EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)

Volume 39 (2020), Number 2

Optimizing Object Decomposition to Reduce Visual Artifacts in 3D Printing

I. Filoscia[1], T. Alderighi[1,2], D. Giorgi[1], L. Malomo[1], M. Callieri[1] and P. Cignoni[1]

[1] ISTI – CNR
[2]University of Pisa

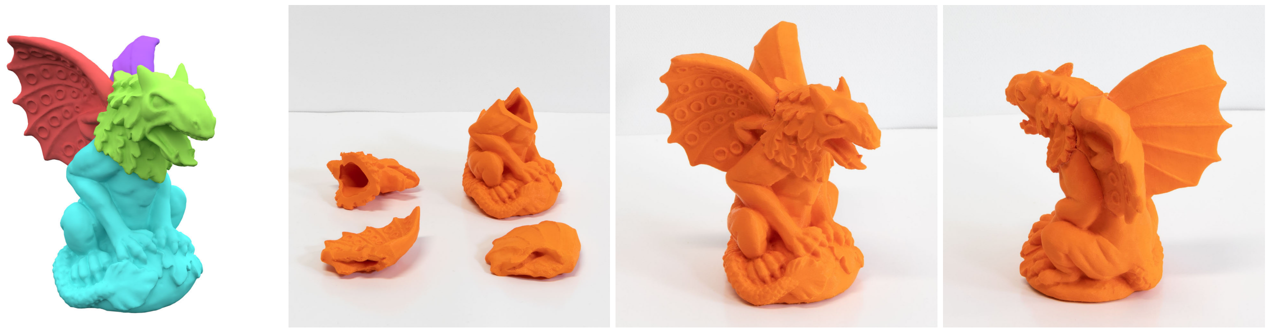


Figure 1: *Our method partitions an input object into a small number of simpler parts, which can be 3D printed with few supports, and such that the gaps separating parts have a low visual impact on the assembled object.*

Abstract

We propose a method for the automatic segmentation of 3D objects into parts which can be individually 3D printed and then reassembled by preserving the visual quality of the final object. Our technique focuses on minimizing the surface affected by supports, decomposing the object into multiple parts whose printing orientation is automatically chosen. The segmentation reduces the visual impact on the fabricated model producing non-planar cuts that adapt to the object shape. This is performed by solving an optimization problem that balances the effects of supports and cuts, while trying to place both in occluded regions of the object surface. To assess the practical impact of the solution, we show a number of segmented, 3D printed and reassembled objects.

CCS Concepts
• ***Computing methodologies*** → ***Shape modeling; Mesh geometry models; Shape analysis;***

1. Introduction

Digital fabrication, and 3D printing in particular, are growing in importance in a variety of fields, from industry to medicine, from cultural heritage to art. We are also witnessing the transition of 3D printing technology from industry to the community of non-professional users, sometimes referred to as the process of *democratization* of 3D printing [BM17]. Therefore, a new discipline has emerged called *computational fabrication*, which investigates manufacturing problems using computational tools, and especially graphics and geometry processing techniques. Whereas for many years the focus in graphics was on the creation and processing of digital 3D representations of real-world objects, the success of fabrication techniques now calls for novel methods for the reverse process, namely, for making tangible what lives in the digital world.

A major challenge is to develop techniques which support creative projects, and enable people to print high-quality, complex objects at home. One of the most common technologies for home use is fused deposition modeling (FDM). The printing material is extruded by a nozzle and deposited, layer by layer, to fabricate the desired object. A major limitation of FDM is that the extruded material must be supported when deposited: additional columns of material called *supports* have to be printed along with the object to

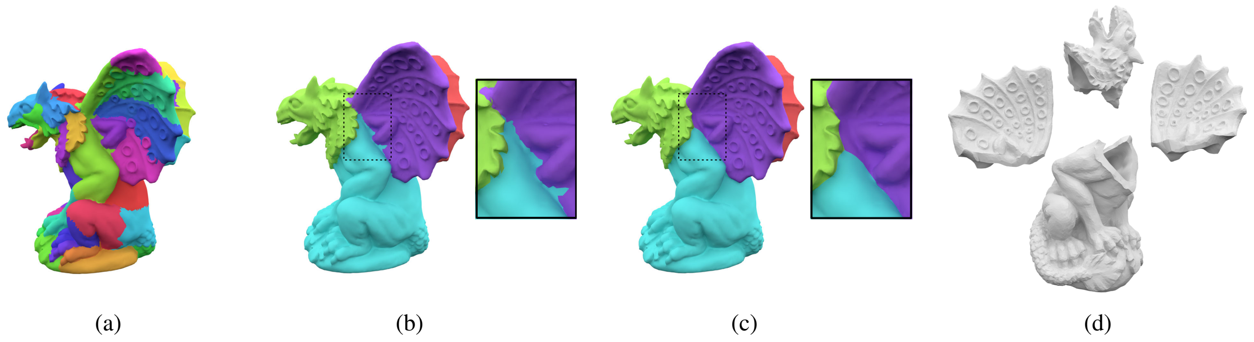


Figure 2: *Our segmentation and fabrication pipeline. (a) Oversegmentation into smartly-shaped patches; (b) merging patches which can be printed in the same direction, to get an initial segmentation; (c) refining the segmentation, by locally moving and smoothing the segmentation boundaries; (d) defining the solid shells corresponding to pieces in the segmentation, and 3D printing them along their optimized orientation.*

prevent overhanging parts from collapsing. The supports are then removed in a post-processing step. Unfortunately, this may cause visual defects on the surface at regions contacted by supporting structures, damage small geometric features, and even break thin parts. Although there exist 3D printers which use a different, soluble material to print supports and make them easier to remove, both the 3D printing hardware and the soluble material can be expensive, and therefore they are not common in domestic environments. Usually, the general approach is to look for the optimal printing direction that minimizes the amount of supports required, as the latter depend on the 3D printing orientation of the model. However, for complex shapes, the amount of supports can be significant even for the optimal printing direction [WCT*15, ZLP*15].

An alternative solution is to segment 3D objects into smaller parts, which can be printed individually with no or limited supports. Then, the parts have to be reassembled and glued together, as it happens for example with expertly-designed, commercial plastic miniatures. The main drawback is that the decomposition into parts introduces *cuts* on the object surface, in correspondence of the boundaries between parts. If placed in correspondence of salient or visible parts, such cuts can be as visually disturbing as the imperfections due to the removal of supports, or even more.

In this context, this paper proposes a solution for the automatic segmentation of 3D objects into parts which can be printed individually via FDM, and then reassembled by preserving the visual quality of the final object. Differently from existing techniques which only cut along planes and across possibly highly-visible regions, we define *non-planar* cuts, which better adapt to the object shape, and minimize the visual impact of the decomposition (Figure 1).

1.1. Contribution

Given an input 3D mesh, we develop a fully-automatic segmentation technique to partition the mesh into a small number of simpler parts, each of which can be printed with no or minimal support, and such that the boundaries between the parts (i.e., where the cuts in the object surface will be) affect the appearance of the printed model as little as possible.

Figure 2 depicts the pipeline of our algorithm. First, we define an oversegmentation into a set of patches using a variant on the isophotic metric [PSH*04, KT03]; the patch boundaries are expected to align with surface features, as they are enforced to lay along concave features (Figure 2a). Then, we merge the patches that can be printed along the same direction with low overhanging areas, to get an initial segmentation into parts that are printable with no or few supports (Figure 2b). The cuts separating parts are expected to be non invasive, according with the *minima rule* which states that human vision defines part boundaries on surfaces along negative minima of the principal curvatures [HR84]. Finally, we locally refine the boundaries of the initial segmentation, by smoothing cuts and encouraging them to move towards less visible regions (Figure 2c). The segmented parts are then printed individually along their optimized printing direction (Figure 2d).

We pose the initial segmentation problem as a multi-labeling problem solved via functional minimization [SY96]. The data points are given by patches, and the labels are potential printing directions. We define an objective function that takes into account the area of supported regions and the length of cuts, as well as the visual impact of both in terms of location on the surface. We formulate this multi-labeling problem as an Integer Linear Program (ILP), which can be solved using standard optimization packages.

