Volumetric Covering Prints-Paths for Additive Manufacturing of 3D Models

Ben Ezair, Saul Fuhrmann, Gershon Elber

Computer Science Dept., Technion,Israel Institute of Technology, Haifa, Israel

Abstract

In additive manufacturing (AM), slicing is typically used to manufacture 3D models, one layer after another. Yet, in recent years quite a few hardware platforms were introduced toward the use of multi-axes AM with general 3D curves as print-paths. This paper presents algorithms for the generation of such general print-paths that can potentially be used to synthesize superior 3D models using AM. In slicing, a 3D model is decomposed into a series of parallel planar sections, which in turn are (usually) decomposed into a set of piecewise linear curves used as print-paths in the AM process. The methods we propose in this work ease this restriction, namely the print-paths are no longer limited to parallel planes. Like slicing, the methods we propose achieve a complete covering of a general volume with print-paths expressed as general curves. However, and unlike slicing, the created print-paths can conform better to the 3D model, its properties, and even user input. We expect that the added flexibility and freedom in the specification of AM print-paths, as opposed to limiting them to planar curves, will enable the synthesis of 3D models (using AM) with superior properties (such as mechanical strength and surface finish). As a proof of concept, we also present examples of 3D models manufactured with a low-end AM hardware and using the algorithms described in this paper.

Keywords: Additive manufacturing, 3D-printing optimization, Volumetric covering
2010 MSC: 00-01, 99-00

1. Introduction

Contemporary additive manufacturing (AM) systems largely use slicing [1]. Slicing deconstructs a three dimensional object (often specified by a polygonal mesh) to a series of two dimensional parallel (to the printing surface) planar sections. These planar sections, in turn, are decomposed into piecewise linear paths for the manufacturing process to use. In general, the slicing planes are not intrinsic to the input object. The surface finish, the strength, and possibly other properties of an object printed using some existing AM techniques, are influenced by the slicing orientation and the print-paths used to create it [2, 3]. One of the conclusions in [2] is that parts fabricated with the expected tensile loads aligned with the fibers (print-paths) would have greater effective tensile strength, and could handle greater loads. Additionally, the experiments in [4] showed that parts manufactured using curved layers that fit the part geometry (as opposed to the flat layers used in slicing) performed better under mechanical stresses. An illustration of the possible advantages of print-paths that conform to the model geometry over slicing can be seen in Figure 1. Both Figure 1 (a) and (b) show the result of printing the same model (a section from the model in Figure 2), on the same printer, using the same resolution (layer-height is 0.3mm). However, in Figure 1 (a) the model was printed using print-paths that conform to the model geometry, while in Figure 1 (b) slicing was used. Figure 1 (c) and (d) show the simulated preview of the printing result for (a) and (b) respectively. As the images show, the slicing result suffers from surface finish issues due to aliasing, the so called the staircase effect, that comes from approximating a curved shape with planar sections. In this effort, we seek to create print-paths that are more dependent on design goals, and less dependent on constraints imposed by the printing process, allowing the creation of superior printed objects in terms of surface finish, strength, etc.

There are quite a few reports on the use of multi-axis robotic hardware platforms in AM, which would allow non-planar 3D printing without using slicing [5, 6, 7, 8]. Such a hardware platform would be able to print along the main feature lines of a 3D object and gain the advantages mentioned before when compared to slicing. Yet, algorithmic support is lacking. We are aware of no algorithm that is capable of covering the volume of an arbitrary 3D object using any general user defined univariate (curve) tool paths, while also establishing a valid printing order, fully exploiting these platforms toward multi-axis AM.

Our main contributions in this paper are:

(1) Presenting an algorithm that can generate covering curves for general 3D objects represented by (possibly trimmed) trivariate volumes, that conform to the geometry of the trivariate, toward AM.

(2) Supporting the use of an additional external direction field that can be used to specify AM printing-paths for any general B-rep (boundary representation) 3D model (including polygonal meshes).

(3) Algorithms that enable the use of general (possibly user defined) 3D printing-paths, while controlling their width, resolving their accessibility, and establishing valid AM printing order and coverage.

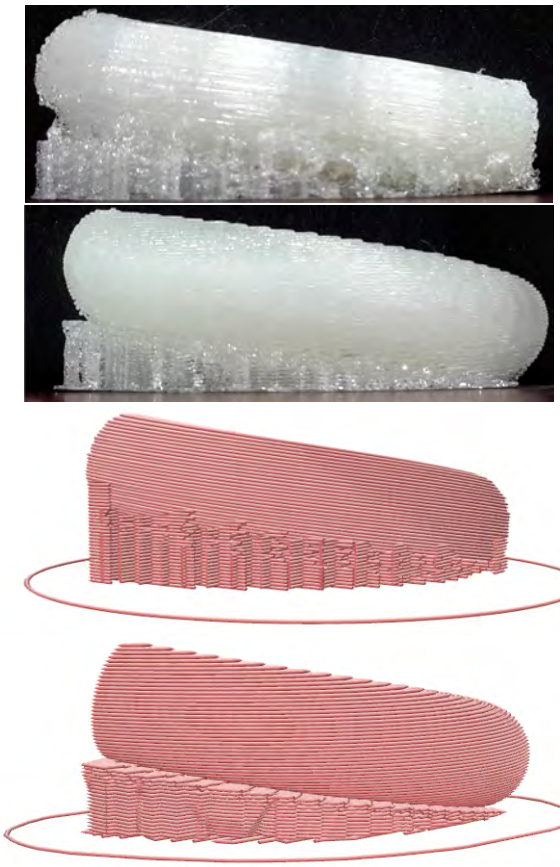


Figure 1: A comparison between slicing and print-paths that conform to the model geometry. In (a), the model is printed using curves that conform to geometry of the model. In (b), traditional slicing is used. In (c) and (d), simulated previews of the printing results (based on the print-paths) are shown for (a) and (b), respectively. The model in both cases is a short section (a quarter of a cycle) from the helical model in Figure 2, and is shown with the generated support structure.

Note that throughout this paper we assume a generic process, similar to the Fused Deposition Modeling (FDM) or Directed Energy Deposition (DED) 3D printing process [9]. This means, we assume some sort of a printing head that extrudes material with a circular cross section. While we do not address low level printing process properties (such as, overlaps between adjacent extrusions and the movement of extruded material), these assumptions were useful enough to allow us to manufacture physical objects using an FDM printer (as we show later).

The rest of this paper is organized as follows, Section 2 discusses previous work that dealt with alternatives to slicing in AM. In Section 3, we examine the main considerations to employ when transforming a description of a 3D object into a description of print-paths (curves) needed to manufacture the object using AM. Section 4 presents our covering algorithms and shows how (geometric) design, rather than the printing process, can be the main consideration in 3D print-path planning. Section 5 outlines how the covering curves (generated based on geometric design) can be used to create a 3D model using AM. Some experimental results are shown in Section 6, while in Section 7, we discuss our results and suggest future work. Finally,

Figure 2: A helical volume created by sweeping a circle along a helix (rotating around the z axis).

in Section 8, we conclude.

2. Previous Work

There have already been several publications that examined the possibility of printing in ways that differ from the traditional AM slicing approach. Curved layers are suggested by Chakraborty et al. [10] as offsets of a parametric surface, which, of course, limits objects to those that can be expressed as a set of offsets from such a surface. In [11], Huang and Singamneni show how non-planar slices can be created and used to print special geometries. Objects in [11] are limited to geometries that can be expressed as offsets of polygonal faces [12], starting from the top (facing up) polygonal surface of the object. Future work mentioned in [11] will seek to handle more complex geometries by subdividing an object into parts that can be printed using non-planar slices, and complex parts that will be printed using traditional slicing. In [13], the upper and lower boundaries of objects are printed as single layers using simple z-height maps to plan the paths, while the interior of the object is printed conventionally. Their implementation uses non-planar lines parallel to the x (or y) axis to fill the upper and lower layers. This method limits the objects to those with distinct upper and lower boundaries that are relatively flat. In the work by Davis et al. [14], intermediate mappings between an initial object (designed by the user) and a final object (the one printed) are the basis for non-planar layers. However, accessibility, coverage, and other manufacturing considerations are ignored and the main focus is on producing intermediate layers (surfaces in 3D). Mueller et al. [15] show how a wire-frame on the boundary of an object can be printed directly, to quickly create a visual representation of the object. They generate the wire-frame so that there would be no need to print downwards in a steeper angle than the slant of the print head itself, and avoid accessibility issues in that way. In [16] and [6], manually planned print-paths are used to create objects without slicing. The focus in [16] and [6] is on implementing and testing the hardware

needed to accomplish the prints. In [8], Gao et al. introduce an additional degree of freedom to 3D printing, and allow printing around a cuboid (box) object and creating slices in six different directions corresponding to the facets of the cuboid. The resulting object in [8] is, in essence, made of six traditionally sliced (and manufactured) objects fused together.

Most of the above previous work either dictate printing directions that are independent of the model geometry ([8, 15]), or limit themselves to specific simple geometries ([11, 16, 6, 13, 10]). For example, to the best of our knowledge, the model in Figure 2 cannot be manufactured using any of the above methods without resorting, at least partially, to slicing. Specifically, none of these methods would enable printing along the helix, which is likely to result in a better surface finish (as in Figure 1), and probably a stronger model [2]. Additionally, none of these method allows the user to easily (if at all) specify the directions of the printing-paths.

AM using slicing is general - the volume of *any 3D closed object* can be covered by piecewise linear curves and manufactured using slicing. In this work, we strive for a similar general covering AM solution for any closed 3D object, while also offering the freedom to use almost any set of 3D print-paths. We expect that by accepting general models as input, and automatically filling (covering) their entire volume with curves that fit a model's specific structure and specific requirements (i.e. stress tensors), better quality models can be manufactured using AM technologies such as FDM and DED.

