Eurographics Symposium on Geometry Processing (2004)
R. Scopigno, D. Zorin (Editors)

# Isotopic Approximation of Implicit Curves and Surfaces

Simon Plantinga and Gert Vegter

Institute for Mathematics and Computing Science
University of Groningen
simon@cs.rug.nl gert@cs.rug.nl

**Abstract**

*Implicit surfaces are defined as the zero set of a function $F\colon \mathbb{R}^3 \to \mathbb{R}$. Although several algorithms exist for generating piecewise linear approximations, most of them are based on a user-defined stepsize or bounds to indicate the precision, and therefore cannot guarantee topological correctness. Interval arithmetic provides a mechanism to determine global properties of the implicit function. In this paper we present an algorithm that uses these properties to generate a piecewise linear approximation of implicit curves and surfaces, that is isotopic to the curve or surface itself. The algorithm is simple and fast, and is among the first to guarantee isotopy for implicit surface meshing.*

## 1. Introduction

Implicit functions provide a convenient representation of smooth surfaces. However, piecewise linear approximations are often required, e.g. for visualization. Meshing algorithms can only compute function values at a finite number of points. Since grid based schemes can miss important detail of the surface, correct topology usually cannot be guaranteed. To capture the global properties of implicit surfaces, we somehow need to extract information about the surrounding of these points. Lipschitz conditions give a bound on the gradient and can therefore sometimes discard the neighbourhood of a point. Another tool that can be used is interval arithmetic.

In this paper we present an algorithm that creates a regularly isotopic approximation for implicit curves in $\mathbb{R}^2$ and for implicit surfaces in $\mathbb{R}^3$. Regularly isotopic means that the approximation is equivalent to the curve or surface under continuous deformation within the embedding space. In particular, the approximation has the same topology as the implicit manifold. Correct topology is important to determine connectedness, and for example in constructing an initial mesh for time-dependent surfaces (e.g. morphing). Our implicit surface meshing algorithm subdivides space using an octree. Interval arithmetic is used to decide which cells of the octree need to be subdivided further. After the subdivision, for each leaf of the tree a local approximation is constructed.

The algorithm given in this paper falls in the category of enumeration methods. A novelty is that we use a fast and simple interval test to decide which cells need subdivision, in such a way that we can guarantee isotopy. Compared to other enumeration methods it adds little overhead, and is therefore quite fast. In particular, we do not need to compute the critical points of the implicit function.

The octree based subdivision is very flexible. For example we could define a minimal subdivision level to improve the Hausdorff distance between the surface and the approximation. A maximal subdivision level could be used to improve the speed of the meshing process and to handle surfaces with singularities. In this case colour coding of the mesh could be used to identify the (arbitrarily small) regions where the approximation might be topologically incorrect.

In section 2 we give an overview of the existing techniques for implicit curve and surface approximation. Since our algorithm is based on interval arithmetic, in section 3 we give a brief overview of this technique. Section 4 explains our approximation algorithm for implicit curves. After giving the algorithm, we will prove that the resulting piecewise linear approximation is isotopic to the curve itself. Since both the algorithm and its proof will be generalized to the three-dimensional case, this section also provides an introduction to the meshing of implicit surfaces. In section 5 we provide a first step towards our meshing algorithm, by adapting the well-known marching cubes algorithm to con-

struct isotopic approximations. Section 6 extends the previous section by using octrees instead of a regular grid, thereby greatly reducing the complexity of both the algorithm and the resulting mesh. Finally, in section 8 we discuss some possible improvements and changes to our approximation scheme.

**Main contribution** This paper presents a new algorithm to approximate not necessarily algebraic regular implicit curves and surfaces. For surfaces, it is one of the first schemes guaranteeing that the resulting mesh is isotopic to the implicit surface. For curves, it is the first practical algorithm giving this guarantee. Since we do not need to compute the critical points of the implicit function, both algorithms are fast enough to be of practical use. Also, they can easily be adapted to improve the accuracy of the approximation and to deal with singular surfaces. We expect that the algorithm generalizes to isotopic approximation of all codimension one manifolds in Euclidean spaces.

## 2. Related work

Several algorithms exist for the approximation of implicit curves and surfaces, but very few can guarantee topological correctness. Since we can only compute function values at a finite number of points, there is a risk of missing some important detail of the curve or surface. Some approximation algorithms depend on a user specified value to give a tradeoff between accuracy and speed. Algorithms that do guarantee a topologically correct result depend either on extra information about the underlying implicit function (e.g. algebraicity of surfaces or bounds on the Lipschitz conditions), or on interval arithmetic.

In this section we will give an overview of the different existing techniques for approximation of implicit curves and surfaces.

**Curves** The approximation schemes for approximation of implicit curves can be roughly divided in continuation methods and adaptive enumeration techniques.

Continuation methods approximate the curve or surface by first finding seed points on all components, and then using these as starting points for tracing these components.

In [BH98] an implicit curve (a contour in $\mathbb{R}^3$) is approximated using a predictor-corrector method. Initial points are found by shooting random rays to the surface. In [PV03] we adapted this method to guarantee topological correctness of an approximation of the contour generator, but to achieve this the relatively slow interval Newton method was required. Also, the guarantee requires a small step size for the tracing process.

Approximation schemes based on adaptive enumeration start with a bounding box and subdivide until the cells are 'small enough', for example using the local curvature. Then, for each cell of the subdivided box a local approximation is constructed. In [LOdF02] interval arithmetic is used to get a good approximation of implicit curves. The size of the subdivided cells depends on the geometry of the curve. However, to terminate the subdivision process, the method requires user-specified bounds on cell size, and on the variation of the normalized gradient within a cell. Although the edge length varies with the curvature of the curve, it cannot guarantee topology.

