Orthogonal Slicing for Additive Manufacturing

Abstract

Most additive manufacturing technologies work by layering, i.e. slicing the shape and then generating each slice independently. This introduces an anisotropy into the process, often as different accuracies in the tangential and normal directions, but also in terms of other parameters such as build speed or tensile strength and strain. We model this as an anisotropic cubic element. Our approach then finds a compromise between modeling each part of the shape individually in the best possible direction and using one direction for the whole shape part. In particular, we compute an orthogonal basis and consider only the three basis vectors as slice normals (i.e. fabrication directions). Then we optimize a decomposition of the shape along this basis so that each part can be consistently sliced along one of the basis vectors. In simulation, we show that this approach is superior to slicing the whole shape in one direction, only. It also has clear benefits if the shape is larger than the build volume of the available equipment.

1. Introduction

Additive manufacturing techniques usually add layer after layer for fabricating a shape. Depending on the underlying process this introduces *direction bias*. The most obvious example for such bias is a different accuracy along the normal direction to a layer and the tangent directions. There are other factors that make the distinction of the directions worthwhile: different tensile strength or strain [1] (i.e. one can increase the stability of the model by choosing the right orientation in each part), different build time [2] (one can save production time by orienting the parts differently), different amounts of support material (i.e. one can save cost / waste by orientating different parts differently), or simply different dimensions of the build volume.

As a running example for our work we focus on the issue of accuracy. While our approach can be generalized to all layered manufacturing methods from 2D slabs laser cutting to high resolution 3D prints, we wish to stress that the improvements one can get from slicing one object into different directions may depend on its scale, the size of the object, and the desired application. The benefits of our method show in particular

- with increasing thickness of layers for laser cutting cardboard or plywood and low resolution 3D prints (i.e. high anisotropy of accuracy), or
- for large objects that cannot be fabricated as a whole because they do not fit the fabrication space.

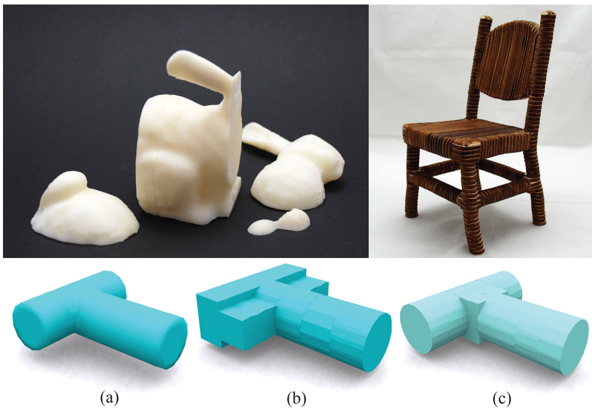

Figure 1: Top: We present a framework that improves additive manufacturing methods across different scales: 3D printing resolutions (left) medium to large scales that might exceed the machine manufacturing volume (right). Bottom: (a) T-shaped object. (b) sliced in one direction (c) decomposed into two partitions after optimization and sliced in two directions.

Additionally multi-material objects that cannot be printed in one run because of the printer limitations or puzzles that are made to be of pieces for manually assembly are interesting applications for our method.

Our goal is to decompose the shape into few pieces so that each piece can be consistently sliced with small geometric error – and that by assembling the pieces one gets a replica with overall small error (see Figure 1). The corresponding optimization problem needs to avoid both extremes: we assume that using one direction is not flexible enough, creates large error, or would simply be