One additional previous work we would like to mention is [17], which is a part of some of the algorithms presented in this paper. In [17], an algorithm for adaptively covering freeform surfaces with curves is presented. The concept behind this algorithm is simple: given two parallel (in the parametric space) isoparametric curves on a surface, we can check if they sufficiently cover the surface area bounded between them. If they do then the curve coverage is sufficient obviously. Otherwise we add isoparametric curves between them (where needed according to some distance measure) and invoke the algorithm recursively for each pairing of the new curve and one of the original curves. This concept of adaptively covering surfaces was also used to generate CNC tool-paths [18]. Similarly, in this work, we show that volume covering by curves can be used for generating AM print-paths.

3. Considerations for AM Print-Path Planning

Given a description of a 3D closed object (such as a 3-manifold volume, or a 2-manifold boundary representation), there are several considerations to observe when generating the print-paths needed to manufacture the object using AM. These considerations are outlined in Sections 3.1 to 3.4.

3.1. Volume Coverage By Curves

We start by defining the notion of *covering*:

Definition 3.1. *Consider a volume V of some closed 3D object and a desired tolerance, $\epsilon \in \mathbb{R}^+$. A valid ϵ* curve-covering *of V is a set of n univariate parametric curves $\boldsymbol{C} = \{C_1(t) \dots C_n(t)\}$, $\boldsymbol{C} \subset V$, so that for any point $p_v \in V$ there exists a point $p_c \in C_i(t)$, $C_i(t) \in \boldsymbol{C}$, for which $\left\|p_c - p_v\right\| \leq \epsilon$.*

Definition 3.1 ensures that when the object is printed using the curves in $\boldsymbol{C}$ as print-paths, the entire volume will be filled (with material), as all points in V are close enough to some print-paths and will be covered by printed material. While all points in the volume should be covered, over-coverage should be avoided. Over-coverage occurs when points in the volume are covered too many times, which will result in an excess of material being deposited (and the resulting 3D object would likely be deformed).

In slicing, coverage is achieved in a trivial way, each slice covers the parts of the object in a certain z value range corresponding to the height of the slice. Internally, each slice is covered by 2D curves (often as a set of lines parallel to the x or the y axis).

3.2. Accessibility and Ordering of the Print-Paths

At any point during the 3D-printing process, parts of the model are already printed, while others still need to be printed. The printing head has a known geometry, meaning it occupies some known physical space. If there is no way to print some unprinted portion, without the printing head penetrating an already printed part, then the printing process cannot succeed: either an unprinted part will never be printed, or an already printed part will be gouged into and destroyed. Ordering (portions of) the print-paths of the model so that all of them can be printed is one of the requirements of the printing process.

In traditional slicing methods, the geometry of the printing head is assumed to occupy the half-space above a plane parallel to the XY plane, and is always at a certain offset (in the z axis direction) above the currently printed part. Since slices are printed in planes parallel to the XY plane, from bottom to top, we are always assured that no penetration of the printed parts by (the geometry of) the printing head will occur.

3.3. Generating a Support Structure

In general, some AM technologies require support [19]. For any part of a manufactured 3D object that will normally collapse while it is being printed, a corresponding part must be created, in advance, to support it and prevent this collapse. The union of all the extra support parts created is called the support structure. The generation of the support structure is closely related to the order in which the model parts are created, as many parts are already supported by previously printed pieces of the 3D object.

Using slicing, support generation is relatively simple: any print-path that extends a certain threshold beyond previously printed slices requires support, and the support is printed in slices, along with the object itself.

3.4. Design Streamlines

Apart from fulfilling the requirements of the printing process, print-paths should also address the design goals of the user:

Definition 3.2. *Given a volume V of some closed 3D object and a set of design goals prescribed by the user, the curves that form a valid coverage of V while fulfilling the design goals in an optimal manner, are denoted* design streamlines *or simply* streamlines *of volume V.*

For example, streamline print-paths, possibly defined as a vector field over V, could be designed so that they create a part with better mechanical strength and/or surface finish.

In slicing, this effort is usually limited to tracing the outline of each slice, to enhance the surface finish, and determining the density with which the slice is filled, to enhance mechanical strength.

4. Generating a Streamline Coverage

In this section, we present algorithms that can be used to incorporate geometric or other streamline design requirements as the main considerations for planning AM print-paths. Definition 3.2 is rather amorphous, as it depends on a set of design goals prescribed by the user. Clearly, given the wide range of possible design goals, creating one algorithm that will satisfy them all would be next to impossible. Instead, in this section, we present algorithms that will provide designers with the tools to achieve their own design goals. Designers will be able to specify general guidelines for covering curves, and the algorithms will automatically generate them, order them for printing, and optionally generate a support structure. Figure 3 illustrates the process. In Section 4.1, we present how a coverage by curves can be generated for (possibly trimmed) trivariate volume objects. Section 4.2 considers a second directional field, possibly defined as a trivariate as well, to direct the coverage and support any B-rep based 3D closed object.

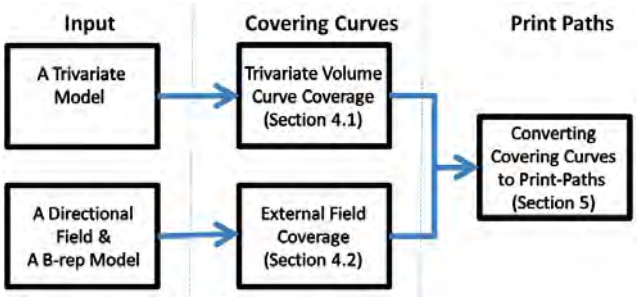


Figure 3: The model created by the designer (in either of two forms of input), is covered by covering curves (using the method described in Section 4.1, or the method presented in Section 4.2), which are then turned into valid AM print-paths using the methods explained in Section 5. The final print-paths can then be used to fabricate the original input model.

4.1. Covering of a Trivariate Using Curves

Let D be a box domain in $\mathbb{R}^3$. Consider a parametric trivariate volume $V(u,v,w)$: $D \to \mathbb{R}^3$ and a desired tolerance, $\epsilon \in \mathbb{R}$.

Definition 4.1. *A valid* ϵ surface-covering *of V is a set of m bivariate parametric surfaces* $\boldsymbol{\mathcal{S}} = \{S_1 \dots S_m\}$, $\boldsymbol{\mathcal{S}} \subset V$, *so that for any point* $p_v \in V(u,v,w)$ *there exists a point* $p_s \in S_i$, $S_i \in \boldsymbol{\mathcal{S}}$, *for which* $\left\|p_s - p_v\right\| \le \epsilon$.

Given Definitions 3.1 and 4.1, we seek to formulate an algorithm that will find a valid covering by curves, for a given trivariate parametric volume $V(u,v,w)$, while limiting the number of redundant curves. The algorithm we present is based on concepts presented in [17]. The algorithm first adaptively covers a given trivariate volume with a set of bivariate parametric surfaces. Once the volume coverage by surfaces is computed, a volume coverage by curves is obtained by using the algorithm from [17] that covers the returned surfaces with curves. The heart of the algorithm stems from an upper bound on the Hausdorff distance between two surfaces, that is based on an *iso-distance* notion:

Definition 4.2. *Consider the (piecewise) polynomial or rational parametric surfaces,* $S_1(u,v)$, *and* $S_2(u,v)$, *sharing a common* (u,v) *domain. The vector field*

$$D(u,v) := S_1(u,v) - S_2(u,v)\,,$$

is denoted the iso-distance *between* S_1 *and* S_2, *as:*

$$\Delta_{iso}(u,v) := \left\|D(u,v)\right\|\,.$$

Because $\Delta_{iso}(u,v)$ is non-rational (as it contains a square root), $\Delta^2_{iso}(u,v)$ (the iso-distance squared, represented as a spline function) will be used instead. The iso-distance is closely related to the Hausdorff distance:

Definition 4.3. *The* Hausdorff distance*, denoted* d_H*, between two surfaces* S_1, S_2, *is:*

$$\mathrm{d}_H(S_1,S_2) = \max\left\{\sup_{a\in S_1}\inf_{b\in S_2}\|a-b\|\,,\sup_{p\in S_2}\inf_{q\in S_1}\left\|p-q\right\|\right\}\,,$$

and the one sided point-surface Hausdorff distance is:

$$\mathrm{d}_{Hs}(a\in S_1,S_2) = \inf_{b\in S_2}\|a-b\|\,.$$

We can now prove the following:

Lemma 4.1. *The iso-distance* $\Delta_{iso}(u_0,v_0)$*, between any corresponding pair of points* $a = S_1(u_0,v_0)$, $b = S_2(u_0,v_0)$, *is an upper bound on the one sided point-surface Hausdorff distance of both* $\mathrm{d}_{Hs}(a\in S_1,S_2)$, *and* $\mathrm{d}_{Hs}(b\in S_2,S_1)$.