As in this paper, [Sny92] uses parametrizability to subdivide a quadtree. This algorithm guarantees topological correctness, but because of a slightly weaker test for parametrizability, it requires a large number of (slow) interval Newton searches for intersection points.

**Surfaces** A continuation algorithm is given in [Har98]. Starting with a seed point on the surface, a triangulation is generated by expanding over the surface, along the boundary of the triangulated part. As with curves, this algorithm depends on seed points and on a user-specified step size. The triangles of the resulting mesh have more or less the same size, independent of the geometry of the surface. An improvement of this algorithm using a dynamic triangle size is given in [AG01].

Enumeration methods use either a regular grid or an octree to subdivide space. Regular grids are used in the well-know marching cubes algorithm [LC87]. The main purpose of these methods is to mesh sampled voxel data, but they can easily be adapted to approximate implicit surfaces. In [Blo88], cubes are constructed along the surface. Although this saves examining a large number of grid cells, there is a risk of missing components of the implicit surface. Another improvement of marching cubes is given in [Che95]. The linear variation along an edge used in marching cubes, is extended to trilinear variation over the cube, using the function values at the eight corners of the cube. The algorithm guarantees that the approximation has the same topology as the isosurface of this trilinear function. For implicit surfaces there is no such guarantee.

For dynamic surfaces, critical points indicate when, where and how the topology changes. This can be used to update the topology of the triangulation, as in [SH97]. Also, the shrinkwrap algorithm [vOW93] can be adapted to use critical points for updating the topology [BNvO96].

An algorithm for isotopic meshing based on critical points is given in [BCSV04]. It is based on Morse theory to determine the topology, and therefore requires a priori knowledge of the critical points of the implicit surface.

Finally, sampling based algorithms construct a sufficiently dense sample of points on the surface, such that surface reconstruction on this point set results in a homeomorphic approximation. To construct such a sampling, in [BO04] bounds on the distance to the medial axis are needed, while in [CDRT04] the critical points of heightfunctions on the

surface and on intersection curves are required. Also, both algorithms assume that the intersection points of a given line segment with the surface can be computed. These Delaunay based algorithms give a better mesh quality, but for generic implicit surfaces it is difficult to meet these assumptions.

## 3. Interval arithmetic

One way to prevent rounding errors due to finite precision numbers is to use interval arithmetic. These interval can be considered as a value, together with an error bound.

An *inclusion function* $\square f$ for a function $f\colon \mathbb{R}^m \to \mathbb{R}^n$ computes for each $m$-dimensional interval $I$ (i.e. an $m$-box) an $n$-dimensional interval $\square f(I)$ such that

$$x \in I \quad \Rightarrow \quad f(x) \in \square f(I)$$

An inclusion function is *convergent* if the width of the output interval converges to 0 when the (largest) width of the input interval shrinks to 0.

For example, if $f\colon \mathbb{R} \to \mathbb{R}$ is the squaring function $f(x) = x^2$, then a convergent inclusion function $\square f([a,b])$ is given by

$$\begin{cases} [\min(a^2,b^2), \max(a^2,b^2)], & a \cdot b \geq 0 \\ [0, \max(a^2,b^2)], & a \cdot b < 0 \end{cases}$$

When using interval arithmetic to prevent rounding errors, the width of the intervals should be as small as possible in order to give accurate results. We will use interval arithmetic in a different way, and compute function values over large intervals. For example, if the function value computed over a large box results in a strictly positive interval, we can conclude that the function has no zeroes within that interval. In other words, the implicit manifold does not intersect this box.

Convergent inclusion functions exist for the basic operators and functions. To compute an inclusion function it is often sufficient to replace the standard number type (e.g. double) by an interval type, using an appropriate interval library, such as `Filib++` [LTvG*].

## 4. Curves

In this section we will introduce an algorithm that constructs a piecewise linear approximation of an implicit curve $S$, such that the approximation and $S$ have the same (regular) isotopy. The construction is based on a quadtree. After subdivision this quadtree is *balanced*, i.e. the tree is further subdivided until two neighbouring squares differ at most a factor two in size.

Possible improvements regarding accuracy and singularities will be discussed in section 8.

Let $S = F^{-1}(0)$ be a bounded implicit curve, where $F\colon \mathbb{R}^2 \to \mathbb{R}$ is a smooth function and 0 is a regular value of $F$, i.e. the gradient $\nabla F$ is non-zero at every point of the curve. The following algorithm constructs an isotopic piecewise linear approximation of $S$. Details are explained right after the presentation of the algorithm.

**Algorithm APPROXIMATECURVE($F, B$)**
*Input.* An implicit function $F$, and a square bounding box $B$.
*Output.* A piecewise linear approximation of the curve $F = 0$.

1. Initialize a quadtree $\mathcal{T}$ to the bounding square $B$ of $F = 0$.
2. Subdivide $\mathcal{T}$ until for all leaves $C$ we have $0 \notin \square F(C) \vee \langle \square \nabla F(C), \square \nabla F(C) \rangle > 0$.
3. BALANCEQUADTREE($\mathcal{T}$)
4. **for** each edge of the quadtree
5. **do if** the signs of $F$ at its two endpoints are opposite
6. **then** construct a vertex at the midpoint of the edge
7. **for** each leaf $C$ of quadtree $\mathcal{T}$
8. **do if** the leaf contains two vertices
9. **then** connect the vertices by a line segment
10. **if** the leaf contains four vertices
11. **then** find the two vertices on the same side and connect each of them to one of the other vertices

Note that a single side of a cell may consist of two quadtree edges, if the neighbouring cell along that side is subdivided. See Figures 14 and 15. Each cell of the quadtree is therefore surrounded by four to eight quadtree edges.

