Toward Support-free for 3D Printing: A skeletal Approach for Partitioning Models

**Abstract:**

We propose an algorithm for partitioning a 3D mesh model into the least pieces of parts for 3D printing without using any support structure.

Minimizing Support structures is crucial in reducing 3D printing material and time, partition-based methods are efficient means in realizing this objective. However, any partition will inevitably induce seams and cracks on the assembled model, which affects the aesthetics and strength of the finished surface. To achieve support-free fabrication while minimizing the effect of the seams, we put forward an optimization system with the minimization of the number of partitioned components and the total length of the cuts, under the constraints of support-free printing angle and the dimension of each part with respect to the printing dimension.

We show that the optimization problem is NP-hard and propose a Monte Carlo method based on training-and-learning data to find an optimal solution to the objectives. We applied our partition method on a number of various 3D models. Finally, we validate our method by carrying out a serial of 3D printing experiments.

**1 Introduction**

3D printing, or additive manufacturing, has drawn growing interests from researchers in computer graphics. Fused deposition modeling (FDM), stereolithographic (SLA), Selective Laser Melting (SLM) and Selective Laser Sintering (SLS) and the four most popular means of 3D printing techniques. Although 3D printing has seen its applications in producing arbitrarily intricate 3D models, the price of the printing materials, especially for those with high quality, are still outrageously high. Therefore, it is desirable to reduce materials used in the fabrication process. Note that this is also a critical operation for reducing production time and therefore the total production cost. For this purpose, an efficient method is the minimizing of support structures, which are removed in the post-processing phase of the fabrication task.

As for minimizing support structures, Autodesk R MeshMixerTM provides a semiautomatic orientation optimization tool to minimize support volume, support area, structural strength, or a combination of these three attributes. However, it requires the operators’ experience in setting the geometric parameters manually. A number of literatures have studied various factors that influence the volume of supports, e.g., optimizing the topology of the support structure [Dumas\_2014, Vanek\_2014\_1], determining an optimal fabrication direction [Charlie\_2015, Hildebrand\_2013, Padhye\_2011]; partitioning any given model into a set of separate parts that preserve nice geometric features [Vanek\_2014\_2, Song\_2015].

However, the partition results of these methods do not simultaneously respect the geometric features and support-free printability of the portioned parts. Further, no existing algorithms ever consider the problem of partitioning a model into the least number of pieces whose fabrication is free of support structures. The least number of portioned parts corresponds to the least number of seams on the final assembled model, which means a nice aesthetics preservation of the model surface; and the support-free fabrication means saving material to the most extent, which is very meaningful in reducing fabrication cost since the prices of printing materials (metal powder, resin, nice plastics, etc.) are still outrageously high. This motivates us to explore an efficient algorithm for solving the issue.

Our solution of the problem draws inspiration from the skeleton of organic models: the topology changes of a natural model can be determined by its skeleton, and each chunk of a mesh corresponding to a segment of the skeleton; further, the mesh chunk is a cylinder-like shape which can be printed free of support structures if the printing direction is parallel to the skeleton.

Automatically partitioning 3D models into components that are consistent with human perception is an extremely intricate due to the lack of semantic information. Therefore, we restrict our focus on organic or man-made models whose shapes are well-defined by their skeletons. Further, support structures are required for the interior and exterior surface of a mesh model during the 3D printing process. For simplicity, our approach assumes that the interior of a mesh model is fully filled, usually the infill can be set as a grid whose porosity can be varied by users, as allowed in almost all existing commercial 3D printing software. This is very useful in preserving the surface quality of any printed object. Therefore, our objective is to partition a model according to the growth of its skeleton. Formally, given a 3D printer, if a facet subtends to a common axis by an angle of less than or equal to **, then the facet can be printed without using any support structure. This inspires us to compute a minimum set of subgraph of the skeleton, such that each arc in any subgraph subtends to a common axis by an angle of less than or equal to **, the corresponding chunk of the mesh is therefore support-free as printed along this axis. In general, a cone of axes satisfies the angle constraint. However, the volume of the chunk should be within the working space of the printer. Further, if the center of mass of the chunk diverts from the support center too much, e.g., the gravitational torque applied at the mass center is too far away from the center of support that it bends the printing model by a layer of thickness, then the surface quality of the model is poor or the printing task fails since the next printing layer cannot be firmly attached to the previous layer.

