CurviSlicer: Slightly curved slicing for 3-axis printers

JIMMY ETIENNE, Université de Lorraine, CNRS, Inria, LORIA
NICOLAS RAY, Université de Lorraine, CNRS, Inria, LORIA
DANIELE PANOZZO, New York University
SAMUEL HORNUS, Université de Lorraine, CNRS, Inria, LORIA
CHARLIE C. L. WANG, The Chinese University of Hong Kong
JONÀS MARTÍNEZ, Université de Lorraine, CNRS, Inria, LORIA
SARA MCMAINS, University of California, Berkeley
MARC ALEXA, Technische Universität Berlin
BRIAN WYVILL, University of Victoria
SYLVAIN LEFEBVRE, Université de Lorraine, CNRS, Inria, LORIA

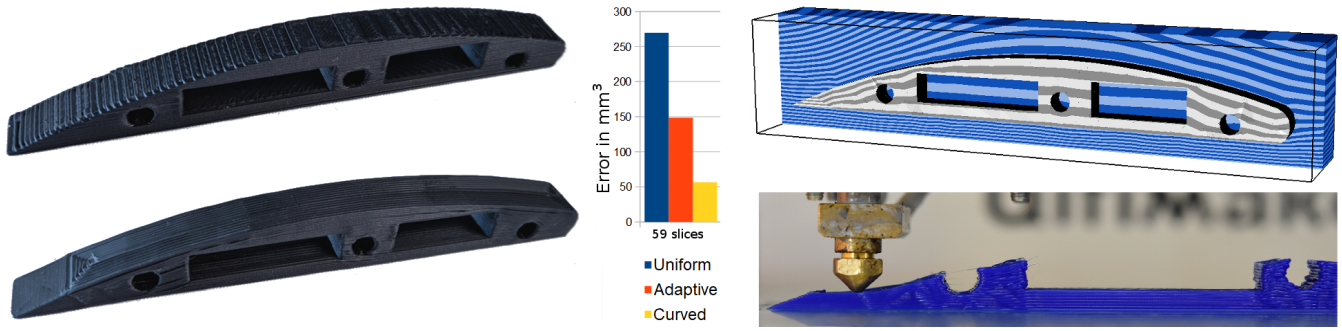


Fig. 1. Our technique curves deposition paths to improve parts printed with fused filament fabrication. Compared to state of the art adaptive slicing (top left) which is limited to planar layers, our print (bottom left) has a smooth surface finish while using the same number of layers (40). The reproduction accuracy is improved overall (middle graph), with a total volume error of 57mm^3 compared to the 149mm^3 of adaptive slicing. Our approach computes a continuous deformation of space (top right) under fabrication constraints (thicknesses, slope). The produced toolpaths are guaranteed to print without collisions on standard 3-axis 3D printers, here an Ultimaker2 (bottom right).

Most additive manufacturing processes fabricate objects by stacking planar layers of solidified material. As a result, produced parts exhibit a so-called staircase effect, which results from sampling slanted surfaces with parallel planes. Using thinner slices reduces this effect, but it always remains visible where layers *almost* align with the input surfaces.

In this research we exploit the ability of some additive manufacturing processes to deposit material slightly out of plane to dramatically reduce these artifacts. We focus in particular on the widespread Fused Filament Fabrication (FFF) technology, since most printers in this category can deposit along slightly curved paths, under deposition slope and thickness constraints.

Our algorithm curves the layers, making them either follow the natural slope of the input surface or on the contrary, make them intersect the surfaces at a steeper angle thereby improving the sampling quality. Rather than directly computing curved layers, our algorithm optimizes for a deformation of the model which is then sliced with a standard planar approach. We demonstrate that this approach enables us to encode all fabrication constraints, including the guarantee of generating collision-free toolpaths, in a convex optimization that can be solved using a QP solver.

We produce a variety of models and compare print quality between curved deposition and planar slicing.

CCS Concepts: • **Computing methodologies → Shape modeling**.

Additional Key Words and Phrases: curved slicing, additive manufacturing

ACM Reference Format:
Jimmy Etienne, Nicolas Ray, Daniele Panozzo, Samuel Hornus, Charlie C. L. Wang, Jonàs Martínez, Sara McMains, Marc Alexa, Brian Wyvill, and Sylvain Lefebvre. 2019. CurviSlicer: Slightly curved slicing for 3-axis printers. *ACM Trans. Graph.* 38, 4, Article 81 (July 2019), 11 pages. https://doi.org/10.1145/3306346.3323022

Authors' addresses: Jimmy Etienne, Université de Lorraine, CNRS, Inria, LORIA; Nicolas Ray, Université de Lorraine, CNRS, Inria, LORIA; Daniele Panozzo, New York University; Samuel Hornus, Université de Lorraine, CNRS, Inria, LORIA; Charlie C. L. Wang, The Chinese University of Hong Kong; Jonàs Martínez, Université de Lorraine, CNRS, Inria, LORIA; Sara McMains, University of California, Berkeley; Marc Alexa, Technische Universität Berlin; Brian Wyvill, University of Victoria; Sylvain Lefebvre, Université de Lorraine, CNRS, Inria, LORIA.

0730-0301/2019/7-ART81 $15.00
https://doi.org/10.1145/3306346.3323022

1 INTRODUCTION

Additive manufacturing processes fabricate physical objects by progressively depositing solidified material, forming a solid. In the vast majority of cases, the deposition is performed layer by layer, where each layer is a planar slab of the object. This constraint sometimes stems from the implementation of the process itself; for instance selective laser sintering technologies melt (and solidify) the flat surface of a powder tank. However, many processes offer additional degrees of freedom. In particular, Fused Filament Fabrication (FFF) allows one to deposit molten plastic along curved paths, as long as it is deposited onto an existing surface [Chakraborty et al. 2008].

This opportunity is used in recent methods tackling the generic problem of free form deposition, e.g. using the 6-DOF of a robotic arm [Dai et al. 2018]. We refer to such methodologies as *curved* slicing and deposition. While offering many advantages over traditional flat deposition, these systems require expensive hardware (6-DOF robotic arms or 5-axis motion platforms), limiting their applicability.

We propose an algorithmic solution to enable curved deposition using standard FFF machines, with the only requirement of having a specific nozzle shape—no flat area around the molten filament exit hole—which is the case on many printers already. Since replacement nozzles for the most common printers are available at a modest price (< 10 USD), our contribution has the potential to be widely adopted by makers and 3D printing companies.

The key idea of our technique is to (1) compute a deformation of the input object, (2) slice the deformed solid using standard uniform planar slicing, and then (3) deform the toolpaths back into the original space. By constraining the volumetric mapping, we guarantee that there will be no collision between the part and the extrusion device and that deposition thicknesses remain within feasible bounds. We propose a specific parameterization of the problem that reduces it to a simple set of constraints which can be solved using a standard QP solver. The mapping is optimized such that during slicing the surface reproduction error is reduced. When mapping back to the initial space, the deposition paths become curved, but the surface quality improvement is preserved.

Our technique significantly improves surface accuracy and finish over traditional FFF, in particular strongly reducing the staircase defects due to planar layering. In fact, our technique compares favorably to previous methods relying on robotic arms, offering a highly effective solution with minimal hardware requirements. To foster adoption of our technique, we will release an open source, reference implementation of our method, in addition to the toolpaths for all our results and detailed instructions for modifying existing FFF printers to achieve the best results.

2 PREVIOUS WORK

As most additive manufacturing processes solidify objects layer by layer, the impact of layering on surface roughness and part accuracy has been extensively studied. For an in-depth review of how processing of a part relates to its final quality we refer to the survey by Livesu et al. [2017].

Approaches fall into different categories: adapting the layer thicknesses, changing the part orientation, splitting parts, curving deposition. We discuss each of these below.

Adaptive slicing. Many additive processes allow for the thickness of an entire layer to be changed within some bounds. Therefore, methods have been proposed to adapt the layer thicknesses to better capture the part geometry [Pandey et al. 2003]. A first approach consists of choosing the thickness using the local surface slope [Dolenc and Mäkelä 1994]. Other methods follow splitting strategies, starting from an initial uniform slicing and then dividing or fusing slices [Hayasi and Asiabanpour 2013; Hope et al. 1997; Kulkarni and Dutta 1996; Sabourin et al. 1996; Tyberg and Bøhn 1999].

Recently, global approaches have been proposed. Wang et al. [2015] formulate a global optimization of the thicknesses, minimizing the worst cusp height [Dolenc and Mäkelä 1994] in each layer. Alexa et al. [Alexa et al. 2017] propose a provably optimal adaptive slicing algorithm, in the discrete setting. This approach minimizes the overall *volume error*, that is the volume incorrectly assigned in the sliced part [Masood et al. 2000; Tata et al. 1998]. We compare our work to optimal adaptive slicing in Section 5.