The construction of the approximating edges handles only a few different cases of vertex placement on the boundary of a cell: two vertices, or four vertices with two of them along one side of the cell. In the correctness proof we will show that these are the only possible cases.

Before proving that this algorithm constructs an isotopic approximation, we will show that the resulting quadtree cells are parametrizable and prove that the algorithm terminates by examining the interval condition:

$$0 \notin \square F(C) \quad \vee \quad \langle \square \nabla F(C), \square \nabla F(C) \rangle > 0.$$

The first clause discards cells where $0 \notin \square F(C)$. Note that this discards boxes of which we are certain that they do not contain part of the curve $S$.

The righthand clause implies that $\langle \nabla F(x), \nabla F(y) \rangle > 0$, for all $x, y \in C$. Hence, the direction of the gradient (and, therefore, of the curve) does not change more than $\pi/2$ over $C$.

The curve $S$ is *parametrizable in direction* $v$, if every line parallel to $v$ intersects $S$ at most once.

If $\langle \square \nabla F(C), \square \nabla F(C) \rangle > 0$, at least one of the two terms $\square F_x(C) \cdot \square F_x(C)$ and $\square F_y(C) \cdot \square F_y(C)$ (where we write $F_x$ for $\frac{\partial F}{\partial x}$) does not contain 0. This implies that $F$ is strictly increasing or decreasing in the $x$ or $y$ direction, and therefore

locally (i.e. within this cell) parametrizable in the direction of one of the axes.

When we continue the subdivision process, the squares shrink towards a point. Since $S$ is a regular curve, at least one of the two clauses converges to a non-zero value. Therefore, the subdivision process terminates.

After these observations we can move on to the following theorem:

**Theorem 4.1** The approximation of $S$ constructed by algorithm APPROXIMATECURVE is (regularly) isotopic to $S$.

*Proof* To prove that the approximation is isotopic to the implicit curve $C$ we will proceed as follows:

- Firstly, we construct an approximation using a regular grid, assuming that $S$ satisfies certain constraints, and show that this approximation is isotopic to $S$.
- Secondly, we will remove the contraints on $S$.
- Finally, we will show that the approximation constructed by the regular grid is equivalent to the one created by APPROXIMATECURVE. To this end, we will show that further subdivision of the quadtree does not change the isotopy of the corresponding approximation. By repeating the subdivision process until the quadtree is complete, we end up with a regular grid. This grid was already shown to be isotopic to the implicit curve.

**Regular grid** For the first part of the proof we will start with a grid-based approximation instead of a quadtree. Let $G$ be a regular grid, such that for each cell $C$ we have $0 \notin \Box F(C) \vee \langle \Box \nabla F(C), \Box \nabla F(C) \rangle > 0$. Furthermore we assume that $F \neq 0$ at the nodes of $G$, and that $S$ intersects each edge of $G$ at most once. This implies that we have to construct at most one vertex at each edge of the grid. Due to the inner product constraint, $S$ is parametrizable in the direction of one of the axes. Therefore we cannot have alternating signs of $F$ at the vertices of $C$, since $F$ would have to increase along one edge, and decrease along the other parallel edge (Figure 1). We conclude that $S$ intersects at most two edges of $C$. For the approximation we connect the two midpoints of these two edges (if any) by a straight line segment $s$.

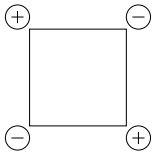


**Figure 1:** *A sign configuration contradicting the inner product constraint. F decreases along the top edge and increases along the bottom edge. It is therefore not parametrizable in the x-direction. A similar argument holds for the y-direction.*

The curve $S$ is locally parametrizable, and therefore cannot contain closed loops inside a cell. Also, since $S$ is a regular curve, there are no self-intersections. This implies that within the cell $C$, if there are two intersection points along the edges, the part of the curve inside $C$ is isotopic to a line segment.

In fact we can easily construct a deformation to move the curve locally to the segment $s$ of the approximation. Suppose for example that $S$ is locally parametrizable in the $y$-direction, i.e. within the grid cell it is of the form $y = f(x)$. If $S$ intersects the left and right edge, the approximation has the same $x$-domain as $S$. By linear interpolation in the $y$-direction we can continuously move this part of $S$ to segment $s$. If $S$ does not intersect the left and right edge, we may need to stretch/shrink the $x$-domain first. An example is given in Figure 2. By mapping $[0,t]$ linearly to $[0,\frac{1}{2}]$ and $[t,1]$ to $[\frac{1}{2},1]$ the intersection point $(t,0)$ moves to $(\frac{1}{2},0)$. Now, both curves have the same $x$-domain, so we can do a vertical interpolation again. Note that an edge of the grid maps onto itself. Therefore, we can stitch the local deformations within each square together, resulting in a global deformation that moves the whole implicit curve to the approximation.