Our methods make the following contributions:

* We partition any given mesh model into the least number of parts that are printable free of support structures; meanwhile, we preserve the aesthetics of the surface finish with the least number of seams whose length is minimized.
* We propose a partition method based on the guide of 1D Laplacian skeleton of any given mesh model;

**2 Related works**

**Shape Optimization for 3D Printing.** A number of literatures in computer graphics have focused on the fabrication of 3D models using 3D printers. Optimization works have been devoted to structural designs with emphasis on saving printing materials while preserving certain strength [Stava\_2012, Zhou\_2013, Umetani\_2013, Wang\_2013, Lu\_2014]. The modeling of some particular features have also been studied, for example, deformation behavior [Skouras\_2013], animated mechanical characters [Coros\_2013, Ceylan\_2013], articulated models with mobile joints [Bächer\_2012, Calì\_2012], models spinnable motions [Bächer\_2014], self-balancing [Prévost\_2013].

**Model Partition for 3D Printing.** A 3D printer cannot directly print a model whose size is larger than the printer's working space, To overcome this practical limitation, Luo et al. [Luo\_2012] proposed a solution to partition a given 3D model into parts for 3D printing and then assemble the parts together. This approach has a few advantages: (1) it is cost-effective in the sense that we only need to print a replacement part for a corresponding broken part; (2 it is convenient for storage and transportation; (3) changing some parts of a model allows innovative designs. Along this line of research, Hao et al. [Hao\_2011] partitioned a large complex model into simpler 3D printable parts by using curvature-based partitioning. Hildebrand et al. [Hildebrand\_2013] addressed the directional bias issue in 3D printing by segmenting a 3D model into a few parts each of which is assigned an optimal printing orientation. Vanek et al. [Vanek\_2014\_2] reduced the time and material cost of 3D printing by hollowing a 3D model into shells and breaking them into parts, a number of parameters including the total connecting area and volume of each segment are considered during the optimization process. Song et al. [Song\_2015] recently develop a novel voxelization-based approach to construct inter-locking 3D parts from a given 3D model. Without using any glue, Xin et al. [Xin] and Song et al. [Song] take a 3D interlocking approach to construct and connect printed 3D parts to form an object assembly. Hu et al. [Hu\_2014] proposed an algorithm for decomposing a solid model into the least number of pyramids. However, their algorithm is not applicable to our mesh modesls.

**3 Overview**

[youyi’s comment for the following part: Here an intuitive overview of the problem and the method is required. e.g., our method focuses on shapes which merit well-defined skeletons, for example, articulated shapes, as in such a setting,]

Observe that the topology of most natural life forms such as trees and animals can be described by their skeletons. Compared to natural skeletons, the medial axis can describe the topology of a mesh model more precisely [Zhang\_2015]. However, a medial axis of a 3D mesh model is a 2D surface which cannot be conveniently applied to describe the critical topology changes of the model. Additionally, the medial axis consists of intersecting pieces of planes and conic surfaces, presenting significant complications to algorithms that attempt to construct 3D medial axes.

Reeb graph provides a possible choice for 1D skeleton. During the generation process of any reeb graph, the slicing direction and the position of the representative node on each slide (a connected region) seriously influence the choice of critical points and therefore generation of the Reeb graph. However, the determination of suitable slicing direction and representative nodes is an intractable problem.

Gown by shrinking the mesh model using Laplacian smoothing, 1D Laplacian skeleton provides an excellent choice for reasonably describing the topology of any 3D model [Au\_2008]. See Figure XXX for an illustration of the Laplacian skeleton.