To assess the practical impact of our solution, we decompose, print and reassemble a number of challenging shapes. Our examples show that – differently from existing algorithms which either decompose into a possibly high number of support-free parts, or cut along planes across possibly highly-visible regions – we strike a good balance between having few supports and nicely-placed cuts.

2. Related work

3D segmentation for efficient fabrication has recently drawn attention from the computer graphics research community. The decomposition of objects into parts, indeed, helps solving different issues related to fabrication. Many works focused on 3D decomposition for the simultaneous fabrication of multiple, smaller, parts that fit into the working volume [CZL*15, VGB*14] and can be easily delivered [Att15]. For most fabrication techniques, such as 3-axis milling [MLS*18] and rigid molding [HMA15], an additional constraint for the shape is to be a height field. Therefore, objects which do not satisfy the height field constraint need to be decomposed into pieces that fulfill the constraint individually, such as pyramidal parts [HLZCO14].

Segmentation for printing quality optimization A number of works explicitly dealt with the decomposition into 3D printable parts, either completely support-free or with a minimal amount of supported areas, to reduce aesthetic defects on the final object. The approach of Wei et al. [WQZ*18] takes advantage of the skeletal graph of the object. The idea is to decompose the Laplacian skeleton of the model into a small number of sub-graphs corresponding to support-free object parts. The authors privilege minimizing the number of parts over the total cut length, and cuts are preferred on concave areas. Nonetheless, cuts are always planar: as a consequence, they do not follow the object shape by design, and are likely to cut across salient areas. Moreover, the use of skeletons restricts the focus on particular classes of models, having local tubular shapes which are captured by the skeletons themselves.

To reduce the amount of supports, Dai et al. [DWW*18] build on a special hardware for multi-directional 3D-printing. The method segments the mesh via planar cuts on clipping planes. The authors start by sampling a high number (15k - 20k) of candidate clipping planes, then use a beam-guided search scheme to look for an optimal segmentation which minimizes the amount of supports. For the remaining supports, a new type of supports is proposed, called *projected supports*, which should minimize the material cost and the impact on the final object. To avoid a fully-randomized search over possible cutting planes, Karasik et al. [KFW19] look for specific separating planes: planar surfaces of the object that can act as a printing bases; planar bases for protruding narrow regions of the object (tips); and planes splitting the object in correspondence of T-junctions. The search procedure is recursive, according with a randomized routine over the remaining non-printable parts of the object. Both the methods cited above only induce planar cuts on the surface, and they do not fully take into account the visibility of cuts on the surface, since the cuts do not explicitly align with the shape geometry and semantics.

Wang et al. [WZK16] aim at improving the visual surface quality of 3D printed objects by segmenting objects into pieces that both minimize supports and reduce the staircase (layer aliasing) effect. The authors use Support Vector Machines to find optimal cutting planes, and 3D Voronoi cells to derive the volumetric decomposition. A post-processing step optimizes the cuts between segments by locally moving them towards areas with higher concavity. However, the visual impact of cuts is only taken into account at the end of the pipeline, resulting in cuts which still cross highly-visible regions. On the contrary, we define from the very beginning non-planar cuts, located along semantic boundaries between parts and on occluded areas.

Segmentation as a labeling problem Segmentation can be seen as a multi-labeling problem: a set of data points have to be assigned to a finite set of labels, so as to minimize an objective function. Points with different labels belong to different segments. The objective function usually includes different terms: a *data cost* which measures the cost of assigning labels to points, and *regularization factors*, promoting smooth boundaries between segments and a limited number of segments.

Segmentation via functional minimization has been widely used in Computer Vision [ZY96], Computer Graphics [SSS*10], and also recently in the context of computational fabrication [HMA15, AMG*18]. In particular, the authors of [AMG*18] propose a method for silicone mold casting, based on the segmentation of objects into moldable pieces. They define an energy which takes into account the cost for detaching the mold from the object, while leveraging on the elasticity of silicone to relax the process-related constraints. Our strategy inherits from [AMG*18] the use of an Integer Linear Programming formulation for the segmentation problem, whereas it completely differs in both the segmentation purpose and the way the energy functional is defined. Indeed, we optimize for the placement of cuts and supports on 3D printed objects, by leveraging on the isophotic metric and on ambient occlusion. The advantage of the ILP approach for segmentation is that it can model different problems within the same algorithmic framework, by defining different objective functions and sets of constraints.

3. Overview

Our goal is to partition an input 3D triangle mesh into a set of pieces, each with its own printing direction, and with the following (possibly conflicting) desiderata:

- the pieces should be support-free along their printing direction, or have small supported areas, possibly in occluded parts;
- the number of pieces should be small;
- the cuts separating pieces should have a limited visual impact on the final assembled object.

To reach this goal, our pipeline works in four steps (Figure 2).

S1 - Oversegmentation First, we produce an oversegmentation of the mesh into a set of patches, whose boundaries are good potential candidates for cuts. To define the patches, we use a multi-source region growing algorithm from a set of seed faces, guided by a variant on the isophotic metric in [PSH*04], which enforces preference for patch boundaries along concave features. Therefore, the candidate cuts align well with shape features (Section 4.1);

S2 - Initial segmentation Then, we aggregate the patches in the oversegmentation that can be 3D printed together, to reach a small number of printable pieces, each with its own printing direction. We define the segmentation problem as a multi-labeling problem solved via functional minimization: we formulate an Integer Linear Problem (ILP) in which the data points are the patches, and the labels are potential printing directions. The objective function takes into account the area of regions affected by supports and the length of cuts, as well as the location of both supports and cuts on visible or occluded regions. The advantages of our formulation are twofold. On the one hand, the number of patches is significantly smaller than the number of mesh faces, thus reducing the computational cost of the problem. On the other hand, since the segmentation boundaries are selected among the smartly-shaped patch boundaries, we obtain cuts that hide into the object folds, thus minimizing the visual impact of the decomposition. The result of the optimization process is a segmentation which strikes a good balance between having few supports and nicely-placed cuts (Section 4.2);

S3 - Refined segmentation Finally, we locally refine the boundaries of the segmentation to get the final decomposition. We define small fuzzy regions near the initial boundaries, and solve a similar ILP problem as in S2, this time using as data points all mesh faces in the fuzzy regions to get smoother cuts, and defining an energy which further pushes boundaries towards occluded regions. Again, as the number of faces in the fuzzy areas is significantly smaller than the original number of mesh faces, we get smooth and well-located boundaries at a reasonable computational cost (Section 4.3). For both steps S2 and S3, we solve the ILP problem using Gurobi [GO19], a commercial optimization package;

S4 - 3D Printing and reassembly of parts Once we have the segmented parts, each with its optimized printing direction, we build their corresponding pieces by propagating the surface partition inside the tetrahedralized volume, and 3D print them. The parts are then reassembled and glued together to get the final object (Section 4.4).

Our method eventually produces *shell models*, which are hollow, and are widely used in mechanical and artistic designs. At any rate, with slight modifications, the method can be easily adapted to obtain solid models as well.

4. Method

Given an input mesh with n faces, the first step is to define an initial library of potential cuts, among which to select the segmentation boundaries. To this end, we generate an oversegmentation into N sensible patches whose borders are good candidates for cuts. We use a a multi-source region growing algorithm from a set of seed faces, following the approach in [KT03], which uses the isophotic metric to drive the growth.

4.1. Oversegmentation

We start by selecting N seed faces via a farthest point sampling approach. We first look for a *central* face, by sampling $m \ll n$ faces with constrained Poisson disk sampling [CCS12], and taking the face that has the smaller value for the sum of geodesic distances to the other m faces. Then, we use a recursive procedure which selects as the next seed the face having the highest geodesic distance to the previous set of seeds. The procedure stops when the maximum geodesic distance between pairs of seeds is below a threshold ε_{patch}. The threshold sets a bound on the minimum size for patches, therefore, a bound for the minimum size of pieces in the decomposition. In our implementation, we set $\varepsilon_{patch} = 1\%$ of the mesh bounding box diagonal, and get $57 \leq N \leq 151$ seeds (hence patches) in the oversegmentations of our tested models (see Section 5, Results).

