Anti-aliasing for fused filament deposition

Hai-Chuan Song, Nicolas Ray, Dmitry Sokolov, Sylvain Lefebvre

Anti-aliasing for fused filament deposition

Hai-Chuan Song[a], Nicolas Ray[a], Dmitry Sokolov[b], Sylvain Lefebvre[a,b]

[a]INRIA, France
[b]Université de Lorraine, France

Abstract

Layered manufacturing inherently suffers from staircase defects along surfaces that are gently slopped with respect to the build direction. Reducing the slice thickness improves the situation but never resolves it completely as flat layers remain a poor approximation of the true surface in these regions. In addition, reducing the slice thickness largely increases the print time.

In this work we focus on a simple yet effective technique to improve the print accuracy for layered manufacturing by filament deposition. Our method works with standard three-axis 3D filament printers (e.g. the typical, widely available 3D printers), using standard extrusion nozzles. It better reproduces the geometry of sloped surfaces *without* increasing the print time.

Our key idea is to perform a local *anti-aliasing*, working at a sub-layer accuracy to produce slightly curved deposition paths and reduce approximation errors. This is inspired by Computer Graphics anti-aliasing techniques which consider sub-pixel precision to treat aliasing effects. We show that the necessary deviation in height compared to standard slicing is bounded by half the layer thickness. Therefore, the height changes remain small and plastic deposition remains reliable. We further split and order paths to minimize defects due to the extruder nozzle shape, avoiding any change to the existing hardware. We apply and analyze our approach on 3D printed examples, showing that our technique greatly improves surface accuracy and silhouette quality while keeping the print time nearly identical.

Keywords: Fused filament fabrication, staircase, anti-aliasing, sub-layer, curved layers.

1. Introduction

Additive manufacturing (AM) technologies produce objects layer-by-layer. The final physical object is thus made of flat slabs of materials stacked on top of each others. As a consequence, only horizontal and vertical planar surfaces can be closely matched. All other regions suffer from approximation errors, which are often referred to as the *staircase effect*. This issue is illustrated in Figure 1 (b) and (c), where a wedge model is sliced into flat layers (layer thickness 0.6 mm) and 3D printed using a filament printer.

Such staircase defects lead to a deviation of the printed result from the input model, where necessary volumes in the input model may be removed in the printed result (red regions in Figure 2), while unnecessary volumes that do not exist in the input model may appear in the printed result (yellow regions in Figure 2). In addition, the final result lacks the smooth geometry of the virtual input model.

A typical approach to minimize staircase defects is to print with thinner layers, which increases the accuracy of the surface approximation. This, however, is achieved at the expense of a large increase in print time (roughly, print time doubles for each reduction of layer thickness by half). The minimal thickness is also limited by material properties – for filament printers the process becomes challenging under $50\mu m$, even though results down to tens of microns have been demonstrated by experts using precisely calibrated hardware.

Many approaches have been proposed to improve this situation, in particular by using different layer thicknesses in different parts of the model (see Section 2). However these approaches still rely on flat layers: the staircase effect is reduced but not removed, and the print time is increased.

We observe that the staircase defect bears similarity with *aliasing* in Computer Graphics, in particular when rasterizing continuous geometry on-screen produces a similar effect. In Computer Graphics, aliasing is often fought by considering information at a sub-pixel scale. We propose to follow a similar idea, and consider sub-*layer* information to produce toolpaths that are subtly
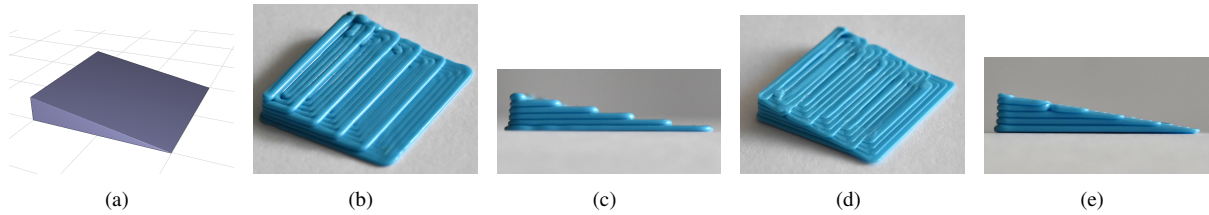
Figure 1: Printing a wedge model clearly reveals the staircase defects that plague 3D printing. (a) Input 3D model ; the bottom edge length is 20 mm and the angle of the incline plane is $10°$. (b) Global view and (c) side view (c) of a standard, flat layer printed result. (d) Global view and (e) side view of our anti-aliased printed result, revealing the improvement in surface accuracy and silhouette smoothness.

curved up and down to follow the surface. Our technique greatly improves surface quality while keeping the print time similar to uniform slicing (see Figure 1 (d) and (e)).

Our contributions are:

- The use of sub-layer information to produce (slightly) curved paths that better reproduce the input geometry.

- A slicing strategy that produces non–flat layers remaining within feasible thickness bounds, never exceeding a minimal thickness threshold.

- A path splitting and ordering approach that minimizes nozzle interference between neighboring paths of different heights during deposition.

Printed examples reveal a significant increase in the accuracy of the printed surfaces, with print times that remain nearly identical to the initial uniform slicing.

2. Previous work

A wide range of approaches have been proposed to minimize the staircase defects. Some apply to different technologies, while others exploit specificities of a particular process. Our work belongs to this second category as our technique focuses on fused filament fabrication: it exploits the possibility to move the extruder up and down during deposition.
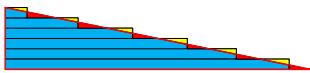

Figure 2: Approximation errors of flat layer slicing in the tease: red lines are the input 3D model surfaces; rectangles are printed layers; volumes removed from the input model are colored red; volumes added to the input model are colored yellow.

Orientation. Part orientation plays a significant role in determining the impact of the staircase effect. For a review of this topic, please refer to [1]. Recent works regarding part orientation have focused on using perceptual models to ensure perceptually important regions are best reproduced [2].

In this work, we assume the object has already been optimally oriented according to application specific criteria, and focus on precise reproduction given this orientation.

Adaptive slicing. Most technologies afford for varying layer thicknesses. Rather than printing with the same uniform thickness throughout the object height, several methods propose to *adapt* the layer thickness to the object geometry [3]. There are essentially three techniques to decide the layer thickness at each height: subdividing into thinner slices from the coarsest uniform slicing [4, 5, 6], merging into thicker slices starting from the thinnest uniform slicing [7], or formulating a global optimization problem [8]. Boschetto et al. [9] perform an inverse adaptation, deforming the input mesh such that the error after slicing is minimized. Different error metrics have been proposed to evaluate the error, e.g. the cusp-height [3], the difference between successive slices [10], or the volume error between the original and sliced geometries [11].

Our approach is compatible with adaptive slicing (see Section 3.1), and will further improve its results.

Locally adaptive slicing. A drawback of adaptive slicing is that it can only adapt to geometric changes along the built direction: the thickness remains constant within the layer. To address this problem several approaches first perform an object decomposition, and then independently slice different regions of the object. The object can be either printed as a single part with careful ordering of the toolpaths [12, 13, 14, 8], or can be printed in several pieces that are later manually assembled [15, 16, 17].

Our approach performs local thickness adaptation of the layers through small z-motions (sub-layer precision), avoiding a subdivision of the geometry into distinct parts.