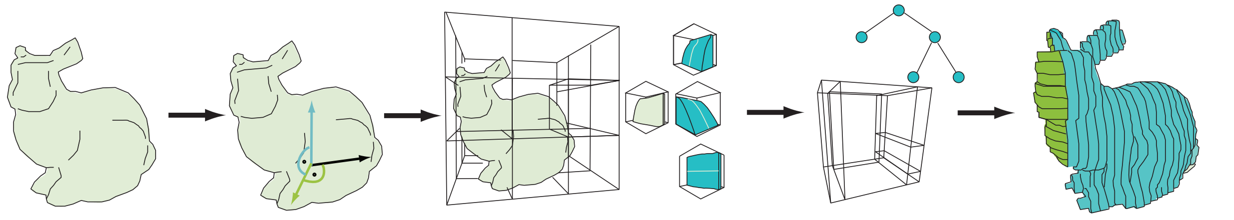

Figure 2: Overview of our method. We start with the input mesh and compute a set of orthogonal manufacturing directions. Afterwards we use these directions to perform a voxelization process where we divide our data into volumetric elements with the size of the width of the material. For each voxel we compute three slicing errors along the directions. We then employ an optimization process to partition the voxel grid. We search for possibly large segments with minimal slicing error while balancing the number of partitions. Our computed segmentation can be cut with a laser cutter or printed with a 3D printer.

impossible; while decomposition into many pieces can clearly make the error small, but the assembly becomes tedious or virtually impossible.

In early experiments, we found that with increasing layer thickness (e.g. > 0.5 mm) partitioning an object using non-orthogonal aligned cuts and printing the parts from their optimal direction would not fit perfectly when the pieces are connected along their aliased direction (resulting in a 'jaggy surface'). The assembly would be difficult, often resulting in connections that cannot be glued together properly. While orthogonality could be achieved locally for some cuts we suggest to solve this problem globally.

Our first modeling decision for this work is, consequently, to *restrict the slicing directions* as well as the normals of the cutting planes to an *orthogonal basis* $B = [\mathbf{b}_0 \mathbf{b}_1 \mathbf{b}_2]$, $B^T B = I$. This approach allows selecting for each part independently an optimal slicing direction b_i while guaranteeing planar connection areas without sampling artifacts between parts (see Section 4).

We model the anisotropy in accuracy (or other properties in the process) as small cubic cells with dimension $d \times \frac{d}{N} \times \frac{d}{N}$, i.e. the thickness of a slice is d, while the accuracy in the tangent directions is N times better than the thickness of a slice. With this basic element, the most natural choice for a smallest element with consistent slicing direction is voxel cell of size d^3. Our idea is to pre-process the shape by decomposing it into voxels, and then find for each voxel its optimal slicing direction and a corresponding contour (see Section 5). We note that only voxels containing parts of the shape's boundary vary in their error depending on the direction.

With this information, we optimize a partition of the voxel set along the voxel faces. The goal is to generate large sets of voxels that are processed along the same direction.

2. Related Work

Computer graphics and related fields in engineering have significantly contributed to computational approaches for computer-aided design that are essential tools in todays digital production pipeline. We will focus on a small subset of this work.

2.1. Manufacturing and Fabrication-oriented design

Additive manufacturing methods are well evaluated and analyzed and show in various research approaches that optimization of the layered manufacturing process is essential. A number of methods address the task of finding an optimal orientation of a single part [3], considering surface finish, evaluate the surface roughness and part deposition time [4],[5]. Danjou and colleagues [2] suggest an optimization procedure based on a genetic algorithm to improve the printing orientation. Masood *et al.* [6] show methodologies for computing the correct orientations based on the minimum volumetric error of basic primitives. Most closely related to our orientation optimization method, Reisner *et al.* [7] propose a method of finding an orthogonal frame. However, none of these approaches considers segmenting the model into sub-parts with different orientations.

In a broader context, Luo *et al.* [8] propose a segmentation algorithm to subdivide a mesh into pieces for the purpose of fitting a large model in a smaller 3D printing volume. This specifically focuses on finding structurally sound and aesthetic pleasing cutlines. In contrast, the goal of our work is to propose a framework to optimize the manufacturing process in accuracy.

By design, our method produces parts that can be simply glued together. There are a variety of approaches that generate specialized connectors used for furniture fabrication [9] or for connecting 3D printed parts [8]. We could easily incorporate this into our framework.