Proof. Combining the definition of the one sided point-surface Hausdorff distance and the iso-distance we get:

$$\mathrm{d}_{Hs}(a\in S_1,S_2) = \inf_{b'\in S_2}\left\|a-b'\right\| \le \|a-b\| = \left\|S_1(u_0,v_0) - S_2(u_0,v_0)\right\| = \Delta_{iso}(u_0,v_0)\,,$$

with a similar result for $\mathrm{d}_{Hs}(b\in S_2,S_1)$. □

In our calculations, we use $\Delta^2_{iso}(u,v)$ instead of the Hausdorff distances that are far more difficult [20] to compute. Algorithm 1 creates a covering of a trivariate volume, V, with isoparametric (trimmed) surfaces of V, in the w direction. We

Algorithm 1 CoverVolumeWithSurfaces

Input:

(1) $V(u,v,w) : D \to \mathbb{R}^3$, a trivariate volume, $D = [0,1]^3$;

(2) m, a minimal subdivision depth to apply to V;

(3) ϵ, the maximum desired distance from a point in the volume to a covering surface;

Output:

(1) $\mathcal{S} = \{S_1(u,v) \dots S_n(u,v)\}$, w-isoparametric (trimmed) surfaces covering V to within ϵ;

1: **CoverVolume**$(V(u,v,w), m, \epsilon)$
2: Return
$\{V(u,v,0)\} \cup$
CoverSubVolume$(V(u,v,w),(0,1),m,\epsilon) \cup$
$\{V(u,v,1)\}$;

3: **CoverSubVolume**$(V(u,v,w),(w_{low},w_{high}),m,\epsilon)$
4: $\mathcal{S} := \varnothing$;
5: $D(u,v) := V(u,v,w_{low}) - V(u,v,w_{high})$;
6: $\Delta^2_{iso}(u,v) := \left\|D(u,v)\right\|^2$;
7: **if** $\Delta^2_{iso}(u,v) > \epsilon^2$ for some (u,v) or $m > 0$ **then**
8: $w_{mid} := \frac{w_{low}+w_{high}}{2}$;
9: **if** $m > 0$ **then**
10: $S^t_{mid}(u,v) := V(u,v,w_{mid})$
11: **else**
12: $S_{mid}(u,v) := V(u,v,w_{mid})$;
13: $D^t(u,v) := \{(u,v) | \Delta^2_{iso}(u,v) > \epsilon^2\}$;
14: $S^t_{mid}(u,v) := \{S_{mid}(u,v) | (u,v) \in D^t(u,v))\}$; // S^t_{mid} is a trimmed surface, with trimming domain $D^t(u,v)$.
15: **end if**
16: $\mathcal{S} :=$
CoverSubVolume$(V(u,v,w),(w_{low},w_{mid}),m-1,\epsilon) \cup$
$\{ S^t_{mid}(u,v) \} \cup$
CoverSubVolume$(V(u,v,w),(w_{mid},w_{high}),m-1,\epsilon)$;
17: **end if**
18: Return $\mathcal{S}$;

will later discuss the problems arising from the lack of tightness in the proposed measure (of $\Delta^2_{iso}(u,v)$ compared to d_H), and how they can be mitigated.

For now assume $m = 0$ (Algorithm 1, input (2)). Once $\Delta^2_{iso}(u,v)$ is computed (line 6 in Algorithm 1), a determination can be made if another surface should be introduced between the two iso-surfaces of V, at $w = w_{low}$ and $w = w_{high}$, to get a valid coverage, following Definition 4.1. If $\Delta^2_{iso}(u,v) < \epsilon^2$, $\forall(u,v)$, no additional surfaces are needed in between and the algorithm terminates. If $\Delta^2_{iso}(u,v) > \epsilon^2$ for some (u,v) values, then a trimmed surface (line 14) is introduced. The tensor product surface $S_{mid}(u,v) = V(u,v,w_{mid})$ is trimmed to only include (u,v) values for which $\Delta^2_{iso}(u,v) > \epsilon^2$. If a middle surface has been introduced then the algorithm is invoked recursively for the newly created pairs of adjacent surfaces, (w_{low}, w_{mid}) and (w_{mid}, w_{high}), to further verify that the volume between them is covered.

Once a valid coverage of V by surfaces is produced using Algorithm 1, a curve coverage is created by covering each trimmed surface with curves. The coverage by curves is realized by using the algorithm described in [17] that functions in a similar manner to Algorithm 1 but with a surface input, starting with two surface boundary curves and recursively adding additional intermediate isoparametric curves, as needed. An example of the full process starting with a volume, V, covering it with surfaces, and then cover the surfaces with curves, can be seen in Figure 4. Note how the surfaces alternate adaptively between full and trimmed surfaces. Figure 4 also shows a coverage of $V(u,v,w)$ by surfaces, for different values of ϵ, and for an alternate parametrization direction, all of which result in different sets of covering iso-surfaces. The ability to choose the parametrization (and ϵ), would allow designers to have a greater control over the resulting covering curves, and ultimately print-paths. Finally, note m is defined to ensure a minimal depth of recursive calls, for example in case $V(u,v,w)$ is a periodic trivariate (i.e. a full torus), where $V(u,v,w_{min}) = V(u,v,w_{max})$.

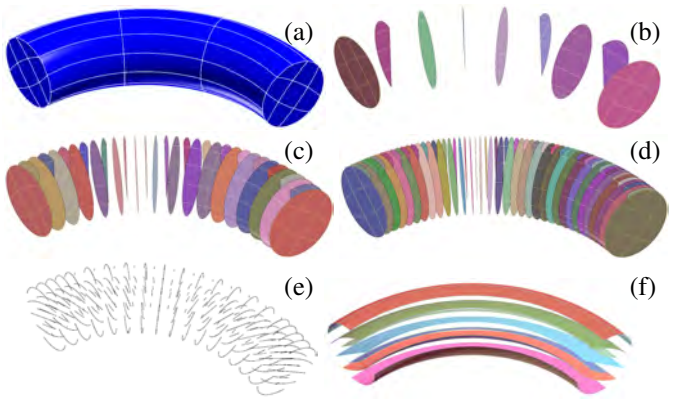


Figure 4: A trivariate, V, of a quarter of a torus (with a major radius 1, and a minor radius of 0.2) (a). Adaptively covered by w-isoparametric surfaces and $\epsilon = 0.4$ (b), adaptively covered by w-isoparametric surfaces and $\epsilon = 0.1$ (c), adaptively covered by w-isoparametric surfaces and $\epsilon = 0.05$ (d), and adaptively covered by curves $\epsilon = 0.1$ (e). In (f), the parametrization direction was changed to v-isoparametric surfaces, to produce a different set of covering surfaces ($\epsilon = 0.1$).

We note that Algorithm 1 can also be adapted to trimmed trivariates [21], by trimming the surfaces according to the trimming information of the trivariate, in addition to the trimming done in the algorithm. However, only points at a distance of more than ϵ from the trimmed boundary are guaranteed to be covered by the same entities that covered them in the untrimmed version. Points closer than ϵ to the trimmed boundary may become uncovered when their covering entities fall outside the trimmed boundary.

If $\Delta^2_{iso}(u_0,v_0) \leq \epsilon^2$, then any point on the line between $V(u_0,v_0,w_{low})$ and $V(u_0,v_0,w_{high})$ is covered, as its distance to one of these surface points is less than ϵ (in fact less than $\frac{\epsilon}{2}$). However, $\Delta^2_{iso} \leq \epsilon^2$ for any (u,v) does not imply all points in the volume $V(u,v,w_r)$, $w_r \in (w_{low}, w_{high})$ are covered. If some point p_w on the w-isoparametric curve $V(u_0,v_0,w_r)$ is sufficiently far from the line between $V(u_0,v_0,w_{low})$ and $V(u_0,v_0,w_{high})$, it's possible that p_w remains uncovered. Figure 5 shows a situa-

tion in which $S_1 = V(u, v, w_{low})$ and $S_2 = V(u, v, w_{high})$ can be to within ϵ, but the volume in between them also contains points (along a w-isoparametric curve between them) that are too far from both. To resolve this issue, *that in regular trivariates can only happen on the boundaries*, one can enhance the measure of Δ^2_{iso}. One possible measure, that can ensure complete coverage, can be created by bounding the arc-length of the w-isoparametric curves. For example, given an isoparametric curve from one covering surface to another, if the arc length of the curve is less than 2ϵ then a full ϵ-coverage is assured for all points on the curve. Establishing such bounds for all isoparametric curves can be used to ensure complete volume coverage. See Appendix A for an explanation and a proof of these claims. This more accurate measure will also eliminate the need for the parameter m, in Algorithm 1.

It should be noted that as long as a situation like the one in Figure 5 does not occur, continuing the recursion with the full surfaces (as done in Algorithm 1) is equivalent to using only the common trimmed portion of the surfaces in the recursion. This is because, given the two input surfaces in some recursion step, as long as the used distance measure decreases monotonically when closer (in the w parametric sense) surfaces are used, any region trimmed by a comparison between the two input (tensor product) surfaces will be similarly trimmed by a comparison between any two (trimmed) surfaces between them that were created by the recursion. For example, Δ^2_{iso} used in Algorithm 1 is not always a monotonic measure, but the suggested measure based on the arc-length of the w-isoparametric curves (i.e. Appendix A) is.

Finally, consider non-regular trivariate volumes (i.e. volumes with a vanishing or negative Jacobean) that contain self intersections. While simple to detect and not very useful in the context of fabrication, Algorithm 1 can handle such volumes as the distance bounds are still valid. However, the generated covering surfaces themselves may also be non-regular, in such cases, and also introduce redundant coverage.

4.2. Object Coverage with an Additional Direction Field

Algorithm 1 and the algorithm in [17] can be used together to cover a trivariate volume using (a subset of) its own isoparametric curves. This, however, limits the coverage to the parametrization of the given volume, and the object description to a trivariate. In cases where a more general set of covering curves and 3D object descriptions are needed, an additional directional vector field (that associates a direction to each point in the volume) can be specified along with the 3D object, to define the desired directions of the covering curves in the object. A good directional field would optimize attributes like desired mechanical strength or surface finish. The field can be manually designed or automatically generated using some form of optimization or analysis. An alternative algorithm can be devised that, given a 3D object and a 3D directional vector field (that encompasses the entire object), will create a set of covering curves for the object that will conform to the prescribed directions.

We've explored several approaches that implement the covering of an object using an additional directional field. The one we found most useful is composed of two stages:

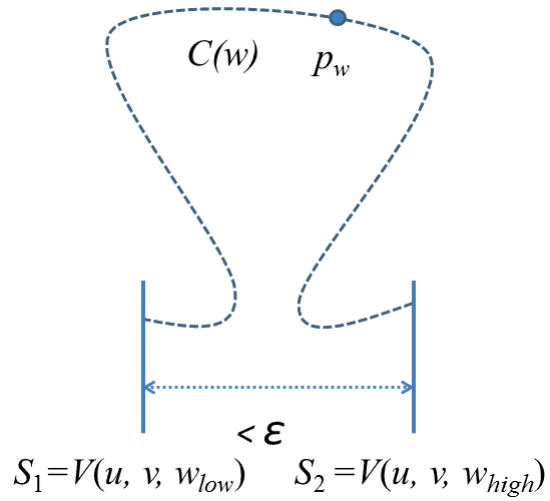


Figure 5: Given two close ($< \epsilon$) surfaces (depicted as vertical solid lines), and one of the iso-curves between them (dashed line) in trivariate $V(u, v, w)$, some points along the iso-curve may remain uncovered using Algorithm 1.