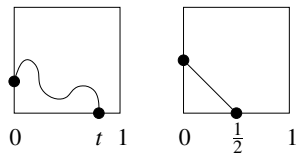


**Figure 2:** *Deformation of the curve (left) to the approximation (right).*

**Removing constraints** In the second part of the proof we remove the constraints on $S$. Suppose $S$ intersects an edge of the grid more than once. In this case the curve must be parametrizable in the direction perpendicular to this edge, for both adjacent cells. For two adjacent intersection points $\alpha_1$ and $\alpha_2$ we look at the curve between these two points. Using the mean value theorem it is easy to see that the inner product constraint prevents $S$ from bending 'too much' (i.e. more than $\pi/2$). Also, recall that the adjacent cells are squares. Therefore, $S$ cannot leave the two cells between $\alpha_1$ and $\alpha_2$ (Figure 3). By a local interpolation in the parametrizable direction between $S$ and the edge, we can continuously deform the part of $S$ between $\alpha_1$ and $\alpha_2$ towards the edge, and 'push' this part of $S$ through the edge, thereby removing the two intersection points. Since we can continue removing pairs of intersection points, $S$ is isotopic to a curve that intersects each edge at most once, and hence it is isotopic to the piecewise linear approximation.

If $S$ passes through a node of the grid, we can again deform $S$, this time by moving it continuously to $F + \varepsilon = 0$. For small $\varepsilon$ this again yields an isotopy. We can take $\varepsilon$ arbitrarily small using a symbolic perturbation, by considering $F$ to be strictly positive at a vertex whenever $F \geq 0$.

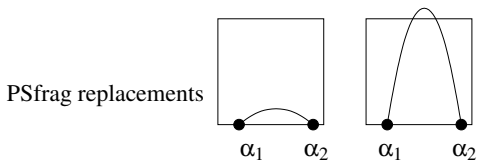


**Figure 3:** *Multiple intersection along an edge. The curve on the right has too much curvature to satisfy the interval constraint.*

**Quadtrees** Now that we have removed the constraints on $F$ we complete the proof by showing that the quadtree approximation is isotopic to the grid approximation. To this end, we consider a leaf $C$ of the quadtree $\mathcal{T}$, together with its approximation edges. When we split $C$, the inner product constraint still holds for its four children. Furthermore, since $S$ is parametrizable within $C$, $S$ is parametrizable in its children, and also in the same direction. The only topological changes can therefore occur if subdivision introduces two new vertices at a single edge of $C$, as shown in Figure 4.

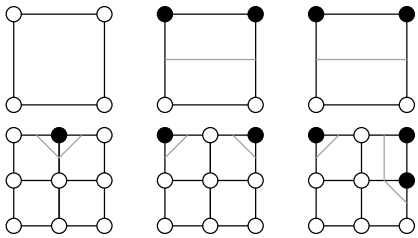

**Figure 4:** *Topological changes after subdivision*

Since two new vertices were created, the neighbouring cell along the edge containing these vertices is not subdivided. After subdivision of $C$, the neighbouring cell will detect the two new vertices and connect them either to each other or to two existing vertices. Both types of change correspond to 'pushing' part of the curve through the edge. In the union of the cell and its neighbour the approximation is isotopically unaltered.

By subdividing the leaves of $\mathcal{T}$ we can turn it into a complete quadtree, with the same approximation as a regular grid. Since this does not change the isotopy of the approximation, the output of algorithm APPROXIMATECURVE is isotopic to $S$. □

## 5. Marching cubes revisited

For the meshing of implicit surfaces we proceed similarly to the two-dimensional case. As a first step towards our meshing algorithm, we approximate the surface on a regular grid, assuming only single intersections on the edges. Later, we remove these constraints. In the next section we will present our meshing algorithm, using a balanced octree instead of a regular grid.

In this section we introduce a slightly modified version of the well-known marching cubes ($\mathcal{MC}$) algorithm. Recall that $\mathcal{MC}$ subdivides space into a uniform cubic grid, constructing a triangulation for each cell of the grid, depending only on the function sign at its eight vertices. (Sometimes the function value at the vertices is used to place a vertex by linear interpolation. Since our main goal is topological correctness we will just put the vertex at the centre of the edge.) The $\mathcal{MC}$ algorithm as introduced by Lorensen and Cline [LC87] cannot guarantee topological correctness. Due to ambiguities in the sign configuration of a cell face, the resulting surface could also contain holes. To fix these problems we will introduce a few assumptions, most of which can later be removed.

Let $S$ be a regular, bounded, implicit surface, given as the zero set of a smooth function $F\colon \mathbb{R}^3 \to \mathbb{R}$, so $S = F^{-1}(0)$. Furthermore, we assume that 0 is a regular value of $F$, i.e., the gradient $\nabla F$ is non-zero at every point of the surface.

For now, we will assume that $S$ does not contain vertices of the uniform grid, and that $S$ intersects each edge of the grid at most once. Furthermore, the following interval condition should hold for each cell $C$:

$$0 \notin \square F(C) \quad \vee \quad 0 \notin \langle \square \nabla F(C), \square \nabla F(C) \rangle.$$

As in the 2-dimensional case, the righthand clause implies:

$$\forall x, y \in C\colon \langle \nabla F(x), \nabla F(y) \rangle > 0.$$

Again we have that for each cell where $0 \in \square F(C)$ the surface $S$ is parametrizable in the direction of one of the axes.

For the signs at the vertices, there are 14 possibilities (up to rotation, mirroring and change of sign), shown in Figure 5.