One drawback of standard adaptive slicing is that it maintains the same thickness everywhere within the layer. To address this limitation, some approaches divide a part in multiple regions, and use different layerings in each [Mani et al. 1999; Sabourin et al. 1997; Tyberg and Bøhn 1998; Wang et al. 2015]. The object still prints as a single part. One difficulty is that the abrupt change of layering thickness results in visible scars along the surface. Curving the layers avoids this problem.

Orientation. The orientation of a shape plays an important role in the final part quality [Livesu et al. 2017]. While orientation impacts many factors, such as structural strength [Umetani and Schmidt 2013] and aesthetics [Zhang et al. 2015], several techniques orient the part to minimize errors due to layering [Cheng et al. 1995; Thrimurthulu et al. 2004].

While we preserve the global orientation of the input, existing techniques for global orientation optimization could be used as a front-end. By curving the layers our approach further locally adapts orientation to the part geometry.

Partitioning. A number of approaches split the input into different parts, fabricated separately and assembled later. Different objectives are sought for: fitting large parts in printers with limited extent [Luo et al. 2012] or boxes of specific sizes [Attene 2015], fabricating large objects with an empty inner core [Song et al. 2016], avoiding support structures [Hu et al. 2014], finding optimal orientations for slicing subparts [Hildebrand et al. 2013; Wang et al. 2016].

Our technique is complementary to these approaches, as it can further improve the quality of each individual part.

Curving layers. Curved layers and their properties have been prototyped by several researchers. [Chakraborty et al. 2008] considers deposition paths along curved surfaces with the objective of obtaining stronger parts. The benefits on structural strength of curved layers are further studied in [Huang and Singamneni 2012; Singamneni et al. 2012]. Existing techniques typically combine flat layers in the core with curved layers on the last few visible surfaces [Allen and Trask 2015; Huang and Singamneni 2015][Ahlers 2018] or overlay thin skins on top of existing models [Thomas et al. 2016]. Lim et al. [2016] perform similar experiments to fabricate large scale

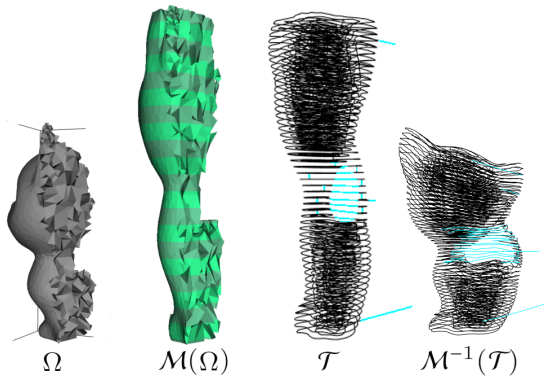


Fig. 2. Overview. We start from a 3D model Ω and optimize a mapping $\mathcal{M}$. We slice the deformed model $\mathcal{M}$ to obtain the toolpaths $\mathcal{T}$, which are mapped back into the initial space as $\mathcal{M}^{-1}(\mathcal{T})$ for fabrication.

concrete panels, with a system akin to fused filament fabrication. Overall, curved layers provide smoother, stronger surfaces.

Printing shell-like, constant thickness curved layers on top of flat layers is very practical but has two drawbacks. First, filament along curved surfaces may sag in the staircases underneath, leaving porosities. Instead our approach progressively curves and thickens layers, avoiding abrupt transitions while leaving no gaps inside the parts. Second, the error cannot improve for slopes exceeding the printable angle. Our algorithm introduces the idea of verticalizing to address such cases (detailed in Section 3.2).

Song et al. [2017] propose a general approach that curves toolpaths by small amounts after slicing: the layers remain flat, but have tiny height variations compensating the staircase defect. However, the technique cannot globally curve the paths nor precisely follow a slanted surface. Ezair et al. [2018] generate curve toolpaths within volumes following a user specified trivariate parameterization. Curved layers are produced along the isolvalue of one parameterization variable, ensuring a proper coverage is obtained—overall equal spacing—by trimming layers in close proximity. The layers are covered with curves in a similar manner. A set of 3-axis toolpaths is obtained by further splitting and ordering the curves to avoid collisions during deposition. We take a different point of view. Our technique optimizes a parameterization that allows to slice the object in the parametric domain using any standard planar slicer. The produced toolpaths are guaranteed to be fabricable when mapped back into the initial domain, without requiring splitting or re-ordering. The parameterization is automatically optimized to obtain smooth surface tops and to globally reduce staircase defects.

Additional degrees of freedom. The approaches we have discussed so far target 3-axis printers. A number of techniques have been proposed to physically realize parts using 5-DOF or 6-DOF systems.

Keating et al. [2013] demonstrated a first prototype fabricating parts of limited complexity with a 6-DOF robotic system. Pan et al. [2014] rely on a multi-axis device to fabricate additional features along existing curved surfaces. Chen et al. [2017] fabricate parts with planar layers but use the rotational capabilities of a robotic arm to change the part orientation mid-print.

Multi-axis systems have also been used to create wireframe models, for example to help prototype shapes quickly [Mueller et al. 2014; Peng et al. 2016]. Algorithms have been proposed for multi-axis toolpath planning of arbitrary wireframe objects [Huang et al. 2016; Wu et al. 2016].

Dai et al. [2018] introduce a general algorithm to fabricate parts with a robotic arm while avoiding the need for supports. Paths are curved throughout the parts, albeit with a constant thickness.

While extremely promising, 6-DOF 3D printing requires special equipment and a relatively complex setup. The part quality is also not currently on-par with that of 3-axis printing, which is wide spread and well understood. Our objective in this paper is to allow for standard 3-axis printers to print curved objects, enabling novel possibilities on the wide variety of printers already installed in workshops, schools, FabLab and homes.

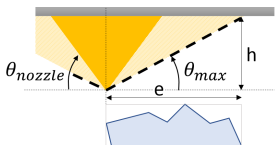
3 OVERVIEW

Our approach starts from a mesh Ω correctly defining a solid (e.g. an STL file for fabrication), and oriented such that the Z axis is the build direction. We then optimize for a mapping $\mathcal{M}$ from the object space to the slicing space. The mesh to be sliced is obtained through the mapping as $\mathcal{M}(\Omega)$. A standard slicer is then called to produce a set of toolpaths $\mathcal{T}$, using uniform slicing. The final toolpaths used for fabrication are obtained through the inverse mapping $\mathcal{M}^{-1}(\mathcal{T})$. Our algorithm optimizes a deformation for both the object inside and surrounding empty space, thus allowing for travel paths and auxiliary structures (e.g. supports) to be properly curved alongside the object. The pipeline is illustrated in Figure 2.

The crux of the problem is how to compute $\mathcal{M}$ (and its inverse) so as to enforce all constraints while minimizing surface defects. In terms of constraints, we have to ensure that no collisions occur and that the deposition thickness remains within minimum and maximum bounds after mapping (Section 3.1). In terms of objective, we seek to improve surface finish, to reduce staircasing and to accurately follow the initial surface (Sections 3.2 and 3.3).

3.1 Fabrication constraints

In this work we face two main fabrication constraints. The first relates to avoiding collision between the extrusion device (nozzle, extruder and carriage) while the second relates to feasible deposition thicknesses. We assume the printer to be equipped with a pointed, conical nozzle without a flat area around the exit hole.



We model the collision constraint as an inverted cone. The Figure inset shows the cone constraint (dashed line), the printer nozzle (orange), the carriage (gray) and the object below (blue). The constraint cone has its apex aligned with the nozzle tip and represents the forbidden space above it (lighter orange hatches). It has to guarantee that if the part remains below the cone's surface, no collisions can occur with any part of the printer—first and foremost the nozzle itself.

The collision cone must at least contain the nozzle, which typically has a conic geometry. We denote the angle of the nozzle cone with

respect to the horizontal by θ_{nozzle} (see inset Figure). We denote by θ_{max} the collision cone angle with respect to the horizontal. It is obtained as $\theta_{max} = min(\theta_{nozzle}, \tan^{-1} \frac{h}{e})$, where h is the distance between the nozzle tip and carriage and e the maximum XY extent of the printed object.

During deposition, we have to ensure that no already printed path enters the forbidden cone. This translates into a local constraint on the slope of the paths after inverse mapping: how quickly they are allowed to raise in Z by units of X and Y.