1. First, a curve coverage for a volume containing the 3D object along the directional vector field is generated. If the direction field is given as a trivariate, then the approach in Section 4.1 can be used. However, since the structure of the direction field can be chosen to make coverage easier, other options may be available. For example if all the desired covering curves were parallel to each other, then creating one 3D curve and uniformly filling the containing volume with 3-space general offsets of that curve (according to its normal and bi-normal) could also be used.

2. Given the curve covering for the containing volume, the curves are clipped to the 3D object we need to cover. The clipping can be done by finding the intersections of the generated covering curves and the object boundary.

The implication of the above approach is that any B-rep model (regardless of the representation) and any set of curves that at least cover it, can be used as a basis for generating AM print-paths. Figure 6 shows how this method allows arbitrary 3D models specified as B-reps (polygonal meshes), to be covered using arbitrary curves. In Figure 6, the curves covering the body of the crocodile (a polygonal mesh) are slightly arched, while the curves covering the legs were shaped to resemble the main axis of the leg. The creation of the print paths in Figure 6 begins by dividing the crocodile (B-rep polygonal closed) mesh into five separate meshes (body and four legs). Each of these meshes is assigned a direction field, and the covering curves of each directional field are clipped to be inside the corresponding mesh, prescribing the resulting print-paths. **The example in Figure 7 goes one step further, applying a single arbitrary field, shown in Figure 7 (a), to the entire model. The crocodile model examples in Figures 6 and 7 show that even curves that are possibly unrelated to the geometry of the ob-**

ject can be used as covering curves. In the next section, we'll see how these curves can be used as print-paths for AM.

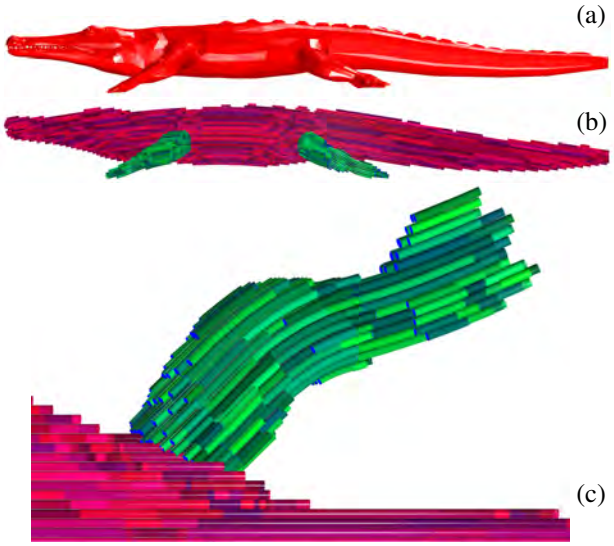


Figure 6: (a) shows a 3D model of a crocodile (a triangle mesh downloaded from https://free3d.com/). (b) shows the same model covered by general 3-space curves. Note the curves covering the body are slightly bent, while the curves covering the legs resemble the general shape of the leg. (c) shows the leg coverage in better detail, while the different colors accentuate the differences between the leg and body covering curves.

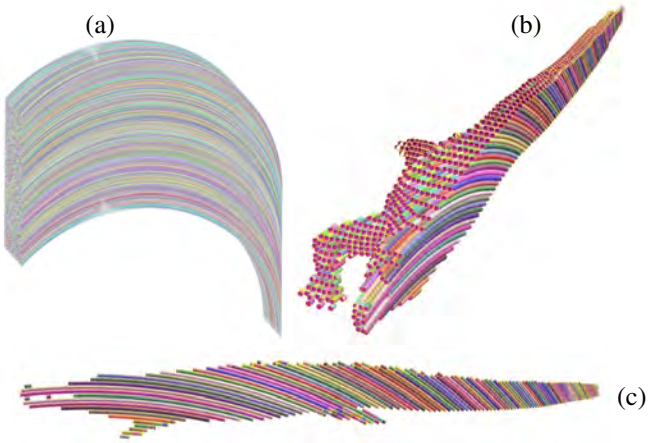


Figure 7: A field created by duplicating offsets of a single arc is shown in (a). This field is then applied to the mesh in Figure 6 (a). Images (b), and (c) show half of the resulting print-paths, exposing the interior.

5. Manufacturing 3D Objects Using Univariate Paths

In this Section, we show how all the coverage algorithms presented so far can be used to create print-paths to be utilized in AM. Covering curves (in the sense of Definition 3.1) cover every point in the volume, and so if they are used as AM print-paths, every location in the volume will be 3D-printed. However, adapting the approaches described in Sections 4.1 and 4.2, to AM, also introduces several challenges we will have to address and are discussed below. Section 5.1 discusses the proper ordering of the covering curves (that are also potentially subdivided) when realizing them as AM print-paths in order to ensure printing accessibility, Section 5.2 explores the amount of material deposition related to each covering curve, and Section 5.3 discusses the generation of the support structure. Finally, Section 5.4 demonstrates how different covering curves can affect the overall printing solution.

5.1. Resolving Accessibility - Curve Ordering

We need to ensure the print-paths can be followed by the printing head without gouging already printed parts, during the entire printing process. We assume printing with a constant orientation (up direction) printing tool:

Definition 5.1. *A curve C_i is* below *curve C_j and denoted $C_i < C_j$, if the geometry of the printing head overlaps with curve C_j when printing curve C_i. Symmetrically, curve C_i is* above *C_j ($C_i > C_j$) if C_j is below C_i.*

Note that Definition 5.1 assumes the build direction is already known, and is assumed to be the direction of the z axis. Following Definition 5.1, at any point in time during the 3D-printing process, only curves that are not yet printed and have no unprinted curves below them may be printed.

Definition 5.2. *Consider a set of n general 3-space curves $\boldsymbol{C} = \{C_1(t) \dots C_n(t)\}$. Construct a directed graph $G_a(\mathcal{V}, \mathcal{E})$ in the following manner: each curve $C_i \in \boldsymbol{C}$ is assigned a vertex $v_i \in \mathcal{V}$. A directed edge $e_{ij} \in \mathcal{E}$, from v_i to v_j, exists if $C_i < C_j$. We denote by G_a the* accessibility graph *for $\boldsymbol{C}$. Any cycle in the graph G_a is called an* inaccessibility cycle*, as it reflects an impossible-to-print set of curves.*

G_a defines the correct printing order for a set of curves. If a path exists from v_i to v_j in G_a, it means curve C_i must be 3D-printed before curve C_j. Any order that complies with the topological order of G_a [22] would be a valid printing order, since no printed curve would interfere with curves printed after it. An inaccessibility cycle means no printing order can be found, and the 3D-printing process can only possibly be realized by subdividing (some of) the curves in $\boldsymbol{C}$ into a new set of curves $\boldsymbol{C}_{new}$, that has an acyclic G_a, as in Figure 8. Clearly, it is better to limit the number of curves we subdivide, because, for example, a continuous printing path is likely to yield a stronger part than one made up of separate printed elements [2].

To build G_a for $\boldsymbol{C}$, we must also be provided with the geometry of the printing head. We have chosen to model the geometry of the printing head as a downward (axis parallel to the z axis) pointing cone (see Figure 9), as this cone can serve as a bounding cone for the shapes of many printing heads. The angle of the cone, θ, and the z-offset of the tip above the currently printed part, z_{offs}, are set according to the geometry of the actual extruder in use. Just like layer-height in slicing, z_{offs} represents the limit of the printer resolution, and anything smaller is considered negligible. Setting z_{offs} to a value that is less than the minimal printing width (or the minimal clearance between two curves), also ensures that no printed curve will exist in the z_{offs} clearance between the extruder and the currently printed curve.

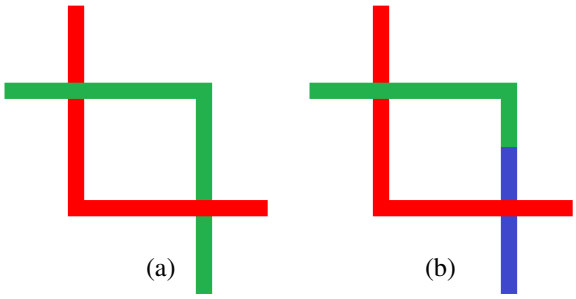


Figure 8: The two space curves in (a) cannot be ordered, as the red curve is both above and below the green curve, creating an inaccessibility cycle. The green curve is subdivided to enable an ordering (blue, red, then green) (b).

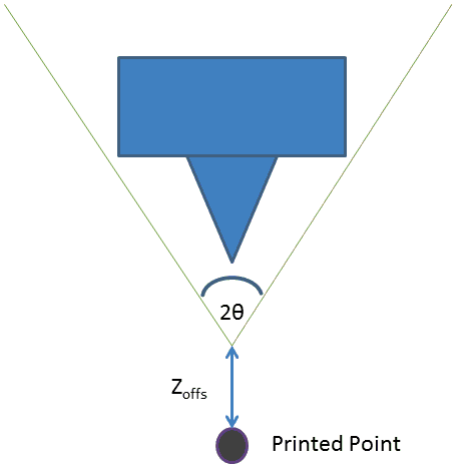


Figure 9: The object used to model the geometry of the printing head (blue), and its bounding cone (green).

To compute if curve $C_i < C_j$, we would need to check whether the volume that encloses the sweep of the cone along curve C_i, intersects C_j (see Figure 10). An intersection means $C_i < C_j$. Doing so for all pairs of curves in $\boldsymbol{C}$ constructs G_a. In practice, this procedure requires every pair of curves to be evaluated and would make the construction of G_a slow, even for a few hundred covering curves, which is not an impractical number. Instead, we calculate a conservative approximation of G_a, $\bar{G}_a$. The conservative approximation can use spatial division acceleration structures, for example a BVH (bounding volume hierarchy), or a z-buffer, and can be done relatively efficiently once a curve is subdivided into sufficiently (for the acceleration structure) spatially compact sub-curves. Currently, we are using a BVH query to find (a conservative approximation of) all sub-curves that are above other sub-curves. Given the information of which sub-curves are above other sub-curves we build $\bar{G}_a$.

