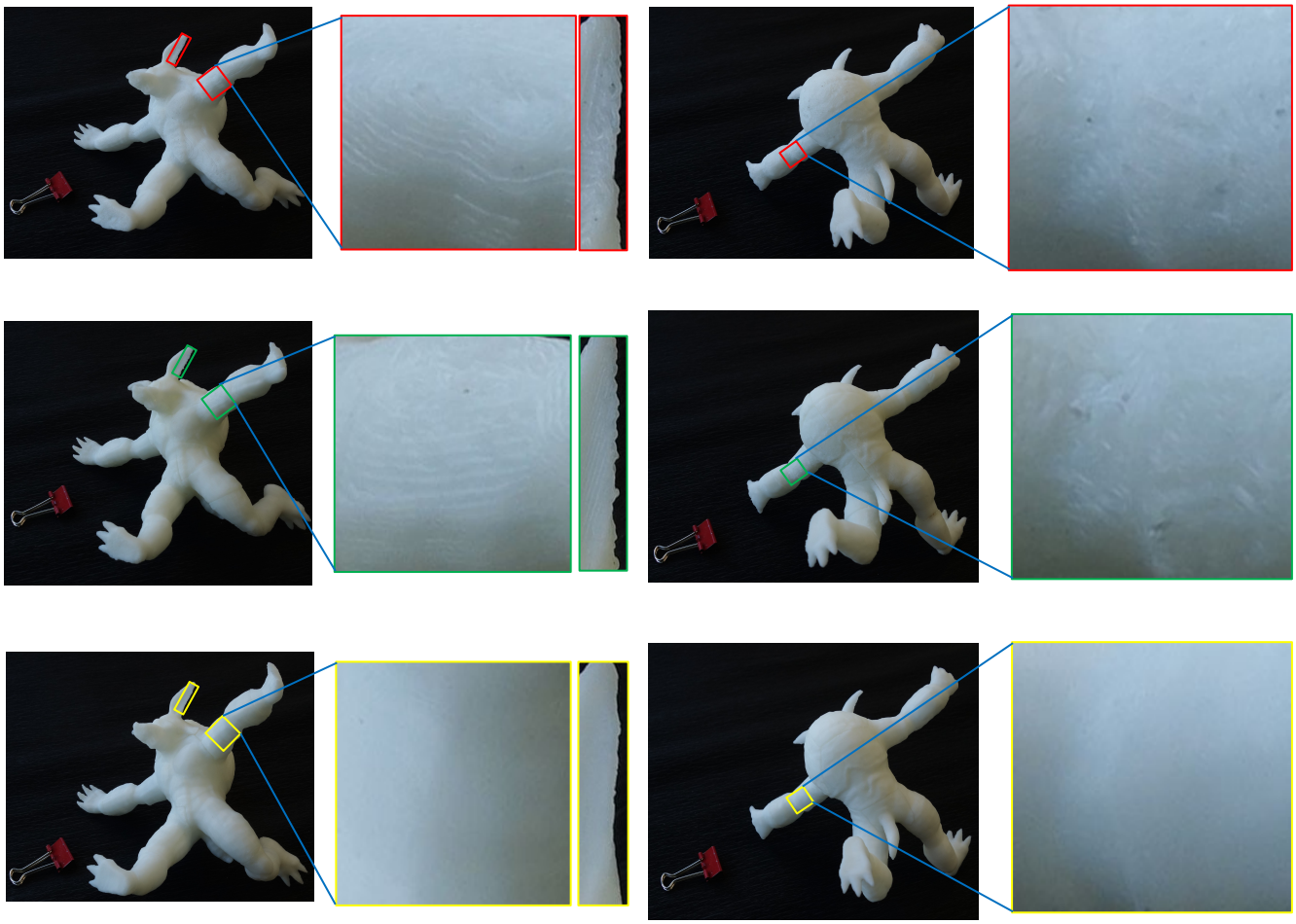

Figure 11: *Comparison of physical prints. From top to bottom we show prints generated by OOD, OSAM, and PDBD (ours). The first column reveals the staircase artifacts resulting from the layered manufacturing and the second column shows damages caused by the removal of support structures.*

9.4. User Feedback

We conduct a brief user survey in order to assess the effectiveness of our method (*PDBD*) in terms of both haptic and visual quality in comparison to a single optimized direction (*OOD*).