To get the patches, we grow from the seeds by computing for each mesh face its distance from the seeds and assigning the face to the patch represented by the nearest seed. As a distance, we rely on a weighted distance which combines the geodesic distance with the *angular distance*, as suggested in [KT03]. The angular distance is a variant of the isophotic distance in [PSH*04], made so that it dilates the distances in correspondence of concave features, and it tends to 0 on convex and planar areas. Formally, if f_i and f_j are two adjacent faces, their *angular distance* is defined as

$$d_a(f_i, f_j) = \eta(1 - cos(\alpha_{ij})), \tag{1}$$

with α_{ij} the angle between the face normals, and η equal to 1 for concave angles and a small value for convex ones. In our implementation we set $\eta = 0.05$. Then, the *weighted distance* is defined as

$$d_\delta(f_i, f_j) = \delta \frac{d(f_i, f_j)}{avg_{geo}} + (1 - \delta)\frac{d_a(f_i, f_j)}{avg_{ang}}, \tag{2}$$

where d is the geodesic distance, avg_{geo} and avg_{ang} are the average geodesic and angular distances between couples of adjacent faces respectively, and $0 \leq \delta \leq 1$ is a parameter controlling the relative weight of the two different distances. After growing the patches according to this metric, their shape is finally refined by iteratively recentering the seeds in their patches.

The weighted distance is expected to produce patches whose borders align with meaningful shape features, according with the *mimima rule* [HR84]. Indeed, the patches tend to be large in flat and convex areas, whereas their borders tend to be located around concave areas (Figure 3). We set $\delta = 0.1$ in our implementation, to emphasize the contribution of the angular distance, while avoiding to get stuck in areas rich of geometric details.

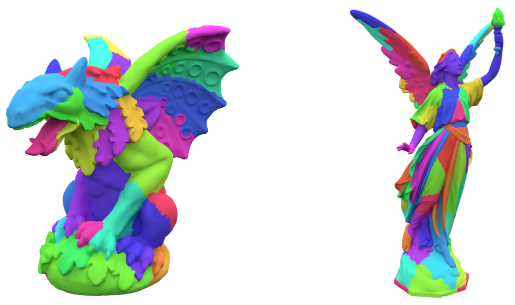


Figure 3: *The patches in the oversegmentation of the Gargoyle (a) and Lucy (b) models. The patches are larger in flat areas, while their borders tend to align with surface features. Therefore, the patch borders constitute a good set of candidate cuts.*

4.2. Initial segmentation

Given the set of N patches $\mathcal{P}_j$, $j = 1 \ldots n$, our aim is to merge those that can be 3D printed in the same direction, to obtain a segmentation with a smaller number of pieces. In other words, we have to assign each patch with one and only one of h candidate printing directions d_i, $i = 1 \ldots h$, which are uniformly sampled on the unit sphere ($h = 512$ in our implementation). Then, the segmentation boundaries will be defined by patch boundaries that separate adjacent patches with different labels.

We formulate the problem as a labeling Integer Linear Program (ILP), where the data points are patches and the discrete labels are candidate directions. The objective function measures the cost of assigning each patch to a given printing direction (*data cost*), and the cost of cuts between patches with different labels (*smoothing cost*). The cost of a patch for a direction is proportional to the support-affected area for that direction; the cost of cuts is proportional to their length and a pre-defined cut width. Both costs are weighted according to the placement of either supports or cuts over visible or occluded areas.

Formally, we formulate the problem as finding the values of the binary variables b_{ij} which globally minimize the function

$$\sum_{i=1}^{h}\sum_{j=1}^{N} w_{ij} b_{ij} + \sum_{i=1}^{h}\sum_{(k,s)\in I} a_{ks}(y_{ks})_i \tag{3}$$

where

$$b_{ij} = \begin{cases} 1 & \text{if patch } \mathcal{P}_j \text{ is assigned to direction } d_i \\ 0 & \text{otherwise} \end{cases}$$

subject to the constraint

$$\sum_{i=1}^{h} b_{ij} = 1 \quad \forall j = 1,\ldots,N$$

to have an assignment problem.

The coefficients w_{ij}s in the first term of Equation 3 (the *data cost*) measure the area of faces which are either supported or footings of supports for patch $\mathcal{P}_j$ and printing direction d_i, weighted by the ambient occlusion of the faces (see Section 4.2.1 for details).

The second term of Equation 3 (the *smoothing cost*) is a pairwise cost penalizing both adjacent patches being assigned to different printing directions, and cuts being placed over visible regions. I is the set of unordered couples (k,s) representing the indices of adjacent patches. The $(y_{ks})_i$s are binary auxiliary variables such that

$$(y_{ks})_i = \begin{cases} 1 & \text{if patches } \mathcal{P}_k \text{ and } \mathcal{P}_s \text{ are adjacent and only} \\ & \text{one of them is assigned to the direction } d_i \\ 0 & \text{otherwise.} \end{cases}$$

The $(y_{ks})_i$s are not variables in the problem, as they are completely determined by the values of b_{ij}, being $(y_{ks})_i = b_{ik}$ XOR b_{is}.

The coefficients (a_{ks})s depend on the length of the boundary between patches $\mathcal{P}_k$ and $\mathcal{P}_s$, and on the mesh ambient occlusion of the boundary location, which are both independent of the printing direction (see Section 4.2.2 for details).

4.2.1. Coefficients for the data cost

The weights w_{ij} in the first term of the ILP formulation (Equation 3) define the *cost* of the patch $\mathcal{P}_j$ for the direction d_i. Since we aim to reduce the visual artifacts due to supports, we define the cost as the area of supported regions, weighted by their ambient occlusion to avoid the placement of supports in visible areas.

The elements needing support, that are the faces which have no supporting geometry in the previously printed layer, include:

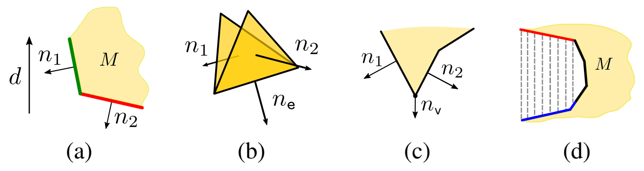


Figure 4: *Elements needing supports: (a) the red face is overhanging with respect to the printing direction d, as its normal n_2 forms an angle with d that exceeds the printer tolerance angle, while the green face with normal n_1 is not; (b) though both faces are not overhanging, their shared edge is; (c) similarly, non-overhanging faces around minima vertices (with down-facing normal) need supports; (d) the effects of supports, on the red faces, extend to footing blue faces where supports grow from.*

a) overhanging faces, i.e., faces whose normal forms an angle with the base which exceeds the printer tolerance angle (55° in our implementation; Figure 4a);
b) faces with overhanging edges (Figure 4b);
c) non-overhanging faces around minima vertices (Figure 4c);

Moreover, additional faces affected by supports include the faces the supports grow from, namely:

d) *footing* faces, computed as the faces first intersected by rays shot from supported faces in the direction opposite to the printing direction (Figure 4d).

We do not consider the impact of supports in the inside of the shell, as the eventual artifacts will be hidden once the object is assembled.

Then, the weight w_{ij} for patch $\mathcal{P}_j$ and printing direction d_i is computed as

$$w_{ij} = \sum_{f\in P_j} c_i(f) \tag{4}$$

with the cost $c_i(f)$ of face f for direction d_i defined as

$$c_i(f) = \begin{cases} area(f)\cdot AO^{\alpha}(f) & \text{if } f \text{ is affected by supports along } d_i \\ 0 & \text{otherwise} \end{cases}$$

where $area(f)$ and $AO(f)$ denote the area and ambient occlusion of face f, respectively. Since ambient occlusion measures the exposition of a surface area to ambient lighting, it gives us information about how visible that area is. So, occluded areas are good candidate locations for supports. The ambient occlusion, computed using the libigl library [JP*18], ranges in the interval $[0,1]$, with 0 standing for completely occluded and 1 for completely visible (Figure 5); we set the exponent $\alpha = \frac{1}{2}$ in our experiments.