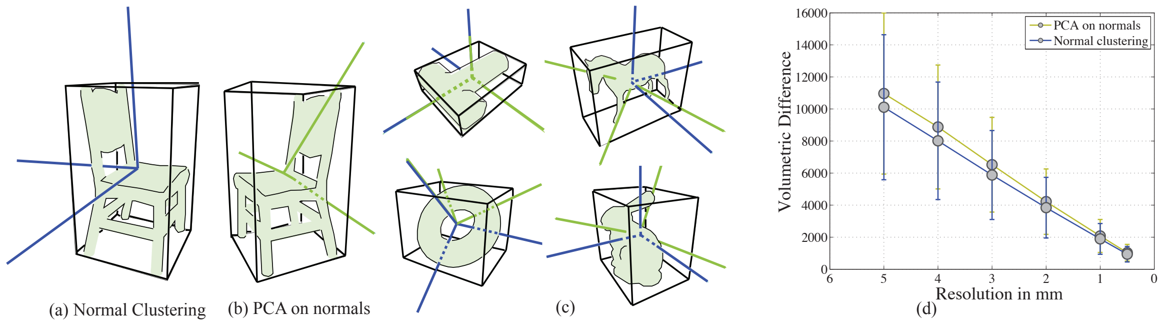

Figure 3: (a) We show the resulting manufacturing directions of the normal clustering (blue) (b) compared to a PCA over the set of normals (green). (c) We show different examples. Note, our method finds an optimal solution for the cylindrical T-shape and that the results of the clustering methods often correspond to the natural upright direction. (d) We evaluated the overall best approximation error over a set of test shapes. We show the mean minimal error and the standard deviation.

2.2. *Slicing and Abstraction*

Planar elements play an essential role in shape analysis, approximation and abstraction. Sellamani *et al.* [10] gathers prominent cross-sections that are also used for mesh segmentation. Recently, [11], [12] and [13] proposed an approach for generating shape abstractions out of a minimal set of planar sections. Décoret *et al.* [14] use billboard clouds as an efficient shape representation. While these approaches rely on mostly unstructured sets of planar elements our proposed framework uses a regular set of stacked layers approximating the shape. Autodesk 123D [15] is able to create custom laser-cut sheets from a 3D shape. In contrast, this approach does not sufficiently take into account the orthogonal fabrication resolution and it is limited to one global slicing direction.

Slicing free-form surfaces was studied in the area of Computer Aided Design for example in the context of finding optimal milling machine paths [16] and [17]. Improving the geometric accuracy of layered manufacturing is proposed by Kulkani *et al.* [18].

3. Overview

Figure 2 illustrates our pipeline to generate partitions that are sliced along good directions.

1. Given an input shape we compute a set of orthogonal directions $B = [\mathbf{b}_0\mathbf{b}_1\mathbf{b}_2]$ that are likely suited for a decomposition of the shape into small parts, each of which can be sliced along one of the three directions with small error (see Section 4). By rotation of the model with B^T we can now consider the canonical directions, i.e. x, y, z.
2. The shape is then decomposed into voxels of size d^3, where d is the desired slice thickness (or, worst accuracy). Each voxel is then decomposed into N^3 sub-voxels, where N is the factor between the thickness of the slice and the accuracy in the tangent directions.
3. For each voxel, the errors for each of the three slicing directions are computed. We use the discrete volumetric difference between the input shape and each of three approximations for a certain direction, computed on the sub-voxel grid. This requires computing approximations that are constant in the direction normal to a slice, yet may vary in tangent direction with the sub-voxel resolution. We explain how to do this consistently for all voxels, yet using only information available in each voxel in Section 5.
4. Based on the per-voxel errors, we compute a decomposition of the voxel grid so that each part can be sliced consistently with small error, yet the total number of pieces remains small. We also consider other factors in this process, such as the maximum size of each part. This process is explained in Section 6.

The result is an orthogonal decomposition of the shape into few pieces, as well as a direction for slicing for each piece.

4. Selection of Manufacturing Direction

Computing a set of orthogonal directions is the first step in our optimization. We base this computation on a simple observation: a planar surface with normal direction $\mathbf{n}$ should be sliced in a direction *orthogonal* to $\mathbf{n}$, because the accuracy in the tangents of a slice is supposed to be significantly higher than normal to a slice.