Haptic: The first questions (Q1 and Q2) are asked while holding the objects in hand but hidden from sight. For question Q2, participants are told that the object is assembled from several pieces.

Q1 : Which object is smoother?

Q2 : Can you feel the gaps on the object generated by PDBD? And if yes, are they disturbing?

Q3 : After looking at the object generated by PDBD, can you feel the gaps on the object? If yes, are they disturbing?

Visual: The participants are asked :

Q4 : Which object has more details, structures and textures?

Q5 : Which object has smaller printing errors and artifacts?

Q6 : What is the more disturbing between bad printing quality and gaps?

Participants of the survey were 21 college students and staffs (2 women and 19 men), results are shown in Figure 12. We can draw the following conclusions: our method *PDBD* presents an improvement in terms of haptic quality (the surface feels smoother and seams are mainly noticed if seen). In terms of visual quality, most participants subjectively found that our method produce more details with smaller printing error but participants are divided when it comes to what is the more disturbing between bad printing quality and gaps.

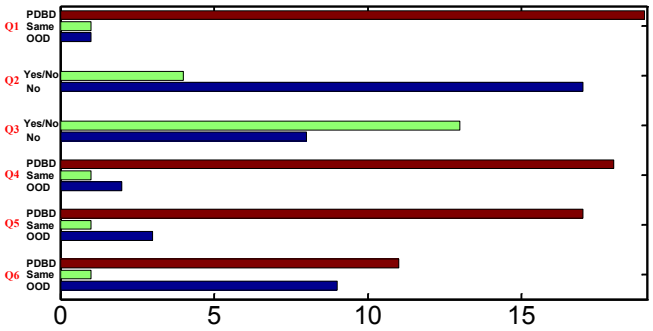

Figure 12: *Results of our user survey. Six groups of bars are corresponding to the answers of questions Q1 to Q6.*

In Figure 13, we present a comparison between painted models : one printed in a single piece with *OOD* and one where the seams have been filled with a modeling putty (we painted the models to avoid the visual softness emerging from the translucency of the printing material). Such fixing does not cause a reduction in details such as when a surface is sanded or smoothed with acetone vapor.

Figure 13: *Comparison between a painted model segmented with our method after filling of the seams (right) and one printed in a single piece with OOD (left).*

9.5. Large Models

Because the printing volume of a printer is limited, some large models cannot be printed [LBRM12]. Our method can be modified to segment these models while guaranteeing an optimal printing quality. After the initial parts generation, we could check that each part fits into the volume of the printer. The parts that are problematic are then partitioned into subparts: three orthogonal cutting planes (one of them having the printing direction as normal) passing through the mass center of the part are used to subdivide the part. By using a recursive subdivision when necessary, final parts are guaranteed to fit in the printer. Because the printing direction stay unchanged, the surface quality is unaffected.

9.6. Printing Time

To save printing time, we can pack the parts into the volume of the printer, as done in [VGB*14b, CZL*15, YCL*15, Att15]. For the Armadillo model, under the same layer thickness, the whole model is printed in 10 hours, while the packed parts generated by our method are printed in about 8 hours. Given that our method allows an error reduction around 50% in average, we could in addition increase the layer thickness by two while keeping a similar surface quality as with OOD. Doing so we would save even more printing time.

10. Discussion

10.1. Seams Optimization

Once we have obtained a decomposition of the model with the associated assembly directions, we can easily adapt existing optimization techniques such as snakes [LL02, BWK05] (polylines which vertices, called snaxels, are defined on mesh edges) to optimize the seams in order to reduce their visibility while preserving the assembability (in Figure 14, we move seams toward minima of mean curvatures).

First, note that all interface between two parts generated by our method can be represented as a height fields in the associated assembly direction $\mathbf{a}_{dir}$. The assembly will remain valid if this property is kept during optimization. Given the fact that a height field defined on the boundary of a 2D domain is sufficient to define an interpolated height field in the domain, it is sufficient to work with the boundary of the interfaces in order to optimize the seams while preserving a valid assembly order. Each interface's boundary is defined by a set of polylines (that live on the surface) and of interior segments (edges of Voronoi cells). The polylines are used to define our snakes while the interior segments are only used to define assembly constraints and are not optimized. Since initial cuts are valid, we just have to restrict the movement of snaxels such that they preserve the assembly constraint. The latter can be defined locally by guaranteeing the absence of flips in a local neighborhood of a boundary point:

$$\mathbf{a}_{dir}{}^{\top}\ \mathbf{n}_{cut} > \varepsilon$$

with $\mathbf{n}_{cut}$ the normal of the interface at the boundary point and the safety margin ε is used to guarantee that the local cut does not include the assembling direction as it is done in section 7. In order to obtain parts of high interior quality (section 6), we would like the interface to meet the model's outer surface orthogonally, i.e. we would like to use $\mathbf{n}_{cut}$ orthogonal to both the surface normal $\mathbf{n}_f$ and to the local tangent direction $\mathbf{t}$ of the interface's boundary. In this case, previous inequality becomes :

$$\mathbf{a}_{dir}{}^{\top}\ \frac{(\mathbf{n}_f \times \mathbf{t})}{\|\mathbf{n}_f \times \mathbf{t}\|} > \varepsilon\ . \tag{7}$$

However, such cut is not always valid, therefore we need a relaxation $\mathbf{n}_r$ that will replace the face normal $\mathbf{n}_f$ in equation (7). Since the new cut should be defined inside the surface we can use the following formula : $\mathbf{n}_r = (1-\alpha)\mathbf{n}_f \pm \alpha \mathbf{n}_f \times \mathbf{t}$ with the sign of $\pm \mathbf{n}_f \times \mathbf{t}$ chosen such that this vector is a valid solution for $\varepsilon = 0$. The optimal relaxed normal is obtained for the smallest value α in the range $[0;1]$ that satisfy equation (7). The latter can be re-arranged into a linear and a quadratic inequalities in $\mathbf{n}_r$ and therefore in α.

During the optimization, a snaxel movement occurs along the associated supporting edge $[\mathbf{v}_{from}, \mathbf{v}_{to}]$, and its movement have to be restricted such that the assembly constraint is always satisfied for the two adjacent snake segments. If the snaxel position on its support-

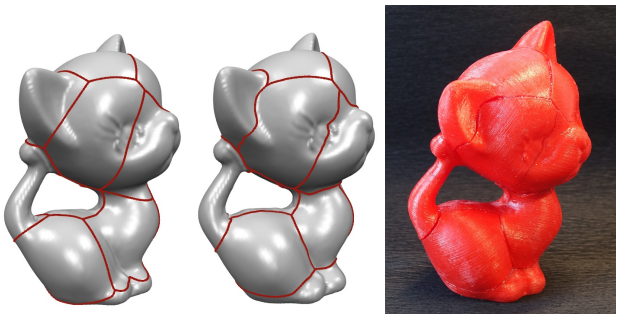

Figure 14: *In order to make the seams less visible, it is possible to move them to areas of negative mean curvature. This can be done by a minimization using snakes whose movements are constrained to preserve validity of assembly. Most of the original seams (left) are moved towards concave areas by the optimization (middle). Right : Photograph of the printed model.*

ing edge is parametrized in $[0;1]$, we can define the tangent $\mathbf{t}$ from equation (7) by $\mathbf{t} = \pm((d(\mathbf{v}_{to} - \mathbf{v}_{from}) + \mathbf{v}_{from} - \mathbf{p}_{adj})$ with $\mathbf{p}_{adj}$ the adjacent snaxel position and the sign is chosen in order to obtain a coherent tangent orientation. Injecting the formula of $\mathbf{t}$ and a fixed relaxed normal $\mathbf{n}_r$ in equation (7) boils down to a linear and a quadratic inequalities in d which give us the range of valid position during an optimization step. Just note that we need to use an over-relaxed normal instead of the optimal one to allow for more freedom during the optimization.

It is important to note that discontinuous snake operation such as snake cleaning [BWK05] cannot break the validity of the assembly. Finally, intersection in the 2D height field domain between segments of the boundary (both interior and snake segments) should be prevented.

Finally, we impose a minimal angle constraint at the junction between snakes that correspond to adjacent interfaces. Once the interface boundaries have been optimized, we can reconstruct the final interface similarly to what is done in section 6 but constraining the movement of vertices to the assembly direction. A photograph of a printed model with optimized seams is shown in Figure 14.

10.2. Limitations

Our method still has some limitations. First, we cannot determine the best number of parts for a given model beyond the self-tuning spectral clustering heuristic. The relationship between printing error and the number of parts should be studied more thoroughly. Second, although the visual and haptic quality of the parts can be considerably improved by our method, some visual artifacts are introduced along the segmentation cuts which are unavoidable when the model is segmented. The artificial gaps that we introduce can at least reduce this effect. Finally, the proposed method does not guarantee the absence of support structures, it only reduce their presence.