4.2.2. Coefficients for the smoothing cost

Optimizing with respect to supports only could still lead to over-segmented meshes, and long, visible cuts. Since we want the cuts to be less invasive as possible, we add a regularizer which promotes smooth cuts and minimal length for the overall segmentation boundary, thus implicitly reducing the number of pieces as well. Indeed, the second term in Equation 3 penalizes neighboring patches

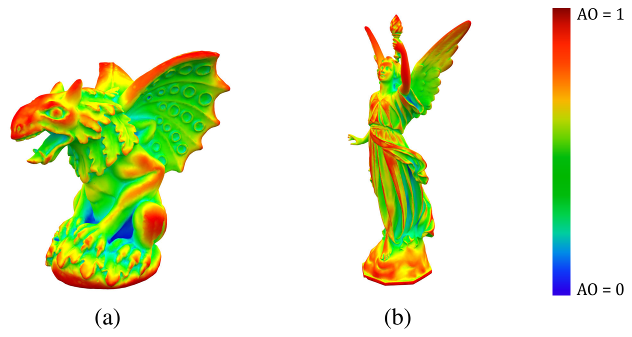


Figure 5: *(a) The ambient occlusion on the Gargoyle (a) and Lucy (b) model, ranging from 0 (blue, occluded areas) to 1 (red, exposed areas). As the ambient occlusion measures the exposition to ambient lighting, it gives information about the visibility of different surface areas. Hence, it is a cue for the visual impact of artifacts over different areas.*

being assigned to different printing directions, with weights proportional to the boundary length. We also keep weighting with the ambient occlusion of edges, so that the most hidden cuts are preferred among the candidate ones in the oversegmentation.

Formally, the weights a_{ks} in Equation 3 are defined as:

$$a_{ks} = t \cdot \sum_{e \in E_{ks}} length(e) \cdot AO^{\alpha}(e) \tag{5}$$

where t is a pre-defined cut width value, E_{ks} denotes the boundary between patches $\mathcal{P}_k$ and $\mathcal{P}_s$, $length(e)$ is the length of the boundary edge e, and $AO(e)$ denotes its ambient occlusion. Again, ambient occlusion helps hiding cuts in less visible regions; we set $\alpha = 0.5$ as in Equation 4.

The width t is the main parameter controlling our algorithm. It stands for the *physical* width of the area around the cut whose aesthetic appearance we consider to be affected. Therefore, it modulates the impact of cuts, and consequently the number of pieces of the decomposition: larger values will result in a segmentation with a lower number of pieces, whereas smaller values will result in more fragmented segmentations. For the examples shown in Figures 1, 2 and 6 we used $t = 3$mm.

4.2.3. Solving the ILP

Finally, the problem in Equation 3 is solved using Gurobi [GO19]. Timings are reported in Table 1 in Section 5. Notice that, for a mesh with n faces, we solve the problem using N patches as variables, with $N \ll n$. Therefore, we can easily deal with meshes with a relatively high number of faces (n is between $67K$ and $220K$ for the models we experimented with).

The result from the solver is a segmentation which satisfies the conditions stated at the beginning of Section 3, namely, a small number of pieces with reduced supported areas and separated by meaningful, non-invasive cuts (Figure 6a). This is thanks to the combined use of the shape-aware metric in Equation 2 – which encourages the patch borders to follow the object features – and the ambient occlusion – which suggests preferred locations for cuts. Since the cuts at this stage depend on the patch borders, they can still be noisy (Figure 6a, close-ups). Therefore, in the third and final step of our pipeline, we refine the cuts by solving a similar ILP problem, where the variables are single mesh faces, but the domain is limited to small *fuzzy regions* around the cuts, as detailed below.

4.3. Refined segmentation

We define *fuzzy regions* around the cuts in which cuts can move across all mesh faces, so as to smoothen and hide them from view as much as possible. The fuzzy regions include all mesh faces whose geodesic distance from the cuts is less than a threshold ε_{fuzzy}, with ε_{fuzzy} depending on the object size. The threshold controls the size of the fuzzy regions: we seek a value that renders this domain sufficiently small not to slow down the solver, yet is big enough for the cuts to freely deform. In our implementation we set $\varepsilon_{fuzzy} = 2\%$ of the object bounding box diagonal (Figure 6c).

Again, we formulate the problem using ILP, for each connected component of the fuzzy regions. In this phase, we only consider in the objective function the smoothing cost, since we are only interested in a local refinement of cuts. The function takes the form

$$\sum_{i=1}^{l} \sum_{(k,s) \in \hat{I}} \hat{a}_{ks} (\hat{y}_{ks})_i \tag{6}$$

where the l printing directions are those among the original candidate directions which have at least one face assigned in the fuzzy component; $\hat{I}$ is the set of unordered couples representing the indices of adjacent mesh faces (instead of patches as in the initial segmentation); $\hat{y}_{ks}$ are binary auxiliary variables defined as in Equation 3 but referring to faces f_k and f_s in the fuzzy region, and

$$\hat{a}_{ks} = length(e_{ks}) \cdot (\exp(\lambda \cdot AO(e_{ks})) - 1) \tag{7}$$

where e_{ks} is the edge shared by f_k and f_s. Here the exponential function is used to reshape the ambient occlusion function so as to push cuts towards highly-occluded regions ($\lambda = 4$ in our implementation). Finally, to further refine the cuts, we perform a smoothing operation which makes the cut polyline cross triangles (Figure 6b), as implemented in the VCG library [Vis19].

4.4. Fabrication and assembly

Once we have the segmented pieces, we build their corresponding shells. We first compute a tetrahedralization of the object whose boundary conforms with the original surface mesh using TetGen [Si15]. Then, we propagate the surface partition inside the tetrahedralized volume, following the approach in [AMG*19], and get a set of solid pieces. After performing a Laplacian smoothing on the interface surfaces between solid pieces, we subtract the insets to solid pieces via Boolean operations, to get the final shell pieces (Figure 7).

The shell pieces are 3D printed along their optimized printing direction, as defined by the ILP solution. Then, the pieces are assembled and glued together to get the final object.

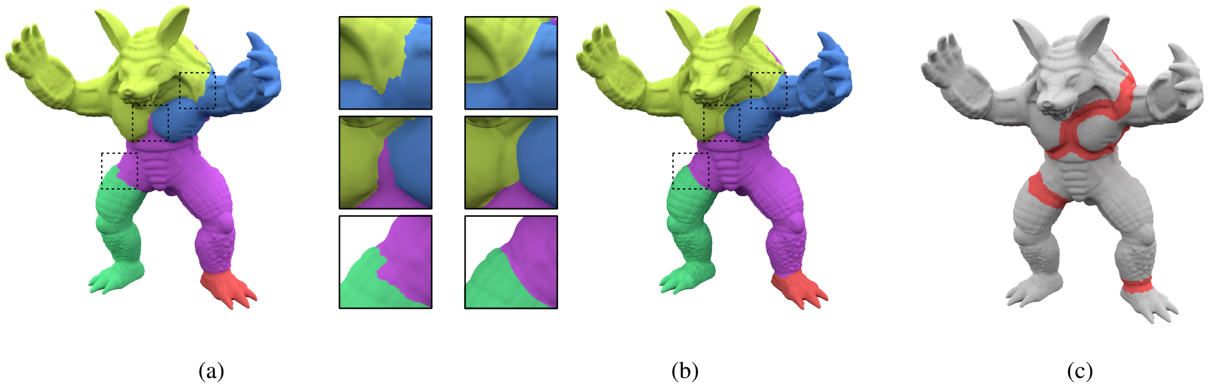


Figure 6: *(a) The initial segmentation of an Armadillo model. Though the cuts are smartly located, they can still be noisy; (b) the refined segmentation with smoother and better located cuts, after the cut optimization in small fuzzy areas around their original location (c).*

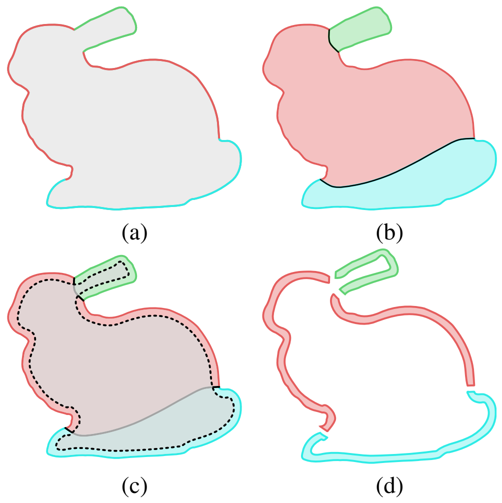


Figure 7: *The shell generation process (2D visualization): (a) we compute a surface conforming tetrahedralization, (b) we propagate the surface partitioning inside the volume; we extract interface surfaces (black lines) on the boundaries between partitions, obtaining a set of solid parts; (c) we then compute an inset surface of the object and (d) through boolean operations we subtract it from the solid parts, obtaining the corresponding shell pieces.*

4.5. Discussions

The results of our algorithm mainly depend on three factors: the width value t in Equation 5, the size of the object to print, and the presence of additional constraints. The role of each factor is discussed below.