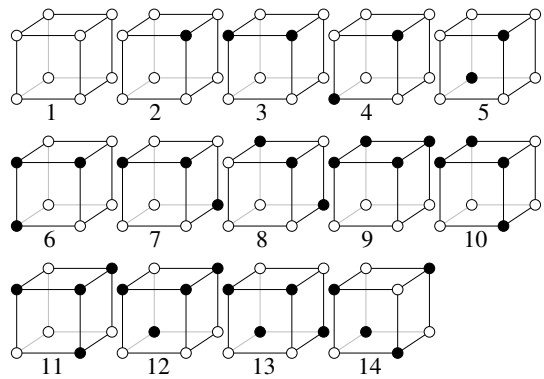


**Figure 5:** *The 14 possible sign configurations.*

Configurations 5, 8, 12, 13 and 14 are impossible, due to the interval condition. For example in configuration 5 (see

Figure 6), on edges $\overline{cd}$ and $\overline{ef}$ the function changes sign in opposite directions. Therefore, $F$ is not parametrizable in this direction. For $\overline{ad}$, $\overline{fg}$ and for $\overline{bf}$, $\overline{dh}$ we find that $F$ is not parametrizable in the direction of the other two axes, contradicting the inner product condition.

For configurations 8, 12 and 14 a similar argument holds. For configuration 13 we have to look at the diagonal pairs $(\overline{ac},\overline{eg})$, $(\overline{bd},\overline{fh})$ and $(\overline{ae},\overline{cg})$, and note that the inner product condition does not depend on a particular orthogonal coordinate system.

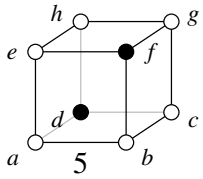


**Figure 6:** *Configuration 5.*

The remaining 9 configurations can now be triangulated. Two of these still contain an ambiguity (case 4 and 6 in Figure 7). To resolve this, note that the ambiguity is due to the sign configuration on a face of the cell. In both cases we have a single face where the function changes sign along all of its four edges, resulting in four vertices of the triangulation. There are two ways to connect these vertices pairwise. We will choose to connect them, such that the resulting segments are parallel to one of the vectors $(1,1,0)$, $(1,0,1)$ and $(0,1,1)$, depending on the orientation of the face. This choice guarantees that the individual triangulations of two adjacent cells fit together. We have to show that this choice does not affect the isotopy.

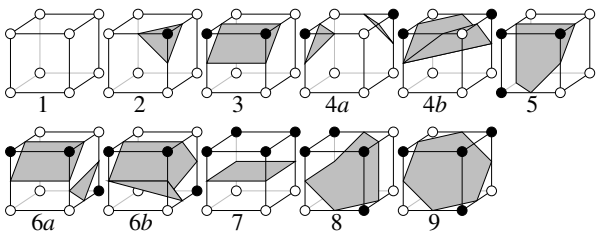


**Figure 7:** *Triangulation of the subcubes.*

Since cells have at most one ambiguous face, we regard the block of two cubes adjacent to such a face (see Figure 8). Note that the border of the union of these two cubes does not contain ambiguities. The sign changes along the edges of the shared face prevent a parametrization in the direction of one of these edges. Both cubes are therefore parametrizable in the vertical direction. Since the vertical component of the gradient does not disappear, the union of these two cells is also parametrizable in the vertical direction. This leads to only three possibilities to join two ambiguous cells, shown in Figure 8.

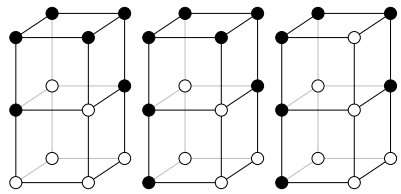

**Figure 8:** *Two cubes sharing an ambiguous face: two type 4 cells, a type 4 and a type 6 cell, and two type 6 cells.*

For the ambiguous face we have two ways to connect the vertices on the edges pairwise. Both possibilities are shown in Figure 9. Since both choices lead to the same isotopic approximation, we can choose either of them, as long as the triangulations for both cells fit together.

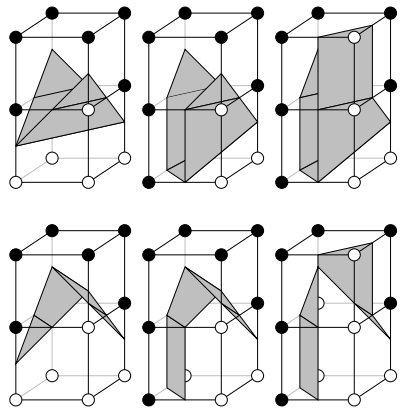

**Figure 9:** *The two different triangulations of the three ambiguous cell combinations.*

If the surface intersects the grid-faces in segments connecting the intersections of $S$ with the edges of the grid, we are done. However, it is possible that $S$ intersects a face of the grid in a closed curve, lying completely withing that face (Figure 10). In this case both cubes adjacent to this face must be parametrizable in the direction perpendicular to this face. Also, the 'bubble' does not leave the cube on the opposite face, since that would require the gradient to change more than $\pi/2$ (see also Figure 3). By linear interpolation we can continuously flatten the bubble towards the face, and push it through this face, thereby removing the intersection loop.

As in the two-dimensional case we have to remove the constraints on the function. The case where the surface passes through vertices of the grid can be handled as with implicit curves, i.e. by considering $F$ to be strictly positive whenever $F \geq 0$.