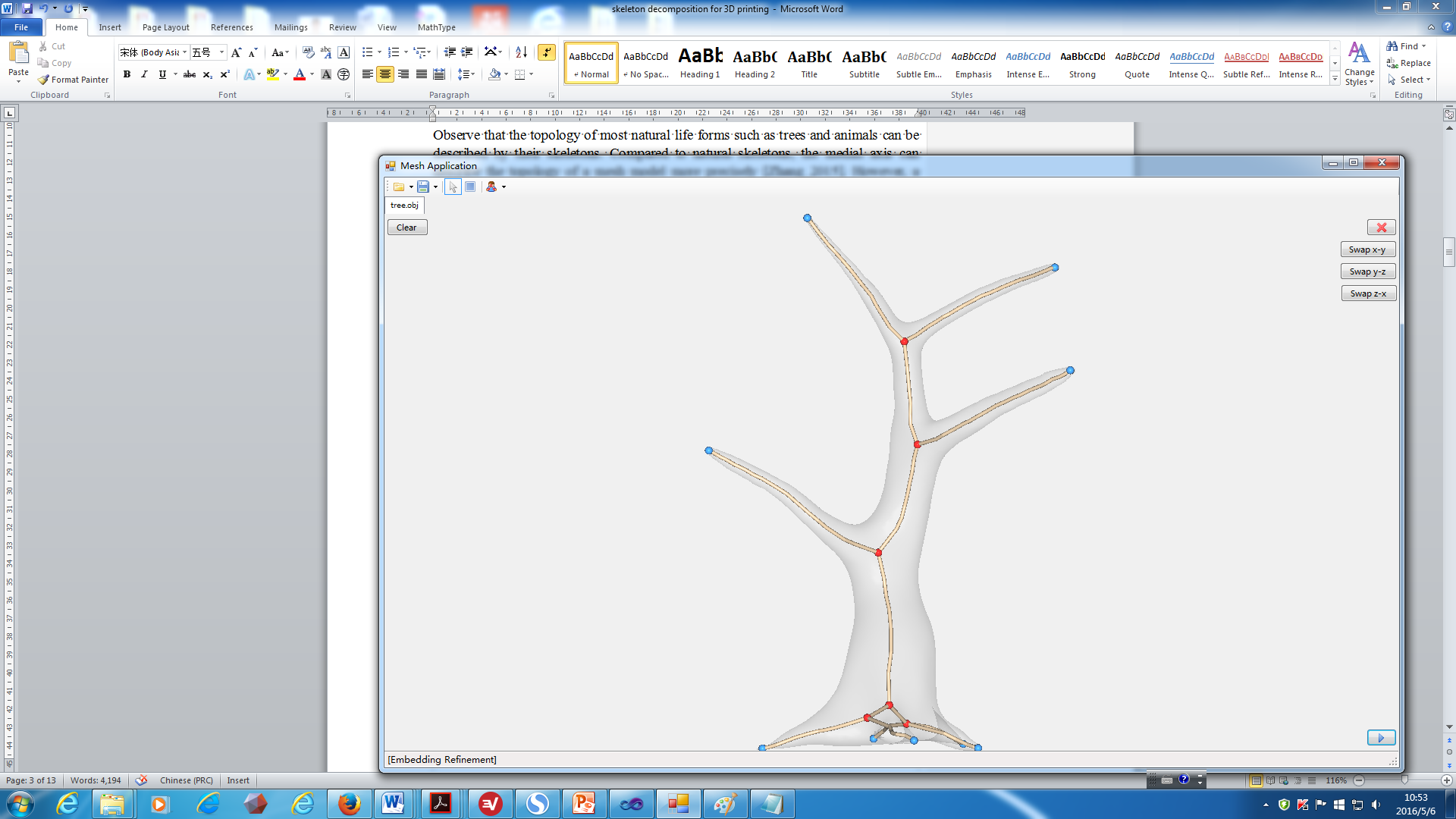


Figure XXX: A 1D skeleton of a tree.

In nature, the geometric features of most organic models are cylinder-liked, e.g., arms, legs, etc. Therefore, an organ segment can be represented by its corresponding 1D skeleton piece. In the remainder of the paper, by *model* we mean an organic mesh model of natural life form or cartoon figures preserving nice topology features of real lives. Based on this property, a chunk of a mesh model can be fabricated free of support if and only if its corresponding skeleton piece subtends to the building direction by an angle of no large than **, where ** is a threshold value determined by experiments.

**Problem Statement**: Decomposing the 1D Laplacian skeleton of the model into the least support-free subgraphs leads to a partition of the model into the least printable parts free of support structures and cracks on the final assembly model. In addition, since support structures result in bumpy supported areas, support-free fabrication also means a nice preservation of the surface quality of the parts. Further, a minimization of the number of cuts and the total cutting length means a minimum amount of seams and their lengths on the assembled model. Therefore, we shall focus on these two problems in this paper.

**The hardness of the problem**: Consider an instant of 1D Laplacian skeleton that is a fork with n vectors sharing a common origin; our objective is partitioning the fork into the least number of sub-forks such that each sub-fork can be packed into a cone of angle 2**. This problem is exactly the problem of packing n items with weights w1, ..., wn into bins of capacity c such that all items are packed into the fewest number of bins, which has been shown to be NP-hard [Fukunaga\_2007].

