A Survey on Implicit Surface Polygonization

B. R. de ARAÚJO, University of Toronto
DANIEL S. LOPES, PAULINE JEPP, and JOAQUIM A. JORGE, INESC-ID Lisboa
BRIAN WYVILL, University of Victoria

Implicit surfaces (IS) are commonly used in image creation, modeling environments, modeling objects, and scientific data visualization. In this article, we present a survey of different techniques for fast visualization of IS. The main classes of visualization algorithms are identified along with the advantages of each in the context of the different types of IS commonly used in computer graphics. We focus closely on polygonization methods, as they are the most suited to fast visualization. Classification and comparison of existing approaches are presented using criteria extracted from current research. This enables the identification of the best strategies according to the number of specific requirements, such as speed, accuracy, quality, or stylization.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Implicit surface, shape modeling, polygonization, surface meshing, surface rendering

ACM Reference Format:
Bruno R. de Araújo, Daniel S. Lopes, Pauline Jepp, Joaquim A. Jorge, and Brian Wyvill. 2015. A survey on implicit surface polygonization. ACM Comput. Surv. 47, 4, Article 60 (May 2015), 39 pages.
DOI: http://dx.doi.org/10.1145/2732197

1. INTRODUCTION

Implicit surfaces are a popular 3D mathematical model used in computer graphics. They are used to represent shapes in modeling, animation, scientific simulation, and visualization [Gomes et al. 2009; Frey and George 2010]. Implicit representations can be extremely compact, requiring only a few high-level primitives to describe complex free-form volumes and surfaces [Velho et al. 2002]. They also present a solution of choice for visualizing scientific and medical data. In particular, they are well suited for representing data gathered from 3D scans (e.g., computed tomography (CT) and magnetic resonance imaging (MRI)) and digitizing complex models.

The work described in this article was partially supported by the Portuguese Foundation for Science and Technology (FCT) through the projects MIVIS PTDC/EIA-EIA/104031/2008, CEDAR PTDC/EIA-EIA/116070/2009, and TECTON-3D PTDC/EEI-SII/3154/2012, and under contract UID/CEC/50021/2013, as well as the Canadian Natural Sciences and Engineering Research Council. The second author is thankful for postdoctoral grant SFRH/BPD/97449/2013.
Authors' addresses: B. R. de Araújo, Department of Computer Science, University of Toronto, 40 St. George St. STE BA4283, Toronto, ON, M5S 2E4, Canada; D. S. Lopes, P. Jepp, and J. A. Jorge, INESC-ID Lisboa, Rua Alves Redol, 9, 1000-021 Lisboa, Portugal; B. Wyvill, Dept. of computer science, University of Victoria, PO Box 3055, STN CSC, Victoria, BC, V8W 3P6, Canada.
Author's current address: B. R. de Araújo , INESC-ID Lisboa.

DOI: http://dx.doi.org/10.1145/2732197

Precise visualization of implicit surfaces (IS) is a difficult and time-consuming process. The most rigorous surface reproduction technique, ray tracing, is also the costliest. Abstract, stylized methods are relatively fast but are used for specific purposes, such as illustrative visualization. The most general and popular approach to represent IS is by using a polygonal approximation. The study presented in this survey is primarily concerned about presenting fast visualization methods. Therefore, although our main focus lies on polygonization techniques, we briefly cover both stylized approaches and ray tracing.

As is common in computer graphics, rendering IS entails a trade-off between accuracy and speed. The traditional speed versus accuracy conflict is compounded in IS polygonization approaches by the desirability of obtaining a high-quality mesh. Such a mesh may be required for reuse in other applications, such as 3D printing and manufacturing. Therefore, besides delivering an accurate approximation to the surface, it must also present a good share of well-shaped triangles. Thus, choosing an appropriate polygonization strategy depends on a primary motivation, which can be speed of interaction, fidelity, or mesh quality.

The literature is rife with approaches to IS polygonization. Although many methods are generally developed with a primary goal in mind, they consider other factors. This survey aims to provide a comparison between different methods so that the interested reader can choose the most relevant approach to polygonizing an IS (see Table I). The algorithms are organized based on their primary motivation, which can either be a fast representation of the surface or an accurate one. In addition, we identify approaches that are also concerned with providing good quality meshes and faithful representations of both the features and topology of the surface.

This article is organized as follows. We start by describing how IS are visualized, discussing the main issues of existing techniques. We discuss how IS are represented, which attributes can be extracted from each formulation, and how IS can be sampled to be visualized. Based on the different sampling strategies, we describe the main basic polygonization approaches in Section 3. Section 4 focuses on polygonization techniques motivated by speed issues. Section 5 delves into quality-oriented methods. In Section 6, we discuss different approaches with respect to relevant discretization features, ranging from topological correctness, to faithful representation of both sharp and smooth features, to quality of the polygonal approximation. We conclude the article in Section 7.

2. VISUALIZING IMPLICIT SURFACES

Visualizing IS involves sampling the space to identify locations on the surface independently of their geometric nature and how they are defined. These samples can be used in two ways: (1) to directly generate a pictorial representation of the surface using direct rendering techniques or (2) to generate a discrete (mesh) representation through a polygonization process. This mesh can be used as an alternative to the IS representation, as it is still best fitted to be displayed on current graphics hardware. This section starts by contextualizing polygonization, presenting its advantages as compared to other IS visualization techniques. We follow through with a definition of IS and discuss their mathematical properties that need to be understood to sample the surface both accurately and efficiently.

2.1. Visualization Methods

Broadly, there are two major visualization categories that render *geometric loci*, which depend on the dimensionality of the object to be rendered. Direct methods allow surfaces to be rendered directly from their representations. These include ray tracing, particle-based methods, non-photorealistic rendering (NPR) techniques, and volume rendering. Indirect methods rely on polygonization to create a discrete polygonal

representation that can then be used for rendering or visualizing the surface for other purposes, such as simulation or modeling applications.

Ray tracing IS is by far the most accurate approach to visualization. This technique samples an IS directly by identifying the intersection point along a viewing ray. Ray–surface intersections are calculated using root finding methods. Spatial data structures, such as octrees, are used for acceleration. This is the most computationally intensive visualization method, and typically it takes the longest time to produce high-quality results [Bloomenthal and Bajaj 1997]. A comprehensive survey of ray tracing for IS is beyond the scope of this article. We refer interested readers to the Ph.D. dissertation of Knoll [2009].

Since the bottleneck in displaying IS (using any approach) is the number of implicit function evaluations used by the sampling process, partial methods such as particle-based or NPR techniques have been used to shorten rendering time. In principle, particle-based approaches query the defining function less often than either ray tracing or polygonization methods. Most particle systems applied to IS rendering [Rösch et al. 1997; Desbrun et al. 1995; Levet et al. 2005] rely on the model proposed by Witkin and Heckbert [1994], which combines attraction and repulsion forces to sample, visualize, and deform an implicit model. Particles can be placed either randomly, uniformly, or through a voxel-based spatial decomposition to capture specific features as proposed by Jepp et al. [2008]. These particles are then updated, attracted to the surface, and repelled from one another until they achieve equilibrium to cover the entire object. In addition, these methods [Witkin and Heckbert 1994; de Figueiredo et al. 1992; Desbrun et al. 1995] are well suited to adaptive sampling and can take into account the smoothness of the surface and adjust to sharp features. They are also very similar to both polygonal vertex relaxation and Laplacian smoothing techniques used to improve the quality of meshes obtained by repolygonization approaches presented in Section 5.1. For a summary of particle-based visualization approaches for IS, please see the Ph.D. dissertation of Jepp [2007].

NPR methods have also been used to visualize IS. These aim to present an accurate yet stylized view of a model or to focus on specific features of a surface. Equally, they can be used to simulate illustration styles or media, such as depiction techniques common in medical and scientific illustration. Although their purpose is not to create a high-fidelity visualization, NPR techniques are particularly suitable to accurately representing feature outlines such as silhouettes and discontinuities [Bremer and Hughes 1998; Belyaev and Anoshkina 2005], ridges and ravines [Belyaev et al. 1998; Ohtake et al. 2005], or singularities [Su and Hart 2005; Rösch et al. 1997], as the field function is sampled directly. Some of these techniques use particle systems as presented by Foster et al. [2005], mimicking pen-and-ink style illustrations to convey shape and form.

Volume rendering methods can be used to directly visualize IS. Nevertheless, these approaches usually depict volumetric data rather than a single surface. These data are often obtained from scans such as CT and MRI sets. The volume is rendered using a transfer function to convolve samples. This is computationally intensive due to the large number of samples, so efficient methods often use parallel graphics processing units (GPUs) [Silva et al. 2005]. For a description and survey on volume rendering with regard to IS, see the Ph.D. dissertation of Sigg [2006] and the survey on octree volume rendering methods by Knoll et al. [2006].

Finally, the most viable real-time visualization method (using commodity hardware) of nonstylized IS uses polygon meshes. As compared to the other techniques, these meshes are both fast and easy to render. Polygonization of IS requires converting from a continuous mathematical description to a discrete linear piecewise approximation. This is a lossy process, as accuracy depends on the sampling frequency, such as the size of triangles. In addition, this process needs to be adjusted to faithfully approximate

surface features and requires updating whenever the IS changes. Nevertheless, polygonal approximations enable us to explore trade-offs between fidelity of representation and interactive performance and are well adapted to commodity graphics hardware. Meshes also can readily be used in many other domains of computer graphics.

2.2. Defining Implicit Surfaces

There is a seemingly infinite number of implicit models tailored to different application domains: 3D modeling, 3D reconstruction, animation, scientific simulation, and visualization, as presented in Gomes et al. [2009]. These models share common properties. This section revisits some of the IS representations, seminal to computer graphics, that demonstrate these common characteristics, which are exploited by polygonization methods. Independently of their origin, whether based on discrete data or a mathematical model, IS are defined by a level-set function $F(X) : R^3 \rightarrow R$ such that the surface is represented by the set of points $\{X \in R^3 : F(X) = cst\}$. Depending on the mathematical model, cst represents the isovalue of the surface.

The implicit function classifies points in space in relation to the surface such that $\{X \in R^3 : F(X) < cst\}$ and $\{X \in R^3 : F(X) > cst\}$ represent points located inside or outside the object, respectively, and $\{X \in R^3 : F(X) = cst\}$ the set of points lying on the surface.

Implicit models can be generated in a number of ways: created from scan data representing either a 3D volume (e.g., MRI and CT sets) or a 2D surface (e.g., 3D laser scans), converted from polygon meshes or point clouds, or mathematically defined from well-defined functions (e.g., constructed from primitives). Whereas surfaces data obtained from 3D scans can be used to visualize an object, piecewise functions can capture the local shape of the surface and are blended to create the larger model. If the discrete sample points are dense enough, then unconnected (nonpolygonal) geometry can be used to approximate the surface. This method also benefits greatly from GPU processing for fast visualization. A survey of point-based rendering techniques including IS appears in the Ph.D. dissertation of Reuter [2003] and in Kobbelt and Botsch [2004].

Implicit models can also be built from mathematical definitions using compositions of functions [Bloomenthal and Bajaj 1997]. Since Blinn introduced the blobby molecule [Blinn 1982] using a weighted sum of density functions, several radial basis formulations have been proposed, such as soft objects [Wyvill et al. 1986], metaballs [Nishimura et al. 1985], the blobby model [Muraki 1991], and generalized distance functions. These representations have been particularly explored in computer graphics for interactive modeling and animation tools and are still one of the most common applications of IS. The following expressions present the mathematical formulation of some of these basis functions:

$$ImplicitFunction : F(X) = \sum_i w_i f_i(X)$$

$$BlobbyMolecule : f(X) = \exp(-ar),$$

where $r = distance(X, X_i)$ and X_i is the location of an atom.

$$Metaballs : f(X) = \begin{cases} 1 - 3\left(\frac{r}{R_i}\right)^2, & 0 \le r \le \frac{R_i}{3} \\ \frac{3}{2}\left(1 - \frac{r}{R_i}\right)^2, & \frac{R_i}{3} \le r \le R_i \\ 0, & r > R_i \end{cases}$$

$$BlobbyModel:\ f(X) = \begin{cases} \left(1 - \left(\frac{r}{R_i}\right)^2\right)^2, & 0 \le r < R_i \\ 0, & r \ge R_i \end{cases}$$

$$SoftObject:\ f(X) = -\frac{4}{9}\left(\frac{r}{R_i}\right)^6 + \frac{17}{9}\left(\frac{r}{R_i}\right)^4 - \frac{22}{9}\left(\frac{r}{R_i}\right)^2 + 1$$

By controlling the blending of these radial basis functions, complex objects can be created by incremental modeling such as BlobTrees [Wyvill et al. 1998; Allègre et al. 2006] or using a reconstruction process such as variational implicit surfaces (VIS) [Turk and O'Brien 1999], FastRBF [Carr et al. 2001], compactly supported radial basis functions [Morse et al. 2001; Ohtake et al. 2005; Tobor et al. 2006], and multilevel partition of unity [Ohtake et al. 2003]. For further reconstruction methods based on implicit fields and their comparison, we refer the interested reader to the recent survey presented by Berger et al. [2014].

An implicit model worth mentioning is the R-function [Rvachev 1963], which has the notable feature of associating the sign of the function to a logical property—that is, negative and positive values can be considered to correspond to logical false and logical true, respectively. This Boolean switching capability is possible since the R-function, by definition, consists of a real-valued function whose sign is completely determined by the signs of its arguments. Such function type have been extensively used in computer graphics and geometric modeling in the context of IS representation [Pasko et al. 1995; Shapiro 2007].

IS are not restricted to model rigid, inflexible geometries. Osher and Sethian [1988] devised the level set method, which relies on the IS representation to model dynamic interfaces and shapes, even if the shapes change topology over time. This method consists of a numerical technique that solves a partial differential equation describing how a surface varies over time (i.e., the level set equation), and whose solutions is the geometric loci of an IS for a given time instant. High-order finite difference schemes are generally applied to solve such evolution equations. Since this method is able to model time-varying objects, it become popular in computational fluid dynamics, combustion, soap bubble dynamics, and among many other disciplines, such as image processing, computer graphics, and computational geometry [Osher and Fedkiw 2003].

Another surface model of interest is the distance-based IS. This model has proven to be useful in reconstructing surface geometry from a set of oriented points [Taubin 2012]. Recently, Calakli and Taubin [2011] revealed that by forcing the implicit function to be a smooth approximation of the signed distance function to the surface, a significantly simpler and easier-to-implement algorithm is conceivable to reconstruct the geometry, topology, and color map of a 3D scene from a finite set of colored-oriented points.

2.3. Mathematical Properties

Using the preceding definition, we can extract interesting features to gather properties from the surface that are described next when $cst = 0$ but can be generalized for any $cst \in R$. The gradient vector $\vec{G}$ and the normal unit vector $\vec{N}$ are defined at point X using the first-order partial derivative of F:

$$\vec{G} = \bigtriangledown F = \left[\begin{array}{ccc} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \end{array}\right]^T \text{ and } \vec{N} = \frac{\vec{G}}{\|\vec{G}\|}$$

In practice, when performing a spatial decomposition with a method such as marching cubes (MC) [Lorensen and Cline 1987], it is often simpler to find a numerical

approximation via sample points around X as in

$$\nabla F = \left[\begin{array}{ccc} \frac{F(X+\epsilon x)-F(X)}{\epsilon x} & \frac{F(X+\epsilon y)-F(X)}{\epsilon y} & \frac{F(X+\epsilon z)-F(X)}{\epsilon z} \end{array} \right]^T, \quad (1)$$

which is the forward difference expression. Although it permits fast calculations of gradient vector estimates, other finite difference schemes exist, such as central and higher-order differences. These methods require additional function evaluations to provide a better vector gradient approximation. Additionally, more sophisticated gradient vector formulas such as Sobel and Scharr masks, which are very popular in image processing and volume graphics as edge detection functions [Hadwiger et al. 2006], provide high-quality numerical differentiations due to their ability to maintain a good approximation of the gradients' rotation-invariance property.

By relying on the Householder transformation, tangent vectors can also be defined at a surface point given the first-order partial derivative of F [Lopes et al. 2013]. The resulting analytical expressions for tangent $\vec{T}$ and binormal $\vec{B}$ vectors are, respectively,

$$\vec{T} = \left[\begin{array}{ccc} -2\frac{\mu\frac{\partial F}{\partial y}}{h^2} & 1-2\frac{\frac{\partial F}{\partial y}^2}{h^2} & -2\frac{\frac{\partial F}{\partial y}\frac{\partial F}{\partial z}}{h^2} \end{array} \right]^T$$

$$\vec{B} = \left[\begin{array}{ccc} -2\frac{\mu\frac{\partial F}{\partial z}}{h^2} & -2\frac{\frac{\partial F}{\partial y}\frac{\partial F}{\partial z}}{h^2} & 1-2\frac{\frac{\partial F}{\partial z}^2}{h^2} \end{array} \right]^T$$

with $\mu = max(\frac{\partial F}{\partial x} - \|\nabla F\|, \frac{\partial F}{\partial x} + \|\nabla F\|)$ and $h = \sqrt{\mu^2 + \frac{\partial F}{\partial y}^2 + \frac{\partial F}{\partial z}^2}$.