Deposition thicknesses are bounded by a maximum and minimum. The maximum stems from the nozzle exit diameter. A general rule of thumb is that the maximum thickness should stay within $[0.1d, 0.75d]$ where d is the nozzle diameter. This gives from 0.04mm to 0.3 mm for a typical 0.4 mm nozzle. The minimal thickness constraint is also impacted by printer mechanical quality and calibration accuracy. Indeed, as thickness decreases, even small calibration errors start to have a large impact. For do-it-yourself printers, the typical minimal thickness is 0.1 mm – even though well calibrated printers reliably print at 0.05 mm or less. We denote by τ_{min} and τ_{max} the minimum and maximum thicknesses.

3.2 Improving surfaces

The main defect produced by planar slicing is the so-called staircasing effect: the emergence of visible steps along the build direction. The defect is more pronounced when the slope of the surfaces decreases: the vertical sampling density of the planar slices is no longer sufficient to prevent large steps from appearing between slices (see Figure 3). This leads to the observation that printed vertical walls exhibit minimal staircase, while gently sloped surfaces suffer the largest staircase defects. While using thinner slices reduces the size of the defect, it remains visible on surfaces with low slopes. In addition, thinner slices imply longer print times.

There is, however, a notable special case: a surface that is exactly flat produces no staircase, and thus is ideally reproduced *if a slice exactly aligns with the surface*.

To improve surface reproduction, we seek to minimize the staircase error of the printed part. The approach we propose is to attempt to make all surfaces either vertical or flat during the slicing step as both cases lead to minimal errors. The effect is to curve the slices, locally adapting to the surface slopes, as illustrated in Figure 4. Of course, due to the fabricability constraints, this ideal objective can only be approximated.

Note that to avoid errors during slicing on flat areas, the surface has to exactly align with the top of a slice. Otherwise, large errors occur due to misalignment. This is a specific source of concern that we address in our approach.

3.3 Choosing whether to flatten surfaces

A key question when computing the deformation is which set of surfaces should be flattened. Our proposal is that ideally we would like to flatten as many faces as possible—these regions later print as curved surfaces with no staircase error, and offer accurate reproduction when well aligned with the tops of slices. However, if we seek to flatten a surface, we have to ensure it ends up being reproduced by a single, well aligned slice. A misalignment or residual slope would have the surface intersect a slicing plane, resulting in a large staircase error.

We therefore initially attempt to flatten all surfaces that could be possibly reproduced under the maximum printable slope θ_{max}. This typically leads to the most accurate results (we provide measurements in Section 5.3). Optionally the user can choose a smaller initial set; this allows for selection of which surfaces are flattened.

The initial set usually leads to an infeasible problem due to fabrication constraints. The freedom required to incline, compress, or stretch slices around large flattened surfaces is often not available.

This leads to a challenging problem: flattening cannot be simply expressed as soft constraints as any small violation immediately results in worst-case stairstepping errors. Instead, we propose an iterative scheme were we eliminate infeasible flattening requirements progressively, turning them into surfaces to be inclined. The scheme is described in Section 4.4.

Note that we never attempt to flatten downward facing triangles, as in general these could not print without adding supports in a 3-axis system. There is one exception. The triangles that are flat in the input—either downward facing or upward facing—are *always* constrained to remain flat. This, in particular, implies that the object's bottom remains flat.

4 ALGORITHM

Prior to any computation, we remesh the input with TetWild [Hu et al. 2018], to obtain a tetrahedral mesh Γ for both the inside and outside volume, as well as the triangle mesh Ω at their interface.

Minimizing staircases amounts to making surfaces either exactly flat, or making them as vertical as possible (Sections 3.2 and 3.3). To

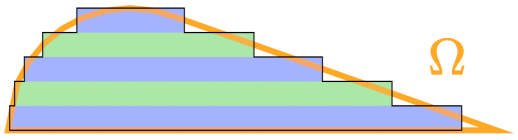


Fig. 3. Slicing a part with uniform slicing, side view (actual result). The model outline to be matched is in orange, the slices are shown in green/blue. The staircase is more pronounced when the slope of the surfaces decreases: the left part suffers less from the staircase defect than the right part.

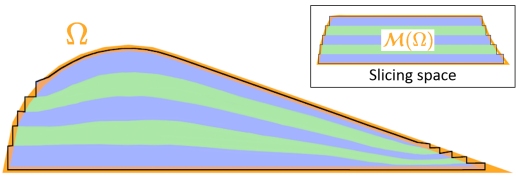


Fig. 4. Slicing a part with curved slicing, side view (actual result). The model outline to be matched is in orange, the slices are shown in green/blue. Our algorithm curves the slices to accurately follow the object surface whenever possible. In this case the slice thicknesses are allowed to vary from 0.1mm to 0.6mm, while the maximum slope is constrained to $\theta_{max} = 30$ degrees. The inset shows the deformed space where slices have been produced. Note the alignment of the top surface with the top of the last slice.

achieve this we optimize for a mapping that deforms the surfaces along the vertical direction only, locally compressing or stretching the initial solid and changing surface slopes. This parameterization of the problem is in line with the constrained motions of a 3–axis system, while leading to a practical optimization scheme only involving minimizing for quadratic objectives under linear constraints.

We represent the mapping $\mathcal{M}$ as a deformation field of Γ, computing a new vertical position $h(p)$ for each of its vertices p. These new positions define a continuous deformation field within the volume of Γ: any point inside can be mapped through linear interpolation from the enclosing tetrahedron vertices. Swapping optimized and original coordinates switches between $\mathcal{M}$ and $\mathcal{M}^{-1}$.

The unknown variables of the problem are thus the values $h(p)$, which will be optimized to define a mapping under the desired objective and constraints.

Additional notation. In the following we denote by $x(p)$, $y(p)$ and $z(p)$ the coordinates of vertices p of Γ in the original, undeformed space. We denote by $\mathcal{F}$ the set of surface triangles in Γ, that is the tetrahedron faces that lie on the boundary of the solid. We denote by $\underline{\mathcal{F}}$ the set of triangles to be flattened and by $\overline{\mathcal{F}} = \mathcal{F} \setminus \underline{\mathcal{F}}$ its complement: the triangles that we seek to incline. Since these sets change during optimization and relaxation, we index them as $\underline{\mathcal{F}}^0, ..., \underline{\mathcal{F}}^i$.

We denote by $t =< p_0, p_1, p_2 >$ a triangle in $\mathcal{F}$ and $\mathbf{n}_t$ its normal. We denote by $\mathbf{z}$ the vertical (build) direction. Faces pointing upwards verify $\mathbf{n}_t \cdot \mathbf{z} > 0$. We similarly denote tetrahedrons as $< p_0, p_1, p_2, p_3 >$ in Γ, and denote by Γ_I the set of inner tetrahedrons and by Γ_O the set of outer tetrahedrons.

Finally, we denote as $\mathcal{M}_h$ the mapping obtained within Γ from a vector of vertical positions h.

4.1 Main algorithm

Our main algorithm is iterative: each iteration starts with a previous solution (initialized with $h^0 = z$) and a set of surfaces to flatten. It then solves for new vertical positions h^{i+1} by minimizing an objective function on Γ under fabrication constraints. The objective attempts to flatten surfaces in $\underline{\mathcal{F}}^i$ and to incline surfaces in $\overline{\mathcal{F}}^i$ towards the vertical. The flattening requirements are often unfeasible if strictly enforced. Thus, we express flattening as a soft constraint (objective with a high weight).

Once h^{i+1} is obtained, the surfaces in $\underline{\mathcal{F}}^i$ — connected components of neighboring triangles — are checked for flatness. If all are sufficiently flat, the loop terminates and we proceed with slicing. If not, we relax the flattening objective by keeping only a subset of triangles from $\underline{\mathcal{F}}^i$ in $\underline{\mathcal{F}}^{i+1}$.

Note that the constrained minimization has at least one solution: the identity deformation $h = z$. Therefore, at worst this process always terminates as the flatness test passes for $\underline{\mathcal{F}} = \emptyset$; or rather, when $\underline{\mathcal{F}}$ contains only the set of surfaces already flat in the input as these are never relaxed. In practice, the algorithm succeeded in flattening more surfaces on all our test cases.

All main steps are described in the next Sections: Fabrication constraints in Section 4.2, optimization of h^i in Section 4.3, relaxation in Section 4.4, slicing and toolpath mapping in Section 4.5.

```
Initialize F^0, h^0 ← z
Setup fabrication constraints on Γ
Loop over i, from i ← 1
    h^i ← arg min_h E(h, h^{i-1}, F^{i-1}, F̄^{i-1}) on Γ
    If surfaces in F^{i-1} using h^i are flat, or F^{i-1} == ∅
        set h ← h^i
        break
    EndIf
    F^i ← Relax(F^{i-1}, h^i)
EndLoop
T ← slice M_h(Ω)
return M_h^{-1}(T)
```

4.2 Fabrication constraints

Thickness constraints. The thickness constraints are captured by limiting the local stretch that the mapping is allowed to introduce. The minimum and maximum allowed stretches are respectively 1 and $\frac{\tau_{max}}{\tau_{min}}$ with τ_{min} and τ_{max} respectively the min/max admissible thicknesses on the target printer. We slice the model at τ_{max} before deformation by $\mathcal{M}^{-1}$.