**High level of our approach**: we shall formulate the problem as an optimization system with both the objectives of the total number of cuts and the cutting length, under the constraint of printing angle of each branch with respect to the build platform, the angle between a cut plane and the printing direction, the dimension of each printed model with respect to the printing space of a given printer, and the area of the base of a printed model. We shall show that the problem is NP-hard; based on a set of training data, we propose a randomized Monte Carlo method to solve the optimization system.

**4 Technical Details**

Let M denote the mesh model, and let S denote the Laplacian skeleton. We propose an algorithm for partitioning S into a minimum set of nice (disjoint) subgraphs, each of which can be fabricated in a 3D printer without using support structures. Decomposing S into two pieces can be done by duplicating a node v and splitting the arcs incident to v properly; however, decomposing M around v requires the determination of the position and normal of a cutting plane; to realize these and guarantee a nice look on the finished surface with shortest seams, we set a constraint of minimizing the peripheral length of a cut.

For realizing the objectives, in addition to the angle constraint for the arcs, we need to consider the dimension constraint: the dimension of the printing model should be within the working space of a given 3D printer.

To summarize, our objectives are the minimization of (1) the number of partitioned components N; (2) the total peripheral length of each cut *Li*, i.e.,.

The constraints of the problem are as follows:

(1) Each arc of the partitioned subgraph of the skeleton subtends to a common axis by an angle of no larger than **, where ** is defined based on printing experiments without using any support structure. This guarantees that the corresponding mesh component is support-free during the printing process.

(2) The dimension of each component of the mesh, denoted by *Di*, corresponding to a partitioned subgraph *Hi*, should be constrained by the dimension of the working space of the printer.

(3) Let the normal direction of the cutting plane *c* (pointing to the exterior of the partitioned part), denoted by *n*(*c*); as the directed arcs of Hi are translated to a common origin, they form a fork, the fork has a central ray, which is the middle axis of the minimum cone that encloses the fork, let Cone(Hi) be the cone, let the central ray of Cone(Hi) be denoted as *r*(*Hi*); in order to guarantee support-free fabrication, the angle between *n*(*c*) and *r*(*Hi*) should satisfy *A*(*n*(*c*), *r*(*Hi*)) ≤ **/2 + **; finally, the base of a printing model should be large enough to gather sticky force from the building platform, such that the model is not deformed during the building process. Formally, let *b*(*Hi*) denote the area of the base of the mesh component corresponding to Hi, let  be a user-defined threshold value., then we have *b*(*Hi*) ≥ . Here  can be determined by some simple printing experiments.

To summarize, we have the following optimization system:

Objective: min N and min (2)

Subject to: *A*(*e*, *r*(*Hi*)) ≤ **, for each *e* *Hi* (3)

*A*(*n*(*c*), *r*(*Hi*)) ≤ **/2+ ** (4)

Di ≤ D; (5)

*b*(*Hi*) ≥  (6)

Assume that we are given a function Tirm\_BFS(*v*, *G*) which travers *G* from *v* in a breath first search manner until all arcs satisfying constraints (2-3) are determined, we have the following algorithm for skeleton decomposition. The main idea of our algorithm is to randomly search the graph using Monte Carlo Method, which randomly chooses a node of S to start traversing and randomly chooses an arc of the current node as the exit path.

**Algorithm: Skeleton\_Mesh\_Decomposition(*S*, M)**

Input: The Laplacian skeleton *S* of a mesh model *M*;

Output: The decomposition of *S* into a set of the least pieces of subgraphs T, each arc of which subtends to an axis by an angle of no larger than **, where (0.5** ** **) is the minimum angle subtends to the build plate that allows a facet being printed free of support structure.

1. *T* = *Ø*; min = +∞; count = 0; max\_iter = a user defined large constant;

// initialization

2. **while** count < max\_iter **do**

3. *G* = *S*; *U* = *Ø*;

4. **while** *G* ≠ *Ø* **do**

5. **for** each node *v*  *S* **do //** Monte Carlo Method

6. *H* = Trim\_BFS(*v*, *S*, **);

7. *S* = *S* \ *H*;

8. *U* = *U*∪*H*;

9. **if** ||*U*|| < min **then**

10. *T* = *U*;

11. min = ||*U*||;

12. count = count +1; // one iteration

13. **return** *T*;