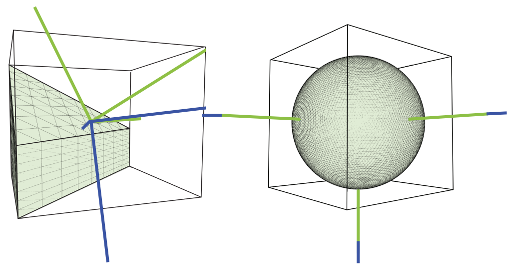

Figure 4: Left: A truncated prism shows that neither PCA directions (green) nor clustered normal directions (blue) are intuitive Right: Directions over the sphere.

Each triangle in the mesh corresponds to a planar piece with area a_i and normal n_i. Our goal is to find three orthogonal directions $B = [\mathbf{b}_0 \mathbf{b}_1 \mathbf{b}_2]$, $B^T B = I$, such that the tangent space $T\mathbf{n}_i = (0,0)^T$ can be well approximated by two of the $\mathbf{b}_c$. Assume these are $\mathbf{b}_0$ and $\mathbf{b}_1$. Then, because B is orthogonal, the normal n_i is well approximated by $\mathbf{b}_2$. This means, it suffices to find an orthogonal B such that all normals are well approximated by one of the $\mathbf{b}_c$.

We first note that it is not sufficient to perform a PCA over the set of the normals. While this gives us one good direction to approximate all normals, it also gives us two more orthogonal directions that are not particular well suited for approximation of normals in the mesh. In other words, the solution is not well balanced.

Our approach is based on clustering of the normals, with the additional requirement of the cluster centers being orthogonal. We start with $B = I$ and then iteratively improve the current solution B as follows:

1. For each normal n find the closest direction $\mathbf{b}_c$ by maximizing $|\mathbf{b}_c \mathbf{n}_i|$. This assigns each normal to one of the three directions and, thus, forms three sets of normal vectors.
2. For each cluster, we perform PCA over the normal vectors *in this cluster* to find the directional center of the cluster. Specifically, let

$$N_c = \sum_{|\mathbf{b}_c \mathbf{n}_i| > |\mathbf{b}_{c'} \mathbf{n}_i|, c' \neq c} a_i \mathbf{n_i} \mathbf{n_i}^T \quad (1)$$

be the covariance matrix of the cluster c. The matrix is symmetric and so has real eigenvalues and orthogonal eigenvectors. We compute the eigendecomposition

$$E_c^T N_c E_c = \Lambda_c \quad (2)$$

use the eigenvector corresponding to the *largest* eigenvalue as the new cluster representative $\tilde{b}_c$.

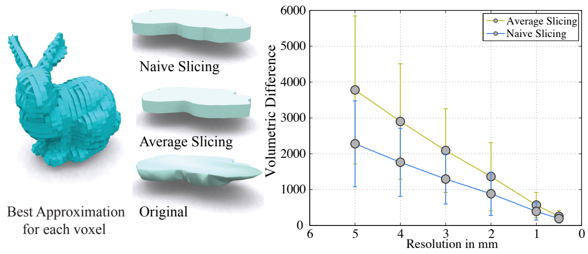

Figure 5: Left: Partitioning the model down to voxel-level would result in the minimal possible error for the manufacturing resolution. Middle: A single slice and two approximations that are constant along the normal to the slice. The first approach intersects the geometry at the center plane of the voxel. We suggest to rather compute average distance values in normal direction of the slice and then extracting the contour as the zero-set of the distance field. Right: the volumetric difference to the original shape as a function of voxel size.

3. The three vectors $\tilde{B} = (\tilde{b}_0, \tilde{b}_1, \tilde{b}_2)$ are generally not orthogonal. We use the SVD to compute the closest orthogonal matrix B from $\tilde{B}$:

$$B = UV^T, \quad \text{where } \tilde{B} = U\Sigma V^T \quad (3)$$

4. We start over with the updated matrix B and repeat until convergence.

Figure 3 shows some results of our method. On the left we illustrate an example of the chair model and the resulting orthogonal basis (a) and the PCA over the set of normals (b). Note that the results of the clustering methods often correspond to the natural upright direction where the PCA averages the normals globally resulting in inefficient fabrication directions. Figure 4 shows that models that do not have a distinguished direction that would result in no error are still meaningful in the sense of minimizing the overall slicing error.