Using this setup, a stretch of 1 in $\mathcal{M}$ will lead to having curved slices with a local thickness of τ_{max}, while a stretch of $\frac{\tau_{max}}{\tau_{min}}$ results in a local thickness along the curved slices of τ_{min}. Any stretch above or below that range would violate the fabrication constraints.

The deformation field is defined by linear interpolation within the tetrahedrons. Within each tetrahedron, the gradient ∇h of the vertical coordinates h is thus constant. We formulate the thickness constraint directly on the gradient ∇h.

Let us consider a tetrahedron with vertices $< p_0, p_1, p_2, p_3 >$. The gradient of a function f defined on the vertices and linearly interpolated within is obtained as $\nabla f = \sum_{i=0}^{2} w_k (f_k - f_3)$, where $f_k = f(p_k)$ and w_k are vector weights in $\mathbb{R}^3$. These weights (nine unknowns) can be computed ahead of time for each tetrahedron, setting $f = x$, $f = y$ and $f = z$ to obtain nine equations. For more details, we refer the reader to literature on linear tetrahedra.

For each tetrahedron $t \in \Gamma_I$ we write the constraints as:

$$1 \quad \leq \quad z(\nabla h_t) \quad \leq \quad \frac{\tau_{max}}{\tau_{min}}$$

where $z(\nabla h_t) = \sum_{k=0}^{2} z(w_k^t) \cdot (h_k - h_3)$ with $z(w_k^t)$ the Z coordinate of the weight k of t and $h_k = h(p_k)$.

Slope constraints. The maximum admissible slope θ_{max} depends on the printer setup and printed part, as detailed in Section 3.1. The constraint is applied to ∇h within each tetrahedron, preventing the gradients along X and Y from varying too fast with respect to the gradient along Z. We write the constraints for each tetrahedron $t \in \Gamma_I$ as:

$$-z(\nabla h_t) \leq \frac{x(\nabla h_t)}{\tan(\theta_{max})} \leq z(\nabla h_t)$$
$$-z(\nabla h_t) \leq \frac{y(\nabla h_t)}{\tan(\theta_{max})} \leq z(\nabla h_t)$$

where $z(\nabla h_t) = \sum_{k=0}^{2} z(w_k^t) \cdot (h_k - h_3)$ with $z(w_k^t)$ the Z coordinate of weight k for the gradient within t and $h_k = h(p_k)$. The terms $x(\nabla h_t)$ and $y(\nabla h_t)$ are similarly defined.

4.3 Objective function

The objective is optimized under strict fabrication constraints (see Section 4.2). We also add a set of constraints to prevent foldovers in Γ_O. This is done by imposing for all $t \in \Gamma_O$ that $z(\nabla h_t) > 0$. Note that foldovers cannot occur in Γ_I thanks to thickness constraints.

Given the set of surfaces $\underline{\mathcal{F}}$ and $\overline{\mathcal{F}}$ and a previous solution h^{i-1}, the objective to minimize is made of four different terms :

$$E(h, h^{i-1}, \underline{\mathcal{F}}, \overline{\mathcal{F}}) = \lambda_f E_{flat}(h, \underline{\mathcal{F}}) + \lambda_a E_{align}(h, h^{i-1}, \underline{\mathcal{F}}) + \lambda_s E_{slope}(h, \overline{\mathcal{F}}) + \lambda_m E_{smooth}(h)$$

E_{flat} attempts to flatten selected triangles while E_{align} encourages flat areas to align with slice tops. E_{slope} rotates other triangles towards the vertical. E_{smooth} regularizes the problem by guiding it towards smooth solutions, in particular in the less constrained empty regions. The λ_i weights control the tradeoff between the terms. For our target object scales (300 mm maximum in extent) and layer thicknesses (0.05 to 0.6 mm) we determined that a good tradeoff is given by $\lambda_f = 30$, $\lambda_a = 1$, $\lambda_s = 0.1$ and $\lambda_m = 0.02$. We use this setup for all results.

If significantly different scales are targeted, the λ_i weights should be rescaled noting that E_{flat} and E_{align} have the scale of τ_{max}^2, E_{slope} has the scale of L^2 with L the maximum object size (printer bed extent), and E_{smooth} is dimensionless.

Flattening. The objective E_{flat} is written as:

$$E_{flat}(h, \underline{\mathcal{F}}) = \sum_{t \in \underline{\mathcal{F}}} \frac{A(\{t\})}{A(\underline{\mathcal{F}})} \Big(\sum_{\substack{i,j \in [0,2] \\ i<j}} \big(h(t_i) - h(t_j)\big)^2 \Big)$$

with t a triangle, t_i its i-th vertex and where $A(.)$ computes the area of a set of triangles in the initial model. This attempts to put all vertices of each triangle in $\underline{\mathcal{F}}$ to be at the same height.

Aligning. The alignment objective encourages flattened areas to be aligned with slice tops. We slice the object after deformation using uniform slicing, therefore the slices are located every $k \cdot \tau_{max}$ with k an integer. This allows to compute an alignment error with respect to the result of the previous iteration h^{i-1}. To perform the alignment we consider connected components c of triangles to be flattened — sets of neighboring triangles in $\underline{\mathcal{F}}$. We attempt to snap the average height of each flat component to a slice top. We compute the height of a component c as a weighted average of its triangle positions, $H(c, h) = \sum_{t \in c} \frac{A(\{t\})}{A(c)} \frac{\sum_{i=0}^{2} h(t_i)}{3}$. We define the snapping position from the previous iteration as:

$$Snap(c, h^{i-1}) = \tau_{max} \left\lfloor 0.5 + \frac{H(c, h^{i-1})}{\tau_{max}} \right\rfloor$$

The alignment objective is defined as:

$$E_{align}(h, h^{i-1}, \underline{\mathcal{F}}) = \sum_{c \in C(\underline{\mathcal{F}})} Flat(c) \frac{A(c)}{A(\underline{\mathcal{F}})} \Big(H(c, h) - Snap(c, h^{i-1}) \Big)^2$$

where $C(\underline{\mathcal{F}})$ are connected components of $\underline{\mathcal{F}}$. $Flat(c)$ selects which alignment objectives are active, returning 1 or 0. An objective is active if 1) the connected component c is *flat* – as defined next – and 2) all *larger* components are also flat. The second rule prevents deciding on an alignment on small components before larger components are properly aligned.

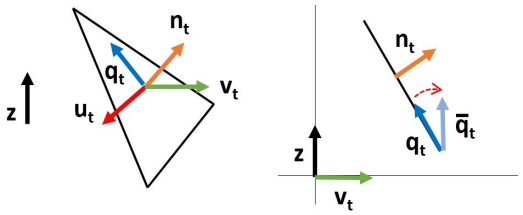


Fig. 5. Left: 3D view of the triangle with normal $\mathbf{n}_t$, tangent frame $\mathbf{u}_t$, $\mathbf{q}_t$. Right: Side view of the $\mathbf{z}$, $\mathbf{v}_t$ plane, showing the rotation of $\mathbf{q}_t$ around $\mathbf{u}_t$.

Component flatness. We evaluate the non-flatness of a component in $\underline{\mathcal{F}}$ by computing the average max height difference between triangle vertices, weighted by triangle area:

$$err(c, h) = \sum_{t \in c} \frac{A(\{t\})}{A(c)} \max_{\substack{i,j \in [0,2] \\ i<j}} \big(h(t_i) - h(t_j)\big)^2$$

This takes into account the fact that smaller triangles contribute less to the final error. A component is said flat if $err(c, h) < (\frac{\tau_{min}}{8})^2$. However, we reject as non flat any component where a triangle would have vertices separated by more than $\frac{\tau_{max}}{2}$: these could cross a slice boundary, producing a staircase.

Changing slope. The objective E_{slope} seeks to make surfaces in $\overline{\mathcal{F}}$ vertical. Given a triangle t with normal $\mathbf{n}_t$, we compute a tangential frame with vectors $\mathbf{u}_t = \frac{\mathbf{n}_t \wedge \mathbf{z}}{||\mathbf{n}_t \wedge \mathbf{z}||}$ and $\mathbf{q}_t = \mathbf{u}_t \wedge \mathbf{n}_t$. When changing the vertices heights $\mathbf{q}_t$ rotates around $\mathbf{u}_t$ (see Figure 5). The objective attempts to align $\mathbf{q}_t$ with $\mathbf{z}$, and thus we define a target vertical direction $\overline{\mathbf{q}}_t = sign(\mathbf{q}_t \cdot \mathbf{z}) \cdot \mathbf{z}$ where $sign(.)$ returns -1 if its argument is negative, 1 otherwise.