Once G_a (or $\bar{G}_a$) is known, we aim to order the curves. If G_a is acyclic (a DAG), any order that concurs with the topological order imposed by G_a can be employed and no subdivisions are needed. For example, we can initially find all accessible curves (that have nothing below them) and add them to a list of available print-paths. We can then keep printing curves from that list while adding new ones as they become accessible. Since the initial set has an acyclic G_a, this process will successfully terminate. Alternatively, if inaccessibility cycles exist, we can pursue either a top down, or bottom up approach to find where to subdivide the curves (in the cycle). In the top down approach, we start with the full coverage curves and subdivide them (in a way that would minimize the total number of subdivisions) and rebuild G_a until no cycles remain. The bottom up approach starts with the curves subdivided into small sub-curves called fragments:

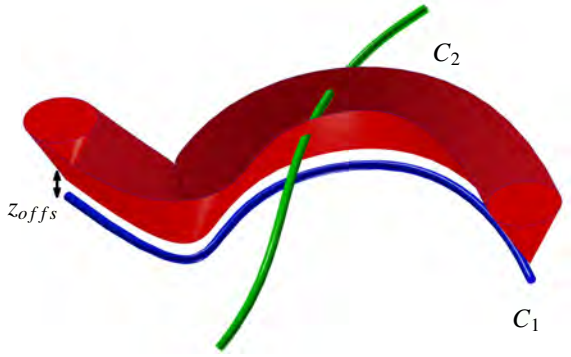


Figure 10: The volume of a sweep of the cone bounding the printing head along the lower curve, C_1, defines which portions of the upper curve, C_2, are above it, denoted $C_2 > C_1$.

Definition 5.3. *Assume $z_{offs} > 0$ (i.e. Figure 9). Any curve whose length is less than z_{offs} will be called a* fragment.

Lemma 5.1. *In a set of fragments, the lowest fragment (the one that contains the point with the lowest z value) can always be printed.*

Proof. Let $\mathcal{F} = \{f_1(t) \dots f_m(t)\}$ be a set of fragments. Assume f_1 is the lowest fragment, and its lowest point has a z value of z_0. Since the length of the fragment is less than z_{offs}, the highest point in f_1 is lower than $z_0 + z_{offs}$. Given Definition 5.1, to be below f_1 another fragment would have to contain a point lower than z_0. Since f_1 contains the lowest point with a z value of z_0, no such fragment exists. As no fragment exists below f_1 (following Definition 5.1), it can be printed. □

Intuitively, Lemma 5.1 states that since the printing head is z_{offs} above the print point, it can always print the lowest fragment without penetrating other fragments. The implication of Lemma 5.1 is that any set of fragments can be printed by sequentially printing the lowest fragment each time and removing it from the set. Hence, the initial set of fragments has an acyclic G_a (since it can be printed). In the bottom up approach, we keep merging pairs of fragments (or the sub-curves resulting from previous merges) while maintaining an acyclic G_a until we end up with a minimal number of curves.

Regardless of which approach we use, finding the optimal subdivision can be shown to be an NP-complete problem by showing that the monotone planar 3-SAT problem can be reduced to the optimal subdivision problem here (the monotone

planar 3-SAT problem is presented in [23]). Given this difficulty in finding the optimal solution, we resort to heuristic solutions. Algorithm 2 gives a short outline of the process:

Algorithm 2 OrderCurves

Input:

(1) $\boldsymbol{C} = \{C_1, \ldots, C_n\}$ a set of curves to be ordered for 3D printing;

(2) Θ, z_{offs}, describe the printing head geometry;

Output:

(1) $\boldsymbol{C}_{ordered}$, the ordered (for 3D printing) list of (subdivided) curves from $\boldsymbol{C}$;

Algorithm:

1: $\boldsymbol{C}_{ordered} := \varnothing$;
2: $\mathcal{F} := Fragments(\boldsymbol{C}, z_{offs})$; // f_i^j marks fragment j in C_i;
3: **while** $\mathcal{F} \neq \varnothing$ **do**
4: $\quad \mathcal{B} := \left\{ f_i^j \mid Below(\Theta, z_{offs}, f_i^j) \cap \mathcal{F} = \varnothing \right\}$;
5: $\quad B_{best} := GetBest(\mathcal{B}) = \left\{ f_i^k, \ldots, f_i^{k+l} \right\} \in \mathcal{B}$;
6: $\quad \mathcal{F} := \mathcal{F} \setminus B_{best}$;
7: $\quad \boldsymbol{C}_{ordered} := \boldsymbol{C}_{ordered} \cup B_{best}$;
8: **end while**
9: Return $\boldsymbol{C}_{ordered}$;

In Line 4 of Algorithm 2 the function $Below(\Theta, z_{offs}, f_i^j)$ returns the set of fragments that are below fragment f_i^j (if any), according to Definition 5.1. Lemma 5.1 proves that at least one fragment (the lowest one) will be included in $\mathcal{B}$. As explained earlier we resort to a heuristic solution in identifying the best sequence of fragments to print each step (Line 5). We choose the fragment sequence that would introduce the least amount of subdivisions in the current step, but may ultimately (because of the greedy nature of the choice) result in more subdivisions. The entire procedure is bound to terminate as at least one fragment is removed from $\mathcal{F}$ every cycle (Line 6). The end result of this greedy process is a list of sub-curves $\boldsymbol{C}_{ordered}$ (with an acyclic G_a) that can be 3D printed (using AM).

Figure 11 shows the result of the process, when applied to the helical object (from Figure 2), covered by nine curves.

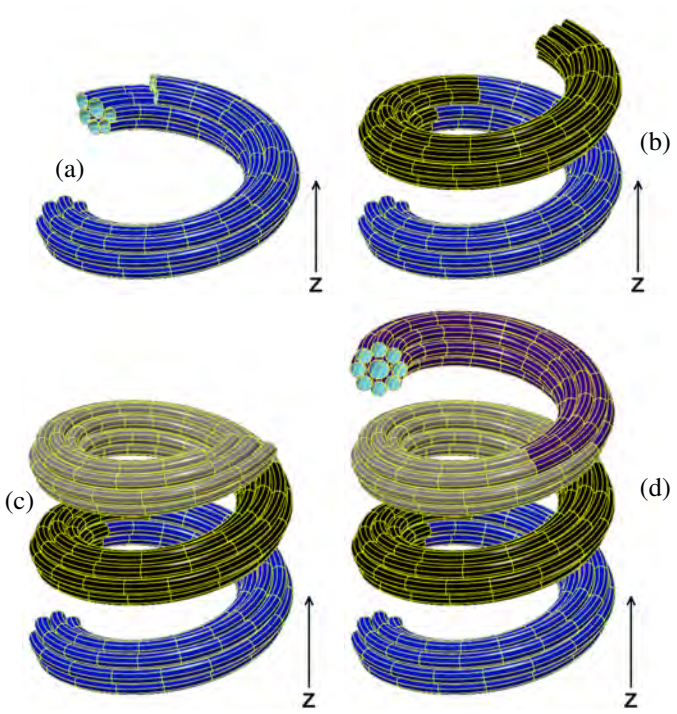


Figure 11: A helical object (from Figure 2) covered and printed by nine curves (shown as tubes), making sure no gouging occurs as the helix is being printed. First 9 sub-curves are printed in (a). First 18 sub-curves are printed (b). First 27 sub-curves are printed (c). All 36 sub-curves are finally printed (d), completing the model. Note the middle tube has a larger radius.

5.2. Setting the Material Deposition Radius Along the Print-Path Curves

Another AM print-paths concern relates to the amount of deposited material along the path. As mentioned in Section 3.1, extruded material should both cover the entire volume of the printed object, while also avoiding over-coverage that results from too much material being extruded. In AM, covering the same location with more than one covering curve can be problematic: when a point is covered more than once, material will also be deposited there multiple times, resulting in excess material being placed.

The print-paths created by the method described in Sections 4.1 and 4.2, are typically not parallel to each other, and the distance between adjacent curves would likely vary along their length. Assuming a circular cross section for the extruded material, the result will be excessive material deposition unless the extrusion radius is modified along the length of the curves. To reduce the amount of excess material extruded, and ultimately control the extrusion feed-rate for the print-paths, the volume each curve is assigned to cover must be set in a way that ensures locations are not covered multiple times, and consequently too much material is applied. Hence, we assign each curve, at any point, an effective coverage radius:

Definition 5.4. *The* material deposition radius *(MDR) function of a print-path or covering curve is the local amount of material that should be extruded along its length. If the deposited material along the print-path traces a virtual varying radius tube, as in Figure 11, the MDR sets the local radius of the tube, along the path.*

The subdivision into fragments, similar to the one used to approximate G_a in Section 5.1, can also be used to determine the MDR along a covering curve. For each fragment, the closest fragments are found, and the radius is set according to those neighboring fragments. This can be done in an iterative process that would expand the radius of each fragment's virtual tube until it touches the virtual tube of another fragment. This sort of process would ensure that no tube can be further expanded when it terminates.

5.3. Generating the Support Structure

There are many alternatives to generate support. Here, we briefly review the simple approach we took to manufacture the

parts presented in Section 6. To generate the support structure, we use ray casting (vertical rays, in a grid, using the boundary surfaces of the object) to identify the necessary volume of the support structure. We then fill the support volume (in a prescribed density) using a grid. To detect which support print-paths should be printed, we use the following procedure: given the next print-path of the object to be printed, the part of the support volume that is below that print-path, should be printed/filled before the object print-path. Recall that the volume below an object print-path, would be the volume enclosed in the sweep of a downward facing cone along the print-path curve. Overall this ordering ensures all of the print-paths of the object are supported, and the printing order of the object and the support print-paths does not cause collisions. Figure 12 shows a schematic example of this procedure. Note how both the support structure and object covering-curves are added in stages.