**Skeleton Partition**:

Next we shall show how Tirm\_BFS(*v*, *S*, **) works to find a maximal subgraph starting at *v* that satisfies the angle constraint. Let *H* be the current subgraph obtained. When an arc *e* of *G* is visited, we need to determine whether it should be included into *H*. If the start of each outgoing arc of *H* is moved to a common site, then the arcs form a fork (Figure QAZ). A naïve method to judge whether *e* should be included is to move the start of *e* to the origin of the fork, and compute the angle between *e* and each arc of the fork, *e* is included if the maximum angle between e and each arc of the fork does not exceed 0.5**. However, this naïve method would require O(*K*2) time, where *K* is the number of the nodes of *S*. To speed up this process, we keep the pair of vectors hat form the largest angle and judge whether a new vector expands the mouth of the fork; if so, determine the other vector (may not be an arc of *H*). See Figure QAZ, let  and  be the units of these two vectors obtained so far, for simplicity, we denoted by F(, ) the fork with the starts of all unit vectors converging at the origin of the coordinate frame, where  and are the pair of unit vectors that form the largest angle in the fork. Let  be the unit of a new vector to be processed next, if  penetrates through the blue circle, then no change need to be made to the fork; otherwise, let *Dij* denote the spherical disk that passes through the endpoints of  and , let *ci,j* be the center of *Di,j*, let *Bi,j* be the boundary circle of *Di,j*, the circle passing through  and *ci,j* (denoted as *O*(, *ci,j*)) intersects *Bi,j* at two points, let *q* the point further away from the endpoint of , then *q* is the endpoint of the other extreme vector ** (in addition to ) that is used in the next iteration. To summarize,

A new edge *ek* is taken by Function Tirm\_BFS if and only if one of the following two conditions is met:

(1) the angle between ** and , denoted as *A*(**, ), satisfies *A*(**, ) ≤ *A*(**, )

(2) *A*(**, ) ≤ ****, where *q* = *O*(, *ci,j*)*Bi,j*.



[Remove the coordinate axes; change Cij into ci,j]

Figure QAZ: Illustration of unit vectors, unit sphere, spherical disks, and the determination of taking a new edge in Trim\_BFS.

**Mesh Partition**:

[Anton: Optimization of cutting sequence]

Although the skeleton partition tells us a rough sketch of the mesh partition, i.e., the cutting plane should be in the vicinity of each node v incident to two distinct subgraphs. Yet we need to determine the exact positions of the cutting planes. For each node v that is incident to at least two distinct subgraphs *Hi* and *Hj*, we process it using the following cutting principle:

**The Cutting Scheme for M**:

See Figure TJK, at the position of node v, define two fillets each of which contains all planes that orthogonal to the vectors in Cone(Hi) or Cone(Hj), let Fillet(Hi) and Fillet(Hj) be the two fillets respectively. Depending on the relative position of Fillet(Hi) and Fillet(Hj), we have the following cases:

* (Fillet(Hi) Fillet(Hj)) / {v} ≠ ;

Then randomly sample a set of cutting planes from (Fillet(Hi) Fillet(Hj)) / {v}, and determine the one achieving the minimum cutting length.

* (Fillet(Hi) Fillet(Hj)) = {v};

In this case, two cuts *c1* and *c2* are inevitable in order to separate the mesh, however, care must be taken as the angle between *c1* and *c2* should be constrained by A(*n*(*c1*), *n*(*c2*)) ≤ **/2+ **, i.e., Eq.(5), if this constraint is violated, one more cut in between *c1* and *c2* is required; while the base of each partitioned component should be constrained by *b*(*Hi*) ≥  and *b*(*Hj*) ≥ , i.e., Eq.(6).

Fillet Example: [zhu lin: 3D drawing]

In each subgraph, if v is neighbored to a single node w in a subgraph H (Figure TJK (a)), and if ||*vw*|| ≤, where  is a user-defined threshold value, then the reflex vertices of M that are incident to v and w are taken into a set R(v), R(v) is truncated by setting a threshold value for the concavity degree of the reflex vertices; on the other hand, if v is neighbored to more than one nodes in H or ||*vw*|| > , then R(v) is only restricted to the reflex vertices of v.

In the above process, the concavity of a reflex vertex is determined by the value of (*vi* - *vj*)(*nj* - *ni*) [Au\_2012]; more precisely, and the concavity of vi is defined as the nonnegative value of (*vi* - *vj*)(*nj* - *ni*); if the value of the term is negative, then vi is a not reflex.