5. Boundary Voxel Optimization

In the following we consider only voxels that are not fully inside (or outside) the shape, i.e. only voxels containing part of the boundary. The slicing direction for all other voxels has no effect of the resulting geometric error. We consider the misclassified (discretized) volume as the approximation error.

Each boundary voxel could be sliced in three directions. We compute the approximation error for all three directions. In the following, we explain the case of slices in the $x-z$ plane, and the y direction is normal to the slice. The other two directions can be computed similarly.

As a first step, we subdivide the voxel into N^3 subvoxels. This follows from the resolution being N times higher in the tangent directions, which are a-priori unknown. For each sub-voxel we compute the signed distance to the original surface. Let the center of a sub-voxel be $\mathbf{s}_{ijk}$ and the corresponding closest point on the surface be $\mathbf{x}(\mathbf{s}_{ijk})$ with surface normal $\mathbf{n}(\mathbf{s}_{ijk})$. Then we get the distance

$$d_{ijk} = sgn\left(\mathbf{n}(\mathbf{s}_{ijk}) \cdot (\mathbf{x}(\mathbf{s}_{ijk}) - \mathbf{s}_{ijk})\right) \|\mathbf{s}_{ijk} - \mathbf{x}(\mathbf{s}_{ijk})\| \quad (4)$$

Our general approach is to compute a new distance function for the $x-z$ plane approximating all distance values in the sub-voxels. Note that this process computes new values for each sub-voxel, so also for the corners, edges, and faces of the voxels. These elements are shared with neighboring voxels. The sign of the value has important topological consequences, namely if a point in space is inside or outside the shape. For topological consistency of the result it is necessary that the signs are identical for shared elements. The only local way to ensure this is to use the same values as input for each resulting value. This means, when we compute a certain value on the $x-z$ plane we can only use the varying y values in this column – and no other sub-voxel in the current voxel. This means, we compute a new distance function $\tilde{d}_{ik} = f(d_{i1k}, \ldots, d_{iNk})$ where we still have freedom in our choice of f. Figure 6 (Left) illustrates how f is evaluated over the distance samples along the y-axis. This defines a new distance field in the x-z plane that is used to extract the final contour over the cell as the zero-set of the field.

The simplest choice would be to pick out a certain value from the column, i.e. $f(\gamma_1, \ldots, \gamma_N) = \gamma_{N/2}$. This is equivalent to intersecting the original geometry with a slice at the height the center of the voxel. We suggest to rather compute a least squares solution for each column, which amounts to taking the average distance value:

$$f(\gamma_1, \ldots, \gamma_N) = N^{-1} \sum_j \gamma_j \quad (5)$$

As we show in Figure 5 this leads to significantly smaller volume differences even in high resolutions.

The new distance field over the x-z slice is extruded along the sub-voxels in y. We define the difference volume $e_\alpha(v)$ between the extruded 3D distance field and its original distances per direction α and voxel v as:

$$e_\alpha(v) = \sum_{ijk} d_{ijk} - \tilde{d}_{ijk} \quad (6)$$

For our optimization we store the difference volumes for each of the three directions for future use.

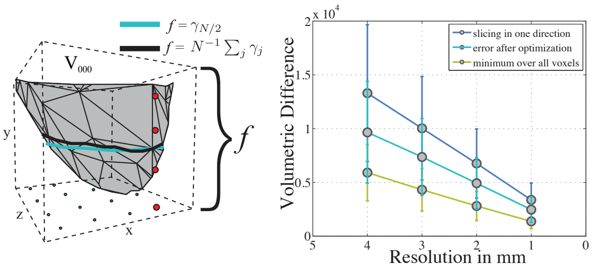

Figure 6: Left: We show a single voxel. f is evaluated along the y direction and influences the illustrated contour along the surface as shown in green and black. Right: We show that partitioning the shape significantly minimizes the volumetric error using just a small set (3-6) of partitions compared to standard one directional slicing.