The curvature information of the function F in R^3 at point X is described by the Hessian matrix H, which contains the second-order partial derivatives of F:

$$H = \left[\begin{array}{ccc} \frac{\partial^2 F}{\partial x^2} & \frac{\partial^2 F}{\partial xy} & \frac{\partial^2 F}{\partial xz} \\ \frac{\partial^2 F}{\partial yx} & \frac{\partial^2 F}{\partial y^2} & \frac{\partial^2 F}{\partial yz} \\ \frac{\partial^2 F}{\partial zx} & \frac{\partial^2 F}{\partial zy} & \frac{\partial^2 F}{\partial z^2} \end{array} \right].$$

The Hessian matrix H represents the curvature evolution of the scalar field F, which supports the implicit function, rather than the surface defined by $F = 0$. Indeed, we need to examine curvature values and directions on the plane tangent to $F = 0$ at X. To compute these, we need to use the matrix C defined by the partial derivatives of the normal $\vec{N}$ instead of the Hessian:

$$C = \left[\begin{array}{ccc} \frac{\partial N_x}{\partial x} & \frac{\partial N_x}{\partial y} & \frac{\partial N_x}{\partial z} \\ \frac{\partial N_y}{\partial y} & \frac{\partial N_y}{\partial y} & \frac{\partial N_y}{\partial z} \\ \frac{\partial N_z}{\partial x} & \frac{\partial N_z}{\partial y} & \frac{\partial N_z}{\partial z} \end{array} \right].$$

[1]A reasonable value for ϵ has been found empirically [Bloomenthal and Bajaj 1997] to be $0.01 * side$, where side is the length of a cube used for the spatial decomposition.

Alternatively, the matrix C can be computed from the Hessian matrix H and the gradient vector $\vec{G}$, as we can see from the following equation:

$$C_{ij} = \frac{H_{ij} * \|\vec{G}\| - \frac{G_i * dot_j}{\|\vec{G}\|}}{\|\vec{G}\|^2},$$

where $dot_j = \vec{G}.[H_{j0}\ \ H_{j1}\ \ H_{j2}]^T$. Since C is defined in terms of the normalized gradient, one of its eigenvalues has zero value and the corresponding eigenvector is normal to the surface (and thus collinear to $\vec{G}$). The other two eigenvalues k_1 and k_2 are called the *principal curvatures* and their respective eigenvectors are the *principal directions* defined to lie in the plane tangent to the surface at X. According to Gray [1996] and Koenderink [1990], additional curvature measures can be computed from k_1 and k_2 beyond maximum and minimum curvature: mean curvature $= \frac{k_1+k_2}{2}$, Gaussian curvature $= k_1 * k_2$, deviation from flatness $= \sqrt{k_1^2 + k_2^2}$, and shape index $= -\frac{2}{\pi} \arctan \frac{k_{max}+k_{min}}{k_{max}-k_{min}}$. Further closed curvature formulas for implicit curves and surfaces can be found in Goldman [2005]. Note that (and as reported by Lopes et al. [2013]) the set of nonlinear differential operators derived from the Householder transformation for calculating the tangent and binormal vector fields can be applied to calculate IS curvatures, namely, principal curvature directions.

2.4. Sampling

Sampling an IS is extremely important for visualization, simulation, and manufacturing of implicit 3D models. Since the surface is defined by $\{X \in R^3 : F(X) = cst\}$, common root finding methods such as bisection, secant, false position, or Newton-Raphson [Press et al. 1986] can be used to sample and identify points of the surface depending on its continuity. Most existing IS visualization algorithms rely on either a Newton-Raphson method or interval analysis. Nevertheless, finding and accurately depicting surface features critically depends on sampling resolution. Certain surface details can only be revealed when samples are spaced closely enough. Thus, guaranteed feature identification requires additional information about the surface being rendered, as discussed in Kalra and Barr [1989]. Indeed, to properly discretize a signal, Nyquist's theorem states that its maximum frequency must be known to define a sampling rate that is greater than or equal to twice this value [Bloomenthal and Bajaj 1997]. Lipschitz constants also require advance information about the maximum rates of change of a function [Bloomenthal and Bajaj 1997]. Therefore, to guarantee that all details are identified, the distance between samples should be related to the size of the smallest surface feature. Still, small discretization steps generate too many samples that entail a considerable time overhead [van Overveld and Wyvill 2004]. Thus, a balance between speed and accuracy is important when rendering intricate, detailed surfaces. To this end, Morse theory [Hart 1997] has been used for critical point analysis, allowing extraction of detailed topological information of a surface from its implicit definition. Still, there is no general method to guarantee sampling rates that work appropriately for any surfaces. Indeed, geometrical and topological information about a surface is necessary to define an adequate sampling rate. Discretizing a surface and visualization can both be carried out in a single step by using ray tracing or particle-based methods. Still, polygonization via surface tracking uses a single step, whereas methods based on spatial decomposition generally perform sampling and rendering separately, as we will discuss in the following section.

3. BASIC POLYGONIZATION APPROACHES

This section describes the fundamental polygonization approaches used for sampling and analyzing the model space for IS. We present the three different sampling strategies used by polygonization methods that can be followed converting from a continuous mathematical description to a discrete linear piecewise approximation. These strategies are differentiated by how they sample the surface or how they guarantee to catch all of the geometric properties of the IS. The first approach—spatial decomposition—regroups all of the polygonization methods sampling the surface by defining a strategy to subdivide the space both in and around the surface. The second encompasses all surface tracking techniques starting from a reduce set of surface samples and sampling the surface based on the analysis of neighborhood of these samples close to the surface. Finally, we describe all of the methods relying on an inflation or a shrinking iterative process that will inflate a polygonal representation from topological singularities or subdivide it to fit closely the surface.

3.1. Spatial Decomposition

To perform a polygonization, the space containing the IS should be organized in a coherent manner. The space is divided into semidisjoint cells such as cubes (voxels: volume elements) or tetrahedra that will enclose the entire object. In general, only the isosurface is of interest, so only cells containing parts of the surface are stored. Stored cells are used for creating a polygonization. Polygon vertices are calculated from surface intersection points along edges of spatial cells using root finding and convergence (see Section 2.4).

There are three main methods of spatial decomposition: subdivision, continuation, and enumeration [Bloomenthal and Bajaj 1997]. We will give a short description of each of these methods in this section. These methods are still used as the foundations for many modern polygonization algorithms. Indeed, advances in polygonization have been focused on improving decomposition speed or using differently shaped decomposition cells.

Subdivision of object space is a recursive process that identifies cells containing the surface [Bloomenthal and Bajaj 1997]. First, a cell containing the entire object is identified. This cell can have a regularly shaped, convex hull or bounding box of the object. This cell is subdivided into equal parts, and any of these cells that are identified to contain part of the surface are then recursively subdivided to an agreed level—in this example, four times. The result is a collection of cells that contain surface intersections. Thus, IS can be approximated by computing cell–surface intersections to yield a triangular mesh. The higher the number of subdivisions, the greater the level of detail. Just as in the bidimensional case, where octrees are used for spatial subdivision of an implicit curve into a set of 2D lines, for 3D objects, octrees are also used to subdivide space—in this case, a cell into 3D polygons. Continuation methods use a predecomposed spatial organization and identify a single seed cell that contains part of the surface [Bloomenthal and Bajaj 1997]. From this seed cell, the surface is tracked through adjoining cells using shared edges with surface intersections. To improve efficiency by avoiding redundant recalculations, the edge surface intersection data or cell data are stored in memory. Stacks, queues, and hash tables are data structures that are commonly used for this purpose. Once edge intersection points have been found, a table is used to polygonize cells. Cell vertices are identified as being “hot” if they are inside the surface and “cold” when outside. A table contains the configurations for polygons based on cell polarity patterns. Wyvill et al. [1986] were the first to publish this method and used several performance improvements, including the hash function to only store cells containing parts of the surface, and

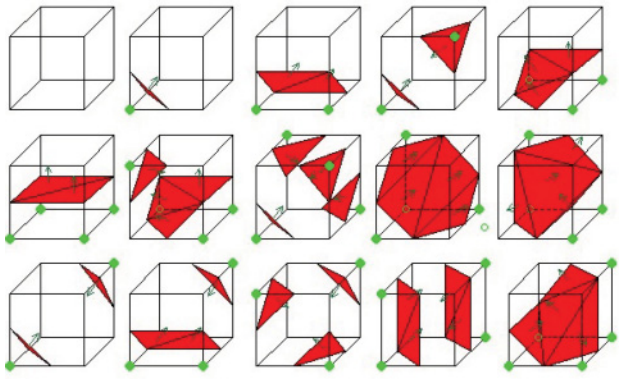

Fig. 1. MC lookup table. The 256 different configurations of vertex cell polarity are generalized by 15 cases due to existing rotations and symmetries [Lorensen and Cline 1987].

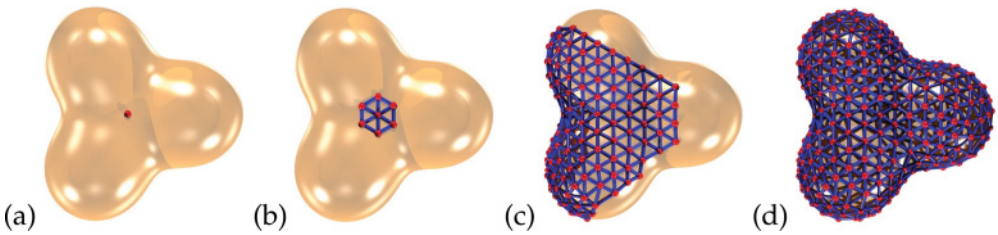

Fig. 2. Surface Tracking iterations on an algebraic IS: initial seed point projection (a), six triangles generated (b), intermediate mesh (c), and final mesh (d).

a smaller lookup table with repetitions and ambiguities removed. The combinations of creating polygons depends on vertices being either inside or outside the surface. When diagonal corners are identified as being inside the surface, there is no clear identification of the behavior of the surface in that cell. One of the simplest ways to gain more information is to sample the field function in the center of the face [Wyvill et al. 1986].

Exhaustive enumeration methods process volume scan data, such as CTs and MRI [Bloomenthal and Bajaj 1997]. Each and every cell is examined in turn to determine the surface intersection points. The MC algorithm [Lorensen and Cline 1987] is the most famous such method. It examines planar slices in turn until the whole volume has been processed. MC uses the same idea of a lookup table (Figure 1) for polygons according to cell vertex polarity, although the table does not have the same ambiguity reductions as the method by Wyvill et al. [1986]. The term *marching cubes* has become synonymous with the continuation method. Cell partitioning or spatial decomposition techniques are popular methods for rendering IS because they are very fast. A reliable implementation of the MC algorithm can be found in the VTK library [VTK 2014].

3.2. Surface Tracking

Alternatively, surface tracking methods do not subdivide the space to create polygons but query the surface directly. Surface tracking designates a family of techniques that generate polygonal approximations by starting from a point lying on the surface and generating triangles by following the surface, as depicted in Figure 2. Marching Triangles was one of the first of such approaches, and many advances are made from this basic technique.

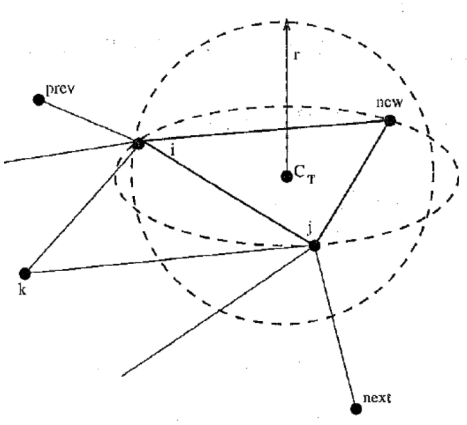

Fig. 3. Border edge expansion by the Marching Triangles applying the 3D surface Delaunay constraint resulting in a new vertex and its triangle [Hilton and Illingworth 1997].

Proposed by Hilton et al. [1996, 1997], Marching Triangles generate a 3D triangulation that verifies the Delaunay constraint—in other words, for each triangle of the mesh, the circumscribed sphere passing through each vertex does not contain other vertices. Applying this constraint generates a mesh with a uniform but not regular vertex distribution over the surface. Marching Triangles can be applied starting from either a seed point (or triangle) or from an incomplete mesh (with holes) to be refined. The algorithm generates new vertices by expanding each triangle edge of the mesh generation boundary as depicted in Figure 3, where the edge ij is expanded to create the vertex new. This vertex is placed such as the new triangle verified the Delaunay constraint. The process is repeated, resulting in the creation of a mesh that covers all of the surface.

3.3. Inflation and Shrinkwrap

The shrinkwrap algorithm [van Overveld and Wyvill 2004] is analogous to the process of using a plastic mouldable film that is shrunk (using heat) to tightly cover an object. In other words, the film is the plastic wrap that is shrunk into shape. The shrinkwrap algorithm starts with an isosurface that is homeomorphic to a sphere, which is calculated by adding a large offset value to the isosurface value. Each iteration of the algorithm reduces the offset value and then corrects vertices so that they lie on the new isosurface, more closely approximating the desired surface at each step. This process is continued until the offset is reduced to zero. During each iteration, new vertices are created to accommodate the surface details. Edges are subdivided and the midpoint is corrected along the direction of the gradient to lie on the surface. Ultimately, the shrinkwrap method converges to the surface as the resolution increases.

Inflation is the opposite approach, in which polygonal components (second image of Figure 4) are created inside the implicit volume and inflated until they create a shape that closely approximates the surface [Stander and Hart 1995], as shown in Figure 4. Components are created, adapted, and joined in relation to critical points: maximum, minimum, and saddle points visible in the left-most image of Figure 4. One of the benefits of the inflation algorithm over the shrinkwrap approach is that internal pockets are identified, although generally these regions are only visible when trimming, clipping, or transparent polygons are used.

As mentioned previously, the three fundamental approaches presented in this section diverge on the way in which they sample the space or the surface to generate a polygonal

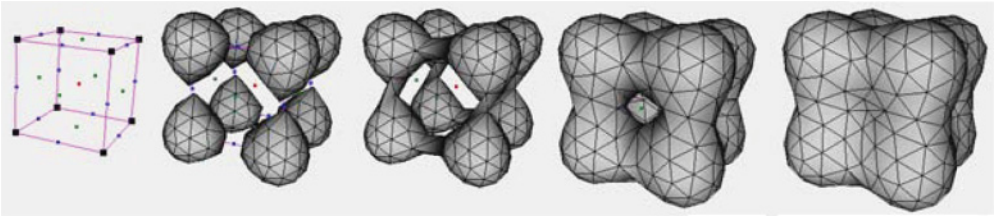

Fig. 4. Polygonization by inflation of the Blobby Cube: successive steps from the critical points analysis on the left to the final mesh on the right [Stander and Hart 1995].

approximation of the IS. Still, the fidelity of polygonal approximations depends on a number of factors: it is a balance of speed, accuracy, and quality. Some polygonization algorithms are primarily motivated by the fastest method of visualization, and others by the fidelity of the representation with respect to features. Other algorithms aim at creating a high-quality mesh that can easily be used or reused in other applications. As current graphics hardware is designed for optimal handling and processing of triangles, real-time visualization is possible for complex models with billions of triangles. Since optimal triangle handling techniques are currently available, more focus of attention can be placed on research topics related to the quality improvement of the mesh and the accuracy of the approximation. The issues of polygonization methods are discussed in the next two sections of this survey, where the methods are classified according to their motivations: to either quickly find the surface and its features (Section 4) or to create a better mesh based on surface properties—that is, motivated by the quality of triangles as a representation of the surface (Section 5).

4. FAST METHODS

As mentioned in Section 3.1, cell partitioning and spatial decomposition techniques are the most popular methods for rendering IS in particular continuation methods exemplified by two papers from the 1980s. The first polygonizer, presented by Wyvill et al. [1986], was based on a uniform voxel grid, used a hash table to avoid storing all of the voxels, a second grid of voxels to restrict the search space, and approximation techniques such as linear interpolation to determine voxel edge intersections, as well as allowing a pure voxel approximation of the surface. In 1987, MC [Lorensen and Cline 1987] was published and became the most popular technique.