For each vertex in R(v), generate a random set of cutting planes (e.g., 30) whose normal direction is within Cone(H); further, for the current node v, we also randomly assign a set of cutting planes whose normal direction is within Cone(H). In both random processes, we explore the Monte Carlo method. We shall select the cutting plane that avoids cutting any other subgraph and is shortest in terms of peripheral length, which corresponds to the shortest seam on the assembled model.

See Figure TJK (c), if two cutting planes form an obtuse angle **> **/2 + **, then we apply an additional cut along the angular center of the two cutting planes originating from the intersection line.

~~~~

~~Figure TJK: 2D illustration of cutting M based on the decomposition of S. The dashed lines indicate the position of cutting planes.~~

~~[the figure has been revised properly]~~

However, do to the existence of various special cases of the relative position of the skeleton nodes and their corresponding mesh features (e.g., concave regions). We still need to consider several special cases with care.

Case 2: A number of dense nodes appear on a path of a subgraph H, and all possible cutting planes respecting Cone(H) cut into some other subgraph. For example, the path from v to w in Hj (Figure EDS). In this case, each cutting planes respecting Cone(Hj) that passes through v incurs an intersection with some other subgraph, which violates our purpose of separating a mesh component that preserve the topology features of its corresponding subgraph. Therefore, we trace the position of the cutting planes by moving from v along the path until a node w that is free of intersection with any other subgraphs.



[change into a 3D version]

Figure EDS: 2D illustration of forward tracing of the cutting planes.

**Cut-subgraph Intersection Detection**: In this process, the intersections of a cut and the subgraphs can be efficiently determined by taking advantage of the correspondence between the mesh vertices and the skeleton nodes: each vertex of M is mapped to a single node of S. Therefore, as a cut goes through the mesh surface, the endpoints of the cut edges of M lend us the information of the cut subgraphs. In our case, we merely need to determine that a cut intended to be applied to subgraph *Hi* is cutting an alternative subgraph (e.g., *Hj* in Figure EDS).

Case 3: Each candidate cut around a node v of S intended to separate *Hi* from S cuts into at least one more subgraph. As this happens, we greedily choose the cut of minimum length; iteratively, we apply a cut to separate each connected component of any other subgraphs (other than *Hi*).

**[insert an example here]**

**Tirm\_BFS(*v*, *S*, **)**

Input: A node v of Laplacian skeleton S, an angle of value .

Output: A maximal subgraph H rooted at v that meet the requirement of angle and dimension constraints.

1. starting from v, initialize F(, ), H = *Ø*;

2. **for** the current arc ek of S picked by the BFS process **do**

3. **if** *A*(**, ) ≤ *A*(**, ) **then goto** line 10**;**

4. **else**

5. *q* = *O*(, *ci,j*)*Bi,j*;

6. **if** *A*(**, ) ≤ **** **then**

**7.**  = ;

8.  = **;

9. update *Bi,j* and *ci,j*;

10. H = H ∪ {ek};

11. call the cutting scheme for M;

12. M= M/MH;

34. **return** H and MH;

In line 2 of Tirm\_BFS, the BFS process randomly chooses an arc incident to v to proceed on. In order to guarantee a greater chance of converging to the optimal result in a short time, we apply a training-and-learning procedure for the first 1000 runs. Formally, let Nv be the number of times an arc e is chosen as the exit arc when node v is visited. Given the data of the first 1000 runs, as a node v is visited, the probability of choosing an arc e as an outgoing arc in the subsequent runs is,

P(v, e) = Nv/1000 (6)

To further speed up the process of Trim\_BFS, we assign a mark that stores the minimum number of subgraphs obtained so far, such that the current branching can be terminated if its output number of subgraphs is larger than the mark.