6. Optimization

It would be possible to fabricate each voxel individually along the direction with smallest error. We call the sum of smallest errors per voxel $e_{\min}$. Figure 6 shows a rendering of this solution for the bunny. Assembling such a model would be very tedious. Simply choosing one direction and slicing all voxels in the same direction is usually far from optimal. Our approach is to rather find a balance between the number of pieces that are sliced consistently and the total volumetric error.

For finding clusters of consistently sliced voxels we chose a decomposition into half-spaces. This has the advantage that the shape can necessarily be assmebled. Note that this is not necessaril true for other decompositions.

We divide the shape by iterating over all possible locations for the split plane. This set is discrete because we consider splitting only between voxel cells. We define a cell Ω (i.e. a box) consisting of voxels and compute the error for a potential split along each of the planes that are consistent with the voxel faces. Our optimization will result in a number of cells, each is sliced along its optimal direction. Let $e_{slc}(\Omega)$ be the minimal error resulting from choosing a consistent slicing direction for a cell. We compute e_{slc} by simply adding the errors of each voxel $e_\alpha(v)$ for the three directions and then taking the minimum sum:

$$e_{slc}(\Omega) = \min_\alpha \sum_{vin\Omega} e_\alpha(v) \quad (7)$$

We define the error function $E_h(\Omega)$ for the cell as

$$E_h(\Omega) = e_{slc}(\Omega) + \mathrm{T}(r) \quad (8)$$

where T defines an additional term for the bound on the

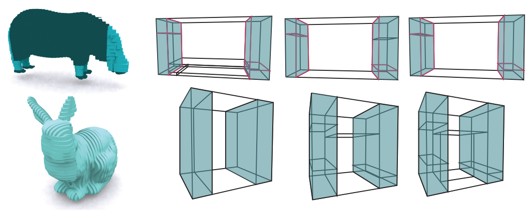

Figure 7: We show that with increasing resolution (5mm, 2mm, 1mm) the partitioning process is stable. Marked are the partitions containing the highest errors over the optimization process.

size of a partition:

$$\mathrm{T}(r_\alpha) = \begin{cases} error_{max} & \text{if } r_\alpha \geq (B(\alpha)) \\ 0 & \text{if } r_\alpha < (B(\alpha)) \end{cases} \qquad (9)$$

where r_α is the size of the bounding box of each partition and $B(\alpha)$ is the maximum production volume size.

The error for a particular plane can be computed from the two errors $E_{h_l}(\Omega)$ and $E_{h_r}(\Omega)$. We add the errors using a particular p-norm, and then minimize the error by trying all possible split planes, leading to

$$E(\Omega) = (E^p_{h_l}(\Omega) + E^p_{h_r}(\Omega))^{1/p}.$$

We solve the global optimization problem using a branch and bound approach in a breadth first manner. The error in each cell is bounded by slicing the whole cell in one direction and then choosing the direction with smallest error which is e_{slc}. We start with Ω as the overall bounding box of the input shape and take the k best options for the choice of the split plane and analyze the next level of splits. In other words, we iterate over all potential splits in Ω computing the errors and keep the best k options in a priority queue. Per iteration each option adds one split to the set of existing cells in its branch. In the next level we analyze the set of cells again finding the best options to advance the next split pruning suboptimal solutions.

This process is repeated until the optimization reaches a maximum number of parts or the total error gets below the threshold

$$L = e_{min}(1 + P/\tau),$$

where P is the number of partitions and τ is a user defined parameter that balances the number of parts with the allowed error. Recall that $e_{\min}$ is the natural lower bound resulting from taking the smallest error in each voxel. We found $\tau \in [2.0, 10.0]$ results in good approximations in a reasonable optimization time and number of parts (between 4-11 parts) as shown in Figure 9. The optimization error e_{opt} is the sum over all partition errors.