MC was oriented toward volume data visualization rather than IS modeling and thus enjoyed an instant success in that community. Unlike the Wyvill paper, MC did not identify the ambiguous cases and stored every cube. Thus, with limited memory machines in the 1980s, this algorithm was comparatively slow. Although the main motivation is speed, the polygonization must also be a close approximation to the surface, and, therefore it must also consider topology and feature sensitivity. Thus, it was important to avoid the ambiguous cases found when replacing cubic voxels by triangles.

A solution to the ambiguity problem was offered by Nielson and Hamann [1991] by decomposing each cube (cell) into six (or, with a different decomposition, five) tetrahedra. This enables a simpler connectivity table because the possible permutations are based on four vertices of the tetrahedra rather than eight for a cube.

To prevent redundant calculations, Bloomenthal [1988] used a technique presented in Wyvill et al. [1986] to support three hash tables, one each for cube centers (to prevent cycling during cell propagation), corners (to prevent recomputing the implicit function value), and edges (to facilitate connectivity). A revised article and accompanying implementation of the Wyvill algorithm, including the avoidance of the redundant

calculations, was published in *Graphics Gems* [Bloomenthal 1994] (see also Bloomenthal and Bajaj [1997]), with the option of subdividing the cubes into tetrahedra. This became a very popular algorithm to convert IS into polygonal meshes. ImplicitMesher is an optimized version of this implementation, written in C++ and specifically developed for real-time IS polygonization [ImplicitMesher 2014].

Ning and Bloomenthal [1993] discuss cell decomposition algorithms and conclude that tetrahedral decomposition produces a consistent topology. However, it uses twice the number of triangles than a single-entry cubical lookup table and thus has the potential to be comparatively slower.

Avoiding the redundant calculations was rediscovered by Triquet et al. [2001, 2003], and by the early 2000s, with larger memory and faster processors, the algorithm could run at *near interactive* rates of polygonization.

Triquet introduced a new pattern table with six new patterns compared to the original Marching Tetrahedra table with 20. This reduced the polygonization time by a factor of six, measured against MC, but not against the Wyvill-Bloomenthal method that already contained many of these optimizations. This work used compactly defined basis functions similar to metaballs and soft objects. Therefore, most of the speedup was achieved by taking advantage of the field function structure and could not be applied to all implicit representations.

A similar approach taken by Cuno et al. [2004] used a hierarchical variant of Marching Tetrahedra to polygonize VIS. The method aimed at minimizing the number of implicit evaluations and quickly pruned the space. This is achieved by simplifying VIS [Turk and O'Brien 1999] models. The pseudo-Euclidean distance computation was changed to be based on the location of the normal and boundary constraints of the VIS model.

Zhang et al. [2006] rediscovered a simple speedup technique that cuts half the meshing time of MC, although it already was reported in Bloomenthal's Ph.D. dissertation [Bloomenthal 1995]. The algorithm rejects empty cells by sampling at the center and comparing the nearest distance to the surface with the length of half the voxel diagonal. This algorithm presents simple and effective speedups with the cost of several assumptions that do not guarantee that the resulting mesh correctly approximates the surface. The algorithm is only targeted at IS that estimate a distance field and in the original paper only compared against unoptimized MC. This method is more suited to achieving a fast coarse representation of IS.

Finally, regarding speed, with the increase in theability of GPUs IS can be represented using graphical shader capabilities. In the method presented by Kipfer and Westermann [2005], performance gain is achieved using two approaches. The first avoids redundant computations of edge surface intersections. The second uses features of the GPU to reformulate the isosurface identification and reduces numerical computations and memory access operations. A span space data structure is also used to avoid processing non–surface-intersecting elements. Kanai et al. [2006] present a fast method to directly render sparse low-degree IS. The approach is based on ray casting with ray intersections and blending operations performed in the fragment program on a GPU. The Accelerated MC method presented by Johansson and Carr [2006a] uses graphics hardware to achieve improvements of as much as 1,300%. The method precomputes the topology for each cell and stores it in a span space structure to avoid redundant CPU computations at runtime. It also improves isosurface acceleration by caching the topology on the GPU.

4.1. Topological Guarantees

There are three sources of topological error when trying to produce an accurate isosurface from scanned data. The first is sampling resolution (see Section 2.4). The

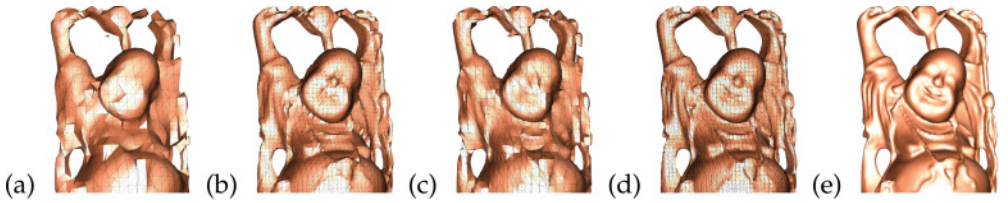

Fig. 5. Polygonization mesh result using MC and Marching Tetrahedra for the Happy Buddha model: MC with 9320 triangles (a), MC with 40740 (b), MT with 28164 (c), MT with 122516 (d), and MT with 508668 (e).

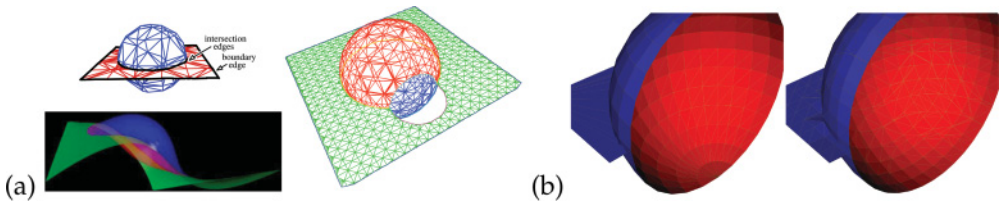

Fig. 6. Polygonization of nonmanifold IS: Bloomenthal and Ferguson [1995] (a) and Yamazaki et al. [2002] (b).

second comes from using a cubical cell classification and its corresponding triangulation (Figure 5). The third originates from using interpolated values (of the scalar data at cubical vertices) rather than actual field values. Although trilinear interpolation is the least expensive to compute, and even though better approximations exist, such as triquadratic [Barthe et al. 2002], accurate topological reconstruction cannot be guaranteed by interpolation alone.

To improve the topological correctness of the polygonal approximation produced by the MC algorithm, several approaches present alternative lookup tables for cubical cell classification [Montani et al. 1994b, 1994a; Chernyaev 1995; Hege et al. 1997; Lopes and Brodlie 2003], as well as for tetrahedral subdivision [Bloomenthal 1994]. Other solutions have proposed additional corrective steps to solve topological ambiguities, which may arise between adjacent cells [Matveyev 1994] or the existence of internal saddle points inside cells [Natarajan 1994]. Nielson [2003] proposes an alternative method that guarantees the topological correctness of isosurfaces at the cost of a more complex lookup mechanism. This mechanism is based on a three-level classification: edges, faces, and cell interiors. More recently, a method based on isosurface critical point analysis was proposed by Renbo et al. [2005]. Triangles for each cell are created by classification of the nature of critical points rather than using lookup tables.

Nonmanifold surfaces create problems for polygonization strategies. Such surfaces arise by combining mixed-dimensional geometric primitives, such as when building an object during constructive solid modeling with volumes and surfaces. Bloomenthal and Ferguson [1995] proposed a strategy for polygonizing nonmanifold IS that used a new tetrahedron classification, as shown in Figure 6. Yamazaki et al. [2002] also proposed an approach for nonmanifold IS by extending the MC algorithm to correctly handle discontinuousfields. Features such as holes, boundaries, and intersections are dealt with by enhancing the distance field, using bounding volumes to simplify the calculation between points. Although features are correctly approximated, the mesh quality depends on a user-defined subdivision level. If the cell size is small enough, then the triangulation is good and identifies sharp features. Otherwise, the mesh is of poor quality. To overcome limitations of most algorithms found in the literature and built-in commercial software packages, Gomes et al. [2010] proposed a general algorithm to polygonize nonhomogeneous and self-intersecting IS that preserve local

and global topological shape. By introducing a new uniform space space partitioning–based algorithm, their algorithm is capable of rendering surfaces with self-intersections and isolated 0 and 1D singularities. In addition, their algorithm can correctly render isolated points, self-intersections, cut points, and isolated and dangling 1D surface patches.

Plantinga and Vegter [2004] present an extension to MC using an octree-based space partitioning. Triangle sizes are adapted to the topology of the surface. Global properties of the implicit function are determined from interval arithmetic and are used as criteria for octree subdivision. Plantinga and Vegter [2007] also present an isotopy for octree-based meshing and extend their approach to apply tetrahedra to MC cells. The results, however, are limited to simple algebraic functions or metaballs. The output mesh does not present a smooth transition of triangle sizes and reveals jagged triangle edges from adaptive subdivision. The method for creating tetrahedra reduces this problem but does not completely solve it. A similar approach was followed by Alberti et al. [2005] to triangulate implicit algebraic surfaces. Instead of using MC, they proposed a subdivision algorithm using a method similar to interval analysis. The method isolates singularities and guarantees the topology in smooth areas. The algorithm results in a topological approximation that produces similar meshes to MC. The topology is guaranteed up to a given threshold due to the spatial partition being defined to identify singularities.

Inflation methods, as presented in Section 3.3, guarantee topological correctness of the polygonal approximation thanks to critical point analysis [Stander and Hart 1997]. The topology is guaranteed by tracking the critical points of any C^2 continuous IS. These points are found using interval analysis, starting from an initial bounding box that is subdivided as an octree data structure using a zero Newton-based search. They can be classified according to the eigenvalues of the Hessian matrix, which is extracted from the implicit function. This classification allows identification of the type of the critical point (i.e., maximum, minimum, or saddle point) and also the sign of the change. The same extension was done regarding shrinkwrap algorithms as demonstrated by the Bottino et al. [1996] extension to support IS with arbitrary topological genus. This is done by adjusting the topological structure of the mesh with critical points.

4.2. Feature-Sensitive Techniques

One of the main concerns regarding MC-style algorithms is their poor ability to correctly approximate sharp features and other discontinuities, including holes. Most of the artifacts are related to the discrete sampling approach followed by the cubical cell subdivision of the space. The problem typically arises in CSG operations, where the intersection of surfaces occurs within a voxel. Wyvill and van Overveld [1996] introduced a numerical technique that converges on the exact point of intersection and then subdivides triangles of an initial mesh accordingly.

Several approaches try to overcome this limitation by improving approximations to the implicit function value. Both extended MC [Kobbelt et al. 2001] and dual contouring [Ju et al. 2002] use spatial partitioning to get a better estimation of the field value, as well as refine cell pattern classifications according to the detection of sharp features. This approach was improved by Azernikov and Fischer [2005], who proposed an adaptive meshing of IS by combining an anisotropic grid for sampling sharp features with dual contouring.

The main problems of the previous works are twofold: they are only able to approximate at most one sharp feature per cell, and thin sharp features are discarded if they are located over an edge cell with no sign change. Varadhan et al. [2003] present an alternative extended dual contouring algorithm that mixes both approaches to generate an accurate polygonization from volumetric data. This new algorithm presents

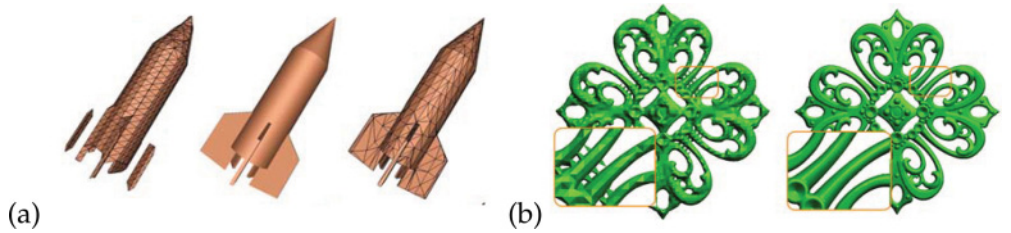

Fig. 7. Dual MC [Schaefer and Warren 2004] result for sharp features approximation spatial subdivision (a) versus Varadhan [Varadhan et al. 2006] visibility maps for an MPU model (b).

a more robust intersection test and applies adaptive subdivision techniques to get a better approximation of sharp features. The authors also used additional criteria for the adaptive octree-based sampling to guarantee one intersection per cell edge and identify any thin features [Varadhan et al. 2004].

Schaefer and Warren [2004] proposed an alternative solution by introducing the dual MC algorithm (Figure 7(b)), which uses dual grids. Compared to dual contouring and extended MC, this approach requires a sparse underlying octree to produce an equivalent contour. The dual grid also provides a better approximation of sharp features when they are not axis aligned. Varadhan et al. [2006] proposed a different solution (Figure 7(b)) for an accurate topological and geometrical polygonal approximation using visibility mapping. The algorithm is based on a spatial subdivision method presented previously by the authors [Varadhan et al. 2004]. The algorithm catches all topological features by subdividing the surface into cells that are star-shaped patches. For each star-shaped patch of the spatial surface decomposition, a visibility mapping is created and triangulated. The simple triangulation of the patch domain over the boundary of the cell is computed and then projected to generate the correct polygonal approximation. This method allows the polygonization of topologically complex surfaces and is applicable to complex implicit models, such as multilevel partition of unity [Ohtake et al. 2003] or CSG-based implicit models. Schaefer and Warren also revisited the dual contouring algorithm in Schaefer et al. [2007]. This extension preserves sharp features and guarantees construction of a manifold while preserving the genus of the original surface. In addition, based on Nielson's ideas [Nielson 2004], Schaefer and Warren extend the original dual contouring to support more than one sharp feature per cell. An adaptive version is also presented using a clustering method that generates a vertex tree between the different levels of a hierarchical subdivision. Note that although these algorithms are feature sensitive, dual contouring and dual MC rely on a previously given adaptive data structure to generate polygons, whereas extended dual contouring builds an adaptive volumetric grid from scratch.

Ho et al. [2005] propose a different solution called *cubical marching squares* to solve together sharp features, topological inconsistency, and intercell dependency when using traditional MC. They analyze each cell by unfolding it as six marching squares. Then they look for ambiguous scenarios on each edge of each square. If an edge ambiguity is detected or it contains a complicated surface (heuristically defined analyzing sample normals), the cell needs to be subdivided such as an octree structure. The process is recursively repeated until no ambiguities are detected or a maximal level of subdivision is achieved. After the subdivision step, resulting cell faces are processed generating line segments based on a lookup table for marching squares. With the help of sample normals, ambiguous scenarios are solved by checking sharp features. Finally, the surface is extracted and the final mesh is generated by connecting the segments of each face per cell. The achieved results show an adaptively refined mesh and present

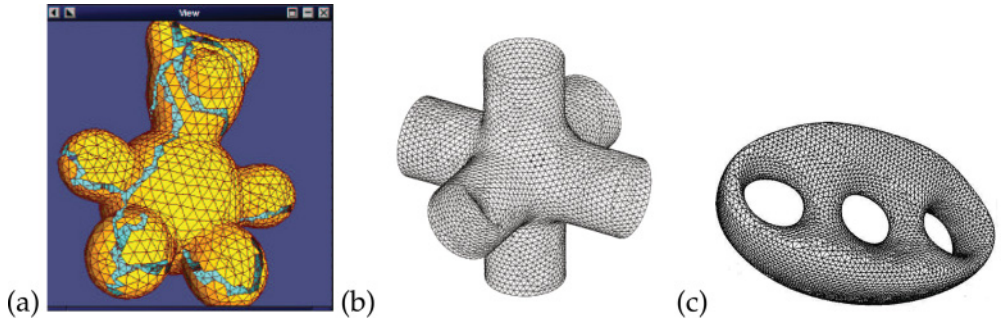

Fig. 8. Polygonization meshes produced by surface tracking approaches. (a) Marching triangle implementation showing the need of overlap testing [Akkouche and Galin 2001]. (b) Hartman algebraic surface [Hartmann 1998]. (c) Edge spinning result [Cermák and Skala 2002].

less average geometric error comparing the mesh to the input data than both extended MC and dual contouring.

Kazhdan et al. [2007] propose an algorithm for generating an adaptive watertight level set from an arbitrary octree by introducing a set of binary edge trees derived from the octree's topology. The edge trees are used to consistently define the positions of isovalue crossings, even when a single edge has multiple isovalue crossings. This allows the extraction of a watertight mesh without the necessity of refining nodes, constraining the topology of the octree or modifying node values.

5. QUALITY-ORIENTED METHODS

The algorithms described in Section 4 are motivated by quickly finding the surface and its features. In this section, we discuss the approaches primarily motivated by creating better polygon meshes.