Curved slicing. Several approaches exploit the ability of filament printers to move freely along the z-axis during deposition. Chakraborty et al. [18] proposed a curved layer deposition, with the objective of strengthening shell-like parts by aligning the toolpaths with the surface. The mechanical properties of the parts are further discussed in [19], and methods combining flat and curved layers are discussed in [20, 21]. In these works the curvature of the paths is large as they flow along the surface (e.g. several millimeters up and down). While this works very well on specific cases, in a general setting this requires specifically designed hardware to avoid interferences between the nozzle and the printed part as discussed [1] in [18, 21] and Figure 12.

Our approach only has to perform small sub-layer adjustments, requiring no change to the printer as the layer interference remains manageable (but still requires careful treatment). However, our work has slightly different objectives since we seek to increase the accuracy of the surfaces, while the aforementioned works attempt to align the filament deposition with the surface curvature to produce stronger parts.

In a more extreme demonstration of the opportunities for non-flat deposition, Mueller et al. [22] propose to print wire-mesh structures, exploiting the fact that extruded filament hardens quickly to print truss-like structures in mid-air.

Process based surface quality improvements. While mechanical means can be used to improve surface quality after fabrication (e.g. abrasion [23] or machining [24]), researchers have proposed a number of software techniques that exploit specificities of a process to reduce the staircase effect. Most notably, in the context of stereo-lithography (SLA) Pan et al. [25, 26] exploit the formation of a meniscus when an object moves out of the resin tank. The meniscus is cured to fill the creases between two layers. Repeating this process produces smooth, accurate surfaces. Park et al. [27] show how dithering can cure resin partially and produce slanted surfaces along a layer.
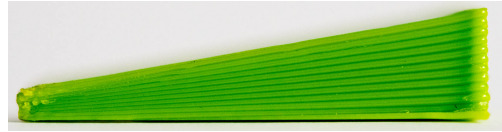

Figure 3: A slope generated by varying layer thicknesses across 12 layers. Each layer thickness varies from 0.07 mm to 0.38 mm uniformly from left to right. The nozzle width is 0.4 mm.

Our work also exploits sub-layer precision, relying on properties specific to fused filament fabrication.

2.1. Printing with varying layer thickness

We provide here some background on the technical aspects of printing with varying layer thicknesses. In printer instructions (G-code) each print path is a sequence of vertices, each having a 3D coordinate (x, y, z) as well as a length of extruded filament e. Thus, varying the layer height involves updating z and e in a synchronized manner such that the width of deposited plastic remains regular. Therefore, if the vertex is displaced vertically by δ the filament extrusion length becomes $\frac{z+\delta}{z} \cdot e$.

We found that reducing the extrusion speed (the length of filament extruded per second) when changing the thickness provides a more regular extrusion. Given an initial speed F_{ini}, a minimum speed F_{min}, the vertical displacements of two neighboring vertices δ_1, δ_2 and a base layer thickness h, we compute the new speed as a linear interpolation $F = F_{ini} + \frac{|\delta_1 - \delta_2|}{h} \cdot (F_{min} - F_{ini})$. This slows down where there are larger changes in height. We use $F_{ini} = 20$ mm/sec and $F_{min} = 13$ mm/sec in all our results. These are standard speeds for good quality prints. Figure 3 illustrates a slope generated by varying layer thicknesses.
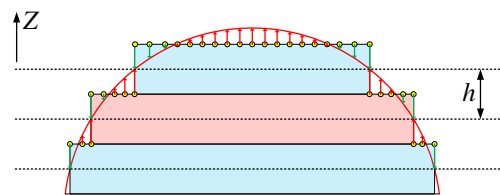

Figure 4: 2D illustration of uniform slicing (side view). The red curve is the input model boundary. The blue and red rectangles are cut view of the layers. Dashed lines are the mid-layer slicing planes. The layers are vertical extrusions of the intersection between the slicing planes and the initial object. The yellow circles are points on the top of the layers, that form the physical object boundary. The arrows reveal the distance between the printed surface and the true surface (red: positive; green: negative).

---

[1]Quote from [18] : "the periphery of the deposition head could interfere with the part being built up (..) deposition head shape would have to be modified (..) to avoid such phenomenon"; Quote from [21]: "A solution to track distortion is to maintain normality between the extruder axis and (..) surface (..). Although not feasible with the current system in use, robots with sufficient degrees of freedom (..)"

3. Toolpath anti-aliasing

The key idea of our technique is better explained by considering the shape of each layer. After uniform slicing, all layers are flat. Conceptually, our approach locally deforms the input flat layers to better approximate the surface. This is detailed in Section 3.1, where we also show that these variations do not need to exceed half the layer thickness, up or down.

In practice however, our method operates primarily on the toolpaths produced by a standard uniform slicing process. Each toolpath is a sequence of vertices, forming a curve along which the extruder deposits plastic. Our algorithm performs the following operations on each layer:

- For each toolpath vertex, the local *vertical* distance to the surface of the input model is computed through an efficient ray-tracing procedure.

- The paths are then deformed (*anti-aliased*) to better capture the sub-layer variations of the surface. This is detailed in Section 3.2.

- Finally, the paths are split and ordered to minimize interferences with the extruder nozzle. This is detailed in Section 4.

The last step is important to the success of our approach: it minimizes cases where a currently printed path collides with (and erases) a previously printed paths that was raised higher.

3.1. Layer anti-aliasing

The input model is sliced into flat layers by intersecting slicing planes with the 3D model. The slicing planes are located at the middle section of each slice, following a standard slicing process. A side view example is shown in Figure 4. The 3D model is represented by the red curve and sliced into three flat layers with the same thickness h. The dashed lines outline the mid-layer slicing planes. The extruder deposits plastic by moving along the top plane of the layers, forming vertical plastic slabs below.

Slicing a curved surface with flat layers produces a staircase defect, clearly visible in Figure 4 : the tops of the printed layers do not follow the original model boundary. To reduce aliasing, we propose to deform the layers and align their boundaries back with the original surface. This changes the layer shapes and produces corresponding z-motions of the extruder.
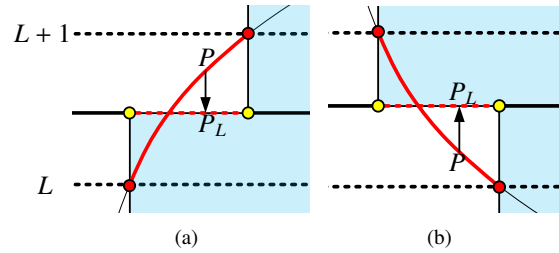

Figure 5: Illustration of the mapping between surface points and layers for the case of respectively top-facing (a) and bottom-facing (b) surfaces. The dashed black lines are mid-layer slicing planes. The dashed red segment is a physical boundary of the printed shape, at the top (a) or bottom (b). Each surface point P vertically projects to a point P_L on this segment. If the mapping is injective we can snap the points P_L back to the surface. The displacement is $\pm \frac{h}{2}$ at most.

Thickness bounds. An important constraint when modifying the layer shapes is to maintain the local thickness within feasible bounds, otherwise plastic extrusion and adhesion fails. For plastic filament deposition, our experiments reveal a minimal thickness of 0.05 mm and a maximum thickness of about $1.1w$, with w the nozzle width (we tested with $w = 0.4$ mm and $w = 0.8$ mm). The lower bound is due to the physical process, but also to errors in the printer assembly and calibration that stop being negligible with small thicknesses.