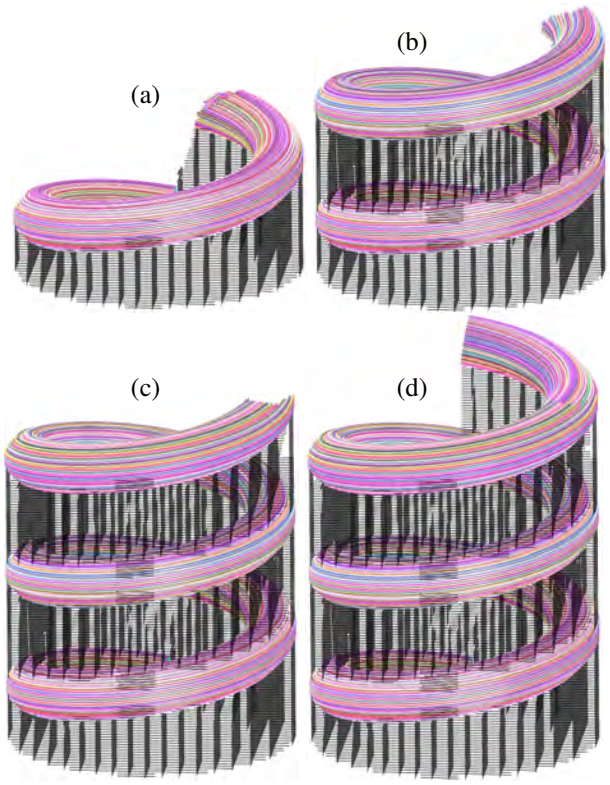


Figure 12: A schematic illustration of the helical object (from Figure 2) covered and printed by 1216 curves (shown as differently colored tubes), presented along with the generated support structure (shown as black lines). First 304 sub-curves and associated support structure are printed in (a). First 608 sub-curves and associated support structure are printed (b). First 912 sub-curves and associated support structure are printed in (c). All 1216 sub-curves and associated support structure are finally printed (d), completing the model. Compare to Figure 16 (b) where an actual print using the same type of support structure is shown.

5.4. Using Alternative Covering Curves

The solutions outlined in Sections 5.1, 5.2, and 5.3 are applicable to any set of covering curves. As discussed in Section 4.2, we are not necessarily limited to the original parametrization when choosing covering curves. For example: given a helical object (as in Figure 2) we can use a similar helical volume (but with a square cross-section), that has a more desired parametrization, as a direction field to achieve more uniformly distributed covering curves. Figure 13 shows an example of this.

In Figure 13 (a) the trivariate from Figure 2 is shown covered by a subset of its own isoparametric curves using the approach discussed in Section 4.1. One can easily notice the varying MDR (Definition 5.4), and the unacceptably thin print-paths. This is the result of the fact that the trivariate in question is actually singular, at the four boundary points of its circular cross-section, with a vanishing Jacobean there. However, we can still use the approach outlined in Section 4.2, and use an external direction field to eliminate this difficulty, and print the model using uniformly distributed covering curves. To do so, we use a direction field similar to the model in Figure 2, except instead of the singular circular cross-section, the direction field now has a square cross-section without any singularities. Covering curves for the model with the square cross-section can easily be generated by uniformly sampling its isoparametric curves in a grid pattern. The covering curves for the square cross-section model can be seen in Figure 13 (b). The generated covering curves that fall outside the boundary of the model in Figure 2 are clipped, as explained in Section 4.2, removing any sub-curves (or full curves) that are outside the boundary. We are left with uniformly distributed covering curves that cover the model from Figure 2, that can be used as the basis for AM print-paths, and are shown in Figure 13 (c). Using this approach, virtually any desired coverage configuration can be achieved, for example we could have easily replaced the grid configuration of the coverage in Figure 13 (b) with a honeycomb arrangement to achieve a tighter coverage.

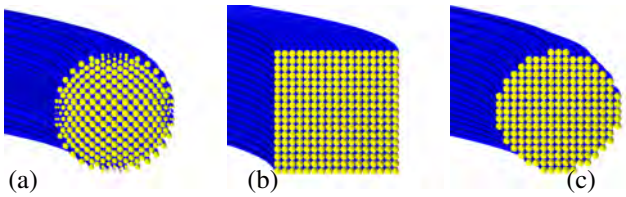


Figure 13: Partial view of the printing curves (tubes representing radius) for the volume in Figure 2. Using the original parametrization (a) resulting in varying radius values for the tubes. The printing curves, all of the same radius, of a direction field (the same volume with a square instead of circle cross-section) (b). Image (c) shows the printing curves of the direction field from (b) clipped to fit the original volume of the helix.

6. Experimental Results

We have implemented in C/C++ the algorithms outlined in this work. We then used these implementations to produce G-code instructions to print 3D objects, using a low-end FDM printer (seen in Figure 14). Unless otherwise noted, in these experiments the nominal print-path covering radius is 0.5 millimeters (as was z_{offs} for the cone), which is coarse but creates

more visible print-paths, and the default cone angle (θ, Figure 9) is 0.4π. We would like to stress these models are here as a proof of concept, showing fabrication of 3D models using the methods we outlined is indeed possible. The actual surface finish quality of the presented models is obviously limited, because of the low-end printer, and the coarse resolution we're using (to make print-paths more visible). For all presented models in this section, the total running times for the algorithms, including the generation of the G-code files needed to print the models, is about $2-10$ minutes on a 3.4 GHz windows 7 machine (single thread). Printing the object in Figure 15 took $4-6$ hours while the the one in Figure 16 took about 48 hours. Printing the model in Figure 16 requires frequent (and relatively long) z axis motions that are particularly slow on our printer that uses lead-screws in z. As mentioned in [15], prints that require frequent z axis motions can be sped up using a delta style 3D printer.

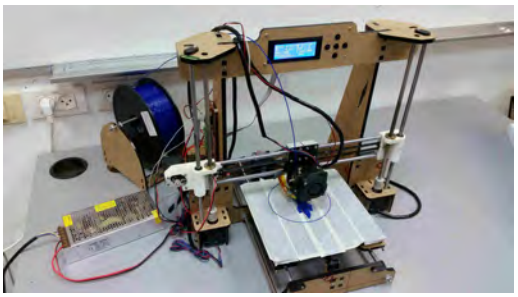


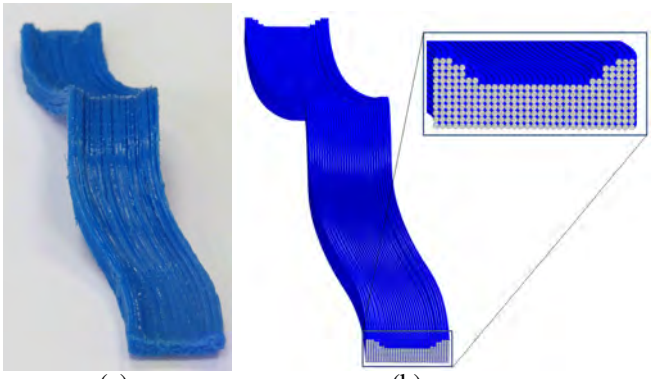
Figure 14: The FDM printer used in our experiments.

The model in Figure 15 was created using the methods outlined in Section 4.2. Specifically using a B-rep (a B-spline surfaces converted to a triangular mesh) of a bridge, and a rectangular cross-section trivariate that specified the external direction field for the print-paths. θ was set to 0.25π so that no subdivisions would be added. Figure 16 shows a manufactured object for the model in Figure 2. The model in Figure 16 (a), was also created using the method in Section 4.2, using a triangular mesh of the model's boundary (created by tessellating the boundary of a trivariate volume) and a square cross-section direction field as shown in Figure 13. Figure 16 also shows the effects accessibility considerations have. Figure 16 (c) shows how a subdivision of the covering curves affects the manufacturing process. The effect is minor, as the overall geometry of the print-paths remain unaffected.

The model in Figure 17, the spout of a Utah teapot, is modeled as a volumetric trivariate with some small varying wall thickness. The model was created using the methods outlined in Section 4.1. The nominal print-path covering radius is 0.3 millimeters, but the actual MDR is determined adaptively and changes along the print-paths.

Figure 18 shows a manufactured object in the shape of a vase. The cross section area of the vase differs substantially along its length. As a result, the number and MDR of the print-paths are changed adaptively in an effort to maintain a constant coverage of the volume.

Figure 19 presents the manufactured result for the model shown in Figure 6. The directional field used to cover the model



(a) (b)

Figure 15: A manufactured model of a twisted bridge, created using the method outlined in Section 4.2 (a) . The print-paths used for the twisted bridge modeled as tubes (and a zoomed view of the cross section) (b).

assigns a single representative guiding curve to each part of the model (body, and each of the four legs). For each part, a containing volume is generated by sweeping a sqaure cross section surface (as shown in Figure 13 (b)) along the representative curve. Then, a set of covering curves is computed by uniformely extracting isoparametric curves from this volume. So while the covering curves for this model match the general shape of the model, they do not match the boundary of the object, resulting in a worse surface finish compared to the other examples.

7. Discussion and Future Work

In this section, we discuss some of the technical issues we encountered in fabricating the models as well as related topics that require further research.

Throughout this work, we've largely neglected the consideration of choosing a build direction. The use of general print-paths, which partially mitigate the importance of the build direction, is the main focus of this work. However, the build direction still plays a major role in determining the quality of the resulting object, as it still affects the support volume and the number of subdivisions in the print-paths as discussed earlier. The determination of the best build orientation, given its many implications, has been studied for traditional AM (i.e. [24]). Similar studies should be performed to asses the best build direction for the AM method we propose.