The objective seeks to obtain the correct distances between the vertices along $\overline{\mathbf{q}}_t$, that is:

$$E_{slope}(h, \eta) = \sum_{t \in \overline{\mathcal{F}}} \frac{A(\{t\})}{A(\overline{\mathcal{F}})} \Big(\sum_{\substack{i,j \in [0,2] \\ i<j}} \big(a_i - a_j + ((a_j - a_i) \cdot \mathbf{q}_t) \cdot \overline{\mathbf{q}}_t\big)^2 \Big)$$

where t is the triangle $< p_0, p_1, p_2 >$ and a_i is a vector using the original vertex XY coordinates and the variable height as the Z coordinate: $(x(p_i), y(p_i), h(p_i))$.

We exclude triangles that are already vertical, since no change of vertex heights can modify their angle. It is logical to exclude them, since there will be no staircase defect on a vertical slope, regardless of the slice thickness.

Smoothness. The smoothness term encourages neighboring tetrahedrons to be modified in a similar manner, that is, to have the same gradient. This is written as:

$$E_{smooth}(h) = \sum_{t \in \Gamma} \sum_{n \in \mathcal{N}(t)} \frac{V(\{t\}) + V(\{n\})}{V(\Gamma)} \Big(\nabla h_t - \nabla h_n \Big)^2$$

where $\mathcal{N}(t)$ is the set of tetrahedrons sharing a face with t, $V(.)$ returns the volume of a set of tetrahedrons, and ∇h_t is the (vector) gradient of t.

Solver. The problem we formulate is a quadratic objective minimization under linear constraints. We use the Gurobi solver [2018] to obtain a solution at each iteration.

4.4 Relaxing flatness constraints

After obtaining a solution h^i for the current iteration, we check whether connected components in $\underline{\mathcal{F}}^{i-1}$ are all properly flattened. If not, we proceed to remove some of the flattening requirements. This process is reminiscent of segmentation techniques in PolyCube optimization [Gregson et al. 2011].

Removing many triangles from $\underline{\mathcal{F}}$ at each iteration would quickly lead to satisfying solutions, but may miss opportunities to flatten some areas. Removing them one by one—canceling the triangle with maximum vertex height differences—would require as many iterations as there are triangles in $\underline{\mathcal{F}}^0$, which is unreasonable.

We propose the following strategy. We observe that in most cases the error is concentrated along the boundaries of the connected components of $\underline{\mathcal{F}}^{i-1}$. We therefore relax all the boundary triangles that exceed the flatness threshold. In this process, we consider as boundary triangles only those whose neighbors were relaxed before. The rationale is that it is best to relax the flattened region progressively along the same front.

If there is no boundary triangle in a component—which is always the case on the first iteration—we relax the triangle with the largest vertex height differences. Intuitively, this opens a tear in the non-flat component that will then grow to relax the problem.

Note that, at each iteration, we relax all components which account for less than 5% of the total area to be flattened.

4.5 Applying the mapping and its inverse

The optimization gives us the mapping $\mathcal{M}_h$ from the initial space to the slicing space. We first apply the mapping to the input model, in order to obtain the mesh that will be sliced with a standard slicer. The mapping has been optimized to use τ_{max} as uniform slicing thickness.

After slicing we obtain a set of toolpaths in the form of G-code for filament printers. We apply the inverse mapping $\mathcal{M}_h^{-1}$ to the toolpath coordinates to obtain the final curved toolpaths.

As the deformation may vary along each toolpath, we first resample the toolpaths using a sampling rate matching the nozzle diameter. We then transform each point back to the inital space.

As the toolpath points lie on the top of each slice, we first offset them down by half a layer thickness, $\frac{\tau_{max}}{2}$. We then locate each point within $\mathcal{M}(\Gamma)$. Having identified the enclosing tetrahedra, we interpolate the z coordinate of the point from the original tetrahedron vertices. This gives us a new point in the inital object space, located in the middle of a slice.

We offset it back to the slice top. This requires knowing the local thickness. This information is directly available from the gradient of h within the enclosing tetrahedra. We similarly adjust the material flow, and adjust the deposition speed to maintain a constant extrusion rate.

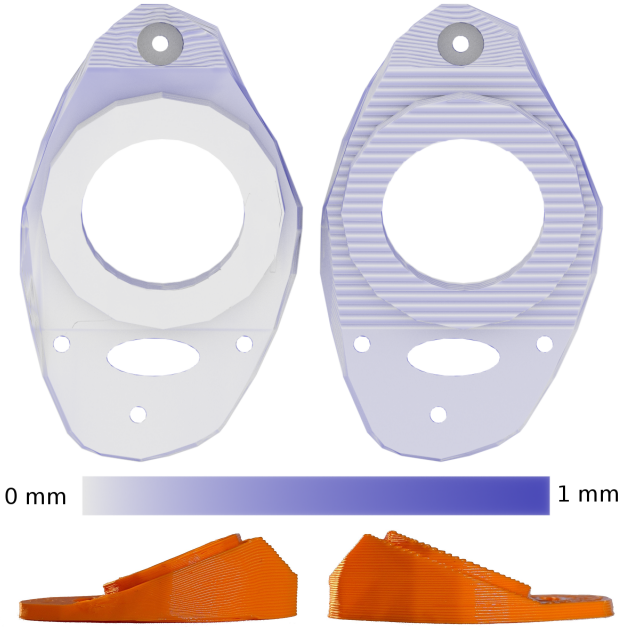


Fig. 6. Anklebase model with different slicing algorithms, all using 38 slices. **Top:** views of distance maps between the input mesh with curved slicing (left) and uniform slicing (right). A lighter color indicates a reduced error. **Bottom:** Sides views of curved slicing print (left) and adaptive slicing (right). All staircasing is eliminated while closely following the input.

4.6 Controlling the number of layers

By attempting to make surfaces vertical, our optimizer tends to make the object as tall as possible in the slicing space – equivalently using the thinnest slices. To control the number of layers we simply constrain the vertices of $\mathcal{F}$ to never exceed a certain height. That is, to obtain n layers we add the constraint:

$$\text{forall } p \in \mathcal{F}, h(p) < h(p_{bottom}) + n \cdot \tau_{max}$$

where p_{bottom} is the lowest vertex of $\mathcal{F}$.

This affords for a time–quality trade-off that can be convenient for the user.

5 RESULTS

We show a variety of prints in Section 5.1, we explore the influence of the input parameters in Section 5.2, and we provide measurements of surface accuracy in Section 5.3, comparing our technique to optimal discrete adaptive slicing [Alexa et al. 2017].

5.1 Printed models

We produced a number of results using an off-the-shelf Ultimaker2 (UM2) printer with minor modifications. It is equipped with a 0.8mm nozzle and we allow thicknesses from 0.6mm down to 0.1mm (1/6 ratio). To allow for increased freedom of motion we removed the metal part holding the fans around the nozzle. We use a wider nozzle to print with thicker layers, so as to clearly reveal the effect of curved slicing. We also used a standard Anet A8 printer for Figure 8.

Table 1. Statistics for all models, with 0.1-0.6 mm layer thickness and $\theta_{max} = 30$. The first column is the number of tetrahedrons, the second column is the number of relaxation iteration and the third column reports total optimization times (Intel i7-4790K, 4 cores).

Model	# Tets	# Iter.	Optimization time
Wing	9 711	4	< 1 minute
Foil cutter	19 740	3	< 1 minute
Anklebase	25 607	3	< 1 minutes
Frog (small)	36 224	3	< 1 minute
Sports car	58 708	17	20 minutes
Kitten	133 838	3	14 minutes
Frog (big)	200 743	3	22 minutes

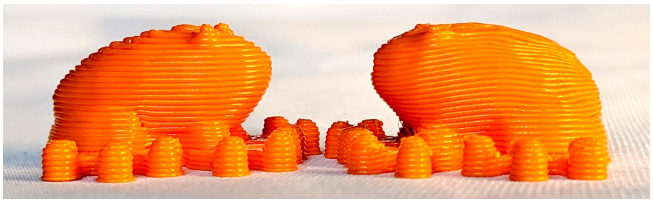


Fig. 7. Frog model, adaptive slicing (left) and curved slicing (right) with 27 slices. Printed on the UM2. Note the difference in the silhouette.

All models use the same 30 degrees angle constraint θ_{max} and the algorithm attempts to flatten all surfaces below this angle, unless otherwise specified. Table 1 summarizes the main statistics and optimization times for the models shown in the paper. Optimization typically takes a few minutes, with the time depending essentially on the input size and number of relaxation iterations.