We deformed, the layers near the boundary may stretch too much and exceed these bounds. A possible approach would be to diffuse the deformation from the boundary layers to the inner layers, producing curved layers throughout the print. This leads back to curved layer slicing (see Section 2) which presents many challenges. Instead, we propose a simpler, yet effective solution that ensures the layer thickness always remains within $[\frac{h}{2}, \frac{3 \cdot h}{2}]$. The new layers properly capture all surface points but for extreme cases (discussed later). This approach is compatible with adaptive slicing.

Deforming the layers. After slicing, each layer is a vertical extrusion of the contour at the mid-layer slicing plane (see Figure 4). We independently deform the top and bottom of the layers to better reproduce respectively the top-facing and bottom-facing parts of the print. Even though we describe how to change the bottom shape of the layers, please note that our current implementation does not treat down-facing surfaces.

For the sake of clarity, let us first consider only the top-facing parts of the surface. We denote P a surface point, and it is uniquely enclosed between the slicing planes of two layers (black dashed lines in Figure 5). We denote the layers L and $L + 1$. We denote P_L the

vertical projection of a point P on the top of layer L (resp. bottom of layer $L + 1$), as illustrated in Figure 5. The vertical distance between P and P_L is $\delta_{P_L} = z(P) - z(P_L)$ where $z(.)$ returns the z axis coordinate. Note that we always have $-h/2 \le \delta_{P_L} \le h/2$.

Let us assume that a unique point P projects to a same point P_L, that is the projection from P to P_L is injective as in Figure 5. This assumption is generally true since most printed objects have a feature size much larger that the layer thickness. The point P lies above the slicing plane of L but below the slicing plane of $L + 1$; thus P_L belongs to a top part of the printed surface, visible in Figure 5 (a) as the dashed red segment. Indeed, layer $L + 1$ will not deposit any material at these locations (all points P project outside the layer contour since they are below the slicing plane). Conversely, all points P project inside the contour of layer L, and layer L deposits the material forming the top surface. Thus, for each P_L, displacing the top surface of layer L by δ_{P_L} will align the layer and the surface. None of these displacements exceed $|h/2|$ in magnitude.

This same observation applies for the down-facing part of the surface, this time displacing the bottom surface of layer $L + 1$, as illustrated in Figure 5 (b). Interestingly, this definition correctly opens a 'tear' between layers where the shape exhibits a vertical concavity, as illustrated in Figure 6 (a). This approach is compatible with adaptive slicing, as illustrated in Figure 7.

With this definition, as long as the mapping between surface points P and layer tops/bottoms P_L remains injective, each surface point gets covered by a displaced layer top/bottom. Figure 6 (b) shows a counter example where the surface folds more that once within the slice. In such a case, the displaced layers cannot capture the surface of the input model.. Note that this limitation is intrinsic to mid-layer slicing, which only produces optimal slices when the object shape is monotonous within the slice [15].

We next discuss our practical implementation of this approach, starting from the toolpaths produced by a standard slicer.

3.2. From layers to toolpaths

Given the definition of anti-aliased layers from Section 3.1, we could proceed with generating toolpaths within these new layers. Instead, we make our approach compatible with existing slicers by deforming the *toolpaths* produced by a standard slicer.

When input a closed, two-manifold non self-intersecting geometry, planar slicing produces 2D closed contours within each layer. From each contour
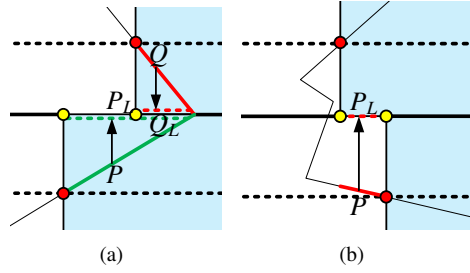

Figure 6: **(a)** Displacing both the bottom and top of layers correctly opens a tear between two adjacent layers. **(b)** Surfaces folding more than once within the slice cannot be properly approximated.
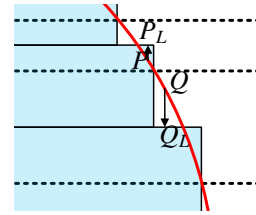

Figure 7: Mapping between surface points P and Q and layer tops/bottoms in adaptive slicing (respectively P_L and Q_L). The magnitude of the displacements remains bounded by half the layer thickness, but the actual value depends on which layer top/bottom is displaced.

the slicer extracts toolpaths for the external perimeters, inner contours, as well as infill paths (see [28] for more details).

We propose to work directly from these toolpaths. Our algorithm starts by re-sampling the paths, inserting vertices in path segments until their length is below the nozzle diameter. We then trace a vertical ray starting from each path vertex towards the model surface above and below. We keep the closest surface point P which gives the distance δ_{P_L}. When P belongs to the top-facing part of the surface and the distance is within the range $[-h/2, h/2]$ we displace the vertex by δ_{P_L}, otherwise we leave it untouched. This effectively displaces the toolpaths towards the top-facing surfaces.

Clearly, this approach can only capture surface details which are bigger than the width of a toolpath (typically w): the toolpath vertices are a subset of the points P_L from Section 3.1 and cannot be used to reproduce an arbitrarily detailed relief. We will see in Section 5 that this results in some limitations. However this improves all surfaces which smoothly vary compared to w – the vast majority of cases, and the cases which produce the worst cases when sliced with flat layers.

Our current implementation does not treat down-facing surfaces. However, the following procedure
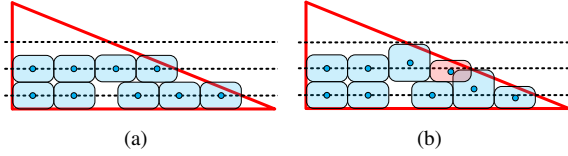
Figure 8: Raising the toolpaths may produce small overlaps where toolpaths do not align across layers. **(a)** Original uniform slice, viewing the cross section of the deposited plastic tracks. **(b)** Anti-aliased layer with displaced plastic tracks, revealing the overlap.
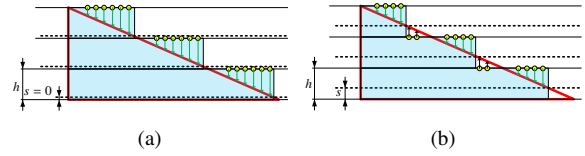

Figure 9: **Left:** $s = 0$ produces only downwards displacements, but exceeds the lower thickness bound. **Right:** Intermediate values less than $\frac{h}{2}$ exploit thinner slices while reducing overlaps across layers.

could be used. Let us assume a dense support from below, sliced using our anti-aliasing algorithm. The support is the complement of the bottom-facing surface, and therefore its top-facing surfaces will have been treated properly by our algorithm. Thus, for the surface we do not need to displace the toolpath vertices. Instead, the plastic flow is adjusted such that the layer is locally thinner. After support removal (e.g. chemical dissolve), the bottom surface will be correctly anti-aliased from below.

3.3. Overlaps and flow adjustment

In general the toolpaths do not align across layers. As a consequence raising a toolpath might generate a small overlap with a toolpath located above, as illustrated in Figure 8. This is again due to the discretization of the layers by the toolpaths: the deformed layers do not overlap themselves.

To avoid any detrimental effect on the final print we adjust the plastic flow. In particular, we detect the overlap between toolpath segments and reduce the flow by the volume of the intersection.

3.4. Slicing plane position

In Section 3.1 we have considered the slicing planes to be in the middle of the layers. This is the typical setup and ensures that our approach is compatible with the toolpaths produced by any slicer. However, we can also change the location of the slicing plane, moving it anywhere within the layer from 0 to h (default being $\frac{h}{2}$).