In slicing, each layer is largely supported by the previous layer, and for sufficiently vertical slopes, no support is required [19]. AM using the methods we describe in this paper means adjacent 'layers' (or rather curves printed one after the other in this case) are no longer as similar to one another and as a consequence a printed part does not offer as much support to the next printed part. This makes support a much more critical issue, and apparently, it seems to be forcing us to use more support than would be required using slicing. Refer to Figure 1 (c), (d) for a comparison of the support structure for our methods, and traditional slicing for a simple model. Overall, we have used a

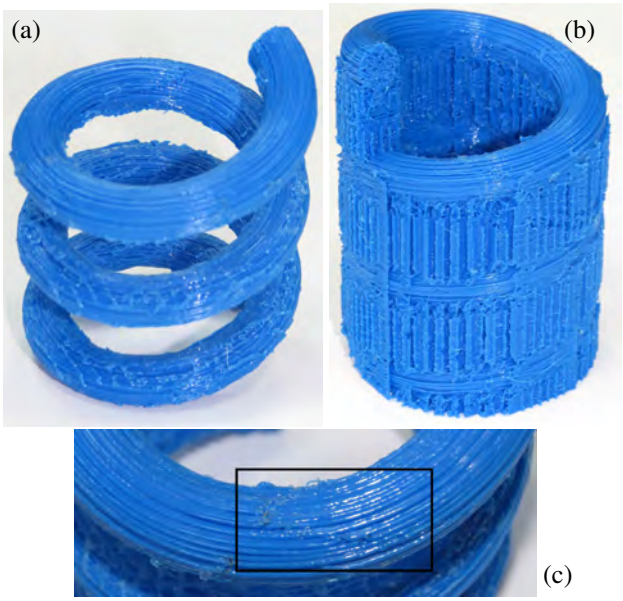


Figure 16: Manufactured object for the helical model presented in Figure 2, created using the method presented in Section 4.2 (a). In (b) the same model is shown before the removal of the support material. Image (c), shows a zoomed view of the transition (subdivision) area created because of accessibility considerations. See also Figure 20

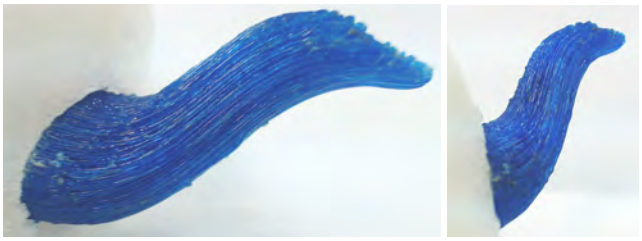


Figure 17: Manufactured object for the spout of a Utah teapot modeled as a volumetric trivariate (from two views), created using the method presented in Section 4.1.

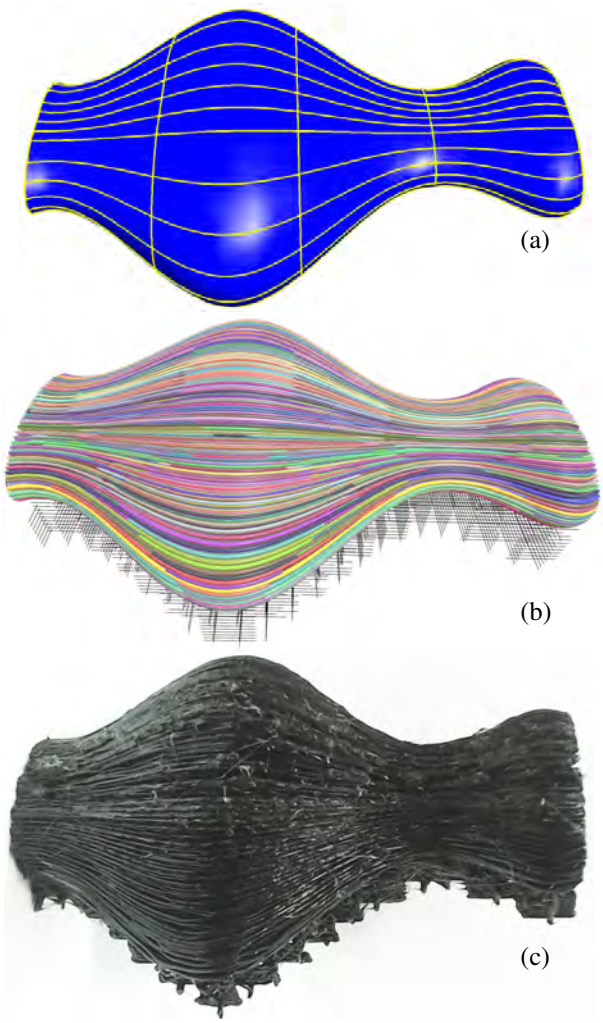


Figure 18: (a) shows the model of a vase. (b) shows a rendering of the print paths (object print-paths as colored tubes, support print-path as black lines) used to manufacture the object. A manufactured object for the vase model (the support structure was not removed), created using the method presented in Section 4.1 is shown in (c). Note how additional shorter print-paths are added in wider regions of the vase.

rather naive approach to generate the support, and more work needs to be done to optimize the support structure.

The solution we found for setting the MDR of print-paths in Section 5.2 may not be the optimal way to do so. For example, setting the radius so that there is some overlap between the virtual tubes of different print-paths may give better results, in certain cases, compensating for the inherent voids between paths with a circular cross-section. Further research needs to be done to examine how to best use the coverage radius of a covering curve to create a uniform and accurate coverage of the printed 3D object. Further, in reality, the extruded material is not circular, and supporting extrusion models with a non circular cross-section also requires further study. More work needs to be done to enable accurate control of the MDR, by either adaptively setting the extrusion rate, in existing printers, or by adding a dedicated valve mechanism to achieve a more accurate extrusion cross-section, in newer printers.

In Section 5.1, we discussed how covering curves sometimes have to be subdivided to create a printable set of curves. However, this subdivision only considers accessibility and may cause problems with other design goals. Figure 20 shows two possible sets of print-paths, modeled as tubes, for a model of a single cycle from the model presented in Figure 2. In Figure 20 (a) only accessibility considerations were used to subdivide the curves. The resulting subdivision creates a mechanical weakness in the model: it can be divided into two pieces held together only by the connection between the endpoints of the print-paths that make up the two pieces. This weakness can be alleviated by bonding or welding the endpoints together. This weakness can also be mitigated by changing the design of the covering curves. By forcing an additional subdivision, we were able to create an entirely different subdivision, seen in Figure 20 (b), that largely eliminated the weakness. Finding a way to automate this procedure toward, possibly, optimal mechani-

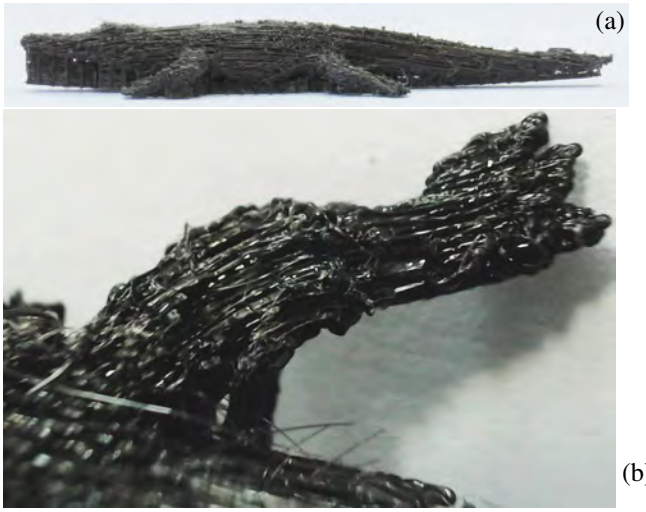


Figure 19: Manufactured object for a model of a crocodile (the support structure was not removed), created using the method presented in Section 4.2. In (a), the whole object is shown. (b) gives a more detailed view of the back right leg. See also Figure 6.

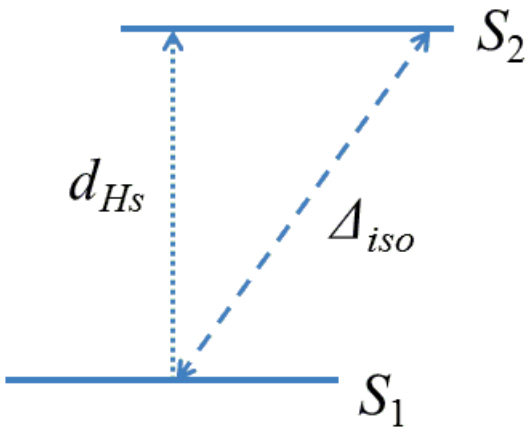


Figure 21: Given these two surfaces (depicted as lines) S_1 and S_2, the iso-distance Δ_{iso} between them (dashed arrow) from Algorithm 1, may be arbitrarily larger than the one sided point Hausdorff distance (dotted arrow).

cal strength, and prevent endpoints from clustering would be a worthwhile research goal.

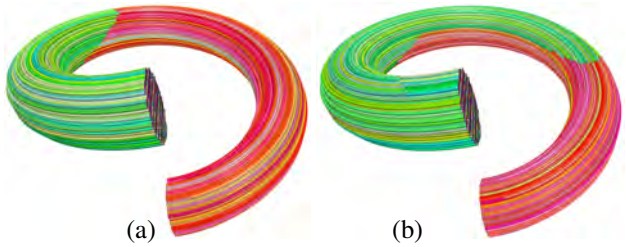


Figure 20: A division to print-paths that only considers accessibility (a). A division to print-paths that also considers the model's mechanical strength (b). In both cases, print-paths are rendered in perspective, modeled as tubes, and colored to highlight the subdivision.

Over-coverage can also occur in Algorithm 1 when Δ_{iso}^2 is overly conservative in relation to the true Hausdorff distance. One such case is illustrated in Figure 21. While rare in practice, reducing this skewing effect is highly desirable. By employing the first fundamental forms [25] of the adjacent surfaces and their relation in the local neighborhood in the trivariate, one can compensate for the variation in these distances between adjacent covering curves.