Piecewise representations, such as MC-style algorithms, create undesired patterns of triangles that are related to the spatial subdivision. Triangles are created within voxel boundaries, so these patterns are evident in the final mesh. Creating meshes without these artifacts requires techniques that focus on mesh quality and regularization. Quality is of particular concern when the mesh is to be used for other applications, such as simulation, modeling, and deformation [Persson 2005; Frey and George 2010].

Mesh quality can be viewed in two ways: either creating regular or adaptive meshes. Regular meshes should have triangles of similar sizes, and vertices should have have similar valences. Adaptive meshes use triangles that can be adapted in size and density to reflect properties of the surface (i.e., small triangles in areas of high curvature and larger triangles in more flat regions).

5.1. Regular Meshing

Regular meshes aim to create a set of evenly sized polygons where vertices have equal valence. Regularization aims to remove the unwanted artifacts associated with spatial subdivision techniques. Regular meshes are also very easy to use in other applications, such as subdivision and texture mapping.

Surface tracking approaches, as depicted in Figure 8, can achieve good results by generating meshes of evenly sized and quasiequilateral triangles. Hartmann [1998] proposes a similar surface tracking algorithm to marching triangles [Hilton et al. 1996; Hilton and Illingworth 1997] (Section 3.2), where points located at mesh boundaries are organized into fronts that are expanded to cover the surface. Starting from a seed point, the first expansion creates a new front formed by boundary vertices. Triangles are created and appended relative to topological characteristics. Fronts need to be split

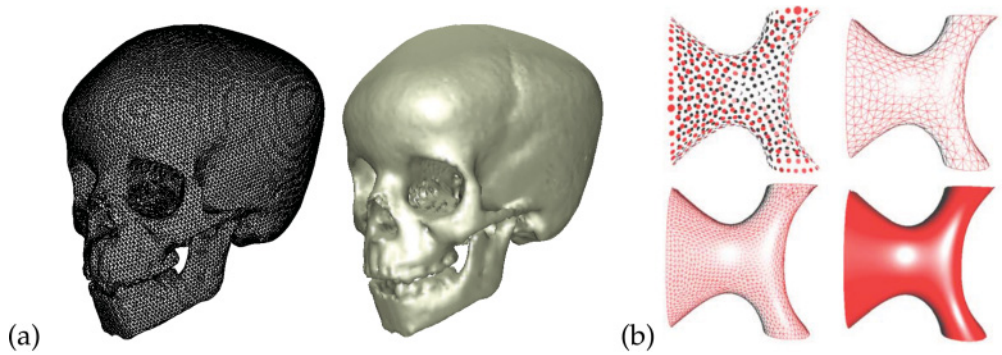

Fig. 9. (a) Regularized marching tetrahedra Treece et al. [1999]. (b) Particle-based optimization proposed by Liu et al. [2005].

or merged with others to avoid overlap. This requires collision tests, which can be costly. More than one triangle can, however, be generated by each point expansion compared to the single edge expansion used by marching triangles.

Cermák and Skala [2002] propose another variant of marching triangles called *edge spinning*. Expansion is applied using an oriented rotation around one of the edge extremities. Each new triangle is tested against existing active edges to avoid mesh overlap. Existing points are reused to create a new triangle after finding the closest point lying in an active edge (boundary). Isosceles triangles are created using a constant radius for edge expansion. The projection of the new edge extremity on the surface is done by binary subdivision. The convergence is slower for binary search, even though it does not require gradient computation, as with a Newton step. The edge spinning algorithm presents more almost-equilateral triangles than marching triangles. The expansion usually creates only one new triangle, although some scenarios can create two using adjacent points or three using the nearest colliding and adjacent points.

The meshes produced by marching tetrahedra (Section 4, [Bloomenthal 1988]) approaches were of poor quality. Chan and Purisima [1998] proposed a simpler subdivision for the tetrahedra to produce more regular triangles. The ratio of longer to shorter edges was reduced, and a smaller number of resulting triangles was observed. There is a marked improvement in the quality of the mesh, as the triangulation is smoother and without a subdivision pattern. The method may not be compatible with scan data, as it requires a sampling inside the voxel, which is not available.

The requirement for an internal voxel sample was partially eliminated by Treece et al. [1999] (Figure 9). A method is presented for regularized marching tetrahedra applied to volume data extraction. This approach combines marching tetrahedra for isosurface extraction and vertex clustering to produce the regularized triangle set. Furthermore, by applying this additional step, the final triangle size of the mesh is reduced by 26% to 30%. Liu et al. [2005] also presents a regular meshing method for algebraic- and skeleton-based IS designedto handle dynamic implicit descriptions (see Figure 9). They use a particle system similar to Witkin and Heckbert [1994] with repulsion and fission (i.e., killing particles) mechanisms to produce a uniform sampling of the surface. Then a ball-pivoting algorithm is used that starts from a triangle created using three initial points, then travels along the surface by pivoting from sample to sample. This process creates new triangles and finishes when all samples are visited, producing a regular mesh.

Recent reviews of traditional MC have produced more regular meshes and avoid small and degenerate triangles. Dietrich et al. [2009b] proposes MC using the edge

transformations (Macet) algorithm by introducing a new stage between the detection of active edges and the intersection calculation. This stage transforms the end points (cell extremities) of the active edge, combining a gradient and a tangential transformation. The algorithm chooses the best of the two transformations by using an iterative process. Finally, a different intersection method is proposed based on bisection, since the assumption of alignment of the edges with the grid is no longer valid. This algorithm has been extended with the notion of edge groups in Dietrich et al. [2009a] producing less skinny triangles than the original Macet algorithm. This is done using a probabilistic analysis that identifies MC configurations that generate skinny triangles. In such scenarios, an extra vertex is added to the cell for triangle generation.

A different approach is used by Moore and Warren [1995] that presents a modified implementation of MC using a mesh displacement with boundary classification. Grid points of the cubic mesh generated by MC are displaced so that vertices lie on or near the appropriate zero set over a cubic grid. Compared to MC, this mesh displacement technique improves the mesh quality by eliminating badly shaped triangles (e.g., small and narrow triangles) produced by MC and reduces the number of triangles by 40% to 50% without significant loss of accuracy. A related approach is used by Raman and Wenger [2008] called *SnapMC*. The scalar field value of intersection points close to grid corners are changed to snap these points onto the grid vertex. Then the traditional MC approach is applied on the new scalar grid using an extended lookup table. Finally, the snapped vertices are moved back to one of the original isosurface intersections. Although this algorithm is two times slower than the original MC, it successfully reduces the number of triangles by 20% to 40%. The method is faster and more robust than the DelIso [Dey and Levine 2007] and Afront [Schreiner and Scheidegger 2006] algorithms, which are presented in Section 5.2.2. However, it does not generate an adaptive mesh.

Besides MC and marching triangles approaches, several IS meshing algorithms are based on constrained Delaunay triangulation. Chew [1993] presents a technique to create high-quality triangular meshes for curved surfaces by generalizing Delaunay triangulation. A high-quality mesh refers to triangles that have angles between 30 and 120 degrees and for which the triangle density can be controlled according to the curvature of the surface. Starting from a crude triangulation, the Delaunay property is checked for all triangles. The mesh is updated by subdivision steps or edge flipping until the desired quality is reached. Rineau and Yvinec [2007] have implemented generic software for meshing algorithms based on the notion of constrained Delaunay triangulation and the Delaunay refinement paradigm, which are available in the CGAL library [CGAL 2014].

One feature that is common to all of the regular approaches mentioned is that they are not adapted to local surface properties. Large numbers of small triangles are required to approximate surfaces with large variations in curvature.

5.2. Adaptive Meshing

Meshes are described as adaptive when triangles are created relative to characteristics of the surface. Size and density of polygons are related to surface properties such as curvature or features. In adaptive polygonization methods, there is a larger number of triangles in regions of high curvature or in the presence of sharp features. Conversely, in more flat regions, triangles are larger and there are fewer of them.

This section describes several algorithms that produce adaptive meshes. First we describe approaches that are based on adapting existing meshes, then we discuss approaches that are designed to create new adaptive polygonizations directly from the surface.

5.2.1. Remeshing. Remeshing approaches create new triangles from a previously defined mesh. This is generally motivated by either speed considerations or availability of data (e.g., medical scans).

There are two main types of remeshing algorithms: those that do not make further calls to a defining function, which are designed to be faster and can be used where no further information is available (e.g., scan data), and those that must make additional evaluations (e.g., with defining functions).

Without further calls to the implicit function. Several methods perform global modification of an existing mesh. Feature-sensitive remeshing techniques [Vorsatz et al. 2001; Wood et al. 2002; Attene et al. 2003] can be applied to meshes generated using MC-style approaches without using any further IS information.

Vorsatz et al. [2001] present several triangle operators that are applied to improve the mesh. This approach was also used by Kobbelt and Botsch [2003] to generalize sensitive remeshing techniques as a reverse engineering tool and complement previous work (i.e., extended MC) [Kobbelt et al. 2001].

Remeshing can be restricted to specific areas, such as around sharp features. Attene et al. [2003] present a remeshing method sensitive to sharp features based on a simple filter called *Edge-Sharpener*. The filter starts by classifying mesh edges, vertices, and triangles. Smooth areas are clustered to identify the triangles located near or at sharp features. Triangles at sharp features are subdivided and their vertices are optimized using a process similar to Ohtake and Belyaev's normal error minimization [Ohtake and Belyaev 2002]. (Ohtake's method works correctly over regular or almost regular meshes, so it cannot be used over adaptive triangulations. In addition, it depends on the initial mesh sampling correctly identifying all sharp features).

Spatial decomposition techniques can also introduce nonexistent topological features, such as unwanted handles, in the polygonal mesh. This problem can be solved by using the technique from Wood et al. [2002] for remeshing. First, handles are found using a Reeb graph representation. Then, the significance of handles is evaluated and erroneous ones are identified and removed. Reducing and removing unnecessary topological features allows a better approximation for other mesh processing techniques, particularly mesh compression algorithms.

Requiring further calls to implicit function. Another approach to generate adaptive meshes of IS is to improve the polygonal approximation at the cost of additional field evaluations. These methods use the result of an existing polygonization algorithm (most of the time using marching tetrahedra). Several steps are performed on the mesh to improve its quality, to be more sensitive to sharp features, and to become adaptive or topologically correct.

Rösch et al. [1997] present a post-remeshing technique based on WH-style particle [de Figueiredo et al. 1992; Witkin and Heckbert 1994] energy minimization, where repulsion and attraction forces are applied to mesh vertices to optimize their placement and therefore the quality of the mesh. Particles are modeled as a surface-constrained mass-spring system. Surface constraints include curvature and singularities.

Another particle-based method was proposed by Persson [2005] to generate unstructured simplex meshes for implicitly defined surfaces. Prior to polygonization, a mesh size function is computed to specify the desired size of the elements. This function is adapted to the curvature and the feature size of the surface. The node locations are calculated by solving a force equilibrium in a truss structure (using piecewise linear force-displacement relations), and the boundary nodes are projected using the IS definition. The mesh topology is then determined by the Delaunay algorithm that triangulates the optimal set of nodes. This algorithm typically produces meshes of very high quality and is simpler than other meshing techniques found in the literature. Due to its iterative nature, the algorithm is well suited for level set method applications in

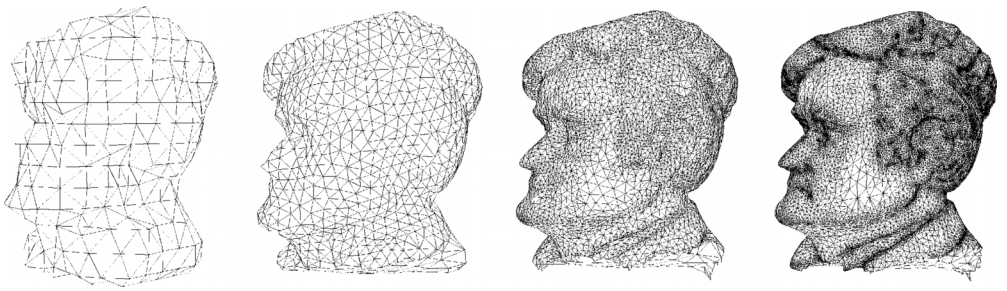

Fig. 10. Adaptive meshing using subdivision over MC triangles [Neugebauer and Klein 1997].

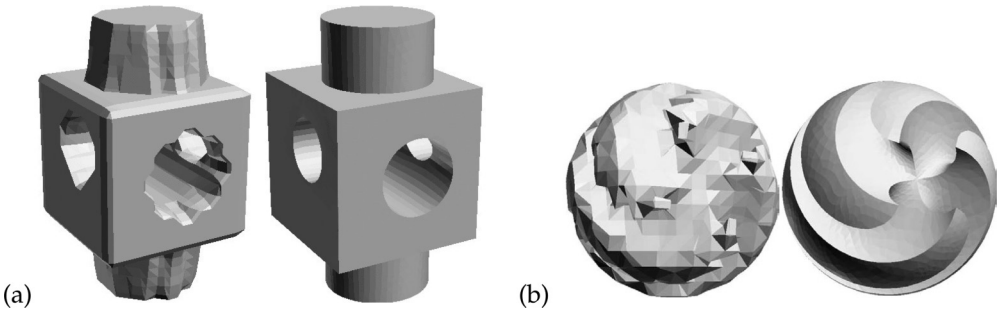

Fig. 11. Ohtake remeshing technique over MC: Ohtake et al. [2001] (a) and Ohtake and Belyaev [2002] (b).

fluid dynamics, shape optimization, and structural deformations. The software of this approach is made publicly available in MATLAB/C++ code under the name of DistMesh [Persson 2014].

Neugebauer and Klein [1997] combine a subdivision-based method with the discrete MC from Montani et al. [1994a], which uses an MC alternative lookup table [Montani et al. 1994b] generating less triangles and solving some ambiguous configurations. The original coarse mesh was subdivided to produce an adaptive triangulation that better approximates the IS. The coarse mesh is improved by shifting the vertices into the center of gravity of surrounding polygons. Then, vertices located on a coplanar surface or along almost collinear edges are removed. Finally, a fixed number of subdivision iterations is applied to mesh triangles, as shown in Figure 10.

Ohtake et al. [2001] used a combination of techniques to produce good approximations of sharp features (Figure 11). The optimization is made on mesh vertices combining the use of three forces similar to a WH particle system. Two forces are used to optimize vertices using the implicit function value and its gradient. A third is applied to improve mesh regularity using a Laplacian smoothing operator. The resulting mesh better approximates sharp features and is of higher quality regarding regularity at the cost of several iterations of the local operator.

Ohtake et al. also proposed a different approach Ohtake and Belyaev [2002] based on the dual primal mesh concept, which consists of two steps. The first creates the dual mesh based on vertices created from triangle centroids that are projecting over the surface. This is in contrast to directly using the vertices obtained by MC as was done by the Neugebauer and Klein [1997] approach. Then, the second step optimizes the modified mesh vertices by minimizing a quadratic energy function using a Garland-Heckbert error metric [Heckbert and Garland 1999]. This step is a common mesh decimation technique that can be applied to any polygonization result to generate an

optimal triangulation better approximating curvature variations. During these two steps, curvature-weighted vertex resampling (similar to the Laplacian smoothing) and adaptive mesh subdivision procedures are performed, depending of the normal deviation. Although this method produces good results, it requires a fine initial mesh to retrieve all shape measures and results in many calls to the implicit function during the post-remeshing process. A similar approach is followed by Peiró et al. [2007] that also mixes Laplacian smoothing with local modifications such as side swapping. It also includes an optimization step using a curvature estimation of the surface for triangle side collapsing.

5.2.2. Meshing. Many adaptive techniques do not need an initial polygonal representation. They examine the definition of the surface directly to create the mesh. The traditional approaches of spatial decomposition and surface tracking are both used to create adaptive meshes. The algorithms, however, are adapted to have more consideration of surface properties and create triangles accordingly.

Spatial decomposition. Velho et al. [1999] present a unified and general tessellation algorithm for parametric and IS. It generates an adaptive mesh using controlled subdivision. The algorithm starts by creating a simplified uniform spatial decomposition creating a coarse triangulation. Afterward, a refinement step is performed, sampling the edges and subdividing the triangulation to better approximate the shape.

Galin and Akkouche [2000] propose a method leading to an adaptive mesh for visualization of skeletal-based IS for modeling operations. The algorithmstarts by creating an octree using a subdivision criteria based on the Lipschitz condition [Kalra and Barr 1989]. The Lipschitz property allows culling of empty cells and identification of cells that needed to be subdivided. Cell polygonization is performed using a lookup table similar to marching tetrahedra [Bloomenthal 1994]. Adjustments are proposed to correctly deal with ambiguities and adaptive cell sizes. When ambiguities are detected, cell subdivisions are required to avoid cracks on the final mesh.