Let us denote $s \in [0, h]$ the position of the slicing plane. We have seen that for $s = \frac{h}{2}$, layer displacements are in $[-\frac{h}{2}, \frac{h}{2}]$. Figure 9 illustrates the displacements that would occur for other values of s.

Interestingly, setting $s = 0$ only requires downwards displacements, i.e. within $[-h, 0]$. This avoids overlaps across layers (see Section 3.3). However, this also produces layers where the thickness goes *below* the lower feasible bound.
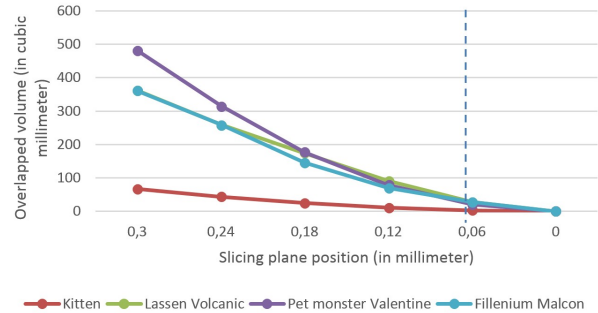

Figure 10: Volume of overlap versus slicing plane position. The dashed line indicates the slicing position with the thinnest possible layers (0.06 mm on our printer). Models are shown in the result section.

Setting s to any intermediate value produces displacements within the range $[s - h, s]$. A benefit of not following the standard procedure would be to select s at the smallest value where deposition is reliable, e.g. 0.06 mm. For our setup, where $w = 0.8, h = 0.6$ this leads to thicknesses within $[0.06, 0.54]$. This fully exploits the capability to produce thin layers while strongly reducing potential overlaps across layers. However, this requires changes to the slicer code to adjust the slicing plane position. Figure 10 reports the potential benefits in terms of overlap reduction for our test models.

We only used the standard setup $s = \frac{h}{2}$ in our results to enforce backward compatibility.

4. Interference avoidance

During plastic deposition the active nozzle moves alongside previously printed paths. With flat layers this is never a problem as the nozzle cannot collide with previously printed paths: all have the same height. However, our algorithm produces toolpaths of varying heights within a same layer. Even though, the height variations are small ($\pm h/2$) they are enough to make the nozzle interfere with printed paths. This is worsen by the nozzle designs which are optimized for printing flat layers and
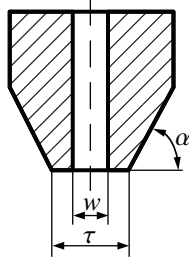
Figure 11: Typical nozzle geometry, sectional view. w is the inner diameter, τ is the outer diameter, α is the nozzle sides inclination.

typically have a wide flat end, as illustrated Figure 11. This has a detrimental effect on our approach, as the heated nozzle printing a low-height path can now interfere with a higher, previously printed path. This will damage the relief, effectively canceling the slope that we are attempting to introduce. This effect is visible in Figure 12. Three neighboring paths A, B, C are printed in two different orders. If higher paths are printed first, then the subsequently printed lower paths damage existing paths. However, if lower paths are printed first, then the subsequently printed higher paths will not interfere.

As just hinted, to print properly the potentially interfering paths have to be ordered from lower to higher heights. In particular, any two paths whose contours have a Hausdorff distance smaller than a threshold might interfere and have to be ordered. We compute the threshold as $\epsilon = \frac{\tau+d}{2} + \Delta h \cdot \cot\alpha$ (this computation is illustrated in Figure 13). To obtain the final threshold we set Δh equal to the layer thickness, as our approach never displaces a tool path by more than the layer thickness. This gives a conservative, larger threshold that detects all possibly conflicting paths. The path width d is generally determined by the inner nozzle width w, and thus we set $d = w$.

Each path can exhibit complex height variations, going up and down several times. Therefore, a path might be lower that its neighbor along a section and become higher later on. To make ordering possible we therefore split neighboring paths into subpaths that can be ordered. An example is shown in Figure 14. In this Figure there are three curved paths: *Path* 1, *Path* 2, *Path* 3. *Path* 1 and *Path* 2 are direct neighbors, as well as *Path* 2 and *Path* 3. They will interfere with one another during printing. *Path* 1 and *Path* 3 are independent as they are too far from one another. As can be seen the paths go up and down several times, and therefore have to be split to allow for an ordering. *Path* 1 is split into A and B, *Path* 2 is split into C, D and E, and *Path* 3 is split
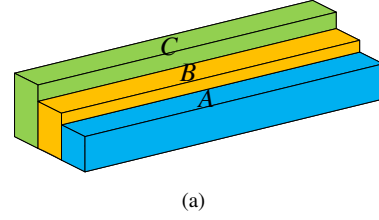
(a)
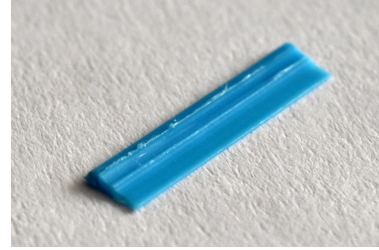
(b)
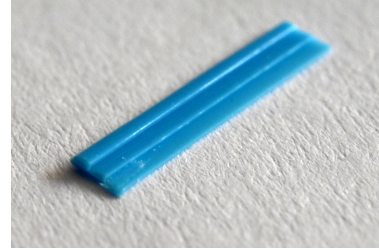
(c)

Figure 12: Interference between neighboring paths. **(a)** The heights of A, B and C are: 0.1 mm, 0.3 mm and 0.6 mm. The distances between the middle axis of A, B and C are $w = 0.8$ mm, which is smaller than the nozzle outer diameter $\tau = 1.25$ mm. **(b)** Printed result in the order C, B, A. **(c)** Printed result in the order A, B, C
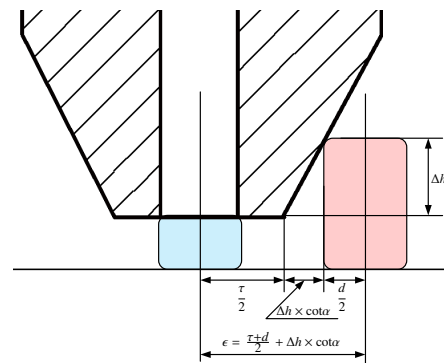

Figure 13: Threshold computation for detecting interferences between paths (cross-section view). The blue rectangle is the cross section of a first path, the red rectangle is the cross section of a second path. τ is the outer diameter of the nozzle, α is the inclination of the nozzle side, d is the path width. Δh is the height difference between the two paths.
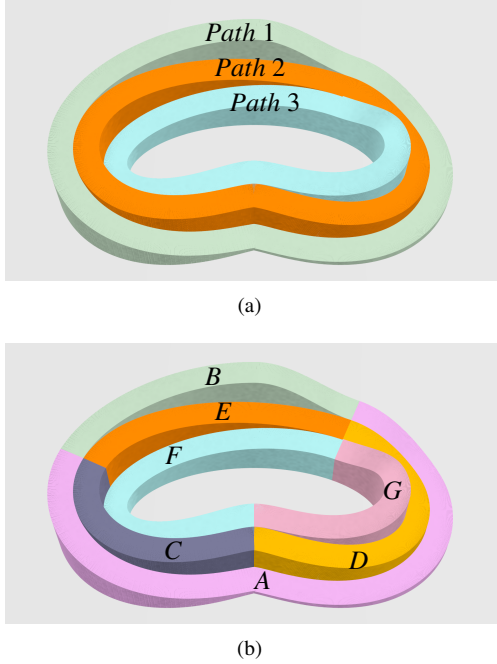
(a)

(b)

Figure 14: Paths are split according to their relative heights. **(a)** Initial three paths. **(b)** Resulting seven subpaths after splitting.
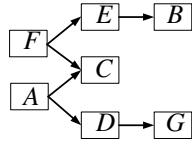

Figure 15: Height constraint graph for the example in Figure 14

into F and G.