7. Evaluation and Results

To show that our framework significantly improves additive manufacturing processes we evaluated our results on a set of 3D objects shown in Figure 9. All models were generated and analyzed for a size of approximately 150mm (chair is over 300mm in height) and resolutions from 5mm-0.5mm. The subsampling was computed with about 150dpi consistently over all resolutions. On a standard desktop computer the processing took from several seconds up to about an hour depending on the resolution. Printed objects were automatically rotated facing with the larger footprint towards the printing platform.

Optimal Slicing Direction. Figure 3 shows that our method outperforms the PCA approach resulting in a lower best approximation error over all voxels. We measure the difference between both methods and show, plotted as mean and standard deviation over all models, that we constantly achieve an error minimization by about 10% over the whole resolution scale. Interestingly, the results of our clustering method often correspond to the natural upright direction. Note, this finding also contributes to additive manufacturing processes in general as it can be used to place an object in the 3D printing volume with the highest resulting accuracy - even without decomposing the object in several parts.

Optimized Contour. While standard additive manufacturing methods intersect the geometry at the center of a slice we propose an optimization by extracting the contour out of the distance field. We show that our method minimizes the volumetric difference error significantly in Figure 5. We plot the mean and standard deviation over increasing voxel resolutions showing that even for high resolutions up to 0.25mm the minimum voxel error improves between 20%-35%. As already mentioned, to compute the volume difference over decreasing resolutions we need to account for the loss in sampling resolution. Therefore we use more subsamples for lower resolutions.

Optimization Evaluation. Our proposed optimization process does not necessarily lead to a globally optimal partitioning solution. However, as shown in Figure 8 with the first five to eight parts the decomposition process lowers the volumetric difference error about 25% percent on all our reference models. The error decreases slowly with further increasing the number of parts. Furthermore, Figure 6 (Right) validates that we significantly improve accuracy compared to a standard one-directional slicing.

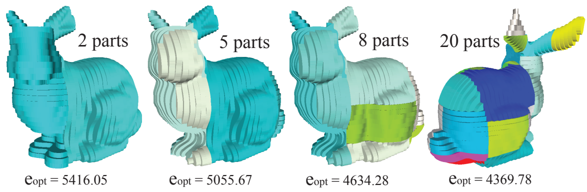

Figure 8: We show an increasing number of parts and the resulting optimization error e_{opt}. The best approximation error for this example is e_{min} = 3178.04. Slicing the object along its direction with the minimal error would result in e_{slc} = 7462.34. This example is generated for a material thickness of 3mm for a model size of 150mm.

Figure 9 show a variety of example 3D input shapes, results that are sliced in its best direction and their optimized handcrafted or rendered results. Additionally, we annotated the minimal volumetric error along the boundary of the shape e_{min}, the error that would result from an one directional slicing along the best slicing direction e_{slc} over the complete model and the error after optimization e_{opt}. We also show the user defined values p and τ used for the results. It can be seen that τ correspond to the number of parts generated.

In some cases our splitting algorithm would prefer to cut through very thin connections that would be tedious to assemble, e.g. a 'toothbrush' shape cutted vertically along the 'brushes' instead of cutting through the toothbrush 'head' horizontally. We suggest to prevent that problem by adding an additional energy term C to equation 8. We define C as the weighted connection cost of cutting area over cutting perimeter $C = \omega \cdot A/P$ weighted by a user defined value ω.

Stable Partitions. While an optimized contour generation has to be performed on the material thickness resolution we can show that the decomposition process is stable to resolution changes. Figure 7 shows (marked in green) that the parts generated in the beginning of the optimization process stay stable with increasing resolution. The optimization process was executed over the resolution from 1mm to 5mm material thickness. Depending on the manufacturing goal we propose that optimizing and decomposing at lower resolutions improves the overall accuracy even if the machine resolution is higher.