Figure 6 shows a mechanical part with a slanted area. Curved slicing closely follows the surfaces, printing the entire part around the hole with a matching slope. Note how the slices go from flat to curved in the bottom part. Figure 6 also compares with the result of uniform and adaptive slicing for the same number of slices.

Figure 7 shows a frog model sliced with our technique and adaptive slicing, using a low number of slices. Figure 8 shows the same model printed with thinner layers on a standard Anet A8 (cooling fan removed for clearance). In both Figures, note the smooth curved top of the frog. A bigger frog model is also shown in Figure 9. Note how the slices match the overall angle of the main body, and how the slices become thicker at the vertical extremity (mouth).

Figure 1 shows a wing cross-section model, revealing how curved slicing can accurately reproduce an entire curved top. The model is more accurately reproduced with our approach.

Figure 10 compares two car model prints, one via curved slicing and the other via adaptive slicing. The curved slices nicely follow the car outline at the top of the part, as well as reproducing the wheels more accurately. This reveals how fewer slices can be used to better reproduce slanted surfaces, allowing the reallocation of other slices to the more detailed parts of the model.

5.2 Influence of optimization parameters

We now study the impact of the optimization parameters. Figure 11 illustrates the effect of the angle constraint θ_{max} on an example model, showing the deformation obtained on printers offering different degrees of freedom in terms of achievable angles. The results use a similar number of slices, but smaller areas are flattened: increased angular freedom leads to larger smooth surfaces.

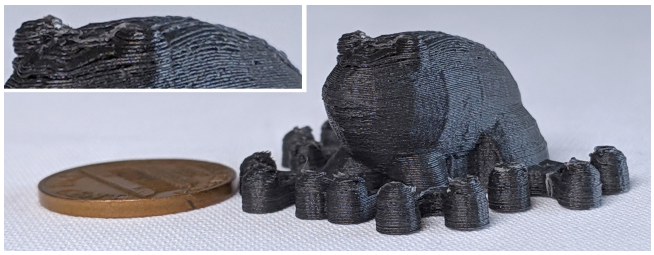


Fig. 8. Small frog model printed with curved slicing on an Anet A8 printer, with a 0.4mm nozzle and 103 layers between 0.3mm and 0.05mm. Layers at the top smoothly follow the curvature.

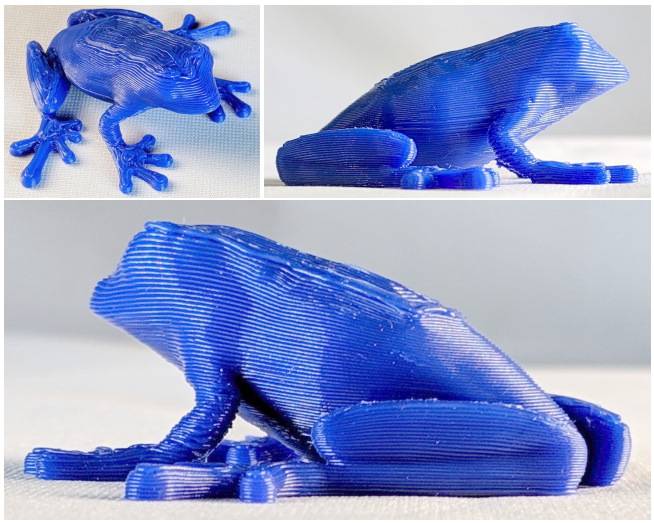


Fig. 9. Big frog printed on the UM2 with 68 curved layers. Note how the layers have been curved to follow the overall shape of the body, and how they become thicker near the mouth.

Figure 12 illustrates the effect of the thickness ratio $\frac{\tau_{max}}{\tau_{min}}$ on the same model. A ratio closer to one leaves little freedom to deform the model upwards, resulting in using fewer slices. However, the same set of surfaces are flattened.

In summary, increasing θ_{max} allows larger curved surfaces, while increasing $\frac{\tau_{max}}{\tau_{min}}$ leads to thinner slices.

Finally, Figure 13 illustrates the use of the number of layers target (Section 4.6). Please note, however, that models obtained this way are usually less accurate (in terms of volume error) than those obtained automatically.

5.3 Quality

We compare the surface accuracy of our results to that of uniform slicing and adaptive slicing in Figure 14. We use the volume assignment error, computed by discretization [Alexa et al. 2017]. We run

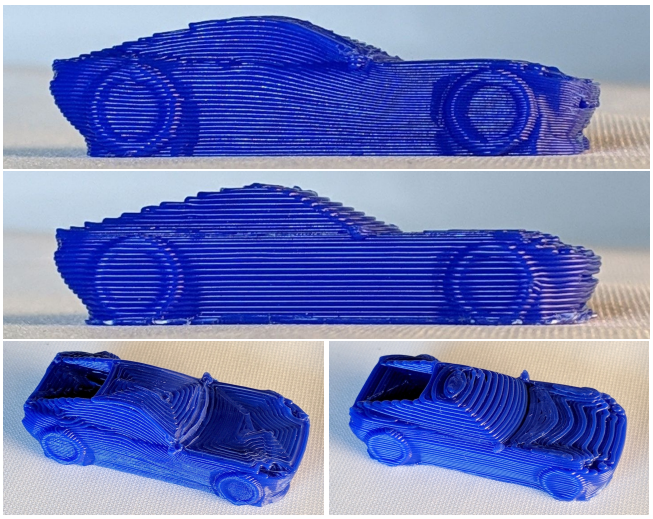


Fig. 10. Sport car model printed on the UM2 using 34 layers, using our approach (top and then left) and adaptive slicing (bottom and then right). The two side views reveal the smooth outline of the curved layers version. Note the improved surface finish of the roof and hood, as well as the way the curved layers uncompress to produce the lateral windows. Adaptive slicing uses most layers around the hood.

an optimization of our models to obtain the reference number of slices for comparison, and run each slicing algorithm on Ω using the same number of slices. As can be seen from the left column of each comparison in Figure 14, curved slicing generally results in superior accuracy.

To evaluate the importance of flattening, we force the optimization to start from artificially smaller sets of surfaces to flatten $\underline{\mathcal{F}}$ (all but first columns of each comparison in Figure 14). As expected, curved slicing is less accurate with less flattening: flattened surfaces — equivalently curved surfaces in the original space — are those that benefit the most from our approach. Nevertheless, thanks to verticalizing, even when little flattening is performed curved slicing produces results which are better than uniform slicing, even though less accurate than adaptive slicing on some models.

5.4 Discussion

Overall we can see that our technique outperforms both uniform and adaptive slicing. This is true both in terms of accuracy (volume error) and surface finish.

It is especially remarkable that our technique produces results that are more accurate than *optimal* adaptive slicing. This is explained by several factors. At low numbers of slices, adaptive slicing has little freedom, whereas we curve the deposition and align with surfaces. Adaptive slicing cannot perform local adaptations, while by curving our algorithm can reallocate slice thicknesses as required. This is true within the same region of the object but also for disconnected components within a slice. Finally, adaptive slicing allocates many slices to low slopes, while we capture low slopes with a single slice.

However, optimal discrete slicing [Alexa et al. 2017] is orders of magnitude faster as it computes slicing plans for all slices at once. Also, while we can target a specific number of slices (Section 4.6), we cannot guarantee that the target is reached.

In many cases the visual aspect of the prints does not reflect the actual volumetric error. This is for instance the case on the cars in Figure 15: The orientation of the top surface fill — the zigzag pattern — changes the perceived smoothness. The surface is nevertheless smoother than that of the planar slicing result. To reduce such issues the slicer could be made aware of a preferred direction for the covers.

It is worth noting that we leave vertical surfaces free in our approach. In the idealized vertical slice model, a vertical wall produces no error regardless of the layering thickness. In practice, on filament printers, slices have a rounded profile that changes the perceived appearance. We could add a penalty requiring vertical surfaces to be as tall as possible (and hence use as thin as possible slices). Nevertheless, we found that leaving them free adds flexibility to the solver and generally results in better solutions.

Finally, we did not consider structural properties in this evaluation. Based on observations from previous work prototyping curved 3D printing, we expect the prints to be stronger [Huang and Singamneni 2012; Lim et al. 2016; Singamneni et al. 2012] than those obtained with planar slicing. As future work we will quantitatively evaluate the change in structural properties.

6 LIMITATIONS AND FUTURE WORK

A first limitation of our approach is the relatively high running time. Since we perform the optimization in a volume, the number of variables grows quickly. As future work, it would be interesting to consider a multi-resolution approach, and use a coarse solution to bootstrap the deformation of a high-resolution version of the model. Note, however, that the number of slices and the mesh optimization are not correlated. Optimizing a scaled version of a model costs the same, and slicing is fast.