Obviously, Trim\_BFS can guarantee the torque constraint and the dimension constraint properly. However, it is not obvious whether the angle constraint is satisfied, since the traversing process assigns a specific direction to each arc that was originally undirected in S. To clarify this, we have the following lemma.

**Lemma 1: H = Tirm\_BFS(*v*, *G*, **) is a maximal subgraph of *G* that satisfies the angle constraint, i.e., each arc of H subtends to an axis by an angle of no larger than **.**

Proof: Suppose to the contrary that *H* violates the angle constraint, there exists a directed arc that does not satisfy the angle constraint. For example, arc (*c*, *a*) (directed from either direction) or directed arc (*b*, *c*) in Figure WSX. Such case is impossible as line XXX of function Trim\_BFS excludes any directed arc that violates the angle constraint of nor larger than ** with respect to the (virtual) center axis.

It remains to prove that H is maximal, i.e., the largest graph rooted at *v* that covers all the arcs that satisfies the angle constraint. Suppose that this is not true, there must exist an arc that was mistakenly discarded due to the direction in which the arc is traversed. Let (b, c) be one of such arcs, as illustrated in Figure WSX (a), as the arc is directed from b to c, it is not included as it violates the angle constraint, but can be included if the arc directed from c to b satisfies the angle constraint. We shall prove in a case-by-case basis.

If c is not reachable from v via a directed path without passing through b (Figure WSX (c)), then c is only reachable from b, arc (b, c) should not be included and line XXX of Function Trim\_BFS correctly handle this case.

Otherwise, c is reachable from v via a directed path without passing through b (Figure WSX (d)), as c is visited, by line XXX of Function Trim\_BFS, each arc leaving c is considered, and the directed arc (c, b) is correctly included into H.

This completes the proof. □



(a) (b)

Figure WSX: (a) a subgraph incorrectly takes directed arc (*c*, *a*) and arc (*c*, *b*); (b) the configuration of (*c*, *a*) and (*b*, *c*) in the fork system.

**5. Results**

[In the examples, show how the training-and-learning procedure saves running time]

We have evaluated our skeletal partition approach on a number of models, including man-made art objects and organic forms. Figures LLL show partition results, and the printed models and their experimental statistics.



[show the printed model without any cutting operation, the partitioned CAD model, the assembled printed model, the statistics on time and material saving, for all 12 models]

**6. Conclusion, Limitations, and Future Works**

Compared with existing partition-based methods, the advantages of our partition method are as follows:

(1) The models are support-free, especially for the 3D printing techniques including SLA, SLM and SLS. For FDM technique, it requires a bed of support that consumes very little volume of materials.

(2) The seams on the assembled model are minimized in terms of cut number and cut length.

**Reference:**

[Luo\_2012] Luo, L., Baran, I., Rusinkiewicz, S., and Matusik, W. 2012. Chopper: Partitioning models into 3D-printable parts. ACM Trans. on Graph 31, 6, 129:1–129:9.

[Zhang\_2015] Zhang X.L., Xia Y., Wang J.Y., Yang Z.W., Tu C.H., Wang W.P.. Medial axis tree—an internal supporting structure for 3D printing. Computer Aided

Geometric Design, 35:149–162, 2015.

[Lo] K.-Y. Lo, C.-W. Fu, H. Li, 3D polyomino puzzle, ACM Tran. on Graphics (SIGGRAPH Asia) 28 (5), article 157.

[Xin] S.-Q. Xin, C.-F. Lai, C.-W. Fu, T.-T. Wong, Y. He, D. Cohen-Or, Making burr puzzles from 3D models, ACM Tran. On Graphics (SIGGRAPH) 30 (4), article 97.

[Song] P. Song, C.-W. Fu, D. Cohen-Or, Recursive interlocking puzzles, ACM Tran. on Graphics (SIGGRAPH Asia) 31 (6), article 128.

[Song\_2015] Peng Song, Zhongqi Fu, Ligang Liu, Chi-Wing Fu. Printing 3D objects with interlocking parts. Computer Aided Geometric Design. Volumes 35–36, May 2015, Pages 137–148.