Paiva et al. [2006] present an algorithm to generate an adaptive mesh that captures the exact topology of the IS. The algorithm starts by building an octree using three subdivision criteria based on the interval analysis of the implicit value and its gradient. The first criterion discards empty cells using interval arithmetic. The second uses the gradient value to identify topological features. Then, the third criterion estimates the curvature from the interval analysis of the gradient. This method is similar to the approach of Azernikov and Fischer [2005] (Section 4.2) that creates an octree. The mesh is then generated using a version of dual MC [Schaefer and Warren 2004] (Section 4.2), which uses a Lewiner et al. [2003] MC implementation. Finally, mesh vertices are shifted by vertex relaxation using the tangent plane defined by the normal and the barycenter of neighboring points. This method presents adaptive meshes with guaranteed topology or point to the user ambiguous parts, which can be solved by further refinements. However, it only applies to algebraic surfaces with C^2 continuity, as its subdivision criteria cannot handle implicit functions with a zero length gradient.

Bouthors and Nesme [2007] present an adaptive meshing method for dynamic implicit models also relying on dual representation. The first is a *mechanical mesh*, which is a coarse mesh approximation generated by MC that is optimized using a particle system. Optimization creates a regular sampling of the surface and better approximates sharp features using the method from Ohtake and Belyaev [2002]. Finally, the visualization is offered through a second mesh, named the *geometric mesh*. This mesh is obtained by applying successive subdivision steps on the mechanical mesh. The subdivision criteria uses the normal vector and implicit gradient analysis. This approach presents smooth and adaptive mesh results as depicted in Figure 12; however, only

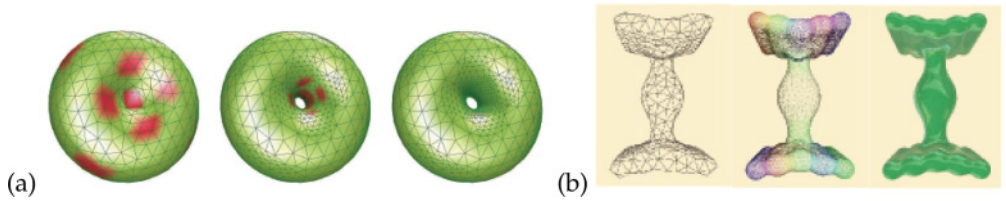

Fig. 12. Dual MC extension proposed by Paiva et al. [2006] (a) and adaptive mesh generated by Bouthors and Nesme [2007] (b).

implicit models with few primitives (i.e., metaballs, convolution, skeleton-based models) can be supported. Major topological changes of the implicit function cannot be approximated by this method (e.g., disjoint elements).

Gois et al. [2008] use a partition of unity implicits method to produce an IS representation. This method produces an adaptive triangulation for complex topological models and correctly identifies sharp features. The method couples the polygonization process with the implicit representation thanks to an adaptive structure named J_1^A triangulation. The J_1^A triangulation allows the association of spatial decomposition with an adaptive surface extraction algorithm. The algorithm starts by subdividing the implicit function domain using the J_1^A triangulation. For each J_1^A block, a local approximation is generated and a recursive subdivision of the domain is performed. Sharp features are detected using the same classification as Kobbelt et al. [2001] and approximated using a method similar to Ohtake's MPU [Ohtake et al. 2003]. The triangulation is obtained from the J_1^A definition, and a mesh enhancement is applied, displacing vertices. This method generates an adaptive representation and correct sharp feature approximation. The triangulation, however, is of poor quality when compared to other similar spatial decomposition techniques due to the nature of the J_1^A triangulation.

Tetrahedral cell decomposition was used to achieve an adaptive mesh by Hall and Warren [1990]. Algebraic IS are polygonized using a recursive tetrahedron-based adaptive spatial subdivision method. This approach was extended by Hui and Jiang [1999] to present an adaptive marching tetrahedral algorithm for bounded implicit patches. The patch is initially enclosed by a tetrahedron that is subdivided according to vertex value classifications. Ambiguous tetrahedron/surface intersections are further subdivided, which results in an adaptive polygonization. Crespin [2002] also proposes an algorithm based on tetrahedra. This method has the limitation of generating a dynamic triangulation for VIS [Turk and O'Brien 1999] using incremental Delaunay tetrahedralization. An extended bounding box is created using the constraint points of the implicit model, which is subdivided into tetrahedra instead of cubical cells. A refinement criterion based on the tetrahedron's circumscribing sphere is used to subdivide the tetrahedron, being triangulated in a method similar to Bloomenthal's approach.

These methods are able to produce adaptive meshes thanks to nonuniform spatial subdivision. The method presented by Paiva et al. [2006] is the only one that uses local implicit characteristics such as curvature as criteria for the subdivision. This is also partially true regarding the method used by Gois et al. [2008], as it generates its own IS representation. The resulting mesh produces ill-shaped triangles due to the nature of the J_1^A triangulation. The other methods try to identify the correct topology of the surface by applying subdivision for a more robust surface/cell intersection classification.

Surface tracking. Surface tracking algorithms such as the marching triangles from Hilton and Illingworth [1997] and Hartman's [1998] approach present more controlled and regular meshes than spatial decomposition methods. This is because the samples

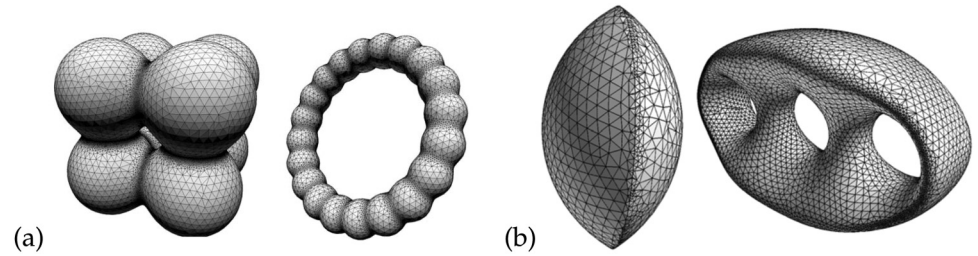

Fig. 13. Adaptive meshes using surface tracking. (a) Karkanis using geodesic-based curvature estimation [Karkanis and Stewart 2001]. (b) Cermák adaptive edge spinning [Cermák and Skala 2004].

evaluate and follow the surface rather than examine the space surrounding the object. Regularity reveals to be a problem when the edge step is not small enough to correctly approximate a high-curvature surface area. On the other hand, it may produce too many triangles on a low-curvature area. The following approaches extended surface tracking methods to produce an adaptive mesh that better approximates the IS.

Akkouche and Galin [2001] present an extended version of Hilton and Illingworth [1997] marching triangles. The algorithm produces an adaptive mesh and supports local repolygonization when minor changes are applied to the implicit function. The algorithm is divided into two phases. The first phase is the growing phase that adds validated, nonintersecting triangles to the mesh when expanding an edge—for instance, there is no triangle in the circumscribing sphere with the same orientation. Then, the second phase closes all cracks that remain from the growing stage. The adaptive mesh is achieved by constraining triangle edge lengths to local surface characteristics using both a midpoint projection heuristic and Delaunay triangulation properties. The drawback of this method is that it does not correctly approximate sharp features.

Karkanis and Stewart [2001] (Figure 13) present a similar method to Akkouche and Galin. They use a local curvature measure to define the edge length of the expansion rather than edge midpoint projection over the surface. The radius of curvature is the local measure that is computed using the minimum geodesic lying in the normal plane. The radius is computed using the angle between the surface normal and geodesic normal. In contrast to Akkouche's method, gap closing can create additional points to produce a smoother edge length transition. It uses vertex relaxation, convex filling, edge flip, and subdivision.

Even with the use of a more reliable curvature estimator, both Akkouche and Karkanis' methods presented earlier are not able to correctly support sharp features. McCormick and Fisher [2002] present an improvement to marching triangles that is more sensitive to sharp features, such as edges and corners. This is done by constraining the marching triangles approach with the detection of C^1 discontinuities. The detection of features creates a set of seed points for the algorithm to create correct approximations using line fitting for edges.

De Araújo and Jorge [2005a] present a surface tracking algorithm to generate adaptive triangulation of smooth VIS. A front-based propagation approach is used, based on Hartmann [1998], instead of marching triangles. In a single step, this algorithm generates a mesh adapted to the local curvature of the IS. Starting from a seed point, the front propagation generates newpoints updating the border of the triangulation. New points are created at a given distance from the front related with an heuristic based on the curvature of the IS. The curvature is extracted from the Hessian matrix, which is computed with the second-order partial derivative of the implicit function. Compared to the Karkanis and Stewart [2001] curvature estimator using the minimum geodesic, this solution provides a more reliable curvature estimator with less computational cost.

On the other hand, the triangle density of the mesh can be controlled using the curvature heuristic scalar; however, it does not correctly reproduce sharp features. This is due to using a fixed step value to create the new points when the curvature cannot be extracted. This method was adapted to support MPU and other algebraic implicit models in de Araújo and Jorge [2005b]. Instead of relying on front propagation, this algorithm is based on point expansion. By doing so, it avoids the cost of managing fronts that can be merged or split, such as in Hartmann's original algorithm. This approach takes advantage of the MPU octree to propose a faster method of collision detection during mesh generation.

Cermák and Skala [2004] also present an adaptive extension of the edge spinning algorithm (Figure 13(b)). The logic of the algorithm remains the same as in the original version. The radius, however, is used to place the new vertex when the edge spin is variable. The radius is computed using a curvature estimation based on normal vectors. A root-finding adjustment is performed to better approximate sharp features computing the intersection of three planes: two from the existing tangent and the other from the circle. The accurate location of the sharp point is made by binary subdivision. As depicted in Figure 13, the resulting mesh has a better approximation of sharp features than other surface tracking methods. The improvements also decrease the number of triangles with a poor aspect ratio. But only few results are presented by the authors, and the method's reliability with more complex IS models is not illustrated. This algorithm has been extended in Cermák and Skala [2007] to polygonize disjointed IS. This is done by sampling the surface using a regular grid with a cell size proportional to the smallest object.

Using the same sampling approach, Cheng et al. [2004] propose an adaptive method relying on Delaunay triangulation to polygonize smooth and compact surfaces without boundaries. First, they start by sampling the surface to define a set of seed points for the algorithm. Points are added to the Delaunay triangulation by using a geometric sampling. The quality of the triangulation is improved using the Chew [1993] approach (Section 5.1). Finally, the adaptive mesh is obtained by smoothing the triangulation according to a threshold. The threshold uses vertex dihedral angles to represent a local estimation of the curvature.

Following the ideas of Cheng et al. [2004], Dey and Levine [2007] proposed an adaptive meshing method for isosurfaces. Starting from an initial 3D Delaunay triangulation, they recover a "rough" version of the surface. Delaunay-based criteria are applied at each vertex insertion, and poles estimate the scale of local features. A refinement process is then performed over the mesh from the Delaunay triangulation. This method does not require that Delaunay is applied at each insertion; only the previous mesh and pole values are used to produce the final adaptive mesh.

Using the approach proposed by Chew [1993] (Section 5.1), Boissonnat and Oudot [2005] proposed an adaptive meshing algorithm for sampled surfaces. It can be applied to IS that are C^2 continuous and for which gradients do not vanish. First, they search for the set of points on the surface that have horizontal tangent planes—that is, the critical points of the function with respect to the height function. These critical points are identified using the generalized normal form modulo if the IS is polynomial or using interval analysis if the function is not. Following a surface tracking approach, triangles are added by finding a point with the smallest radius of curvature. At each step, the Delaunay property is certified over the incremental mesh. This method presents a topologically correct adaptive meshing; however, no global timing is presented in the article—only comparative relationships with the Delaunay triangulation. The algorithm does not correctly approximate sharp features.

A similar approach is presented by Schreiner and Scheidegger [2006] with their Afront algorithm. First, they sample the surface according to the curvature using a

guidance field function [Schreiner et al. 2006]. Then, these points are used as seeds for a front propagation algorithm. The resulting mesh combines high-quality triangles, adaptivity, and fidelity. Unfortunately, depending on the implicit definition, models polygonized within seconds using MC require minutes using this approach. Following a surface tracking approach as well, Xi and Duan [2008] produced a curvature-dependent semiregular mesh approximating sharp features correctly while supporting disjoint implicit components. Point seeds are created on all disconnected elements of the surface by a regular grid sampling. Then, sharp features are detected, creating new point seeds. These seed are prioritized depending of their curvature and are subsequently expanded, generating the triangular mesh. For each new triangle, the Delaunay face property is verified, forcing the property that the intersection of the local Delaunay sphere with the isosurface is a topological disk. This is done by evaluating the normal variation among several points inside the Delaunay sphere.

The algorithm of Meyer et al. [2007] generates an adaptive mesh with nearly regular triangles mixing Delaunay criterion with a dynamic particle system. Starting from an isosurface description, the medial axis of the surface is extracted using all grid points. A sizing field is then created based on the maximum curvature extracted from the Hessian matrix of an implicit function approximating the isosurface. The sizing field is sampled, generating particles that are distributed with a density related to the curvature of the surface and are projected on the surface using the iterative Newton-Raphson gradient descent method. Finally, the distribution of particles is triangulated using the Delaunay triangulation method. Several examples of biological datasets were tested, producing high-quality adaptive meshes. The drawback consists of the method being time consuming (complex models require several hours) and unsuited to reproducing sharp features, as the sizing field is smoothed and the implicit representation is continuous.

Gelas et al. [2009] introduced a two-stage adaptive mesh algorithm that minimizes distance error while preserving topology and sharp features. The algorithm applies a Delaunay triangulation to a random sampling of the surface. This is then refined by minimizing the local geodesic distance using a quadric error metric and flipping the edges that fail the Delaunay criterion. This algorithm only uses both implicit and gradient values and estimates the geodesic by analyzing the normal of several sampled vertices. The final result is an adaptive mesh with good triangle edge ratios and good reproduction of sharp features.

6. DISCUSSION ON POLYGONIZATION

Table I presents a classification of existing polygonization approaches comparing primary motivations: speed, topology reproduction ability, quality of sharp feature approximation, surface curvature approximation (smoothness), and the quality of the mesh. We choose a qualitative classification focusing on each of these topics to compare the different methods, as more objective metrics are hard to achieve. This is mainly due to the lack of available implementation and different hardware configurations used to assess the performance of each method. In addition, most of the existing approaches are not demonstrated using a consistent IS dataset—that is, the complexity of the surface is heterogeneous as well as its provenance (algebraic or volume data) and the underlying IS representation. Based on the algorithmic sampling strategy, on the results presented in each paper and the data (IS) used to demonstrate the approach, we devised a relative scale for each primary motivation and classify the examined methods whenever possible, providing an overview of the existing work on polygonization presented in this survey. Each topic presented in this table is discussed separately in the following sections, allowing us to recommend some methods based on these primary criteria. Finally, the discussion of these topics appears summarized in Table II,

Table I. Comparison of Polygonization Approaches

Note: Scale: $--- < -- < - < \simeq < + < ++ < +++$. The Mesh Type column describes patterns from spatial decomposition (P), adaptive (A), and regular (R). For Speed, the approach is considered variable (VA) when the process cannot be compared to finite spatial decomposition. The classification is empty when it is not possible to evaluate the category due to the lack of support. (Abbreviations: Curv., curvature; MC, marching cubes; MTet., marching tetrahedra; MTri., marching triangle, ReMesh., remeshing; Subdiv., subdivision; S.D., spatial decomposition; Topo., topology.)