A practical limitation relates to the range of 3D printers where this technique is applicable. Some printers have very short extrusion nozzles, and in such cases there is very little angular freedom (see Figure 11 to see the impact of a reduced Θ_{max}). Another practical concern, on devices mounting relatively heavy plates on the Z axis—such as the UM2—is that the speed has to be limited to obtain a smooth Z motion and deposition. We typically printed twice as slow as for a normal UM2 print. The lighter carriage of the A8 did not impose lower speeds. A delta printer would be ideal [Allen and Trask 2015]. Another practical difficulty relates to abrupt flow variations, which can result in some defects where they occur.

Our iterative scheme favors flattening. This is a reasonable choice as revealed in Figure 14: the accuracy is generally better with increased flattening. However, it is clear that in some situations locally relaxing flattening could improve the overall solution. Our optimizer cannot currently detect such cases. A possible approach would be to relax flattened areas that do not align well with the slice tops. Allowing the user to select which faces to keep flattened would be a good alternative, enabling application-dependent choices.

Finally, a better integration with the slicer would help further improve perceived surface quality, in particular aligning the fill patterns with the slope direction (see Section 5.4). Another potential

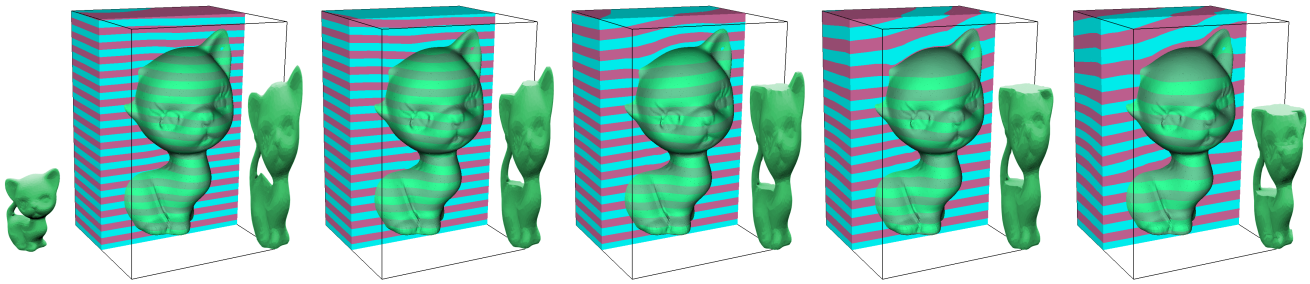


Fig. 11. Optimization of kitten model for θ_{max} = 10, 20, 30, 40, and 50 degrees. This shows both the object and the empty space around (cut open). The stripes correspond to slices. The 3D models shown inset are obtained after mapping $\mathcal{M}(\Omega)$. The original model is shown in the leftmost bottom corner. Note how additional angular freedom results in more areas being curved and smoothly reproduced.

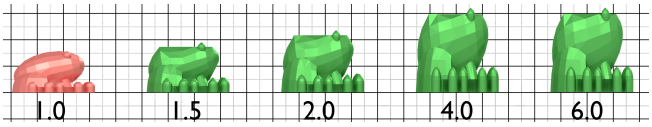


Fig. 12. Optimization of the frog model for different min/max thickness ratios. The same surfaces are flattened, but the additional freedom leads to using thinner layers (taller models imply more layers). The red frog (ratio 1.0) is deemed unfeasible by the optimizer ; this is due to numerical issues as we request all slices to have exactly the same thickness.

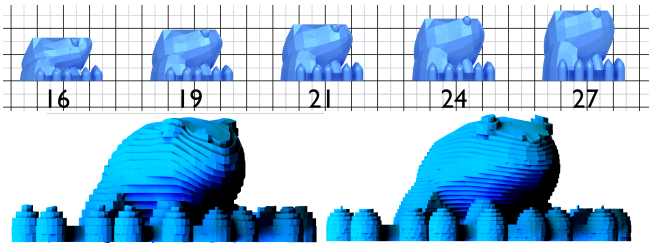


Fig. 13. Optimization of the frog model targeting 16, 19, 21, 24, and 27 layers of 1mm, using the approach from Section 4.6. The renderings show the models for 16 and 27 slices. Lighting enhanced to highlight slices.

approach is the *ironing* technique, where the print head performs multiple passes to heat and smooth surfaces.

We hope our technique will encourage further adoption of curved slicing on standard 3-axis machines. We expect this technique will lead to future work regarding curved slicing using additional degrees of freedom.

ACKNOWLEDGMENTS

This research was initiated at the 17th Bellairs Workshop on Computational Geometry, February 10-16, 2018, co-organized by S. Lazard and S. Whitesides. The work was partly supported by Lorraine Université d'Excellence, ANR-15-IDEX-04-LUE, by ANR MuFFin ANR-17-CE10-0002, the NSF CAREER award IIS-1652515, the NSF grant OAC:1835712, and the CUHK Direct Research Grant 4055094.

3D MODELS

- wing https://www.thingiverse.com/thing:95502
- anklebase (50% scale) http://inmoov.fr/inmoov-stl-3d/?bodyparts=Legs-Ankle&parts=AnkleBaseV1.stl
- frog (small) https://www.thingiverse.com/thing:3284
- frog (large) https://www.thingiverse.com/thing:182144
- kitten https://www.thingiverse.com/thing:12694
- sportscar https://www.thingiverse.com/thing:1587558

REFERENCES

Daniel Ahlers. 2018. *3D Printing of Nonplanar Layers for Smooth Surface Generation.* Master's thesis. University of Hamburg.

Marc Alexa, Kristian Hildebrand, and Sylvain Lefebvre. 2017. Optimal discrete slicing. *ACM Trans. Graph.* 36, 1 (2017), 1 – 16.

Robert J.A. Allen and Richard S. Trask. 2015. An experimental demonstration of effective Curved Layer Fused Filament Fabrication utilising a parallel deposition robot. *Additive Manufacturing* 8 (2015), 78–87.

Marco Attene. 2015. Shapes In a Box: Disassembling 3D Objects for Efficient Packing and Fabrication. *Comput. Graph. Forum* 34, 8 (2015), 64–76.

Debapriya Chakraborty, B. Aneesh Reddy, and A. Roy Choudhury. 2008. Extruder Path Generation for Curved Layer Fused Deposition Modeling. *Comput. Aided Des.* 40, 2 (2008), 235–243.

W. Cheng, J.Y.H. Fuh, A.Y.C. Nee, Y.S. Wong, H.T. Loh, and T. Miyazawa. 1995. Multi-objective optimization of part- building orientation in stereolithography. *Rapid Prototyp J.* 1, 4 (1995), 12–23.

Chengkai Dai, Charlie C. L. Wang, Chenming Wu, Sylvain Lefebvre, Guoxin Fang, and Yong-Jin Liu. 2018. Support-free Volume Printing by Multi-axis Motion. *ACM Trans. Graph.* 37, 4 (2018), 134:1–134:14.

André Dolenc and Ismo Mäkelä. 1994. Slicing procedures for layered manufacturing techniques. *Comput. Aided Des.* 26, 2 (1994), 119–126.

Ben Ezair, Saul Fuhrmann, and Gershon Elber. 2018. Volumetric covering print-paths for additive manufacturing of 3D models. *Comput. Aided Des.* 100 (2018), 1 – 13.

J. Gregson, A. Sheffer, and E. Zhang. 2011. All-Hex Mesh Generation via Volumetric PolyCube Deformation. *Comput. Graph. Forum* 30, 5 (2011), 1407–1416.

Gurobi. 2018. Gurobi Optimizer Reference Manual, Gurobi Optimization, LLC.

Mohammad T Hayasi and Bahram Asiabanpour. 2013. A new adaptive slicing approach for the fully dense freeform fabrication (FDFF) process. *Journal of Intelligent Manufacturing* 24, 4 (2013), 683–694.

Kristian Hildebrand, Bernd Bickel, and Marc Alexa. 2013. Orthogonal slicing for additive manufacturing. *Computers & Graphics* 37, 6 (2013), 669 – 675. Shape Modeling International (SMI) Conference 2013.

R.L. Hope, R.N. Roth, and P.A. Jacobs. 1997. Adaptive slicing with sloping layer surfaces. *Rapid Prototyp J.* 3, 3 (1997), 89–98.

Ruizhen Hu, Honghua Li, Hao Zhang, and Daniel Cohen-Or. 2014. Approximate Pyramidal Shape Decomposition. *ACM Trans. Graph.* 33, 6 (2014), 213:1–213:12.

Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. 2018. Tetrahedral Meshing in the Wild. *ACM Trans. Graph.* 37, 4 (2018), 60:1–60:14.

B Huang and S Singamneni. 2012. Alternate Slicing and Deposition Strategies for Fused Deposition Modelling of Light Curved Parts. *J. of Achievements in Materials and Manufacturing Engineering* 55, 2 (2012), 511–517.

B. Huang and S. Singamneni. 2015. A Mixed-Layer Approach Combining Both Flat and Curved Layer Slicing for Fused Deposition Modelling. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 229, 12 (2015),

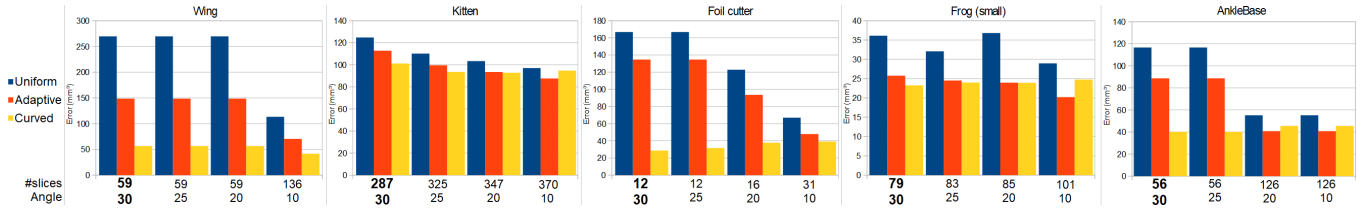


Fig. 14. Comparison of uniform slicing (blue), optimal adaptive slicing (orange) and curved slicing (yellow) for five different models. We measure the volume error in mm^3. Lower is better. We run our approach first and then slice with uniform and adaptive slicing using the same number of slices. The result obtained by our method in the normal setup is the leftmost in each graph (angle = 30 degrees). The other columns show the result obtained when starting from smaller sets $\underline{\mathcal{F}}$, selecting only faces whose slopes are below the indicated angle (in degrees). Please refer to the text for discussion.

Fig. 15. Different toolpath orientations for the top layers in curved slicing result in different surface finish. The visible stripes are shallower than the layer thickness, yet they create a visual effect that is detrimental to perceived quality. The effect also depends on lighting and view angle.

2238–2249.

Yijiang Huang, Juyong Zhang, Xin Hu, Guoxian Song, Zhongyuan Liu, Lei Yu, and Ligang Liu. 2016. FrameFab: Robotic Fabrication of Frame Shapes. *ACM Trans. Graph.* 35, 6 (2016), 224:1–224:11.

Steven Keating and Neri Oxman. 2013. Compound fabrication: A multi-functional robotic platform for digital design and fabrication. *Robotics and Computer-Integrated Manufacturing* 29, 6 (2013), 439–448.

Prashant Kulkarni and Debasish Dutta. 1996. An accurate slicing procedure for layered manufacturing. *Comput. Aided Des.* 28, 9 (1996), 683 – 697.

Sungwoo Lim, Richard A Buswell, Philip J Valentine, Daniel Piker, Simon A Austin, and Xavier De Kestelier. 2016. Modelling Curved-Layered Printing Paths for Fabricating Large-Scale Construction Components. *Additive Manufacturing* (2016).

Marco Livesu, Stefano Ellero, Jonàs Martínez, Sylvain Lefebvre, and Marco Attene. 2017. From 3D models to 3D prints: an overview of the processing pipeline. *Computer Graphics Forum* 36, 2 (2017).

Linjie Luo, Ilya Baran, Szymon Rusinkiewicz, and Wojciech Matusik. 2012. Chopper: Partitioning Models into 3D-printable Parts. *ACM Trans. Graph.* 31, 6 (2012).

K Mani, P Kulkarni, and D Dutta. 1999. Region-based adaptive slicing. *Comput. Aided Des.* 31, 5 (1999), 317 – 333.

H. S. Masood, W. Rattanawong, and P. Iovenitti. 2000. Part Build Orientations Based on Volumetric Error in Fused Deposition Modelling. *The International Journal of Advanced Manufacturing Technology* 16, 3 (2000), 162–168.

Stefanie Mueller, Sangha Im, Serafima Gurevich, Alexander Teibrich, Lisa Pfisterer, François Guimbretière, and Patrick Baudisch. 2014. WirePrint: 3D Printed Previews for Fast Prototyping. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. 273–280.

Y. Pan, C. Zhou, Y. Chen, and J. Partanen. 2014. Multitool and multi-axis computer numerically controlled accumulation for fabricating conformal features on curved surfaces. *Journal of Manufacturing Science and Engineering* 136, 3 (2014).

PM Pandey, N Venkata Reddy, and Sanjay G Dhande. 2003. Slicing procedures in layered manufacturing: a review. *Rapid Prototyp J.* 9, 5 (2003), 274–288.

Huaishu Peng, Rundong Wu, Steve Marschner, and François Guimbretière. 2016. On-The-Fly Print: Incremental Printing While Modelling. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*.

Emmanuel Sabourin, Scott A Houser, and Jan Helge Bøhn. 1996. Adaptive slicing using stepwise uniform refinement. *Rapid Prototyp J.* 2, 4 (1996), 20–26.

Emmanuel Sabourin, Scott A. Houser, and Jan Helge Bøhn. 1997. Accurate exterior, fast interior layered manufacturing. *Rapid Prototyp J.* 3, 2 (1997), 44–52.

Sarat Singamneni, Asimava Roychoudhury, Olaf Diegel, and Bin Huang. 2012. Modeling and evaluation of curved layer fused deposition. *Journal of Materials Processing Technology* 212, 1 (2012), 27–35.

Hai-Chuan Song, Nicolas Ray, Dmitry Sokolov, and Sylvain Lefebvre. 2017. Anti-aliasing for Fused Filament Deposition. *Comput. Aided Des.* 89, C (2017), 25–34.

Peng Song, Bailin Deng, Ziqi Wang, Zhichao Dong, Wei Li, Chi-Wing Fu, and Ligang Liu. 2016. CofiFab: Coarse-to-fine Fabrication of Large 3D Objects. *ACM Trans. Graph.* 35, 4 (2016), 45:1–45:11.

Kamesh Tata, Georges Fadel, Amit Bagchi, and Nadim Aziz. 1998. Efficient slicing for layered manufacturing. *Rapid Prototyp J.* 4, 4 (1998), 151–167.

Llewellyn-Jones Thomas, Allen Robert, and Trask Richard. 2016. Curved layer fused filament fabrication using automated tool-path generation. *3D Printing and Additive Manufacturing* 3 (2016), 236–243.

K Thrimurthulu, Pulak M Pandey, and N Venkata Reddy. 2004. Optimum part deposition orientation in fused deposition modeling. *International Journal of Machine Tools and Manufacture* 44, 6 (2004), 585 – 594.

Justin Tyberg and Jan Helge Bøhn. 1998. Local Adaptive Slicing. *Rapid Prototyp J.* 4, 3 (1998), 118–127.

Justin Tyberg and Jan Helge Bøhn. 1999. FDM systems and local adaptive slicing. *Materials & design* 20, 2 (1999), 77–82.

Nobuyuki Umetani and Ryan Schmidt. 2013. Cross-sectional Structural Analysis for 3D Printing Optimization. In *SIGGRAPH Asia 2013 Technical Briefs (SA '13)*. 5:1–5:4.

Weiming Wang, Haiyuan Chao, Jing Tong, Zhouwang Yang, Xin Tong, Hang Li, Xiuping Liu, and Ligang Liu. 2015. Saliency-Preserving Slicing Optimization for Effective 3D Printing. *Comput. Graph. Forum* 34, 6 (2015), 148–160.

W. M. Wang, C. Zanni, and L. Kobbelt. 2016. Improved Surface Quality in 3D Printing by Optimizing the Printing Direction. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics (EG '16)*. 59–70.

Chenming Wu, Chengkai Dai, Guoxin Fang, Yong-Jin Liu, and Charlie C. L. Wang. 2017. RoboFDM: a robotic system for support-free fabrication using FDM. In *Proceedings of IEEE International Conference on Robotics and Automation*.

Rundong Wu, Huaishu Peng, François Guimbretière, and Steve Marschner. 2016. Printing Arbitrary Meshes with a 5DOF Wireframe Printer. *ACM Trans. Graph.* 35, 4 (2016), 101:1–101:9.

Xiaoting Zhang, Xinyi Le, Athina Panotopoulou, Emily Whiting, and Charlie C. L. Wang. 2015. Perceptual Models of Preference in 3D Printing Direction. *ACM Trans. Graph.* 34, 6 (2015), 215:1–215:12.