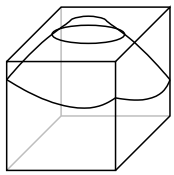

**Figure 10:** *Surface S intersects a face of the regular grid in a closed loop.*

Now assume that $S$ intersects an edge $e$ of the grid more than once, say at two consecutive points $\alpha$ and $\beta$ (Figure 11). Since $F$ changes sign at $\alpha$ and at $\beta$, there is a point $p$ between $\alpha$ and $\beta$ where $\nabla F(p)$ is perpendicular to $e$. This gradient lies inside the gradient cone of all four cubes adjacent to $e$. (Due to the interval constraint all gradients over a cell lie within a cone with top angle $\pi/2$.) Therefore, in the union of these four cubes, the surface is parametrizable in this direction. Consider the plane through $e$ in the direction of $\nabla F(p)$. Each of the two half-planes bounded by $e$ intersect one of the cubes. Since $\nabla F(p)$ lies inside the cone of gradients of this cube, the projection of $\nabla F(x)$ does not change more than $\pi/2$ for $x$ in the intersection of this half-plane with the cube. Therefore in one of these two half-planes, the intersection with $S$ consists of a connected curve between $\alpha$ and $\beta$. By linear interpolation in direction $\nabla F(p)$ between this curve and $e$ we can remove the pair $\alpha, \beta$.

After removing all pairs of edge intersections we have continuously deformed the surface to one with only single edge intersections, for which the grid triangulation was shown to be correct.

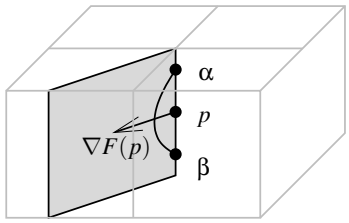


**Figure 11:** *Surface S intersects an edge of the regular grid more than once.*

We conclude that, the triangulation of the regular grid is isotopic to the implicit surface.

## 6. Octrees

In the previous section we have shown that a regular grid satisfying an interval constraint can be used to create an isotopic approximation of an implicit surface. Now we show how a balanced octree can be used to create an isotopic mesh. Since we can subdivide the octree approximation to a complete octree, we have isotopy between the surface and the octree approximation, similarly to the two-dimensional case.

The subdivision process is identical to the two-dimensional case. Starting with an octree initialized to a bounding box $B$, we subdivide the leaves until for each leaf $C$ we have

$$0 \notin \square F(C) \quad \vee \quad \langle \square \nabla F(C), \square \nabla F(C) \rangle > 0.$$

After the subdivision process, we balance the octree.

Once we have constructed the octree, we can construct the approximation. First we put vertices at the center of each edge of the octree that has opposite function signs at its endpoints. Then, for each face of the octree we connect these vertices pairwise with segments. For each cell of the octree, these segments form closed chains on the boundary of the cell. If we have two nested loops, we connect these two loops with triangles, resulting in an annulus (this case is shown in Figure 12). In all other cases, we close each loop with triangles, constructing a topological disk for each closed chain of edges.

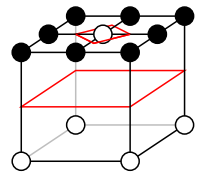

**Figure 12:** *The top face indicates the only case where two nested loops occur, resulting in an annulus. All other configurations are triangulated as a set of disks.*

To make sure that further subdivision does not change the isotopy, we have to be careful how we connect the vertices. The subdivision can push the approximation up and down in the parametrizable direction. We shall assume a vertical parametrizable direction. For the top and bottom faces, it does not matter how we connect the vertices. Since the parametrizable direction is perpendicular to these faces, changing the connections corresponds to moving a saddle point through the surface. In the union of the adjacent cubes the isotopy does not change (similar to the regular grid case in Figure 9).

For the faces of the octree along the side of a cell, the intersection of the surface with such a face is parametrizable, but the $\pi/2$ bound on the gradient restricted to the face does not hold. The intersection curve can therefore pierce through two opposite edges (cf. Figure 3). This prevents us from uniquely determining the topology of the intersection curve. However, a change in topology corresponds to moving the surface sideways through such a side face. Although

the intersection curve changes, the topology of the surface itself is not affected.

To construct an isotopic approximation, we examine the newly created vertices on the boundary of an octree face. If the vertex configuration could correspond to a parametrizable intersection curve, we connect the vertices such that the piecewise linear approximation is parametrizable as well. For other vertex configurations we connect the vertices pairwise, starting at an arbitrary vertex. Some examples are shown in Figure 13.

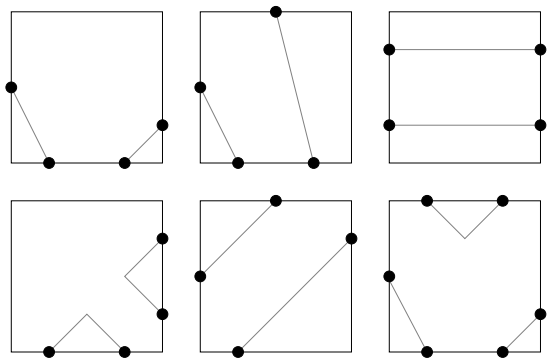

**Figure 13:** *Examples of how to connect the vertices along the faces of the octree. The top row corresponds to potentially parametrizable intersections. Note that for example the middle figure could be parametrizable, since the vertices do not represent the exact intersection point. The bottom row is not parametrizable and can be connected arbitrarily.*

## 7. Results

Figure 14 and 15 show two examples of the curve approximation algorithm. The quadtree is also shown in the images. Both examples took 0.04 seconds to compute.