Visual artifacts along the splits. Our method generates results optimized for accuracy but also introduces additional visual artifacts along the segmentation cuts, especially for high resolution 3D prints. However, with increasing layer thickness we have found that thin structures might also suffer visually from slicing in the wrong direction. For example the horse model in Figure 9 top row is best sliced along the direction of its torso. This results in the legs being represented badly. While the shape is reproduced with overall small geometric error - because of the small volume and surface area of these structures - the result is still visually displeasing. Slicing along the legs results in visual artifacts on the torso. Our partitioning method optimally represents the geometry by decomposing the model.

References

[1] A. Bagsik, V. Schöppner, Mechanical properties of fused deposition modeling parts manufactured with ultem*9085, in: Proceedings of ANTEC 2011, 2011.

[2] S. Danjou, P. Köhler, Determination of optimal build direction for different rapid prototyping applications, in: A. Bernard (Ed.), Proceedings of the 14th European Forum on Rapid Prototyping, Ecole Centrale Paris, 2009.

[3] P. Alexander, S. Allen, D. Dutta, Part orientation and build cost determination in layered manufacturing, Computer-Aided Design 30 (5) (1998) 343–356.

[4] K. Thrimurthulu, P. M. Pandey, N. V. Reddy, Optimum part deposition orientation in fused deposition modeling, International Journal of Machine Tools and Manufacture 44 (6) (2004) 585 – 594.

[5] V. Canellidis, V. Dedoussis, N. Mantzouratos, S. Sofianopoulou, Pre-processing methodology for optimizing stereolithography apparatus build performance, Computers in Industry 57 (5) (2006) 424 – 436.

[6] S. Masood, W. Rattanawong, P. Iovenitti, Part build orientations based on volumetric error in fused deposition modelling, International Journal of Advanced Manufacturing Technology 16 (2000) 162 – 168.

[7] I. Reisner-Kollmann, C. Luksch, M. Schwärzler, Reconstructing buildings as textured low poly meshes from point clouds and images, in: N. Avis, S. Lefebvre (Eds.), Eurographics 2011 - Short Papers, 2011, pp. 17–20.

[8] L. Luo, I. Baran, S. Rusinkiewicz, W. Matusik, Chopper: partitioning models into 3d-printable parts, ACM Trans. Graph. 31 (6) (2012) 129:1–129:9.

[9] M. Lau, A. Ohgawara, J. Mitani, T. Igarashi, Converting 3D furniture models to fabricatable parts and connectors, ACM Trans. Graph. 30 (4) (2011) 85:1—-85:6.

[10] S. Sellamani, PCS: Prominent Cross-Sections for Mesh Models, Computer-Aided Design and Applications 7 (4) (2010) 601.

[11] J. McCrae, K. Singh, N. J. Mitra, Slices: a shape-proxy based on planar sections, ACM Trans. Graph. 30 (6) (2011) 168:1–168:12.

[12] K. Hildebrand, B. Bickel, M. Alexa, crdbrd : Shape Fabrication by Sliding Planar Slices, Computer Graphics Forum (Eurographics 2012) 31 (2).

[13] Y. Schwartzburg, M. Pauly, Fabrication-aware design with intersecting planar pieces, Computer Graphics Forum (Proceedings of Eurographics 2013) 32 (2).

[14] X. Décoret, F. Durand, F. X. Sillion, J. Dorsey, Billboard clouds for extreme model simplification, ACM Trans. Graph. 22 (3) (2003) 689.

[15] Autodesk, 123D, http://http://www.123dapp.com/ (2012).

[16] T. S. Smith, R. T. Farouki, Gauss map computation for freeform surfaces, Computer Aided Geometric Design 18 (2001) 831—-850.

[17] T. S. Smith, R. T. Farouki, Optimal slicing of free-form surfaces, Computer Aided Geometric Design 19 (2002) 43–64.

[18] P. Kulkarni, D. Dutta, An accurate slicing procedure for layered manufacturing, Computer-Aided Design 28 (9) (1996) 683 – 697.

Figure 9: We show the input 3D shapes (left), a result that is sliced in its best direction (middle) and the handcrafted or 3D printed results or rendered images (right). We also show the best approximation error, the error that would result from one directional slicing and the error after optimization. Additionally parameter settings and the number of resulting subparts are annotated.