Dependence on the width value The main parameter controlling our technique is the cut width t (Section 4.2.2), which modulates the relative impact of supports and cuts on the printed object. Large values of t result in segmentations in a small number of pieces with possibly larger supported areas, whereas small values produce more fragmented segmentations in less supported pieces (Figure 8). Therefore, the user is left free to choose the parameter according to its needs and preferences. Additional examples are shown in Section 5, Results.

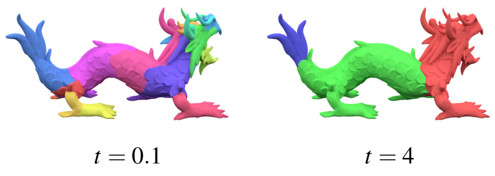


Figure 8: *Segmentation of the Dragon with different values for the cut width parameter t, which denotes the width of the area around the cut that we consider to be affected by the cut itself. The parameter controls the number of pieces in the segmentation, and therefore the balance between the presence of cuts and the supported area.*

Dependence on size. The value of the cut width parameter t is set in millimeters. Therefore, the same value will have a different result depending on the desired size of the fabricated model: fewer cuts on smaller objects, more cuts on larger objects (Figure 9). The rationale is that a higher number of cuts only makes sense for bigger objects, whereas smaller objects are better printed and assembled with fewer pieces.

Optional additional constraints The energy in the ILP formulation can be easily customized by setting pre-defined costs for specific faces/edges. For example, one could set the face cost $c_i(f)$ (Equation 4) to 0 for a surface portion to imply that artifacts over that portion would not be visible, thus favouring solutions which place supports there. This may make sense for example for objects with a large flat base. Figure 10 shows the segmentation results with and without considering the flat base on the Gargoyle model. The base faces can be either manually selected by the user, or automatically detected by analysing the co-planarity of faces and the object up-right direction [FCODS08].

Model	**# Faces**	**Bounding Box Size** (*mm*)	**S1**	**S2**	**S3**	**# Pieces**	Figure
Armadillo	177k	$127 \times 151 \times 115$	1681	2513	25	5	Figure 11
Bunny	111k	$112 \times 111 \times 86$	704	2572	4	4	Figure 11
Dragon	121k	$265 \times 118 \times 176$	714	1229	78	6	Figure 11
Gargoyle	67k	$87 \times 94 \times 67$	379	790	3	4	Figure 1
Lucy	100k	$186 \times 318 \times 107$	718	975	11	4	Figure 11
Alwin	200k	$137 \times 144 \times 67$	2877	4110	2110	13	Figure 12
Dancing Children	120k	$226 \times 167 \times 124$	790	1053	6	8	Figure 12
Thundercrab	155k	$133 \times 123 \times 120$	2200	360	298	11	Figure 12
Vidri	201k	$67 \times 150 \times 61$	3073	3503	9	2	Figure 12
Wartortle	220k	$142 \times 199 \times 145$	1395	2107	2928	10	Figure 12
Running Armadillo	152k	$179 \times 151 \times 176$	957	1669	33	4	Figure 13

Table 1: *Statistics for our example models: number of mesh triangles; dimension of the bounding box; timings (in seconds) for the different steps of our pipeline: (S1) oversegmentation, (S2) initial segmentation, (S3) refined segmentation; the number of pieces of the decomposition.*

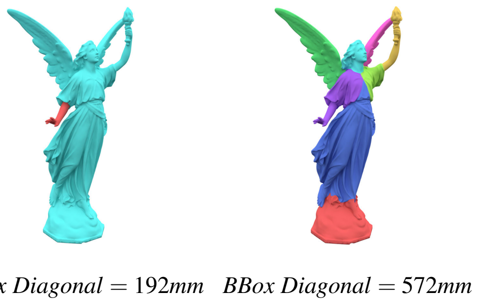


Figure 9: *Segmentation of the Lucy model with different dimensions for the printed object, using the same parameter $t = 3mm$. For the smaller Lucy (left), few pieces are preferred. More pieces are allowed for the bigger Lucy (right), as the pieces would be easier to 3D print and assemble.*

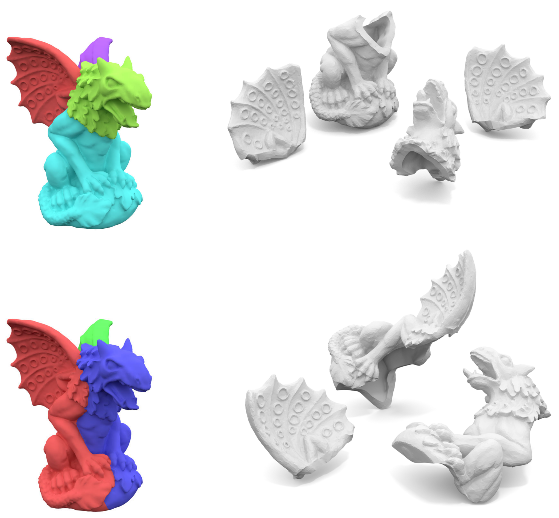


Figure 10: *Decomposition and printing of the Gargoyle with (top) and without (bottom) setting to* 0 *the cost of the base faces. In the first case, the ILP solution places supports all over the base, while in the second case, a vertical cut is preferred. The shell pieces are displayed according to their optimized printing direction.*

5. Results

We successfully segmented and fabricated a number of objects. The shell pieces were printed using PLA on different 3D printers (Ultimaker 2+, Ultimaker 5S). Before the assembly steps, the shell pieces were cleaned by removing support residues, and lightly sanding the shell borders to ease adhesion between connecting pieces. Indeed, given the manufacturing tolerance of home-use 3D printers, the interface surfaces between pieces are not guaranteed to have a perfect fit. The pieces were glued with superglue. Finally, the small gaps between pieces were filled with superglue and baking soda (a technique used in model making). Sprue removal, light sanding and gaps filling are the standard steps when assembling multi-part plastic kits for hobby, scale models and do-it-yourself crafts.

Table 1 reports the statistics about the models and timings to complete the entire computational pipeline, using unoptimized code, on a regular desktop machine (Intel Core i7-6700K 4GHz). We also point out that our primary goal was appearance preservation, and not printing time and material saving.

Figure 11 shows the 3D printed and assembled models, whereas Figure 12 shows the segmentations of additional models. For all models, we used the empirically selected value $t = 3mm$, which stroke a good balance between assembly and need for supports.

Figure 13 shows a comparison between previously proposed methods and our approach. While in [HLZCO14, KFW19, WZK16] (Figures 13a, 13b and 13c respectively) the authors focus on the definition of planar cuts to decompose the object into 3D printable support-free parts, cutting along visible areas, our method realizes a good tradeoff between the aesthetic appearance of the cuts and the minimization of the overall visual impact of cuts and supports combined. Moreover, while the authors of [WZK16] (Figure 13c) only locally refine the cuts once they are computed, our method accounts for cuts visibility from the very beginning, therefore it is able to generate cuts that better hide into occluded regions of the object.