In Figures 16, 17, 18 and 19, four examples of implicit surface approximation are shown. The algorithm was implemented in `gcc`. The table below shows the number of leaves in the octree before and after subdivision, the number of triangles in the mesh, and the running time in seconds on a Pentium 667 MHz, running Linux.

| Surface | Octree | Balanced | Triangles | Time |
|---|---|---|---|---|
| Tangle | 24648 | 24816 | 8704 | 0.8 |
| Chair | 232072 | 233402 | 43014 | 6.2 |
| Bear | 497596 | 688794 | 366746 | 113.0 |
| Non-alg | 29275 | 40293 | 24612 | 2.0 |

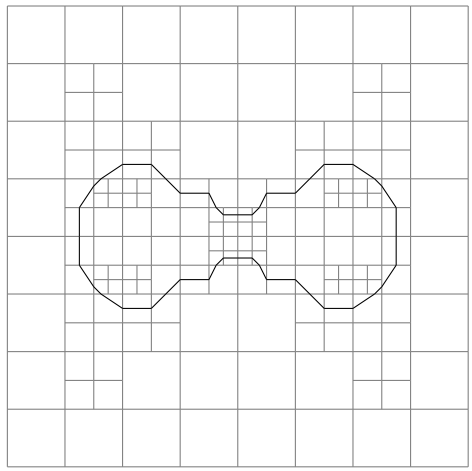

**Figure 14:** *Implicit curve approximation of* $f(x,y) = x^2(1-x)(1+x) - y^2 + 0.01$

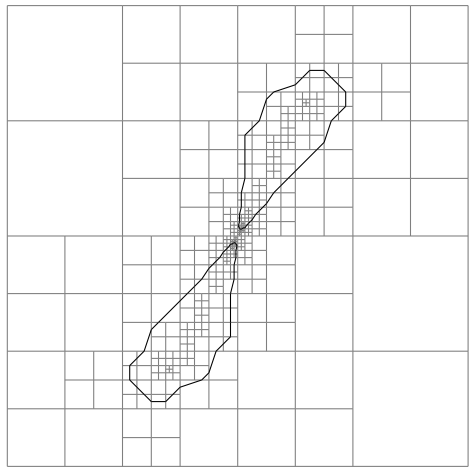

**Figure 15:** *Implicit curve approximation of* $f(x,y) = x^2 - xy + y^4 + 0.0001$

## 8. Conclusion

We have presented a simple and fast algorithm for meshing implicit surfaces, that is among the first to guarantee isotopy between the mesh and the surface. The current implementation does not try to create a good approximation in terms of Hausdorff distance. An open problem remains how to improve the mesh quality without loosing isotopy. Other improvements to the algorithm are given below.

**Improvements** Both algorithms use balanced trees. This balancing is not necessary for an isotopic result. Although the approximation for the leaves becomes more complicated, removal of the balancing would produce a slightly smaller approximation.

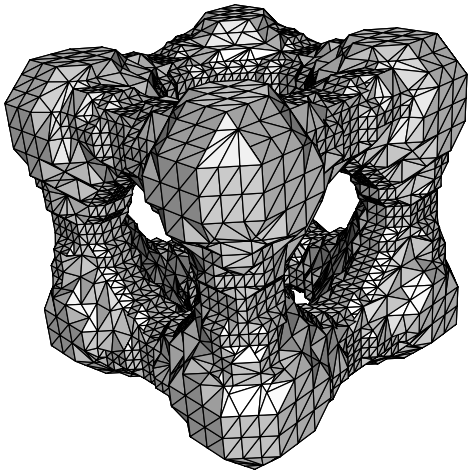

**Figure 16:** *Tangle cube* $f(x,y,z) = x^4 - 5x^2 + y^4 - 5y^2 + z^4 - 5z^2 + 10$

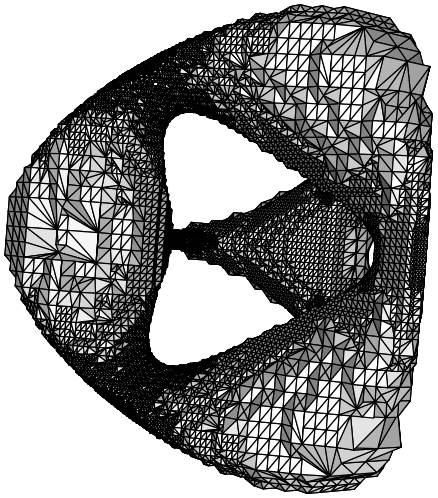

**Figure 17:** *Chair* $f(x,y,z) = \left(x^2+y^2+z^2-ak^2\right)^2 - b\left((z-k)^2-2x^2\right)\left((z+k)^2-2y^2\right)$, *for* $k = 5$, $a = 0.95$ *and* $b = 0.8$

The octree based approach allows for local updates if $S$ changes only locally, for example when using blobs or metaballs. For example, adding a single metaball does not change the implicit function outside the influence region of the metaball. Therefore, we only need to remesh the part of the octree intersecting the new metaball.

Although the resulting approximation is isotopic to $S$, it does not have to be close in terms of Hausdorff distance. To get a closer approximation the algorithm could be extended by subdividing cells where $0 \in \square F(C)$ to a given minimal level, before starting the subdivision process (recall that the surface does not leave the neighbouring cell).