Note that paths which were not modified by the anti-aliasing procedure do not need to be ordered and are printed first, in their initial ordering (even if they may be slightly damaged, these paths are hidden from view).

4.1. Height constraint graph

In order to search for an ordering we build a *height constraint graph*. Each node of this directed graph corresponds to a printing path. Node A has an edge pointing to node B if the Hausdorff distance between path A and path B is smaller than τ *and* path A is lower than path B. The edge direction implies that A has to be printed before B. The graph is shown in Figure 15 for the example of Figure 14.

Paths are ordered by performing a topological sort on the height constraint graph. This is an iterative process that at each step selects a node with no edge pointing in

(0 in-degree), adds it to the ordering and deletes all the edges going pointing out of it. The order in which we select the nodes is important for print quality and will be discussed next. This process always terminates as after splitting the graph is acyclic.

4.2. Minimizing the number of seams

Reducing interferences between paths is not the only concern to achieve a good surface quality. After printing each path the extruder has to travel to the next. If the start vertex Q of the following path is far away from the end vertex P of the current path, there will be small gaps left on P and Q, respectively. These gaps are due to the interruption of the plastic extrusion, and produce visible seams on the final print. However, if P and Q approximately coincide, the transition will remain inconspicuous. Since we are splitting paths, the number of potential gaps increases. Therefore we aim to find an ordering sequence leaving a minimum number of gaps, from all possible topological sorts of the constraint height graph.

Let N be the number of all path nodes in the height constraint graph. We use the following objective to represent the number of seams generated by a given path order $S = \{S_i | i = 1, 2, ..., N\}$:

$$E(S) = \sum_{i=1}^{N-1} NumGaps(S_i, S_{i+1}, G) \qquad (1)$$

where $NumGaps(S_i, S_{i+1}, G)$ is the number of gaps generated from path S_i to S_{i+1} given the set of already produced gaps $G = Gaps(S_1, ..., S_{i-1})$. This is a set tracking the location of already introduced gaps. It lets us count gaps only once, giving the opportunity to exploit a previously introduced gap at no cost later in the sequence.

Seam cost. $NumGaps(S_i, S_{i+1}, G)$ evaluates to 0 if the endpoint of path S_i is near the entry point of S_{i+1}, as specified by a user specified tolerance ϵ (we use $\epsilon = 4 \cdot w$). Otherwise, it evaluates to the number of produced gaps – we later describe an improvement where we weight the cost of the gaps, but for the sake of clarity we leave this out for now. Up to two gaps may appear (endpoint of S_i, entry point of S_{i+1}). These are only counted if they are not already in the set G (e.g. they have not been already introduced earlier in the sequence). Therefore, *NumGaps* returns a value in $\{0, 1, 2\}$.
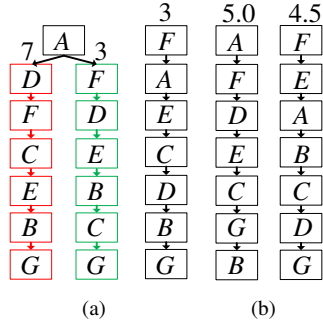
8

Figure 16: Ordering of paths in Figure 14. **(a)** the order starting from *F* and the green order starting from *A* have the smallest number of gaps (3). The red order starting from *A* has the largest (7). **(b)** When considering the concavity of the surface at the gaps, the order starting from *A* has a score of 5.0 while the order starting from *F* has a smaller score of 4.5.

Searching for the best ordering. The algorithm for searching the best ordering enumerates possible orders, tracking the one having minimal cost under Equation 1. The number of orderings is limited since the only choices are between independent paths (e.g. A,F and E,C,D in Figure 15) and only paths that have been touched by the anti-aliasing process are considered. Nevertheless, to avoid a brute force search the algorithm prunes branches as early as possible. There are two pruning opportunities. The first occurs when the cost exceeds the current best. The second occurs when no two paths in the remaining set have matching exit/entry points. In such a case, there is no opportunity to factor gaps and the cost is the remaining number of exit/entry points (twice the number of remaining paths).

Examples are shown in Figure 16 (a) for the case of Figure 14. The path orders {A,F,D,E,B,C,G}, {F,A,E,C,D,B,G} both lead to the smallest number of gaps (3), while {A,D,F,C,E,B,G} gives the worst number of gaps (7). Let us denote *AB* the end point of *A* which is also the start point of *B*. The locations of the gaps generated by the two optimal path orders are respectively {*AB*, *BA*, *GF*} and {*DE*, *FG*, *GF*}. All other transitions do not introduce gaps.

Hiding the gaps. So far we have considered all gaps equivalent. However, some gaps have a stronger visual impact than others. We draw inspiration from works on hiding start/end point of perimeter paths [29] to penalize gaps that are too visible. In particular, gaps are easily noticed if positioned along smooth and convex regions, while they are less visible in concave regions of high curvature, as shown in Figure 17.

We therefore modify *NumGaps* to return a sum of the 'cost' of each gap, which is defined as $(1 + \frac{\theta}{2\pi})$ with θ the angle opening to the outside of the model at the gap location. In Figure 16 (b), the orders {A,F,D,E,C,G,B} and {F,E,A,B,C,D,G} would normally both receive a cost of 5.0. However, the second receives 4.5 as the model is more concave at the produced gaps.

5. Results

We evaluate our technique on a variety of examples. In all cases we use IceSL [30] to generate flat layer paths (but any other slicer could be used). An Ultimaker 2 with PLA filament is used to print. It is fitted with a nozzle having an inner nozzle diameter $w = 0.8$ mm, an outer nozzle diameter $\tau = 1.25$ mm, and inclination $\alpha = 45$ degrees. The base layer thickness is 0.6 mm. All objects are printed with 100% infilling. The z-motions are obtained by adding a z coordinate to all G1 motions in the G-code sent to the printer.

We perform all computations on an Intel Core i7, CPU 4.0 GHz, 32G memory. All the slicing and printing times are recorded in Table 1.

We compare our algorithm with uniform slicing using flat layers at thicknesses 0.6 mm and 0.2 mm, which we denote respectively with labels flat06 and flat02 for compactness.

Wedge. This first example is shown in Figure 1. It reveals the improvement of layer anti-aliasing for a simple, elementary slope. Our result compares favorably to the print with 0.2 mm layers (flat02) which takes *twice* as long to print (using flat06 as reference, our print is only 6% slower, while flat02 is 234 % slower – see Table 1). It takes 30 milliseconds for our anti-aliasing algorithm to process 327 vertices on 51 paths in 6 layers.
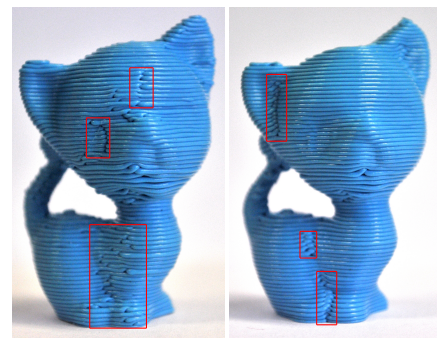

Figure 17: Hiding seams. Seams are highlighted with red rectangles. **Left:** No optimization of seams. **Right:** Result after our algorithm. Most of the visible gaps exist on the print without our technique (extrusion start/stops, also called zippers).