Motivation	References	Mesh Type	Speed	Topology	Features	Smoothness	Mesh Quality	Techniques	Notes
Speed	Wyvill et al. [1986]	P	+	−		−	−	S.D.	
	Lorensen and Cline [1987]	P	≃	−−		−	−	M.C.	
	Bloomenthal [1988, 1994]	PA	+	−		−	−−	MTet.	
	Hall and Warren [1990] and Hui and Jiang [1999]	PA	≃	++	−	+	−	S.D. Tet	
	Ning and Bloomenthal [1993]	PR	+	−		−	−	MTet. Subdiv.	
	Montani et al. [1994b]	P	+	++		−	−	MC. Lookup	For isosurface data
	Bloomenthal and Ferguson [1995]	P	+	++	+++	−	−−	MC. NoLookup	Nonmanifold
	Chernyaev [1995], Lewiner et al. [2003], and Lopes and Brodlie [2003]	P	+	+++		−	−	MC. Lookup	For isosurface data
	Velho [1996] and Siqueira et al. [1998]	PA	+	+		+	−	MTet.	
	Hege et al. [1997]	P	+	++		−	−	MC. Lookup	For isosurface data
	Chan and Purisima [1998]	P	++	−		+	−	MTet. Subdiv.	Lesser triangles
	Kobbelt et al. [2001]	P	≃	+	++	+	−	Extend. MC	Distance fieldsimpl.
	Triquet et al. [2001] and Triquet et al. [2003]	P	++			−	−	MTet. Speedup	For blobs-like IS
	Ju et al. [2002]	PA	≃	++	++	+	+	Dual Contour	
	Yamazaki et al. [2002]	P	+	++	+++	−	−−	MC. Lookup	Nonmanifold
	Nielson et al. [2003]	P	≃	+++		−	−	MC. NoLookup	For isosurface data
	Varadhan et al. [2003] and Varadhan et al. [2004]	PA	≃	++	+++	+	+	Ext. Dual Contour Topo.	N feat. per cells
	Cuno et al. [2004]	P	++			−	−	MTet. Speedup	For VIS
	Plantinga and Vegter [2004, 2007]	PA	+	++	−	−−	−	MTet. Topo.	
	Schaefer and Warren [2004]	P	≃	+	++	+	−	Dual MC.	N feat. per cells
	Alberti et al. [2005]	PA		++		+	−	S.D.	
	Azernikov and Fischer [2005]	PA	≃	+	+	+	+	Dual Contour	
	Rembo et al. [2005]	P	+	+++	+	−	−	MC. NoLookup	
	Schmidt et al. [2005]	P	+++			−	−	MTet. Speedup	For BlobTree
	Varadhan et al. [2006]	PA	≃	+++	+++	+	+	Visibility Map	N feat. per cells

Continued

Table I. Continued

Motivation	References	Mesh Type	Speed	Topology	Features	Smoothness	Mesh Quality	Techniques	Notes
Speed	Zhang et al. [2006]	P	++			−	−	MTet. Speedup	For distance field IS
	Schaefer et al. [2007]	PA	≥	++	+++	+	++	Dual Contour	
	Kazhdan et al. [2007]	PA	++	++	+++	+	++	S.D.	Edge trees
	Raman and Wenger [2008]	PA	+	++	+	−	−	SnapMC	
	Dietrich et al. [2009b] and Dietrich et al. [2009a]	PA	+	+	+	−	−	MACET	
	Gomes et al. [2010]	PA	+	+++	+++	−	+	S.D.	Nonmanifold
Mesh Quality	Hilton et al. [1996]	R	VA			+	++	MTri.	
	Hartmann [1998]	R	VA			+	++	MTri. like	
	Treece et al. [1999]	R	≥	−	+	+	++	MTet.	
	Cermák and Skala [2002]	R	VA			+	++	MTri. like	
	Nielson [2004]	R	≥	+++		++	++	MC. Lookup	For isosurface data
	Liu et al. [2005]	R	≥	−	−	+	++	Particles	For skeleton-based IS
	Bottino et al. [1996]	A		++	+	+	++	Shrinkwrap	
	Neugebauer and Klien [1997]	A	++		++	++	+++	MC. Remesh.	Dual mesh
	Rösh et al. [1997]	A	+		++	++	+++	MC. Remesh.	Particle
	Stander and Hart [1997]	A		+++	+	+	++	Inflation	Requires C2 function
	Velho et al. [1999]	A	≥	++	+	++	+	S.D.	
	Galin and Akkouche [2000]	A	≥	−	−	+	+	S.D.	For skeleton-based IS
	Akkouche and Glain [2001]	A	VA			+++	++	MTri. like	Use Curv. estimation
	Karkanis and Stewart [2001]	A	VA			+++	+++	MTri. like	Curv info: geodisc
	Ohtake et al. [2001]	A	+		++	++	+++	MTet. ReMesh.	Like particle system
	Vorsatz et al. [2001] and Kobbelt and Botsch [2003]	A	+		+++	+++	+++	Full ReMesh.	Particle
	Crespin [2002]	A	≥			+	++	S.D. Delaunay	For VIS
	McCormick and Fisher [2002]	A	VA		+++	+++	++	MTri. like	Use Curv. estimation
	Ohtake and Belyaev [2002]	A	++		+++	+++	+++	MTet. ReMesh.	Dual mesh
	Wood et al. [2002]		≥	+	+	++	++	Full ReMesh.	Reeb graph
	Attene et al. [2003]	A	+		+++	++	++	Partial ReMesh.	Filter based; edges improv.

Continued

Table I. Continued

Motivation	References	Mesh Type	Speed	Topology	Features	Smoothness	Mesh Quality	Techniques	Notes
	Cermák and Skala [2004] and Cermák and Skala [2007]	A	VA		++	+++	+++	MTri. like	Use Curv. estimation
	Cheng et al. [2004]	A	−−	++	+	+	++	Delaunay	
	van Overveld and Wyvill [2004]	A		+	−	+	++	Shrinkwrap	For skeleton-based IS
	de Araújo and Jorge [2005a] and de Araújo and Jorge [2005b]	A	VA		++	+++	+++	MTri. like	Use Hessian Curv.
	Boissonat and Oudot [2005]	A	−−	+++	+	+	++	Delaunay	Use Curv. estimation
	Persson [2005]	A	+	+	+++	++	+++	ReMesh	Particle
	Paiva et al. [2006]	A	≷	++	++	++	++	Dual MC.	Requires C2 function
	Schreiner and Scheidegger [2006]	A	−	++	++	++	++	MTri. like	AFront
	Bouthors and Nesme [2007]	A	≷	−	++	++	++	MC. Remesh.	For skeleton-based IS
	Dey and Levine [2007]	A	≷	++	+	++	++	Delaunay	Curv info: Amenta poles
	Meyer et al. [2007]	A	− − −	++	++	+++	+++	Delaunay	Dynamic particles
	Peiró et al. [2007]	A	+		++	++	+++	MTet. ReMesh.	Laplacian
	Xi and Duan [2008]	A	−	++	++	+++	+++	Delaunay	
	Gois et al. [2008]	A	VA	+	++	≷	≷	Reconstrution	J_1^A triangulation
	Gelas et al. [2009]	A	≷	+	+++	+++	+++	MTri. like	Delaunay

Table II. Exemplar Methods for Each Category

Topology	Features
Stander and Hart [1997]	Vorsatz et al. [2001]
Nielson [2003]	Kobbelt and Botsch [2003]
Nielson [2004]	Ohtake and Belyaev [2002]
Boissonnat and Oudot [2005]	Attene et al. [2003]
Renbo et al. [2005]	Varadhan et al. [2004]
Varadhan et al. [2006]	Varadhan et al. [2006]
	Kazhdan et al. [2007]
	Gelas et al. [2009]
Smoothness	**Mesh Quality**
Akkouche and Galin [2001]	Karkanis and Stewart [2001]
Karkanis and Stewart [2001]	de Araújo and Jorge [2005a]
de Araújo and Jorge [2005a]	Cermák and Skala [2007]
Cermák and Skala [2007]	Ohtake et al. [2001]
Vorsatz et al. [2001]	Ohtake and Belyaev [2002]
Kobbelt and Botsch [2003]	Peiró et al. [2007]
Ohtake and Belyaev [2002]	Vorsatz et al. [2001]
Meyer et al. [2007]	Kobbelt and Botsch [2003]
Xi and Duan [2008]	Meyer et al. [2007]
Gelas et al. [2009]	Gelas et al. [2009]

presenting recommended methods regarding each topic. We would like to stress that this summary does not aim to limit or constrain choice to a particular approach. However, we seek to present exemplar methods addressing a given polygonization requirement according to the research described in this survey.

6.1. Speed Issues

If the primary motivation is for a fast polygonization, we can see that the fastest algorithms belong to the spatial decomposition set of algorithms.

Surface tracking algorithm performance times highly depend on the surface being polygonized. Timings are not consistent, and the algorithm ends when the complete surface is covered by the mesh. Depending on the simplicity of the model, there are numerous surface tracking algorithms that can be as fast as spatial decomposition algorithms. Fewer IS evaluations are needed for simple surfaces, and a better triangle distribution is produced. The main concern regarding surface tracking algorithms is to guarantee that mesh overlap is avoided during expansion. This is achieved using fast collision detection tests.

Algorithm speeds for remeshing approaches depend on the complexity of the mesh and the area to be covered. The number of triangles, or vertices, and the quality of the initial mesh are important. A full spatial decomposition does not always need to be performed before applying remeshing techniques. Traditional spatial subdivision requires a standard, surface-wide level of detail to identify small features of the surface. Some remeshing approaches use a lower-resolution subdivision and target specific areas of the surface. Therefore, some approaches may have comparable timing results to spatial decomposition techniques [Neugebauer and Klein 1997; Ohtake and Belyaev 2002].

6.2. Topology Issues

Topology concerns are particularly important for isosurfaces extracted from scan data. Polygonization algorithms cannot solely guarantee the isotopy between an IS and its polygonal representation. The most successful spatial subdivision algorithms concerning topology have been specifically designed with scan data in mind [Bloomenthal

1994; Montani et al. 1994b; Chernyaev 1995; Hege et al. 1997; Lopes and Brodlie 2003; Nielson 2003; Renbo et al. 2005]. Wood et al. [2002] present the best remeshing technique, although it does not provide any guarantees.

Topological guarantees are mostly offered by methods studying the critical points of the IS [Stander and Hart 1997; Bottino et al. 1996]. Regarding mesh generation for isosurfaces, this problem is overcome by using space decomposition improvements to deal with the topological ambiguities [Lewiner et al. 2003; Nielson 2003, 2004; Renbo et al. 2005; Raman and Wenger 2008; Dietrich et al. 2009b].

When considering topology, our analysis has also included the potential for algorithms to identify nonconnected elements. Some spatial decomposition techniques [Wyvill et al. 1986; Bloomenthal 1994; Lorensen and Cline 1987] depend on a seed cell that intersects the surface to start the polygonization process. They cannot identify disconnected surfaces given volume data, but they can when searching from every implicit field generator, such as when using implicit skeletons. For an in-depth discussion on this point, please see Bajaj et al. [1999].

In addition, some surface tracking techniques use the same principle as far as identifying a point on the surface to start the calculation, and also do not correctly identify all topological concerns [Hilton et al. 1996; Cermák and Skala 2002; Akkouche and Galin 2001; Karkanis and Stewart 2001]. The topological basis used by the Delaunay-based methods [Boissonnat and Oudot 2005; Cheng et al. 2004; Dey and Levine 2007] could be used to overcome these issues.

6.3. Sharp Features

Implicit polygonization techniques often miss important qualities of the surface, particularly sharp features. Some methods from both spatial subdivision and adaptive mesh approaches are designed specifically to identify and adapt the polygonization of features. Spatial subdivision techniques were extended to specifically identify sharp features, and some techniques have excellent results [Chernyaev 1995; Wyvill and van Overveld 1996; Lewiner et al. 2003; Lopes and Brodlie 2003; Nielson 2003; Varadhan et al. 2003, 2004, 2006; Kazhdan et al. 2007]. Remeshing techniques are often designed specifically to improve sharp feature approximation [Vorsatz et al. 2001; Ohtake and Belyaev 2002; Kobbelt and Botsch 2003; Attene et al. 2003]. From the surface tracking approaches,only Cermák and Skala [2007] and Gelas et al. [2009] propose an adaption to improve sharp features approximation.

6.4. Smoothness

Spatial decomposition techniques and regular meshing methods are rarely concerned with the curvature of the surface and representing it by varying triangle sizes. Adaptive meshing techniques are precisely motivated by capturing local shape characteristics. Generation of polygons is guided by curvature estimation. Improved techniques are designed with curvature as a motivation for polygonal construction [Akkouche and Galin 2001; Karkanis and Stewart 2001; Vorsatz et al. 2001; Kobbelt and Botsch 2003; McCormick and Fisher 2002; Ohtake and Belyaev 2002; de Araújo and Jorge 2005a; Cermák and Skala 2007].

Most methods approximate curvature rather than calculate it from the Hessian matrix of the implicit function. Although the Hessian matrix can produce robust curvature values, the cost is differential geometry calculations, which are as expensive as an implicit evaluation [de Araújo and Jorge 2005a]. Most correct topological approximation of IS relies on critical point analysis [Stander and Hart 1997; van Overveld and Wyvill 2004; Boissonnat and Oudot 2005]. Several approaches have used Hessian-based curvature analysis for other purposes, such as feature line extraction [Pasko et al. 1988; Bogaevski et al. 2003; Ohtake et al. 2004].

6.5. Mesh Quality

Referencing Tables I and II, we can highlight two classes of techniques that produce the best polygonal approximations in term of mesh quality. These are surface tracking algorithms [Karkanis and Stewart 2001; Akkouche and Galin 2001; McCormick and Fisher 2002; de Araújo and Jorge 2005a; Cermák and Skala 2007; Xi and Duan 2008; Gelas et al. 2009] and remeshing techniques over marching tetrahedra or MC triangulations [Ohtake and Belyaev 2002; Vorsatz et al. 2001; Kobbelt and Botsch 2003; Rösch et al. 1997; Neugebauer and Klein 1997; Ohtake et al. 2001; Peiró et al. 2007; Attene et al. 2003]. These algorithms present better adaptive meshes and do not exhibit the triangulation pattern created by cubical space decomposition. It also is possible to produce such quality and reproduce sharp features correctly, as it is done by Gelas et al. [2009]. Surface tracking techniques produce well-defined adaptive meshes and are more suited to use local shape information such as curvature. On the other hand, remeshing techniques are identified to be the most complete methods, allowing the production of an adaptive mesh as well as improving the sharp feature approximation [Vorsatz et al. 2001; Ohtake and Belyaev 2002; Kobbelt and Botsch 2003; Attene et al. 2003]. Finally, particle-based methods combined with Delaunay have shown high-quality meshes in Meyer et al. [2007]. However, this method is slow and does not handle discontinuities correctly.

7. CONCLUSIONS

In this survey, we have discussed visualization methods for IS with a focus on fast approaches. Although ray tracing is a high-fidelity and direct visualization technique that produces the most faithful representations of IS, it is also the slowest and generally not appropriate for fast interactive visualization. Stylized representations using NPR and particle systems are applicable in specific circumstances, such as illustrative visualization, but are not widely used for general purposes.

The most common techniques for fast visualization are based on polygonization, supported by current graphics hardware focusing on processing of triangles. Thus, polygonization techniques strive to maintain an acceptable balance between generation speed and mesh accuracy as a representation of the IS. Therefore, we characterized the techniques by their primary motivation: either speed of visualization or reusability of the generated mesh.

IS polygonization algorithms have focused on improving conversion to a linear piecewise representation. In this article, we have discussed issues related to visualization, such as topological correctness, feature sensitivity, smoothness, and visualization or conversion quality.

Polygonal approaches are able to satisfy concerns related to creating an accurate surface representation. First, they can guarantee topological correctness and create a high-quality approximation of surface features such as edges and corners. They can also create adaptive representations that conform to local shape features such as curvature. The quality of generated meshes is generally good and can be used for alternative representations by several applications (i.e., in the scope of computer graphics or scientific simulation).

We have presented the results of this investigation to help in choosing appropriate algorithms for specific purposes. Table I presents our findings for comparison and evaluation purposes when the primary concern is either speed or mesh quality.

From the existing methods, spatial decomposition is generally considered the fastest, whereas surface tracking and post-remeshing techniques result in the most usable representations. Although remeshing specific areas of the surface can produce adaptive meshes quickly, these come at the cost of an extra processing step. Surface tracking

is the most appropriate strategy, which uses local implicit information during mesh generation to create a polygonal representation in a single step.

GPU-based polygonizers have already been proposed [Hansen and Hinker 1992; Johansson and Carr 2006b; Kipfer and Westermann 2005].As for future polygonizers, these can take advantage of high-throughput processing of many parallel operations [Shirazian et al. 2012], since graphics hardware is moving away from special-purpose architectures and toward multicore processing with large cache memories. Thus, new interest and research will likely come from the granularity of future hardware. Eventually, one can foresee a time when there will be no need for polygonizers, as direct visualization methods, such as ray tracing, will prove to be fast enough. More recent approaches [Gomes et al. 2010] based on interval approximations and purely numerical methods may, however, yield practical techniques to render and interact with complexor ill-defined surfaces.

REFERENCES

CGAL. 2014. The Computational Geometry Algorithms Library Home Page. Retrieved April 1, 2015, from http://www.cgal.org.

ImplicitMesher. 2014. ImplicitMesher Home Page. Retrieved April 1, 2015, from http://www.dgp.toronto.edu/~rms/software/ImplicitMesher/index.html.

VTK. 2014. VTK Home Page. Retrieved April 1, 2015, from http://www.vtk.org/.