Model	A_{whole}	A_{seg} $(+cut)$	A^{AO}_{whole}	A^{AO}_{seg} $(+cut)$
Armadillo	8.5%	2.5% (5.5%)	5.9%	1.7% (3.7%)
Bunny	6.4%	0.8% (4.2%)	5.8%	0.7% (3.0%)
Dragon	8.4%	6.7% (7.6%)	5.3%	4.1% (5.0%)
Gargoyle	11.7%	10.4% (13.6%)	7.3%	2.1% (3.2%)
Lucy	10.4%	2.4% (3.5%)	3.8%	1.2% (1.5%)
Alwin	13.7%	3.5%(10.5%)	7.8%	0.9%(2.6%)
D. Children	8.0%	2.0% (4.7%)	5.7%	1.2% (3.3%)
T. Crab	12.7%	2.3% (6.6%)	8.6%	1.2% (3.1%)
Vidri	5.7%	4.9% (6.7%)	3.8%	2.7% (3.4%)
Wartortle	16.7%	7.5%(11.2%)	8.7%	3.9%(5.1%)
R. Armadillo	9.3%	1.8%(2.7%)	7.6%	1.3%(1.9%)

Table 2: *A_{seg} and A_{whole} are the supports affected areas for the segmented and whole object, printed along the direction minimizing its supported areas, while A^{AO}_{seg} and A^{AO}_{whole} denote the same measures weighted by ambient occlusion. All quantities are normalized by the total surface area. The values between brackets include the additional impact of cuts affecting the surface using a width $t = 3mm$. It can be seen that the areas affected by supports are considerably smaller for the segmented objects, especially when considering their weighted visibility cost.*

To quantitatively demonstrate how we balance the presence of cuts and supports, Table 2 reports for each segmented model the supported area A_{seg} (computed as the sum of the areas of supported faces) and supported area weighted by ambient occlusion A^{AO}_{seg}, both normalized with the total surface area to remove scale dependency. For a fair evaluation, the measures A_{whole}, A^{AO}_{whole} are also reported for the model printed as a whole along its optimal printing direction, i.e. the direction that minimizes the supported area. Figure 14 shows a visual comparison between an object printed as a whole and the same object printed using our proposed method.

Finally, Figure 15 illustrates the effects of various choices for the main parameter of our technique, the cut width t, from smaller to larger values. Small values of t produce segmentations into a possibly high number of pieces with a smaller fraction of supported areas (up to support-free pieces), under the assumption that cuts are preferred over supports. On the contrary, large values of t induce segmentations in a smaller number of pieces, possibly with larger supported areas, under the assumption that cuts are to be avoided in favour of supports. Therefore, a tradeoff value needs to be defined depending on the user needs.

6. Conclusions

We devised a method to automatically segment 3D objects into parts which can be 3D printed individually via fused deposition modeling (FDM), and then reassembled by preserving the visual quality of the final object. Our aim was to make it easier for people to print high-quality objects at home, even objects of complex shape. Therefore, we defined a segmentation technique which takes into account by design both the printability of parts and the aesthetic impact of cuts on the printed surface. We segmented, fabricated and assembled a number of objects, with diverse shapes and sizes, and demonstrated how we succeed in reducing the visual impact of gaps between pieces. Our method relies on the assumption that objects have features where to hide cuts and supports. As such, the method is mainly tailored to geometrically complex objects.

A limitation of our technique is that we do not take explicitly into account the assemblability of pieces, in terms of assembly order and part gluing, which could be difficult in case of parts with small contact interfaces. We did not encounter difficulties in assembling the models we fabricated, since our method strives for a small number of pieces and smooth interfaces between them. Though, it is still possible that for complex shapes our technique produces pieces which are difficult to assemble. The definition of an assembly criterion and assembly order to avoid collisions between pieces has been addressed in [ACA*19], and their approach could be adapted to our scenario. We also plan to add connectors inside the shell pieces, which would aid the gluing and assembly process.

Another limitation of our technique is that we do not impose explicit constraints on the maximum size of pieces. Whereas the dimension of pieces is partly regulated by the cut width parameter, which determines the number of pieces in the decomposition, there are no explicit guarantees that each piece would fit for example into the printing chamber, or into a box for better packing. A possible solution would be to impose additional constraints to prevent merging patches according to their size.
Additional improvements could include the use of perceptual metrics to further assess the impact of cuts [CLL*13] and the inclusion of terms depending on structural and mechanical properties [ZPZ13].

Acknowledgements

The Armadillo, Lucy, Dragon and Bunny models are courtesy of the Stanford 3D Scanning Repository. The Gargoyle and Dancing Children models are courtesy of the AIM@SHAPE Shape Repository. The Thundercrab model by Dan Pressman is available at www.thingiverse.com/thing:50207. The Vidri model is courtesy of Martin Nikolov (https://mnikoloff.com/), the Wartortle model is courtesy of Mickael Lelièvre (https://lelievremickael.artstation.com/), the Alwin model is courtesy of Thomas Lishman (https://tlishman.artstation.com/). This research was partially funded by the EU H2020 Programme EVOCATION: Advanced Visual and Geometric Computing for 3D Capture, Display, and Fabrication (grant no. 813170), and by the Italian PRIN project DSURF (grant no. 2015B8TRFM).

References

[ACA*19] Araújo C., Cabiddu D., Attene M., Livesu M., Vining N., Sheffer A.: Surface2volume: Surface segmentation conforming assemblable volumetric partition. *ACM Trans. Graph. 38*, 4 (July 2019). doi:10.1145/3306346.3323004. 9

[AMG*18] Alderighi T., Malomo L., Giorgi D., Pietroni N., Bickel B., Cignoni P.: Metamolds: Computational design of silicone molds. *ACM Trans. Graph. 37*, 4 (July 2018), 136:1–136:13. doi:10.1145/3197517.3201381. 3