9

Kitten. Our second example is the "Kitten" model, shown in Figure 18. The print time of our result is only 2 % slower than that of flat06, while flat02 is *three times longer* to print. Our result is more accurate than flat06 along silhouettes and gently slopped surfaces. Compared to flat02 silhouettes and gently slopped regions are either similar or slightly better: this is for instance visible in the top row of Figure 18 (arrow).

The surface quality of our prints is affected by the additional gaps we introduce. However their number and impact is kept small as described in Section 4.2. Figure 17 shows the effect of our optimization on the result. Very careful calibration of the print parameters could also further reduce the gaps, but we generally assume this is too high a requirement from all but the most expert users.

It takes 314 milliseconds for our anti-aliasing algorithm to process 105466 vertices in 3078 paths and 146 layers.

Lassen Volcanic. Our third example is a landscape with low slopes, as shown in Figure 19. We printed a subset of the whole model – the red rectangle region in Figure 19, left. As shown in this figure, strong staircases exists in flat06. The surface quality is significantly improved in our result and is comparable with flat02. It even slightly improves where staircases still exist in flat02, see regions highlighted by red rectangles in Figure 19, right. In this example, it takes 9.358 seconds for our anti-aliasing algorithm to process 211508 vertices on 10261 paths in 62 layers.

The print time of our result is nearly identical to that of flat06 (only 2% slower), while flat02 takes more than twice as long to print (226% slower than flat06).

Rendered results. We propose in Figures 20, 21 results which are rendered by our print simulator. This allows to appreciate the effect of the technique without defects due to printer and filament calibration.

Error maps. To reveal the benefits in terms of surface error we compute error maps between the original surface and the surfaces produced by the toolpaths. The error maps are obtained by computing the distance from sampling points on the original model to the closest printed path, accounting for the deposited filament width and height. The distances range from 0.0 mm to 0.3 mm and are color coded.

Figure 22 compares the error maps of flat06, our technique and flat02 for the "Kitten", "Lassen Volcanic", "Pet monster Valentine", and "Fillenium Malcon" models. As can be seen our technique greatly reduces the error along gently curved surfaces compared to flat06 (e.g.
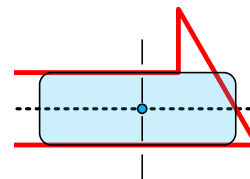

Figure 23: The blue rectangle is a cross-section of a printed plastic path. The red outline is the shape of the anti-aliased layer. The only degree of freedom to deform the path is the blue vertex. In such a case, the toolpaths cannot be modified to reproduce the sharp features of the anti-aliased layers.

the top and back of "Kitten", the plateaus of "Lassen Volcanic"). These areas are also improved when compared to flat02, even though regions at greater angles still have more error (this is also visible on the printouts, e.g. the kitten ear in Figure 18 top row).

The error map of our result for "Kitten" (denoted (b) in Figure 22) reveals a limitation of our technique. The top of the model has a greatly reduced error, but the error increases again when the angle grows beyond a threshold. This is due to the fact that the deposited plastic tracks are too large in these regions to reproduce the sharp features of the anti-aliased layer relief, as illustrated in Figure 23 and discussed in Section 3.2.

Generally the distance between neighboring paths is determined by the inner nozzle diameter w. Let the original layer thickness to be h. For an arbitrary vertex on the model surface, the angle between the tangent plane and the XY plane should be at most $\theta = \arctan(\frac{h}{w})$ to make infill paths available for anti-aliasing. With our setup this gives a critical angle of 37 degrees ($w = 0.8, h = 0.6$).

Nevertheless, our technique greatly reduces the error where it is worst in flat06, while preserving a nearly identical print time.

6. Conclusion

We proposed a technique to reduce the staircase aliasing defects that plague filament deposition. Our approach proposes to exploit small, sub-layer z-motions to better reproduce gently sloped areas. It does not require to globally curve the layers, nor does it require specific changes to existing printers. In particular our technique considers within-layer nozzle interference, splitting and reordering paths to minimize its detrimental effect.

The main limitation of our technique stems from the discretization of the layers into contours (plastic tracks) of a same width, as discussed in Section 3.2 and Section 5. A second drawback of our approach is that it

Table 1: Slicing time (in seconds) / printing time (in seconds) / G-Code file size (in MBs) comparisons between: flat06, flat02 and our algorithm. #Order is the number of complete orders considered by our algorithm.

Cases	flat06	flat02	our algorithm	#Order
Wedge	0.625 / 61	1.927 / 143	0.625+0.03 / 64	38
Kitten	57.772 / 4767	156.690 / 14234	57.772+0.314 / 4859	40293
Lassen Volcanic	342.474 / 25083	1039.691 / 66731	342.474+9.358 / 30434	256219
Pet monster Valentine	421.455 / 35637	1418.160 / 105470	421.455+25.815 / 36107	164494
Fillenium Malcon	183.858 / 15602	573.254 / 46860	183.858+25.515 / 15614	56363

introduces additional gaps onto the print surface, even though their number of location is optimized to reduce their impact. Overall, the improvement in accuracy and visual quality of the silhouettes is very significant.

Finally, it is worth noting that our technique can be applied on the output of any slicer, as long as both the G-Code and the original mesh are available. Therefore, we hope to see a wide adoption of this approach, which produces more accurate and visually pleasing surfaces at negligible cost in print time.

Acknowledgments

References

[1] M. Taufik, P. K. Jain, Role of build orientation in layered manufacturing: A review, International Journal of Manufacturing Technology and Management 27 (1/2/3) (2013) 47.

[2] X. Zhang, X. Le, A. Panotopoulou, E. Whiting, C. C. L. Wang, Perceptual models of preference in 3d printing direction, ACM Transactions on Graphics 34 (6) (2015) 1–12.

[3] A. Dolenc, M. I., Slicing procedures for layered manufacturing techniques, Computer-Aided Design 26 (2) (1994) 119–126.

[4] E. Sabourin, S. A. Houser, J. H. Bhn, Adaptive slicing using stepwise uniform refinement, Rapid Prototyping Journal 2 (4) (1996) 20–26.

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

[6] R. L. Hope, R. N. Roth, P. A. Jacobs, Adaptive slicing with sloping layer surfaces, Rapid Prototyping Journal 3 (3) (1997) 89–98.

[7] M. T. Hayasi, B. Asiabanpour, A new adaptive slicing approach for the fully dense freeform fabrication (fdff) process, Journal of Intelligent Manufacturing 24 (4) (2013) 683–694.

[8] W. Wang, H. Chao, J. Tong, Z. Yang, X. Tong, H. Li, X. Liu, L. Liu, Saliency-preserving slicing optimization for effective 3d printing: Saliency-based slicing optimization, Computer Graphics Forum 34 (6) (2015) 148–160.

[9] A. Boschetto, L. Bottini, Triangular mesh offset aiming to enhance fused deposition modeling accuracy, The International Journal of Advanced Manufacturing Technology 80 (1-4) (2015) 99–111.

[10] Z. Zhao, Z. Luc, Adaptive direct slicing of the solid model for rapid prototyping, International Journal of Production Research 38 (1) (2000) 69–83.

[11] K. Tata, G. Fadel, A. Bagchi, N. Aziz, Efficient slicing for layered manufacturing, Rapid Prototyping Journal 4 (4) (1998) 151–167.

[12] E. Sabourin, S. A. Houser, J. H. Bhn, Accurate exterior, fast interior layered manufacturing, Rapid Prototyping Journal 3 (2) (1997) 44–52.