Samir Akkouche and Eric Galin. 2001. Adaptive implicit surface polygonization using marching triangles. *Computer Graphics Forum* 20, 2, 67–80.

Lionel Alberti, Georges Comte, and Bernard Mourrain. 2005. Meshing implicit algebraic surfaces: The smooth case. In *Mathematical Methods for Curves and Surfaces: Tromso '04,* L. L. Schumaker, M. Maehlen, and K. Morken (Eds.). Nashboro, 11–26.

Rémi Allègre, Eric Galin, Raphaëlle Chaine, and Samir Akkouche. 2006. The HybridTree: Mixing skeletal implicit surfaces, triangle meshes and point sets in a free-form modeling system. *Graphical Models* 1, 42–64. http://liris.cnrs.fr/publis/?id=1946.

Marco Attene, Bianca Falcidieno, Jarek Rossignac, and Michela Spagnuolo. 2003. Edge-sharpener: Recovering sharp features in triangulations of non-adaptively re-meshed surfaces. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP'03)*. 62–69.

Sergei Azernikov and Anath Fischer. 2005. Anisotropic meshing of implicit surfaces. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI'05)*. IEEE, Los Alamitos, CA, 94–103. DOI:http://dx.doi.org/10.1109/SMI.2005.5

Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. 1999. Accelerated isocontouring of scalar fields. In *Data Visualization Techniques*. Trends in Software, Vol. 6. John Wiley and Sons, 31–47.

Loic Barthe, Benjamin Mora, Neil Dodgson, and Malcolm Sabin. 2002. Interactive implicit modelling based on C^1 continuous reconstruction of regular grids. *International Journal of Shape Modeling* 8, 2, 99–117.

Alexander G. Belyaev, Alexander A. Pasko, and Tosiyasu L. Kunii. 1998. Ridges and ravines on implicit surfaces. In *Proceedings of the Computer Graphics International Conference (CGI'98)*. 530. DOI:http://dx.doi.org/10.1109/CGI.1998.694306

Alexander G. Belyaev and Elena V. Anoshkina. 2005. Detection of surface creases in range data. In *Mathematics of Surfaces*. Lecture Notes in Computer Science, vol. 3604. Springer, 50–61.

Matthew Berger, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Joshua A. Levine, Andrei Sharf, and Claudio T. Silva. 2014. State of the art in surface reconstruction from point clouds. In *Eurographics 2014—State of the Art Reports*, S. Lefebvre and M. Spagnuolo (Eds.). Eurographics Association. DOI:http://dx.doi.org/10.2312/egst.20141040

James F. Blinn. 1982. A generalization of algebraic surface drawing. *ACM Transactions on Graphics* 1, 3, 235–256. DOI:http://dx.doi.org/10.1145/357306.357310

Jules Bloomenthal. 1988. Polygonization of implicit surfaces. *Computer Aided Geometric Design* 5, 4, 341–355. DOI:http://dx.doi.org/10.1016/0167-8396(88)90013-1

Jules Bloomenthal. 1994. An implicit surface polygonizer. In *Graphics Gems IV*, Paul Heckbert (Ed.). Academic Press, Boston, MA, 324–349.

Jules Bloomenthal. 1995. *Modelling Natural Forms*. Ph.D. Dissertation. University of Calgary.

Jules Bloomenthal and Chandrajit Bajaj. 1997. *Introduction to Implicit Surfaces*. Morgan Kaufmann, San Francisco, CA.

Jules Bloomenthal and Keith Ferguson. 1995. Polygonization of non-manifold implicit surfaces. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'95)*. ACM, New York, NY, 309–316. DOI:http://dx.doi.org/10.1145/218380.218462

Ilia Bogaevski, Veronique Lang, Alexander Belyaev, and Tosiyasu L. Kunii. 2003. Color ridges on implicit polynomial surfaces. In *Proceedings of GraphiCon 2003*. 161–164.

Jean-Daniel Boissonnat and Steve Oudot. 2005. Provably good sampling and meshing of surfaces. *Graphical Models* 67, 5, 405–451. DOI:http://dx.doi.org/10.1016/j.gmod.2005.01.004

Andrea Bottino, Wim Nuij, and Kees Van Overveld. 1996. How to shrinkwrap through a critical point: An algorithm for the adaptive triangulation of iso-surfaces with arbitrary topology. In *Proceedings of Implicit Surfaces '96*. 53–72.

Antoine Bouthors and Matthieu Nesme. 2007. Twinned meshes for dynamic triangulation of implicit surfaces. In *Proceedings of Graphics Interface 2007 (GI'07)*. ACM, New York, NY, 3–9. DOI:http://dx.doi.org/10.1145/1268517.1268521

David J. Bremer and John F. Hughes. 1998. Rapid approximate silhouette rendering of implicit surfaces. In *Proceedings of Implicit Surfaces '98*. 155–164.

Fatih Calakli and Gabriel Taubin. 2011. SSD: Smooth signed distance surface reconstruction. *Computer Graphics Forum* 30, 7, 1993–2002. DOI:http://dx.doi.org/10.1111/j.1467-8659.2011.02058.x

Jonathan C. Carr, Rick K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and T. R. Evans. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01)*. ACM, New York, NY, 67–76. DOI:http://dx.doi.org/10.1145/383259.383266

Martin Cermák and Václav Skala. 2002. Polygonization by the edge spinning. In *Proceedings of the 16th Conference on Scientific Computing (Algoritmy'02)*.

Martin Cermák and Václav Skala. 2004. Adaptive edge spinning algorithm for polygonization of implicit surfaces. In *Proceedings of the Computer Graphics International Conference (CGI'04)*. IEEE, Los Alamitos, CA, 36–43.

Martin Cermák and Vaclav Skala. 2007. Polygonisation of disjoint implicit surfaces by the adaptive edge spinning algorithm of implicit objects. *International Journal of Computer Science Engineering* 3, 1, 45–52. DOI:http://dx.doi.org/10.1504/IJCSE.2007.014464

Shek Ling Chan and Enrico O. Purisima. 1998. A new tetrahedral tesselation scheme for isosurface generation. *Computers and Graphics* 22, 1, 83–90.

Siu-Wing Cheng, Tamal K. Dey, Edgar A. Ramos, and Tathagata Ray. 2004. Sampling and meshing a surface with guaranteed topology and geometry. In *Proceedings of the 20th Annual Symposium on Computational Geometry (SCG'04)*. ACM, New York, NY, 280–289. DOI:http://dx.doi.org/10.1145/997817.997861

Evgeni Chernyaev. 1995. *Marching Cubes 33: Construction of Topologically Correct Isosurfaces*. Technical Report Technical Report CN/95-17. CERN, Geneva, Switzerland. http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz.

L. Paul Chew. 1993. Guaranteed-quality mesh generation for curved surfaces. In *Proceedings of the 9th Annual Symposium on Computational Geometry (SCG'93)*. ACM, New York, NY, 274–280. DOI:http://dx.doi.org/10.1145/160985.161150

Benoit Crespin. 2002. Dynamic triangulation of variational implicit surfaces using incremental Delaunay tetrahedralization. In *Proceedings of the IEEE Symposium on Volume Visualization and Graphics (VVS'02)*. IEEE, Los Alamitos, CA, 73–80.

Alvaro Cuno, Claudio Esperanca, Antonio Oliveira, and Paulo Roma Cavalcanti. 2004. Fast polygonization of variational implicit surfaces. In *Proceedings of the XVII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'04)*. IEEE, Los Alamitos, CA, 258–265.

Bruno Rodrigues de Araújo and Joaquim Armando Pires Jorge. 2005a. Adaptive polygonization of implicit surfaces. *Computers and Graphics* 29, 5, 686–696. http://dblp.uni-trier.de/db/journals/cg/cg29.html#AraujoJ05.

Bruno Rodrigues de Araújo and Joaquim Armando Pires Jorge. 2005b. A calligraphic interface for interactive free-form modeling with large datasets. In *Proceedings of the XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'05)*. IEEE, Los Alamitos, CA, 333. DOI:http://dx.doi.org/10.1109/SIBGRAPI.2005.2

Luiz Henrique de Figueiredo, Jonas de Miranda Gomes, Demetri Terzopoulos, and Luiz Velho. 1992. Physically-based methods for polygonization of implicit surfaces. In *Proceedings of the Conference on Graphics Interface '92*. 250–257.

Mathieu Desbrun, Nicolas Tsingos, and Marie-Paule Gascuel. 1995. Adaptive sampling of implicit surfaces for interactive modeling and animation. In *Proceedings of Implicit Surfaces '95*.

Tamal K. Dey and Joshua A. Levine. 2007. Delaunay meshing of isosurfaces. In *Proceedings of the Shape Modeling International Conference*. 241–250.

Carlos A. Dietrich, Carlos E. Scheidegger, João L. D. Comba, Luciana P. Nedel, and Cláudio T. Silva. 2009a. Marching cubes without skinny triangles. *Computing in Science and Engineering* 11, 2, 82–87. DOI: http://dx.doi.org/10.1109/MCSE.2009.34

Carlos A. Dietrich, Carlos E. Scheidegger, John Schreiner, João L. D. Comba, Luciana P. Nedel, and Cláudio T. Silva. 2009b. Edge transformations for improving mesh quality of marching cubes. *IEEE Transactions on Visualization and Computer Graphics* 15, 1, 150–159. DOI: http://dx.doi.org/10.1109/TVCG.2008.60

Kevin Foster, Pauline Jepp, Brian Wyvill, Mario Sousa, Callum Galbraith, and Joaquim Jorge. 2005. Pen-and-ink for BlobTree implicit models. *Computer Graphics Forum* 24, 3, 267–276.

Pascal Jean Frey and Paul-Louis George. 2010. *Meshing Implicit Curves and Surfaces*. ISTE, London, UK. DOI: http://dx.doi.org/10.1002/9780470611166.ch16

Eric Galin and Samir Akkouche. 2000. Incremental polygonization of implicit surfaces. *Graphical Models* 62, 1, 19–39.

Arnaud Gelas, Sébastien Valette, Rémy Prost, and Wieslaw L. Nowinski. 2009. Technical section: Variational implicit surface meshing. *Compututers and Graphics* 33, 3, 312–320. DOI: http://dx.doi.org/10.1016/j.cag.2009.03.016

Joao P. Gois, Valdecir Polizelli-Junior, Tiago Etiene, Eduardo Tejada, Antonio Castelo, Luis G. Nonato, and Thomas Ertl. 2008. Twofold adaptive partition of unity implicits. *Visual Computer* 24, 12, 1013–1023. DOI: http://dx.doi.org/10.1007/s00371-008-0297-x

Ron Goldman. 2005. Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design* 22, 7, 632–658. DOI: http://dx.doi.org/10.1016/j.cagd.2005.06.005

Abel Gomes, Irina Voiculescu, Joaquim Jorge, Brian Wyvill, and Callum Galbraith. 2009. *Implicit Curves and Surfaces: Mathematics, Data Structures and Algorithms*. Springer.

Abel J. P. Gomes, Sérgio Dias, and José F. M. Morgado. 2010. Polygonization of non-homogeneous non-manifold implicit surfaces with tentative topological guarantees.. In *Proceedings of the IEEE Congress on Evolutionary Computation*. IEEE, Los Alamitos, CA, 1–8. http://dblp.uni-trier.de/db/conf/cec/cec2010.html#GomesDM10.

Alfred Gray. 1996. *Modern Differential Geometry of Curves and Surfaces with Mathematica*. CRC Press, Boca Raton, FL.

Markus Hadwiger, Joe M. Kniss, Christof Rezk-salama, Daniel Weiskopf, and Klaus Engel. 2006. *Real-Time Volume Graphics*. A. K. Peters, Ltd., Natick, MA.

Mark Hall and Joe Warren. 1990. Adaptive polygonalization of implicitly defined surfaces. *IEEE Computer Graphics and Applications* 10, 6, 33–42. DOI: http://dx.doi.org/10.1109/38.62694

Charles D. Hansen and Paul Hinker. 1992. Massively parallel isosurface extraction. In *Proceedings of the 3rd Conference on Visualization '92 (VIS'92)*. 77–83. DOI: http://dx.doi.org/10.1109/VISUAL.1992.235223

John C. Hart. 1997. Morse theory for implicit surface modeling. In *Mathematical Visualization*. Springer-Verlag, 257–268.

Erich Hartmann. 1998. A marching method for the triangulation of surfaces. *Visual Computer* 14, 2, 95–108.

Paul S. Heckbert and Michael Garland. 1999. Optimal triangulation and quadric-based surface simplification. *Computational Geometry: Theory and Applications* 14, 1–3, 49–65. DOI: http://dx.doi.org/10.1016/S0925-7721(99)00030-9

Hans-Christian Hege, Martin Seebas, Detlev Stalling, and Malte Zockler. 1997. *A Generalized Marching Cubes Algorithm Based on Non-Binary Classifications*. Technical Report.

Adrian Hilton and John Illingworth. 1997. *Marching Triangles: Delaunay Implicit Surface Triangulation*. Technical Report 01. University of Surrey, Guidford, CVSSP.

Adrian Hilton, Andrew J. Stoddart, John Illingworth, and Terry Windeatt. 1996. Marching triangles: Range image fusion for complex object modelling. In *Proceedings of the International Conference on Image Processing*. 381–384.

Chien-Chang Ho, Fu-Che Wu, Bing-Yu Chen, Yung-Yu Chuang, and Ming Ouhyoung. 2005. Cubical marching squares: Adaptive feature preserving surface extraction from volume data. *Computer Graphics Forum* 24, 3, 537–545.

Kin Chuen Hui and Z. H. Jiang. 1999. Tetrahedra based adaptive polygonization of implicit surface patches. *Computer Graphics Forum* 18, 1, 57–68.

Pauline Jepp. 2007. *Using MAS for Illustrative Rendering of Implicit Surfaces*. Ph.D. Dissertation. University of Calgary, Calgary, Alberta.

Pauline Jepp, Jörg Denzinger, Brian Wyvill, and Mario Costa Sousa. 2008. Using multi-agent systems for sampling and rendering implicit surfaces. In *Proceedings of the Brazilian Symposium on Computer Graphics and Image Processing*. 255–262. DOI: http://dx.doi.org/10.1109/SIBGRAPI.2008.18

Gunnar Johansson and Hamish Carr. 2006a. Accelerating marching cubes with graphics hardware. In *Proceedings of the Conference of the Center for Advanced Studies on Collaborative Research (CASCON'06)*. ACM, New York, NY, 39. DOI:http://dx.doi.org/10.1145/1188966.1189018

Gunnar Johansson and Hamish Carr. 2006b. Accelerating marching cubes with graphics hardware. In *Proceedings of the Conference of the Center for Advanced Studies on Collaborative Research (CASCON'06)*. ACM, New York, NY, Article No. 39. DOI:http://dx.doi.org/10.1145/1188966.1189018

Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. 2002. Dual contouring of hermite data. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'02)*. ACM, New York, NY, 339–346. DOI:http://dx.doi.org/10.1145/566570.566586

Devendra Kalra and Alan H. Barr. 1989. Guaranteed ray intersections with implicit surfaces. In *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'89)*. ACM, New York, NY, 297–306. DOI:http://dx.doi.org/10.1145/74333.74364

Takashi Kanai, Yutaka Ohtake, Hiroaki Kawata, and Kiwamu Kase. 2006. GPU-based rendering of sparse low-degree implicit surfaces. In *Proceedings of the 4thInternational Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia (GRAPHITE'06)*. ACM, New York, NY, 165–171. DOI:http://dx.doi.org/10.1145/1174429.1174455

Tasso Karkanis and A. James Stewart. 2001. Curvature-dependent triangulation of implicit surfaces. *IEEE Computer Graphics and Applications* 21, 2, 60–69. DOI:http://dx.doi.org/10.1109/38.909016

Michael Kazhdan, Allison Klein, Ketan Dalal, and Hugues Hoppe. 2007. Unconstrained isosurface extraction on arbitrary octrees. In *Proceedings of the 5th Eurographics Symposium on Geometry Processing (SGP'07)*. 125–133. http://dl.acm.org/citation.cfm?id=1281991.1282009

Peter Kipfer and Rüdiger Westermann. 2005. GPU construction and transparent rendering of iso-surfaces. In *Proceedings of Vision, Modeling, and Visualization 2005*. 241–248.

Aaron Knoll. 2009. *Ray Tracing Implicit Surfaces for Interactive Visualization*. Ph.D. Dissertation. University of Utah.

Aaron Knoll, Ingo Wald, Steven G. Parker, and Charles D. Hansen. 2006. Interactive isosurface ray tracing of large octree volumes. In *Proceedings of the IEEE Symposium on Interactive Ray Tracing*. 115–124.