11. Conclusions and Future Work

In this work, we propose a printing direction based segmentation method to partition a given 3D model into several pieces so that each piece can be printed with a small error in its own printing direction. In order to allow a simple assembly, an order and directions of assembly are computed. The connectors generated from this data ensure that the parts do not fall apart and match at the boundaries when put together. A number of experimental results illustrate the practicability and robustness of our method and demonstrate that our algorithm can reduce printing error and/or printing time compared to previous methods.

In the future, we plan to study in more details the relationship between the number of parts, the printing quality and the visual impact of the seams so that a balanced solution can be found. We also want to propose an algorithm dedicated to decomposing large models such that usage of supporting material and printing quality are simultaneously optimized.

Acknowledgements

We would like to thank the reviewers for their detailed comments and suggestions which greatly improved the manuscript. We are grateful to LIAN et al. [LGB*11] for their 3D databases and to all the participants of our user survey.

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement n°340884 and Natural Science Foundation of China (61370143,61432003).

References

[AAD98] ALEXANDER P., ALLEN S., DUTTA D.: Part orientation and build cost determination in layered manufacturing. *Computer-Aided Design 30*, 5 (1998), 343–356. 3

[ACP*14] ALEMANNO G., CIGNONI P., PIETRONI N., PONCHIO F., SCOPIGNO R.: Interlocking Pieces for Printing Tangible Cultural Heritage Replicas. In *Eurographics Workshop on Graphics and Cultural Heritage* (2014), Klein R., Santos P., (Eds.), The Eurographics Association. doi:10.2312/gch.20141312. 2

[APH*03] AGRAWALA M., PHAN D., HEISER J., HAYMAKER J., KLINGNER J., HANRAHAN P., TVERSKY B.: Designing effective step-by-step assembly instructions. In *ACM Transactions on Graphics (TOG)* (2003), vol. 22, ACM, pp. 828–837. 3

[Att15] ATTENE M.: Shapes in a box: Disassembling 3d objects for efficient packing and fabrication. 64–76. 2, 10

[BBJP12] BÄCHER M., BICKEL B., JAMES D. L., PFISTER H.: Fabricating articulated characters from skinned meshes. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 47. 2

[BWBSH14] BÄCHER M., WHITING E., BICKEL B., SORKINE-HORNUNG O.: Spin-it: optimizing moment of inertia for spinnable objects. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 96. 2

[BWK05] BISCHOFF S., WEYAND T., KOBBELT L.: Snakes on triangle meshes. In *Bildverarbeitung für die Medizin 2005*. Springer, 2005, pp. 208–212. 10, 11

[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 Transactions on Graphics (TOG) 31*, 6 (2012), 130. 2

[CL96] COLEMAN T. F., LI Y.: An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on optimization 6*, 2 (1996), 418–445. 5

[CLM*13] CEYLAN D., LI W., MITRA N. J., AGRAWALA M., PAULY M.: Designing and fabricating mechanical automata from mocap sequences. *ACM Trans. Graph. 32*, 6 (2013), 186. 2

[CPMS14] CIGNONI P., PIETRONI N., MALOMO L., SCOPIGNO R.: Field-aligned mesh joinery. *ACM Transactions on Graphics (TOG) 33*, 1 (2014), 11. 3

[CZL*15] CHEN X., ZHANG H., LIN J., HU R., LU L., HUANG Q., BENES B., COHEN-OR D., CHEN B.: Dapper: decompose-and-pack for 3d printing. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 213. 2, 10

[DHL14] DUMAS J., HERGEL J., LEFEBVRE S.: Bridging the gap: automated steady scaffoldings for 3d printing. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 98. 2

[Dim14] DIMENSION: Rapid prototyping and 3D printing. http://www.alphacam.de, 2014. 3, 7

[DPW*14] DEUSS M., PANOZZO D., WHITING E., LIU Y., BLOCK P., SORKINE-HORNUNG O., PAULY M.: Assembling self-supporting structures. *ACM Trans. Graphics (Proc. ACM Siggraph Asia) 33*, 6 (2014). 3

[FF95] FRANK D., FADEL G.: Expert system-based selection of the preferred direction of build for rapid prototyping processes. *Journal of Intelligent Manufacturing 6*, 5 (1995), 339–345. 3