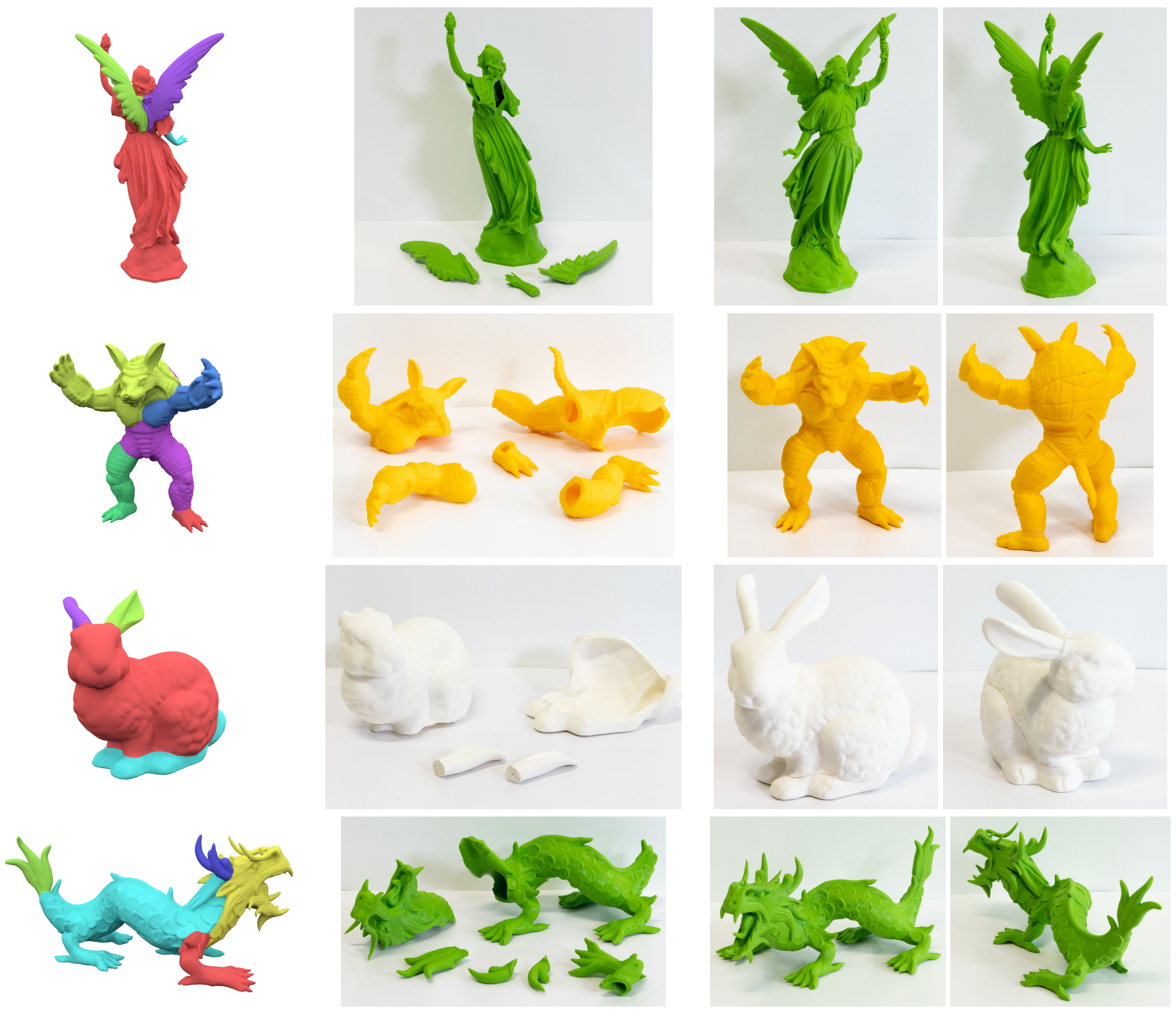


Figure 11: *Some successful examples of segmented, fabricated and assembled objects: (a) segmentations; (b) 3D printed and oriented shell pieces; (c) the assembled model. It can be seen how cuts tend to hide into the object features.*

[AMG*19] ALDERIGHI T., MALOMO L., GIORGI D., BICKEL B., CIGNONI P., PIETRONI N.: Volume-aware design of composite molds. *ACM Trans. Graph. 38*, 4 (July 2019), 110:1–110:12. doi:10.1145/3306346.3322981. 6

[Att15] ATTENE M.: Shapes in a box: Disassembling 3d objects for efficient packing and fabrication. *Computer Graphics Forum 34*, 8 (2015), 64–76. doi:10.1111/cgf.12608. 2

[BM17] BAUDISCH P., MUELLER S.: Personal fabrication. *Foundations and Trends® in Human Computer Interaction 10*, 3âĂŞ4 (2017), 165–293. doi:10.1561/1100000055. 1

[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. doi:10.1109/TVCG.2012.34. 4

[CLL*13] CORSINI M., LARABI M. C., LAVOUÉ G., PETŘÍK O., VÁŠA L., WANG K.: Perceptual metrics for static and dynamic triangle meshes. *Computer Graphics Forum 32*, 1 (2013), 101–125. doi:10.1111/cgf.12001. 9

[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 Trans. Graph. 34*, 6 (Oct. 2015), 213:1–213:12. doi:10.1145/2816795.2818087. 2

[DWW*18] DAI C., WANG C. C. L., WU C., LEFEBVRE S., FANG G., LIU Y.-J.: Support-free volume printing by multi-axis motion. *ACM Trans. Graph. 37*, 4 (July 2018), 134:1–134:14. doi:10.1145/3197517.3201342. 3

[FCODS08] FU H., COHEN-OR D., DROR G., SHEFFER A.: Upright orientation of man-made objects. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 42:1–42:7. doi:10.1145/1360612.1360641. 7

[GO19] GUROBI OPTIMIZATION L.: Gurobi optimizer reference manual, 2019. URL: http://www.gurobi.com. 4, 6

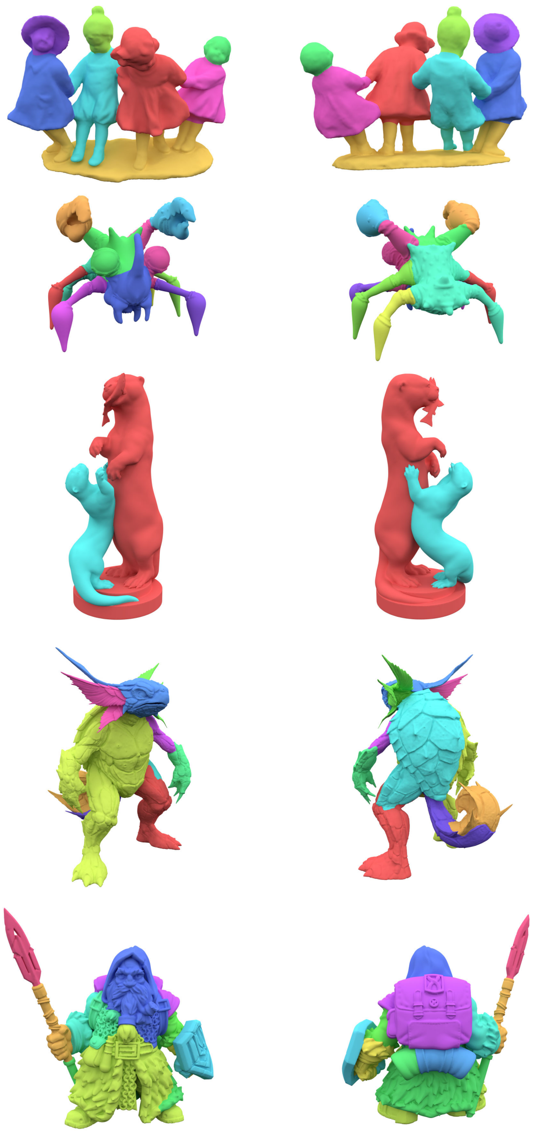


Figure 12: *Additional segmentation examples, showing how cuts are placed along object features. From top to bottom: Dancing Children, Thundercrab, Vidri, Wartortle, Alwin.*

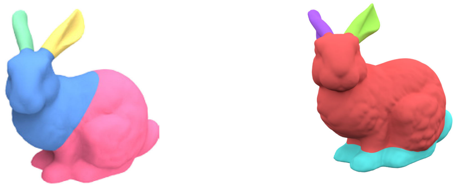
[HLZCO14] HU R., LI H., ZHANG H., COHEN-OR D.: Approximate pyramidal shape decomposition. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 213:1–213:12. doi:10.1145/2661229.2661244. 3, 8, 11

[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. doi:10.1111/cgf.12556. 2, 3

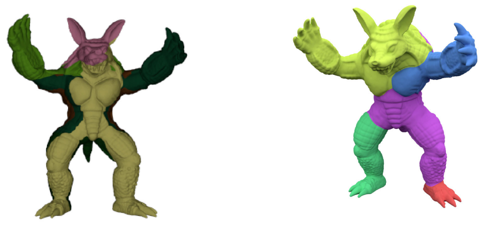
[HR84] HOFFMAN D., RICHARDS W.: Parts of recognition. *Cognition 18*, 1 (1984), 65 – 96. doi:10.1016/0010-0277(84)90022-2. 2, 4

[JP*18] JACOBSON A., PANOZZO D., ET AL.: libigl: A simple C++ geometry processing library, 2018. URL: https://libigl.github.io/. 5

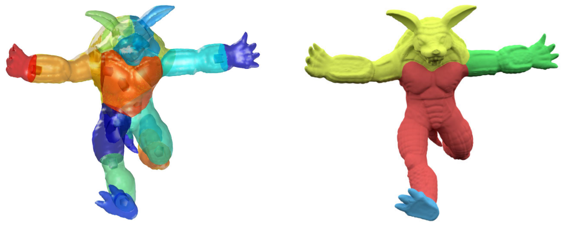
[KFW19] KARASIK E., FATTAL R., WERMAN M.: Object partitioning



(a) Bunny segmentation in [HLZCO14] and ours.



(b) Armadillo segmentation in [KFW19] and ours.



(c) Running Armadillo segmentation in [WZK16] and ours.