The ability to print non-planar curves is also applicable in the context of AM using functionally graded materials. One can alter material properties along print-paths, much like the MDR.

In this work, we have considered a constant orientation for both the printing head and the model. Printing using the same general print-paths presented, but using a multi-axis AM machine (such as in [5]) that can alter the orientations, would probably give superior result. Such a machine would be able to consider multi-axis accessibility options (rather than the single direction we considered), and reduce the need for subdivision of curves.

8. Conclusions

Given the wide range of desired covering curve properties (and the balance between them) no single method for generating covering curves would answer all possible needs. While we did present a method for generating covering curves for trivariate volumes, since there is a natural set of covering curves in that case, any set of covering curves can be used as the basis for print-paths.

We have shown how the slicing approach in AM can be augmented by algorithms to generate print-paths that follow any desired 3D directions. We expect that the added flexibility and freedom that the presented methods allow in the specification of general covering curves, will enable the synthesis of 3D models (using AM) with superior properties (such as mechanical strength and surface finish).

Acknowledgements

We thank the reviewers for their comments and suggestions that helped improve this work.

References

[1] P. Mohan Pandey, N. Venkata Reddy, S. G. Dhande, Slicing procedures in layered manufacturing: a review, Rapid prototyping journal 9 (5) (2003) 274–288.

[2] S.-H. Ahn, M. Montero, D. Odell, S. Roundy, P. K. Wright, Anisotropic material properties of fused deposition modeling abs, Rapid Prototyping Journal 8 (4) (2002) 248–257.

[3] P. Kulkarni, A. Marsan, D. Dutta, A review of process planning techniques in layered manufacturing, Rapid Prototyping Journal 6 (1) (2000) 18–35.

[4] 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.
[5] S. Keating, N. Oxman, Compound fabrication: A multi-functional robotic platform for digital design and fabrication, Robotics and Computer-Integrated Manufacturing 29 (6) (2013) 439–448.
[6] Y. Pan, C. Zhou, Y. Chen, J. Partanen, Multitool and multi-axis computer numerically controlled accumulation for fabricating conformal features on curved surfaces, Journal of Manufacturing Science and Engineering 136 (3) (2014) 031007.
[7] X. Song, Y. Pan, Y. Chen, Development of a low-cost parallel kinematic machine for multidirectional additive manufacturing, Journal of Manufacturing Science and Engineering 137 (2) (2015) 021005.
[8] W. Gao, Y. Zhang, D. C. Nazzetta, K. Ramani, R. J. Cipra, Revomaker: Enabling multi-directional and functionally-embedded 3d printing using a rotational cuboidal platform, in: Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology, ACM, 2015, pp. 437–446.
[9] I. Gibson, D. Rosen, B. Stucker, Additive manufacturing technologies: 3D printing, rapid prototyping, and direct digital manufacturing, Springer, 2014.
[10] 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.
[11] B. Huang, S. B. Singamneni, R. I. Campbell, I. Gibson, Curved layer adaptive slicing (clas) for fused deposition modelling, Rapid Prototyping Journal 21 (4).
[12] X. Qu, B. Stucker, A 3d surface offset method for stl-format models, Rapid Prototyping Journal 9 (3) (2003) 133–141.
[13] R. J. 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.
[14] J. D. Davis, M. D. Kutzer, G. S. Chirikjian, Algorithms for multilayer conformal additive manufacturing, Journal of Computing and Information Science in Engineering 16 (2) (2016) 021003.
[15] S. Mueller, S. Im, S. Gurevich, A. Teibrich, L. Pfisterer, F. Guimbretière, P. Baudisch, Wireprint: 3d printed previews for fast prototyping, in: Proceedings of the 27th annual ACM symposium on User interface software and technology, ACM, 2014, pp. 273–280.
[16] D. Bourell, B. Stucker, Y. Chen, C. Zhou, J. Lao, A layerless additive manufacturing process based on cnc accumulation, Rapid Prototyping Journal 17 (3) (2011) 218–227.
[17] G. Elber, E. Cohen, Adaptive isocurve-based rendering for freeform surfaces, ACM Transactions on Graphics (TOG) 15 (3) (1996) 249–263.
[18] G. Elber, E. Cohen, Tool path generation for freeform surface models, in: Proceedings on the second ACM symposium on Solid modeling and applications, ACM, 1993, pp. 419–428.
[19] J. Vanek, J. A. Galicia, B. Benes, Clever support: Efficient support structure generation for digital fabrication, in: Computer Graphics Forum, Vol. 33, Wiley Online Library, 2014, pp. 117–125.
[20] G. Elber, T. Grandine, Hausdorff and minimal distances between parametric freeforms in R^2 and R^3, in: International Conference on Geometric Modeling and Processing, Springer, 2008, pp. 191–204.
[21] F. Massarwi, G. Elber, A b-spline based framework for volumetric object modeling, Computer-Aided Design 78 (2016) 36 – 47, {SPM} 2016.
[22] T. H. Cormen, Introduction to algorithms, MIT press, 2009.
[23] M. De Berg, A. Khosravi, Optimal binary space partitions for segments in the plane, International Journal of Computational Geometry & Applications 22 (03) (2012) 187–205.
[24] P. Alexander, S. Allen, D. Dutta, Part orientation and build cost determination in layered manufacturing, Computer-Aided Design 30 (5) (1998) 343–356.
[25] M. P. Do Carmo, G. Fischer, U. Pinkall, H. Reckziegel, Differential geometry, in: Mathematical Models, Springer, 2017, pp. 155–180.

Appendix A. Arc-Length Limit of isoparametric Curves in Trivariates

Lemma Appendix A.1. *Let $p_1 = C(0)$ and $p_2 = C(1)$ be the endpoints of a 3-space parametric curve $C(t)$, $t \in [0,1]$, and let L be the arc-length of C. If $L \le 2\epsilon$ and $p_c \in C$ then*

$$\min\left(\left\|p_1 - p_c\right\|, \left\|p_2 - p_c\right\|\right) \le \epsilon\,.$$

Proof. L is the arc-length of a curve from p_1 to p_2 that passes through p_c, so we know that:

$$\left\|p_1 - p_c\right\| + \left\|p_2 - p_c\right\| \le L \le 2\epsilon\,.$$

Then, $\min\left(\left\|p_1 - p_c\right\|, \left\|p_2 - p_c\right\|\right) \le \epsilon$. □

Consider a Bézier, or a B-spline, trivariate:

$$V(u,v,w) = \sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\sum_{k=0}^{n_w} P_{ijk}B_i(u)B_j(v)B_k(w),\ u,v,w \in [0,1]\,.$$

Given i, j, consider one complete control polygon of a w-isoparametric curve from the control mesh of V, $\mathcal{P}_{i,j} = \{P_{ijk} \mid k \in 0 \dots n_w\}$. Then, let $L^k_{i,j}$ be the length of the kth segment in control polygon $\mathcal{P}_{i,j}$, $L^k_{i,j} = \|P_{i,j,k+1} - P_{ijk}\|$.

Lemma Appendix A.2. *Let $S_1(u,v,0)$ and $S_2(u,v,1)$ be the two w boundary surfaces in a trivariate $V(u,v,w)$. The arc-length of any w-isoparametric curve $C(w)$ in V from S_1 to S_2 is bounded by $\sum_{k=0}^{n_w-1}\max_{i,j}(L^k_{i,j})$.*

Proof. The arc-length of a Bézier or a B-spline curve can not exceed the arc-length of its control polygon. For constant values of $u = u_0$ and $v = v_0$, $V(u_0, v_0, w)$ is a parametric (univariate) curve:

$$C(w) = V(u_0, v_0, w) = \sum_{k=0}^{n_w}\sigma_k B_k(w),$$

where $\sigma_k = \sum_{i=0}^{n_u}\sum_{j=0}^{n_v} P_{i,j,k}B_i(u_0)B_j(v_0)$.

Then, the length of the kth segment of the control polygon of $C(w)$, L^k:

$$\begin{aligned}
L^k &= \|\sigma_{k+1} - \sigma_k\| \\
&= \left\|\sum_{i=0}^{n_u}\sum_{j=0}^{n_v} P_{i,j,k+1}B_i(u_0)B_j(v_0) - \sum_{i=0}^{n_u}\sum_{j=0}^{n_v} P_{i,j,k}B_i(u_0)B_j(v_0)\right\| \\
&= \left\|\sum_{i=0}^{n_u}\sum_{j=0}^{n_v}(P_{i,j,k+1} - P_{i,j,k})B_i(u_0)B_j(v_0)\right\| \\
&\le \sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\left\|P_{i,j,k+1} - P_{i,j,k}\right\| B_i(u_0)B_j(v_0) \\
&\le \sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\max_{i,j}\left\|(P_{i,j,k+1} - P_{i,j,k})\right\| B_i(u_0)B_j(v_0) \\
&= \max_{i,j}\left\|(P_{i,j,k+1} - P_{i,j,k})\right\| \\
&= \max_{i,j}(L^k_{i,j})\,.
\end{aligned}$$

The bound on L^k also establishes a bound on the arc-length of any isoparametric curve $C(w)$ from S_1 to S_2:

$$ArcLength(C(w)) \leq \sum_{k=0}^{n_w-1} L^k \leq \sum_{k=0}^{n_w-1} \max_{i,j}(L^k_{i,j}).$$

□

Lemma Appendix A.2 establishes that we can efficiently find an upper bound on the maximum length of any w-isoparametric curve between $S_1(u,v) = V(u,v,0)$ and $S_2(u,v) = V(u,v,1)$ using the control mesh of V. Assume we calculate that bound and find that all w-isoparametric curves are at most of length $L \leq 2\epsilon$. Then, any interior point $p \in V(u,v,w)$ ($w \in [0,1]$), is on some w-isoparametric curve, $p \in C(w) = V(u_0, v_0, w)$, and, following Lemma Appendix A.1 is at most ϵ distance away from either S_1 or S_2. By ensuring that any $p \in V$ is at most ϵ distance away from either S_1 or S_2, we ensure that the volume V is covered by surfaces S_1 and S_2.