[HBA12] HILDEBRAND K., BICKEL B., ALEXA M.: crdbrd: Shape fabrication by sliding planar slices. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 583–592. 3

[HBA13] HILDEBRAND K., BICKEL B., ALEXA M.: Orthogonal slicing for additive manufacturing. *Computers & Graphics 37*, 6 (2013), 669–675. 3, 8

[HFW11] HAO J., FANG L., WILLIAMS R. E.: An efficient curvature-based partitioning of large-scale stl models. *Rapid Prototyping Journal 17*, 2 (2011), 116–127. 2, 3

[HL14] HERGEL J., LEFEBVRE S.: Clean color: Improving multi-filament 3d prints. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 469–478. 2

[HLZCO14] HU R., LI H., ZHANG H., COHEN-OR D.: Approximate pyramidal shape decomposition. *ACM Transactions on Graphics (TOG) 33*, 6 (2014), 213. 2

[HMA15] HERHOLZ P., MATUSIK W., ALEXA M.: Approximating Free-form Geometry with Height Fields for Manufacturing. *Computer Graphics Forum (Proc. of Eurographics) 34*, 2 (2015), 239–251. doi:10.1111/cgf.12556. 2

[LBRM12] LUO L., BARAN I., RUSINKIEWICZ S., MATUSIK W.: Chopper: partitioning models into 3d-printable parts. *ACM Trans. Graph. 31*, 6 (2012), 129. 2, 3, 10

[LDPT13] LAN Y., DONG Y., PELLACINI F., TONG X.: Bi-scale appearance fabrication. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 145. 2

[LFL09] LO K.-Y., FU C.-W., LI H.: 3d polyomino puzzle. *ACM Transactions on Graphics (TOG) 28*, 5 (2009), 157. 3

[LGB*11] LIAN Z., GODIL A., BUSTOS B., DAOUDI M., HERMANS J., KAWAMURA S., KURITA Y., LAVOUÉ G., VAN NGUYEN H., OHBUCHI R., ET AL.: Shrec'11 track: Shape retrieval on non-rigid 3d watertight meshes. *3DOR 11* (2011), 79–88. 7, 11

[LL02] LEE Y., LEE S.: Geometric snakes for triangular meshes. In *Computer Graphics Forum* (2002), vol. 21, Wiley Online Library, pp. 229–238. 10

[LSZ*14] LU L., SHARF A., ZHAO H., WEI Y., FAN Q., CHEN X., SAVOYE Y., TU C., COHEN-OR D., CHEN B.: Build-to-last: Strength to weight 3d printed objects. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 97. 2

[MRI00] MASOOD S., RATTANAWONG W., IOVENITTI P.: Part build orientations based on volumetric error in fused deposition modelling. *The International Journal of Advanced Manufacturing Technology 16*, 3 (2000), 162–168. 3

[MYY*10] MITRA N. J., YANG Y.-L., YAN D.-M., LI W., AGRAWALA M.: Illustrating how mechanical assemblies work. *ACM Transactions on Graphics-TOG 29*, 4 (2010), 58. 3

[PRD07] PANDEY P., REDDY N. V., DHANDE S.: Part deposition orientation studies in layered manufacturing. *Journal of materials processing technology 185*, 1 (2007), 125–131. 3

[PTM*15] PRZEMYSLAW M., THOMAS A., MICHEAL B., MICHEAL W., LEIF K.: Reduced-order shape optimization using offset surfaces. *ACM Transactions on Graphics (TOG) 34*, 4 (2015). 2

[PWLSH13] PRÉVOST R., WHITING E., LEFEBVRE S., SORKINE-HORNUNG O.: Make it stand: balancing shapes for 3d fabrication. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 81. 2

[RCM*14] REINER T., CARR N., MĚCH R., ŠT'AVA O., DACHSBACHER C., MILLER G.: Dual-color mixing for fused deposition modeling printers. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 479–486. 2

[SFCO12] SONG P., FU C.-W., COHEN-OR D.: Recursive interlocking puzzles. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 128. 3

[SFLF15] SONG P., FU Z., LIU L., FU C.-W.: Printing 3d objects with interlocking parts. *Computer Aided Geometric design (Proc. of GMP 2015)* (2015). 2, 3