[13] J. Tyberg, J. H. Bhn, Local adaptive slicing, Rapid Prototyping Journal 4 (3) (1998) 118–127.

[14] K. Mani, P. Kulkarni, D. Dutta, Region-based adaptive slicing, Computer-Aided Design 31 (5) (1999) 317–333.

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

[16] R. Hu, H. Li, H. Zhang, D. Cohen-Or, Approximate pyramidal shape decomposition, ACM Trans. Graph. 33 (6) (2014) 213–1.

[17] W. M. Wang, C. Zanni, L. Kobbelt, Improved surface quality in 3d printing by optimizing the printing direction, Computer Graphics Forum 35 (2) (2016) 59–70.

[18] D. Chakraborty, B. A. Reddy, A. R. Choudhury, Extruder path generation for curved layer fused deposition modeling, Computer-Aided Design 40 (2) (2008) 235–243.

[19] S. Singamneni, A. Roychoudhury, O. Diegel, B. Huang, Modeling and evaluation of curved layer fused deposition, Journal of Materials Processing Technology 212 (1) (2012) 27–35.

[20] B. Huang, S. Singamneni, Alternate slicing and deposition strategies for fused deposition modelling of light curved parts, J. of Achievem in mat and manuf 55 (2) (2012) 511–517.

[21] R. J. A. Allen, R. S. Trask, An experimental demonstration of effective curved layer fused filament fabrication utilising a parallel deposition robot, Additive Manufacturing 8 (2015) 78–87.

[22] S. Mueller, S. Im, S. Gurevich, A. Teibrich, L. Pfisterer, F. Guimbretire, P. Baudisch, Wireprint: 3d printed previews for fast prototyping, in: Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST 14, Association for Computing Machinery (ACM), 2014.

[23] R. E. Williams, V. L. Melton, Abrasive flow finishing of stereolithography prototypes, Rapid Prototyping Journal 4 (2) (1998) 56–67.

[24] P. M. Pandey, N. V. Reddy, S. G. Dhande, Virtual hybrid-fdm system to enhance surface finish, Virtual and Physical Prototyping 1 (2) (2006) 101–116.

[25] Y. Pan, X. Zhao, C. Zhou, Y. Chen, Smooth surface fabrication in mask projection based stereolithography, Journal of Manufacturing Processes 14 (4) (2012) 460–470.

[26] Y. Pan, Y. Chen, Smooth surface fabrication based on controlled meniscus and cure depth in microstereolithography, Journal of Micro and Nano-Manufacturing 3 (3) (2015) 031001.

[27] I. B. Park, Y. M. Ha, S. H. Lee, Dithering method for improving

the surface quality of a microstructure in projection microstereolithography, The International Journal of Advanced Manufacturing Technology 52 (5-8) (2011) 545–553.

[28] H. Q. Dinh, F. Gelman, S. Lefebvre, F. Claux, Modeling and toolpath generation for consumer-level 3d printing, in: ACM SIGGRAPH 2015 Courses, SIGGRAPH '15, 2015, pp. 17:1–17:273.

[29] J. Hergel, S. Lefebvre, Clean color: Improving multi-filament 3d prints, in: Computer Graphics Forum, Vol. 33, Wiley Online Library, 2014, pp. 469–478.
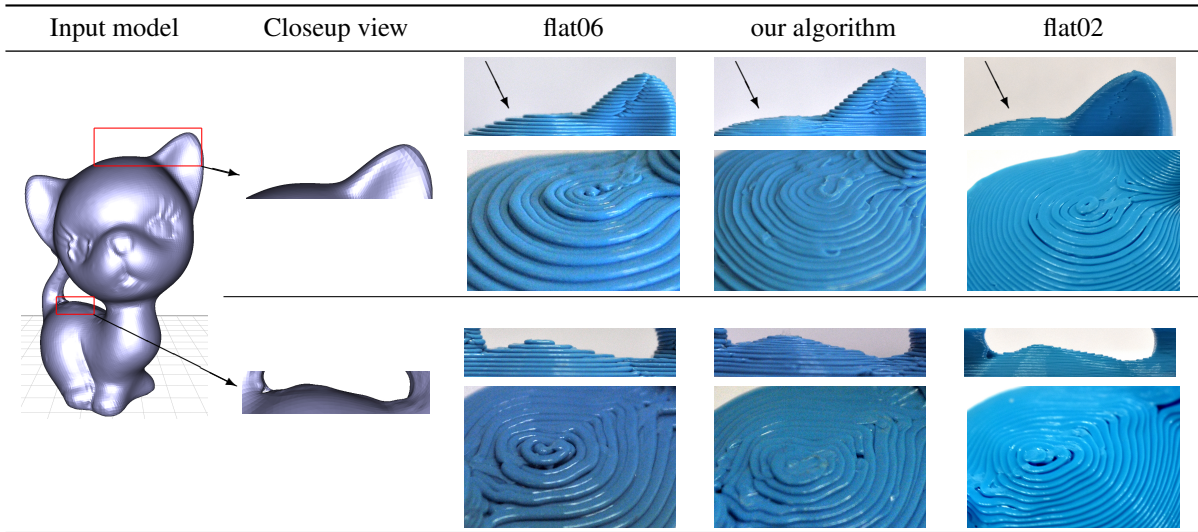
[30] http://www.loria.fr/~slefebvr/icesl/.

Figure 18: Real printouts of "Kitten" (thing: 12694). Leftmost column: input 3D model and closeup regions outlined with red rectangles. Second column: 3D model, closeup views. Third column: flat06, side and top views on real printout. Fourth column: our result, side and top views. Fifth column: flat02, side and top views.
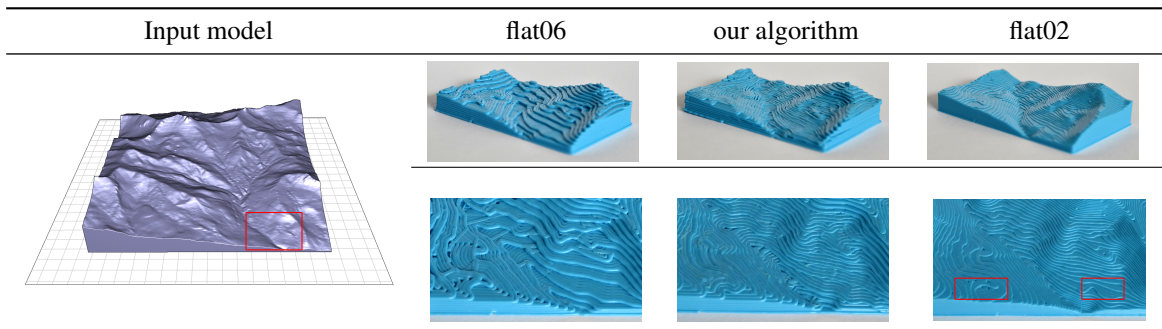