Figure 13: *Comparison between the results of previous methods (left) and our approach (right). Our cuts are better hidden thanks to their non-planarity and positioning along occluded areas.*

for support-free 3d-printing. *Computer Graphics Forum 38*, 2 (2019), 305–316. doi:10.1111/cgf.13639. 3, 8, 11

[KT03] KATZ S., TAL A.: Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM Trans. Graph. 22*, 3 (July 2003), 954–961. doi:10.1145/882262.882369. 2, 4

[MLS*18] MUNTONI A., LIVESU M., SCATENI R., SHEFFER A., PANOZZO D.: Axis-aligned height-field block decomposition of 3d shapes. *ACM Trans. Graph. 37*, 5 (Oct. 2018), 169:1–169:15. doi:10.1145/3204458. 2

[PSH*04] POTTMANN H., STEINER T., HOFER M., HAIDER C., HANBURY A.: The isophotic metric and its application to feature sensitive morphology on surfaces. In *Computer Vision - ECCV 2004* (Berlin, Heidelberg, 2004), Pajdla T., Matas J., (Eds.), Springer Berlin Heidelberg, pp. 560–572. doi:10.1007/978-3-540-24673-2_45. 2, 3, 4

[Si15] SI H.: Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw. 41*, 2 (Feb. 2015), 11:1–11:36. doi:10.1145/2629697. 6

[SSS*10] SHAPIRA L., SHALOM S., SHAMIR A., COHEN-OR D., ZHANG H.: Contextual part analogies in 3d objects. *International Journal of Computer Vision 89*, 2 (Sep 2010), 309–326. doi:10.1007/s11263-009-0279-0. 3

[SY96] SONG CHUN ZHU, YUILLE A.: Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18*, 9 (Sep. 1996), 884–900. doi:10.1109/34.537343. 2

[VGB*14] VANEK J., GALICIA J. A. G., BENES B., MĚCH R., CARR N., STAVA O., MILLER G. S.: Packmerger: A 3d print vol-

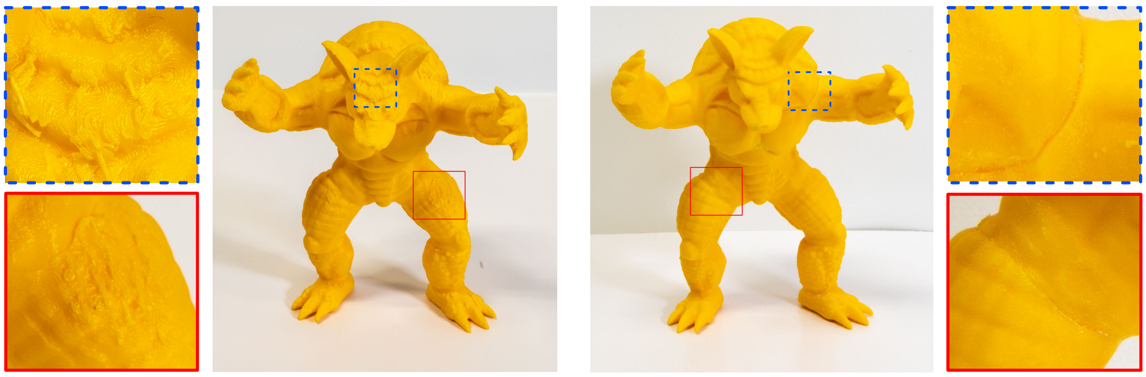


Figure 14: *Comparison between an object printed as a whole, along the direction minimizing A^{AO}_{whole} (left), and the same model printed and glued together following our method (right). We can clearly see the damage caused by supports removal in highly visible and wide areas (left), whereas we both reduce the total supported area and limit the visual impact of seams on the assembled object (right).*

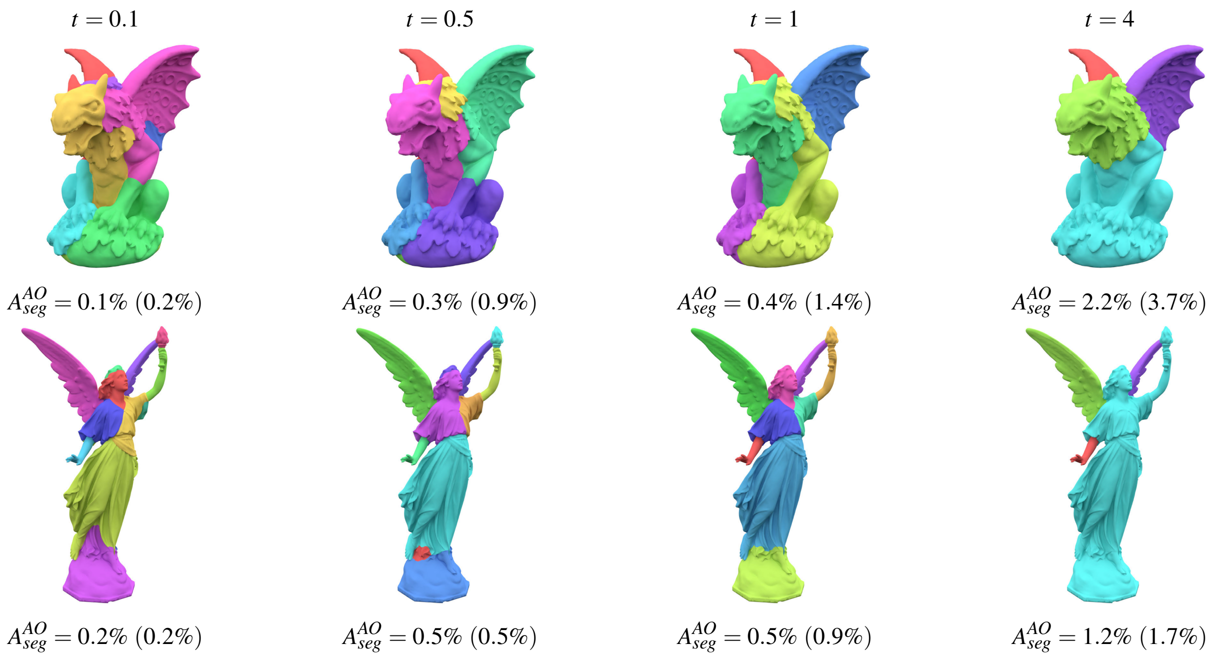

$t = 0.1$
$t = 0.5$
$t = 1$
$t = 4$
$A^{AO}_{seg} = 0.1\%$ (0.2%)
$A^{AO}_{seg} = 0.3\%$ (0.9%)
$A^{AO}_{seg} = 0.4\%$ (1.4%)
$A^{AO}_{seg} = 2.2\%$ (3.7%)
$A^{AO}_{seg} = 0.2\%$ (0.2%)
$A^{AO}_{seg} = 0.5\%$ (0.5%)
$A^{AO}_{seg} = 0.5\%$ (0.9%)
$A^{AO}_{seg} = 1.2\%$ (1.7%)

Figure 15: *Segmentations for increasing values of the cut width t. For small values of t the area affected by supports is very limited, however the number of pieces is high. Conversely, higher values of t produce a smaller number of pieces.*

ume optimizer. *Computer Graphics Forum 33*, 6 (2014), 322–332. doi:10.1111/cgf.12353. 2

[Vis19] VISUAL COMPUTING LAB: The vcg library, 2019. URL: http://vcg.isti.cnr.it/vcglib/. 6

[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. *Comput. Graph. Forum 34*, 6 (Sept. 2015), 148–160. doi:10.1111/cgf.12527. 2

[WQZ*18] WEI X., QIU S., ZHU L., FENG R., TIAN Y., XI J., ZHENG Y.: Toward support-free 3d printing: A skeletal approach for partitioning models. *IEEE Transactions on Visualization and Computer Graphics 24*, 10 (Oct 2018), 2799–2812. doi:10.1109/TVCG.2017.2767047. 3

[WZK16] WANG W. M., ZANNI C., KOBBELT L.: Improved surface quality in 3d printing by optimizing the printing direction. *Computer Graphics Forum 35*, 2 (2016), 59–70. doi:10.1111/cgf.12811. 3, 8, 11

[ZLP*15] ZHANG X., LE X., PANOTOPOULOU A., WHITING E., WANG C. C. L.: Perceptual models of preference in 3d printing direction. *ACM Trans. Graph. 34*, 6 (Oct. 2015), 215:1–215:12. doi:10.1145/2816795.2818121. 2

[ZPZ13] ZHOU Q., PANETTA J., ZORIN D.: Worst-case structural analysis. *ACM Trans. Graph. 32*, 4 (July 2013). doi:10.1145/2461912.2461967. 9

[ZY96] ZHU S. C., YUILLE A.: Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. 18*, 9 (Sept. 1996), 884–900. doi:10.1109/34.537343. 3