[Hao\_2011] J. Hao, L. Fang, R. E. Williams, An efficient curvature-based partitioning of large-scale stl models, Rapid Prototyping Journal 17 (2) (2011) 116-127.

[Hildebrand\_2013] K. Hildebrand, B. Bickel, M. Alexa, Orthogonal slicing for additive manufacturing, Computers & Graphics (SMI) 37 (6) (2013) 669-675.

[Zhou\_2013] Zhou, Q., Panetta, J., Zorin, D., 2013. Worst-case structural analysis. ACM Trans. Graph. (SIGGRAPH)32 (4), article137.

[Lu\_2014] Lu, L., Sharf, A., Zhao, H., Wei, Y., Fan, Q., Chen, X., Savoye, Y., Tu, C., Cohen-Or, D., Chen, B., 2014. Build-to-last: strength to weight 3D printed objects. ACM Trans. Graph. (SIGGRAPH)33 (4), article 97.

[Wang\_2013] Wang, W., Wang, T.Y., Yang, Z., Liu, L., Tong, X., Tong, W., Deng, J., Chen, F., Liu, X., 2013. Cost-effective printing of 3D objects with skin-frame structures. ACM Trans. Graph. (SIGGRAPH Asia)32 (6), article177.

[Au\_2008] Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, Tong-Yee Lee. Skeleton extraction by mesh contraction. ACM Transactions on Graphics. Volume 27 Issue 3, August 2008. Article No. 44.

[Skouras\_2013] Skouras, M., Thomaszewski, B., Coros, S., Bickel, B., Gross, M., 2013. Computational design of actuated deformable characters. ACM Trans. Graph. (SIG-GRAPH)32 (4), article82.

[Coros\_2013] Coros, S., Thomaszewski, B., Noris, G., Sueda, S., Forberg, M., Sumner, R.W., Matusik, W., Bickel, B., 2013. Computational design of mechanical characters. ACM Trans. Graph. (SIGGRAPH)32 (4), article83.

[Ceylan\_2013] Ceylan, D., Li, W., Mitra, N.J., Agrawala, M., Pauly, M., 2013. Designing and fabricating mechanical automata from mocap sequences. ACM Trans. Graph. (SIGGRAPH Asia)32 (6), article186.

[Bächer\_2012] Bächer, M., Bickel, B., James, D.L., Pfister, H., 2012. Fabricating articulated characters from skinned meshes. ACM Trans. Graph. (SIGGRAPH)31 (4), article47.

[Bächer\_2014] Bächer, M., Whiting, E., Bickel, B., Sorkine-Hornung, O., 2014. Spin-It: optimizing moment of inertia for spinnable objects. ACM Trans. Graph. (SIGGRAPH)33 (4), article96.

[Calì\_2012] Calì, J., Calian, D.A., Amati, C., Kleinberger, R., Steed, A., Kautz, J., Weyrich, T., 2012. 3D-printing of non-assembly, articulated models. ACM Trans. Graph. (SIGGRAPH Asia)31 (6), article130.

[Prévost\_2013] Prévost, R., Whiting, E., Lefebvre, S., Sorkine-Hornung, O., 2013. Make it stand: balancing shapes for 3D fabrication. ACM Trans. Graph. (SIGGRAPH)32 (4), article81.

[Stava\_2012] Stava, O., Vanek, J., Benes, B., Carr, N., Mˇech, R., 2012. Stress relief: improving structural strength of 3D printable objects. ACM Trans. Graph. (SIGGRAPH)31 (4), article48.

[Dumas\_2014] Dumas, J., Hergel, J., and Lefebvre, S. 2014. Bridging the gap: Automated steady scaffoldings for 3d printing. ACM Trans. Graph. 33, 4 (July), 98:1–98:10.

[Vanek\_2014\_1] Vanek, J., Galicia, J. A. G., and Benes, B. 2014. Clever support: Efficient support structure generation for digital fabrication. Computer Graphics Forum 33, 5, 117–125.

[Vanek\_2014\_2] J. Vanek, J. A. G. Galicia, B. Benes, R. M\_ech, N. Carr, O. Stava, G. S. Miller, PackMerger: A 3D print volume optimizer, Computer Graphics Forum 33 (6) (2014) 322-332.

[Au\_2012] Oscar Kin-Chung Au, Youyi Zheng, Menglin Chen, Pengfei Xu, and Chiew-Lan Tai. Mesh Segmentation with Concavity-Aware Fields. IEEE Trans. Vis. Comput. Graph. 18(7):1125-1134 (2012).

[Padhye\_2011] Padhye, N., and Deb, K. 2011. Multi-objective optimization and multi-criteria decision making in SLS using evolutionary approaches. Rapid Prototyping Journal 17, 6, 458–478.

[Charlie\_2015] Zhang, X., Le, X., Panotopoulou, A., Whiting, E., Wang, C.C.L., 2015. Perceptual models of preference in 3D printing direction. ACM Trans. Graph. 34 (6), 215.

[Fukunaga\_2007] Fukunaga, A. S. ; Korf, R. E. Bin Completion Algorithms for Multicontainer Packing, Knapsack, and Covering Problems. Journal Of Artificial Intelligence Research, Volume 28, pages 393-429, 2007.

[Hu\_2014] Ruizhen Hu, Honghua Li, Hao Zhang, Daniel Cohen-Or. Approximate Pyramidal Shape Decomposition. ACM Transactions on Graphics, Vol. 33, No. 6, Article 213, 2014.