[Sha08] SHAMIR A.: A survey on mesh segmentation techniques. In *Computer graphics forum* (2008), vol. 27, Wiley Online Library, pp. 1539–1556. 2

[SKS00] SCHNEIDER R., KOBBELT L., SEIDEL H.-P.: Improved bi-laplacian mesh fairing. *Mathematical Methods for Curves and Surfaces, Oslo* (2000), 445–454. 5

[SM00] SHI J., MALIK J.: Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 22*, 8 (2000), 888–905. 4

[SP13] SCHWARTZBURG Y., PAULY M.: Fabrication-aware design with intersecting planar pieces. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 317–326. 3

[SU14] SCHMIDT R., UMETANI N.: Branching support structures for 3d printing. In *ACM SIGGRAPH 2014 Studio* (2014), ACM, p. 9. 2

[SVB*12] STAVA O., VANEK J., BENES B., CARR N., MĚCH R.: Stress relief: improving structural strength of 3d printable objects. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 48. 2

[SVGDB*02] SUYKENS J. A., VAN GESTEL T., DE BRABANTER J., DE MOOR B., VANDEWALLE J., SUYKENS J., VAN GESTEL T.: *Least squares support vector machines*, vol. 4. World Scientific, 2002. 5

[TPR04] THRIMURTHULU K., PANDEY P. M., REDDY N. V.: Optimum part deposition orientation in fused deposition modeling. *International Journal of Machine Tools and Manufacture 44*, 6 (2004), 585–594. 3, 8

[Ult15] ULTIMAKER: Ultimaker 3d printer. https://ultimaker.com/, 2015. 3

[VGB14a] VANEK J., GALICIA J., BENES B.: Clever support: efficient support structure generation for digital fabrication. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 117–125. 2

[VGB*14b] VANEK J., GALICIA J., BENES B., MĚCH R., CARR N., STAVA O., MILLER G.: Packmerger: A 3d print volume optimizer. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 322–332. 2, 10

[VWRKM13] VIDIMČE K., WANG S.-P., RAGAN-KELLEY J., MATUSIK W.: Openfab: A programmable pipeline for multi-material fabrication. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 136. 2

[WCT*15] WANG W., CHAO H., TONG J., YANG Z., TONG X., LI H., LIU X., LIU L.: Saliency-preserving slicing optimization for effective 3d printing. *Computer Graphics Forum 34*, 6 (2015), 148–160. 2

[WWY*13] WANG W., WANG T. Y., YANG Z., LIU L., TONG X., TONG W., DENG J., CHEN F., LIU X.: Cost-effective printing of 3d objects with skin-frame structures. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 177. 2

[XLF*11] XIN S., LAI C.-F., FU C.-W., WONG T.-T., HE Y., COHEN-OR D.: Making burr puzzles from 3d models. *ACM Transactions on Graphics (TOG) 30*, 4 (2011), 97. 3

[XWL*97] XU F., WONG Y., LOH H., FUH J., MIYAZAWA T.: Optimal orientation with variable slicing in stereolithography. *Rapid Prototyping Journal 3*, 3 (1997), 76–88. 3

[XXY*15] XIE Y., XU W., YANG Y., GUO X., ZHOU K.: Agile structural analysis for fabrication-aware shape editing. *Computer Aided Geometric Design 35* (2015), 163–179. 2

[YCL*15] YAO M., CHEN Z., LUO L., WANG R., WANG H.: Level-set-based partitioning and packing optimization of a printable model. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 214. 2, 10

[YSO14] YAMANAKA D., SUZUKI H., OHTAKE Y.: Density aware shape modeling to control mass properties of 3d printed objects. In *SIGGRAPH Asia 2014 Technical Briefs* (2014), ACM, p. 7. 2

[ZLP*15] ZHANG X., LE X., PANOTOPOULOU A., WHITING E., WANG C. C.: Perceptual models of preference in 3d printing direction. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 215. 2

[ZMP04] ZELNIK-MANOR L., PERONA P.: Self-tuning spectral clustering. In *Advances in neural information processing systems* (2004), pp. 1601–1608. 4, 7, 8

[ZPZ13] ZHOU Q., PANETTA J., ZORIN D.: Worst-case structural analysis. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 137. 2

[ZXS*12] ZHU L., XU W., SNYDER J., LIU Y., WANG G., GUO B.: Motion-guided mechanical toy modeling. *ACM Trans. Graph. 31*, 6 (2012), 127. 2