Figure 19: Real printouts of "Lassen Volcanic" (thing: 667269). Leftmost column: input 3D model and closeup regions outlined with red rectangle. Second column: flat06, general and closeup views. Third column: our result, general and closeup views. Fourth column: flat06, general and closeup views.
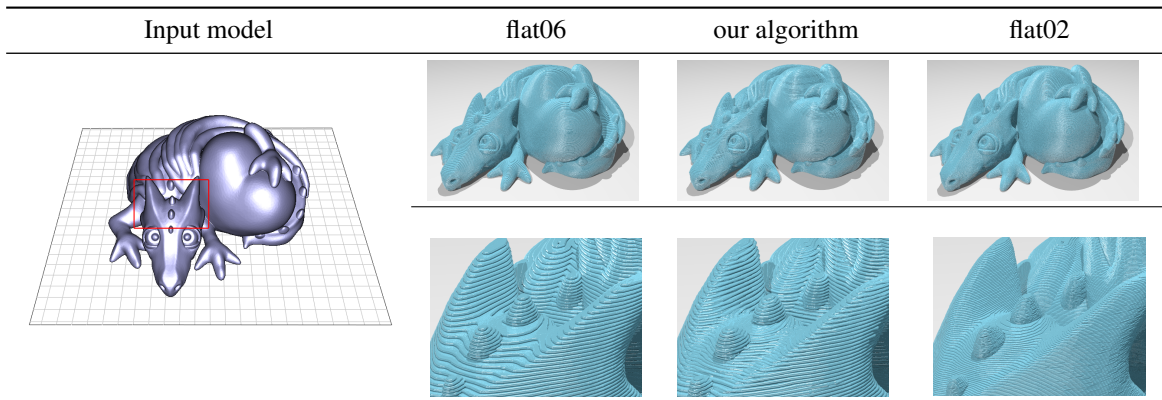

Figure 20: **Rendered** printouts of "Pet monster Valentine" (thing: 17204). Leftmost column: input 3D model and closeup regions outlined with red rectangle. Second column: flat06, general and closeup views. Third column: our result, general and closeup views. Fourth column: flat02, general and closeup views.
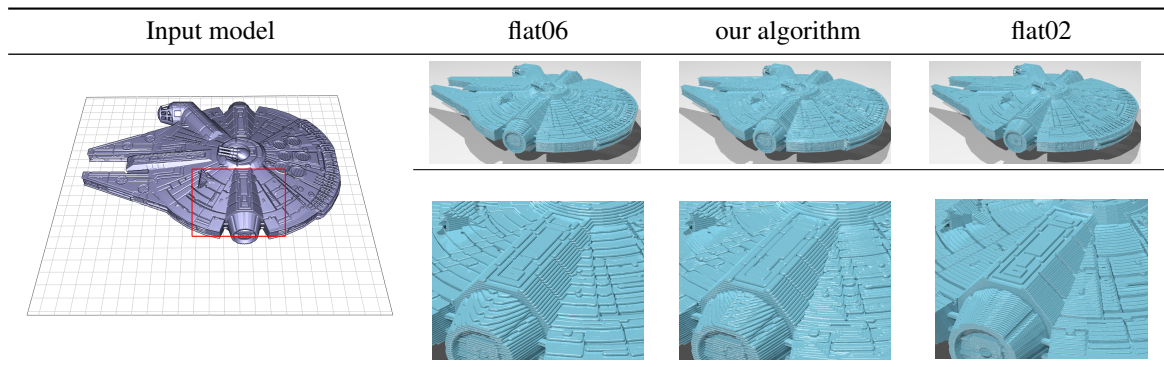
Input model	flat06	our algorithm	flat02

Figure 21: **Rendered** printouts of "Fillenium Malcon" (thing: 919475). Leftmost column: input 3D model and closeup regions outlined with red rectangle. Second column: flat06, general and closeup views. Third column: our result, general and closeup views. Fourth column: flat02, general and closeup views.
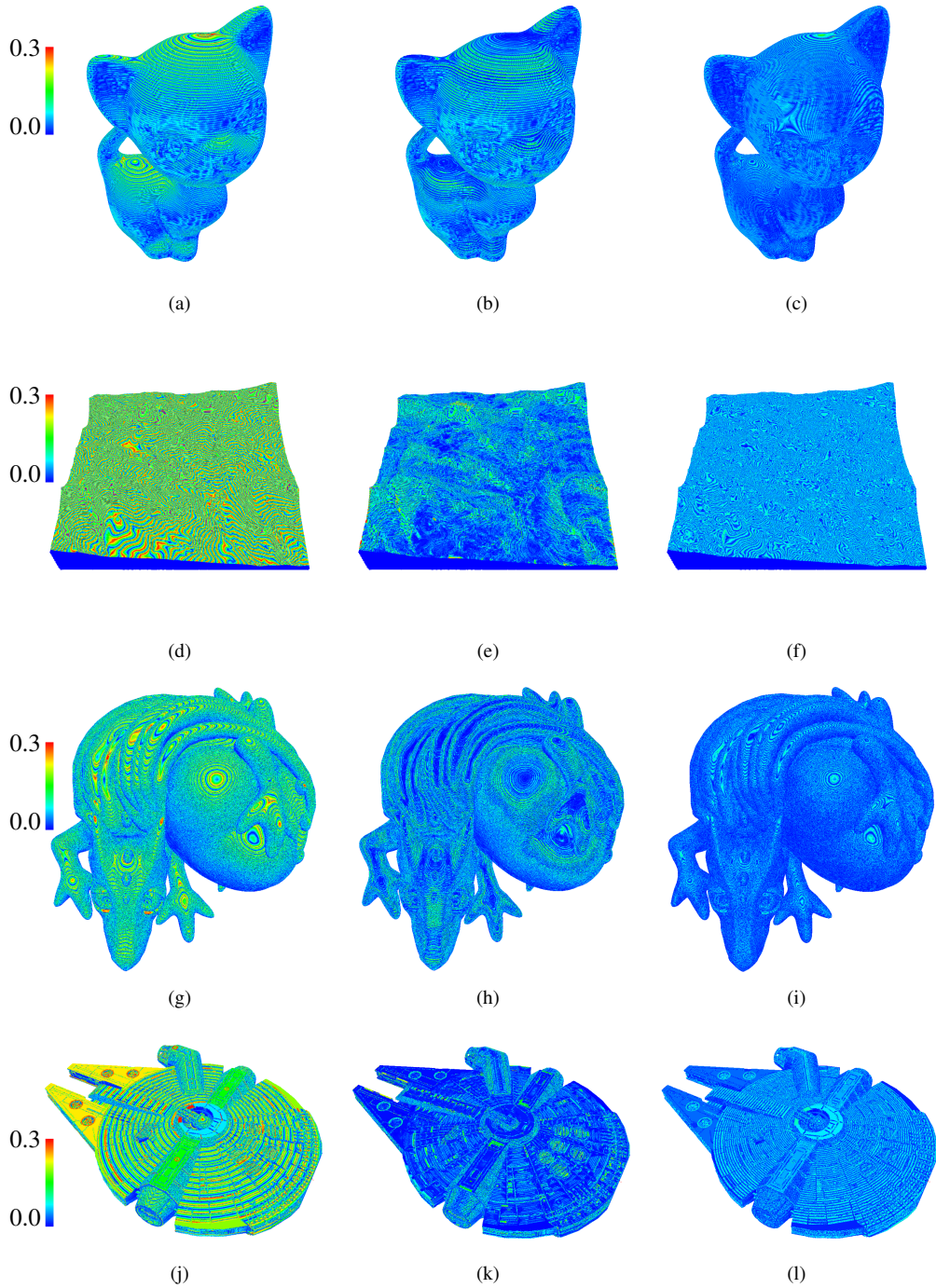
Figure 22: Error map of "Kitten": (a) flat06, (b) our algorithm, (c) flat02; "Lassen Volcanic": (d) flat06, (e) our algorithm, (f) flat02; "Pet monster Valentine": (g) flat06, (h) our algorithm, (i) flat02; "Fillenium Malcon": (j) flat06, (k) our algorithm, (l) flat02