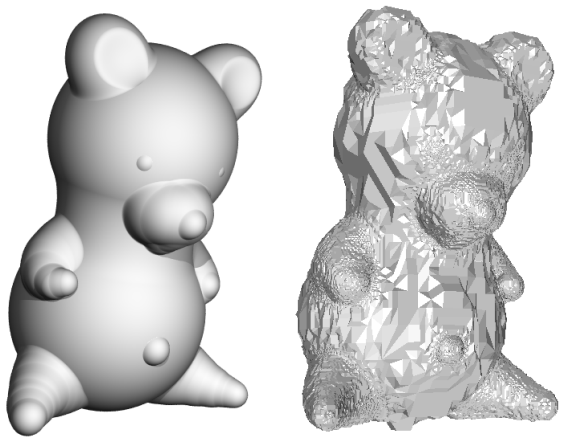

**Figure 18:** *An implicit teddy bear together with its approximation. The bear consists of 48 metaballs.*

For curves or surfaces containing singularities, the subdivision does not terminate. By using a maximum depth for the subdivision level we can mesh these curves and surfaces. Although the isotopy close to a singularity is not guaranteed, this results in an isotopic approximation outside arbitrarily small bounding boxes around the singular points. This can also be used to speed up the approximation, since the algorithm does not continue to refine the tree in difficult areas. Although isotopy is not guaranteed, we can identify the leaves where the interval condition holds, that is where the approximation is isotopically correct. The remaining (arbitrarily small) leaves where we are not yet sure of the topology could be identified, for example by triangulating these areas in a different colour. This way the user can not only easily identify where the problematic areas are, but also can decide to refine locally until the isotopy holds or the approximation is close enough. Also, other methods can be used to examine the behavior of the surface inside these cells, e.g. algebraic methods.

The algorithm can also be used for unbounded curves and surfaces. Recall that the undetected parts of the curves and surfaces cannot pierce through opposite edges or faces of a cell. By making the initial bounding box larger than the area of interest, and discarding the outer cells afterwards, we can construct an isotopic approximation for an unbounded manifold within a given box.

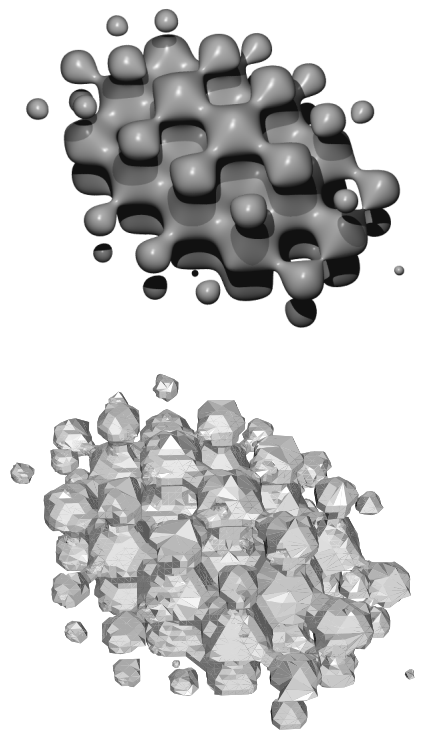

**Figure 19:** *The non-algebraic surface* $-0.4\,(\sin(5x)+\sin(5y)+\cos(5z)) + 0.1x^2 + 0.3y^2 + 0.2z^2 - 0.5$

## References


[AG01] AKKOUCHE S., GALIN E.: Adaptive implicit surface polygonization using marching triangles. In *Computer Graphics Forum*, Duke D., Scopigno R., (Eds.), vol. 20(2). Blackwell Publishing, 2001, pp. 67–80.

[BCSV04] BOISSONNAT J., COHEN-STEINER D., VEGTER G.: Isotopic implicit surface meshing. In *Proceedings Thirty-Sixth Annual ACM Symposium on Theory of Computing* (Chicago, 2004).

[BH98] BREMER D., HUGHES J. F.: Rapid approximate silhouette rendering of implicit surfaces. In *Proc. Implicit Surfaces* (1998).

[Blo88] BLOOMENTHAL J.: Polygonization of implicit surfaces. *Computer Aided Geometric Design 5* (1988), 341–355.

[BNvO96] BOTTINO A., NUIJ W., VAN OVERVELD K.: How to shrinkwrap through a critical point: an algorithm for the adaptive triangulation of isosurfaces with arbitrary topology. In *Proc. Implicit Surfaces* (1996), pp. 55–72.

[BO04] BOISSONNAT J., OUDOT S.: An effective condition for sampling surfaces with guarantees. In *Proceedings ACM Symposium on Solid Modeling* (2004).

[CDRT04] CHENG S., DEY T., RAMOS E., T.RAY: Sampling and meshing a surface with guaranteed topology and geometry. In *Proceedings Symposium on Computational Geometry* (2004).

[Che95] CHERNYAEV E. V.: *Marching cubes 33: Construction of topologically correct isosurfaces*. Tech. Rep. CERN CN 95–17, 1995.

[Har98] HARTMANN E.: A marching method for the triangulation of surfaces. *The Visual Computer 14*, 3 (1998), 95–108.

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (1987), ACM Press, pp. 163–169.

[LOdF02] LOPES H., OLIVEIRA J. B., DE FIGUEIREDO L. H.: Robust adaptive polygonal approximation of implicit curves. *Computers and Graphics 26*, 6 (2002), 841–852.

[LTvG*] LERCH M., TISCHLER G., VON GUDENBERG J. W., HOFSCHUSTER W., KRÄMER W.: Filib++ interval library. http://www.math.uni-wuppertal.de/wrswt/software/filib.html.

[PV03] PLANTINGA S., VEGTER G.: Contour generators of evolving implicit surfaces. In *Proceedings of the eighth ACM symposium on Solid modeling and applications* (2003), ACM Press, pp. 23–32.

[SH97] STANDER B. T., HART J. C.: Guaranteeing the topology of an implicit surface polygonization for interactive modeling. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (1997), ACM Press/Addison-Wesley Publishing Co., pp. 279–286.

[Sny92] SNYDER J. M.: Interval analysis for computer graphics. *Computer Graphics 26*, 2 (1992), 121–130.

[vOW93] VAN OVERVELD C., WYWILL B.: *Shrinkwrap: An adaptive algorithm for polygonizing an implicit surface*. Tech. Rep. 93/514/19, University of Calgary, Mar. 1993.