Leif Kobbelt and Mario Botsch. 2003. Feature sensitive mesh processing. In *Proceedings of the 19th Spring Conference on Computer Graphics (SCCG'03)*. ACM, New York, NY, 17–22. DOI:http://dx.doi.org/10.1145/984952.984956

Leif Kobbelt and Mario Botsch. 2004. A survey of point-based techniques in computer graphics. *Computers and Graphics* 28, 6, 801–814. DOI:http://dx.doi.org/10.1016/j.cag.2004.08.009

Leif P. Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. 2001. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01)*. ACM, New York, NY, 57–66. DOI:http://dx.doi.org/10.1145/383259.383265

Jan J. Koenderink. 1990. *Solid Shape*. MIT Press, Cambridge, MA.

Florian Levet, Julien Hadim, Patrick Reuter, and Christophe Schlick. 2005. Anisotropic sampling for differential point rendering of implicit surfaces. In *Proceedings of the Winter School of Computer Graphics Conference (WSCG'05)*. 109–116.

Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. 2003. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of Graphics Tools* 8, 1–15. http://cgal.inria.fr/Publications/2003/LLVT03.

Shengjun Liu, Xuehui Yin, Xiaogang Jin, and Jieqing Feng. 2005. High quality triangulation of implicit surfaces. In *Proceedings of the 9th International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*. IEEE, Los Alamitos, CA, 133–138. DOI:http://dx.doi.org/10.1109/CAD-CG.2005.46

Adriano Lopes and Ken Brodlie. 2003. Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing. *IEEE Transactions on Visualization and Computer Graphics* 9, 1, 16–29. DOI:http://dx.doi.org/10.1109/TVCG.2003.1175094

Daniel S. Lopes, Mauro T. Silva, and Jorge A. Ambrósio. 2013. Tangent vectors to a 3-D surface normal: A geometric tool to find orthogonal vectors based on the Householder transformation. *Computer-Aided Design* 45, 3, 683–694. DOI:http://dx.doi.org/10.1016/j.cad.2012.11.003

William E. Lorensen and Harvey E. Cline. 1987. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'87)*. ACM, New York, NY, 163–169. DOI:http://dx.doi.org/10.1145/37401.37422

Sergey V. Matveyev. 1994. Approximation of isosurface in the marching cube: Ambiguity problem. In *Proceedings of the Conference on Visualization '94 (VIS'94)*. IEEE, Los Alamitos, CA, 288–292. http://portal.acm.org/citation.cfm?id=951087.951140

Neil H. McCormick and Robert B. Fisher. 2002. *Edge-Constrained Marching Triangles*. Technical Report EDI-INF-RR-0188. Division of Informatics, University of Edinburgh.

Miriah Meyer, Robert M. Kirby, and Ross Whitaker. 2007. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE Transactions on Visualization and Computer Graphics* 13, 6, 1704–1711. DOI:http://dx.doi.org/10.1109/TVCG.2007.70604

Claudio Montani, Riccardo Scateni, and Robert Scopigno. 1994a. Discretized marching cubes. In *Proceedings of the Conference on Visualization '94 (VIS'94)*. IEEE, Los Alamitos, CA, 281–287.

Claudio Montani, Riccardo Scateni, and Roberto Scopigno. 1994b. A modified look-up table for implicit disambiguation of marching cubes. *Visual Computer* 10, 6, 353–355. http://www.crs4.it/vic/cgi-bin/bib-page.cgi?id='Montani:1994:MLT'

Doug Moore and Joe Warren. 1995. *Mesh Displacement: An Improved Contouring Method for Trivariate Data*. Technical Report.

Bryan S. Morse, Terry S. Yoo, David T. Chen, Penny Rheingans, and Kalpathi R. Subramanian. 2001. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Proceedings of the SMI International Conference on Shape Modeling and Applications (SMI'01)*. 89–98. DOI:http://dx.doi.org/10.1109/SMA.2001.923379

Shigeru Muraki. 1991. Volumetric shape description of range data using Blobby model. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'91)*. ACM, New York, NY, 227–235. DOI:http://dx.doi.org/10.1145/122718.122743

Balas K. Natarajan. 1994. On generating topologically consistent isosurfaces from uniform samples. *Visual Computer* 11, 1, 52–62. DOI:http://dx.doi.org/10.1007/BF01900699

Peter Neugebauer and Konrad Klein. 1997. Adaptive triangulation of objects reconstructed from multiple range images. In *Proceedings of the Conference on Visualization '97 (VIS'97)*.

Gregory M. Nielson. 2003. On marching cubes. *IEEE Transactions on Visualization and Computer Graphics* 9, 3, 283–297. DOI:http://dx.doi.org/10.1109/TVCG.2003.1207437

Gregory M. Nielson. 2004. Dual marching cubes. In *Proceedings of the Conference on Visualization '04 (VIS'04)*. IEEELos Alamitos, CA, 489–496. DOI:http://dx.doi.org/10.1109/VIS.2004.28

Gregory M. Nielson and Bernd Hamann. 1991. The asymptotic decider: Resolving the ambiguity in marching cubes. In *Proceedings of the 2nd Conference on Visualization '91 (VIS'91)*. IEEE, Los Alamitos, CA, 83–91. http://dl.acm.org/citation.cfm?id=949607.949621.

Paul Ning and Jules Bloomenthal. 1993. An evaluation of implicit surface tilers. *IEEE Computer Graphic and Applications* 13, 6, 33–41. DOI:http://dx.doi.org/10.1109/38.252552

Hiromitsu Nishimura, Masashi Hirai, Tsuyoshi Kawai, Tory Kawata, Isao Shirakawa, and Kengo Omura. 1985. Object modelling by distribution function and a method of image generation. *Transactions of the IEICE Japan* J68-D, 4, 718–725.

Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. 2003. Multi-level partition of unity implicits. In *ACM SIGGRAPH 2003 Papers*. ACM, New York, NY, 463–470. DOI:http://dx.doi.org/10.1145/1201775.882293

Yutaka Ohtake, Alexander Belyaev, and Alexander Pasko. 2001. Dynamic meshes for accurate polygonization of implicit surfaces with sharp features. In *Proceedings of the SMI International Conference on Shape Modeling and Applications*. 74–81. DOI:http://dx.doi.org/10.1109/SMA.2001.923377

Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. 2004. Ridge-valley lines on meshes via implicit surface fitting. *ACM Transactions on Graphics* 23, 3, 609–612. DOI:http://dx.doi.org/10.1145/1015706.1015768

Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. 2005. 3D scattered data interpolation and approximation with multilevel compactly supported RBFs. *Graphical Models* 67, 3, 150–165. DOI:http://dx.doi.org/10.1016/j.gmod.2004.06.003

Yutaka Ohtake and Alexander G. Belyaev. 2002. Dual/primal mesh optimization for polygonized implicit surfaces. In *Proceedings of the 7th ACM Symposium on Solid Modeling and Applications (SMA'02)*. ACM, New York, NY, 171–178. DOI:http://dx.doi.org/10.1145/566282.566308

Stanley Osher and Ronald P. Fedkiw. 2003. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, New York, NY.

Stanley Osher and James A. Sethian. 1988. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics* 79, 1, 12–49.

Afonso Paiva, Hélio Lopes, Thomas Lewiner, and Luiz Henrique de Figueiredo. 2006. Robust adaptive meshes for implicit surfaces. In *Proceedings of the 19th Brazilian Symposium on Computer Graphics and Image Processing*. 205–212. http://www.mat.puc-rio.br/~tomlew/adaptive_implicit_sibgrapi.pdf.

Alexander Pasko, Valery Adzhiev, Alexei Sourin, and Vladimir Savchenko. 1995. Function representation in geometric modeling: Concepts, implementation and applications. *Visual Computer* 11, 8, 429–446.

Alexander A. Pasko, Victor V. Pilyugin, and V. N. Pokrovsky. 1988. Geometric modeling in the analysis of trivariate functions. *Computers and Graphics* 12, 3–4, 457–465.

Joaquim Peiró, Luca Formaggia, Mattia Gazzola, Alessandro Radaelli, and V. Rigamonti. 2007. Shape reconstruction from medical images and quality mesh generation via implicit surfaces. *International Journal for Numerical Methods in Fluids* 53, 8, 1339–1360. DOI:http://dx.doi.org/10.1002/fld.1362

Per-Olof Persson. 2005. *Mesh Generation for Implicit Geometries*. Ph.D. Dissertation. Massachusetts Institute of Technology, Cambridge, MA.

Per-Olof Persson. 2014. DistMesh—A Simple Mesh Generator in MATLAB. Retrieved April 1, 2015, from http://persson.berkeley.edu/distmesh/.

Simon Plantinga and Gert Vegter. 2004. Isotopic approximation of implicit curves and surfaces. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP'04)*. ACM, New York, NY, 245–254. DOI:http://dx.doi.org/10.1145/1057432.1057465

Simon Plantinga and Gert Vegter. 2007. Isotopic meshing of implicit surfaces. *Visual Computer* 23, 1, 45–58. DOI:http://dx.doi.org/10.1007/s00371-006-0083-6

William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. 1986. *Numerical Recipes—The Art of Scientific Computing*. Cambridge University Press, Cambridge, MA.

Sundaresan Raman and Rephael Wenger. 2008. Quality isosurface mesh generation using an extended marching cubes lookup table. *Computer Graphics Forum* 27, 3, 791–798.

Xia Renbo, Liu Weijun, and Wang Yuechao. 2005. A robust and topological correct marching cube algorithm without look-up table. In *Proceedings of the 5th International Conference on Computer and Information Technology (CIT'05)*. IEEE , Los Alamitos, CA, 565–569. DOI:http://dx.doi.org/10.1109/CIT.2005.44

Patrick Reuter. 2003. *Reconstruction and Rendering of Implicit Surfaces from Large Unorganized Point Sets*. Ph.D. Dissertation. Université Bordeaux 1, France.

Laurent Rineau and Mariette Yvinec. 2007. A generic software design for Delaunay refinement meshing. *Computational Geometry* 38, 1–2, 100–110. DOI:http://dx.doi.org/10.1016/j.comgeo.2006.11.008

Angela Rösch, Matthias Ruhl, and Dietmar Saupe. 1997. Interactive visualization of implicit surfaces with singularities. *Eurographics Computer Graphics Forum* 16, 5, 295–306.

Vladimir L. Rvachev. 1963. On the analytical description of some geometric objects. *Reports of Ukrainian Academy of Sciences* 153, 4, 765–767.

Scott Schaefer, Tao Ju, and Joe Warren. 2007. Manifold dual contouring. *IEEE Transactions on Visualization and Computer Graphics* 13, 3, 610–619. DOI:http://dx.doi.org/10.1109/TVCG.2007.1012

Scott Schaefer and Joe Warren. 2004. Dual marching cubes: Primal contouring of dual grids. In *Proceedings of the 12th Pacific Conference on Computer Graphics and Applications (PG'04)*. IEEE, Los Alamitos, CA, 70–76.

Ryan Schmidt, Brian Wyvill, and Eric Galin. 2005. Interactive implicit modeling with hierarchical spatial caching. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI'05)*. IEEE, Los Alamitos, CA, 104–113. DOI:http://dx.doi.org/10.1109/SMI.2005.25

John Schreiner and Carlos Scheidegger. 2006. High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics* 12, 5, 1205–1212. DOI:http://dx.doi.org/10.1109/TVCG.2006.149

John M. Schreiner, Carlos Eduardo Scheidegger, Shachar Fleishman, and Cláudio T. Silva. 2006. Direct (re)meshing for efficient surface processing. *Computer Graphics Forum* 25, 3, 527–536. http://dblp.uni-trier.de/db/journals/cgf/cgf25.html#SchreinerSFS06.

Vadim Shapiro. 2007. Semi-analytic geometry with R-functions. *Acta Numerica* 16, 239–303. DOI:http://dx.doi.org/10.1017/S096249290631001X

Pourya Shirazian, Brian Wyvill, and Jean-Luc Duprat. 2012. Polygonization of implicit surfaces on multi-core architectures with SIMD instructions. In *Eurographics Symposium on Parallel Graphics and Visualization*, H. Childs, T. Kuhlen, and F. Marton (Eds.). Eurographics Association, 89–98. http://dblp.uni-trier.de/db/conf/egpgv/egpgv2012.html#ShirazianWD12.

Christian Sigg. 2006. *Representation and Rendering of Implicit Surfaces*. Ph.D. Dissertation. ETH Zurich, Zurich, Switzerland.

Cláudio T. Silva, Joo L. D. Comba, Steven P. Callahan, and Fabio F. Bernardon. 2005. A survey of GPU-based volume rendering of unstructured grids. *Brazilian Journal of Theoretic and Applied Computing* 12, 9–29.

Marcelo F. Siqueira, S. R. Freitas, A. C. Filho, and G. Tavares. 1998. Speeding up adaptive polygonization. In *Proceedings of the West Side Computer Graphics Conference (WSCG'98)*. 11–12.

Barton T. Stander and John C. Hart. 1995. Interactive re-polygonization of blobby implicit curves. In *Proceedings of the Western Computer Graphics Symposium*.

Barton T. Stander and John C. Hart. 1997. Guaranteeing the topology of an implicit surface polygonization for interactive modeling. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'97)*. ACM, New York, NY, 279–286. DOI:http://dx.doi.org/10.1145/258734.258868

Wen Y. Su and John C. Hart. 2005. A programmable particle system framework for shape modelling. In *Proceedings of the International Conference on Shape Modeling and Applications*. 114–123.

Gabriel Taubin. 2012. Smooth signed distance surface reconstruction and applications. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Lecture Notes in Computer Science, Vol. 7441. Springer, 38–45. DOI:http://dx.doi.org/10.1007/978-3-642-33275-3_4

Ireneusz Tobor, Patrick Reuter, and Christophe Schlick. 2006. Reconstructing multi-scale variational partition of unity implicit surfaces with attributes. *Graphical Models* 68, 1, 25–41. DOI:http://dx.doi.org/10.1016/j.gmod.2005.09.003

Graham M. Treece, Richard W. Prager, and Andrew H. Gee. 1999. Regularised marching tetrahedra: Improved iso-surface extraction. *Computers and Graphics* 23, 4, 583–598.

Frederic Triquet, Laurent Grisoni, Philippe Meseure, and Christophe Chaillou. 2003. Realtime visualization of implicit objects with contact control. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques in Australia and South East Asia (GRAPHITE'03)*. http://www.anzgraph.org/graphite2003.

Frederic Triquet, Philippe Meseure, and Christophe Chaillou. 2001. Fast polygonization of implicit surfaces. In *Proceedings of the 2001 WSCG International Conference (WSCG'01)*. 283–290. http://wscg.zcu.cz.

Greg Turk and James F. O'Brien. 1999. Shape transformation using variational implicit functions. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'99)*. ACM, New York, NY, 335–342. DOI:http://dx.doi.org/10.1145/311535.311580

Kees van Overveld and Brian Wyvill. 2004. Shrinkwrap: An efficient adaptive algorithm for triangulating an iso-surface. *Visual Computer* 20, 6, 362–379. DOI:http://dx.doi.org/10.1007/s00371-002-0197-4

Gokul Varadhan, Shankar Krishnan, Young J. Kim, and Dinesh Manocha. 2003. Feature-sensitive subdivision and isosurface reconstruction. In *Proceedings of the 14th IEEE Visualization Conference (VIS'03)*. IEEE, Los Alamitos, CA, 14. DOI:http://dx.doi.org/10.1109/VISUAL.2003.1250360

Gokul Varadhan, Shankar Krishnan, Tvn Sriram, and Dinesh Manocha. 2004. Topology preserving surface extraction using adaptive subdivision. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP'04)*. ACM, New York, NY, 235–244. DOI:http://dx.doi.org/10.1145/1057432.1057464

Gokul Varadhan, Shankar Krishnan, Liangjun Zhang, and Dinesh Manocha. 2006. Reliable implicit surface polygonization using visibility mapping. In *Proceedings of the Eurographics Symposium on Geometry Processing*. 211–221. DOI:http://dx.doi.org/10.2312/SGP/SGP06/211-221

Luiz Velho. 1996. Simple and efficient polygonization of implicit surfaces. *Journal of Graphic Tools* 1, 2, 5–24.

Luiz Velho, Luiz Henrique de Figueiredo, and Jonas Gomes. 1999. A unified approach for hierarchical adaptive tesselation of surfaces. *ACM Transactions on Graphics* 18, 4, 329–360. DOI:http://dx.doi.org/10.1145/337680.337717

Luiz Velho, Jonas Gomes, and Luiz Henrique de Figueiredo. 2002. *Implicit Objects in Computer Graphics*. Springer.

Jens Vorsatz, Christian Rössl, Leif Kobbelt, and Hans-Peter Seidel. 2001. Feature sensitive remeshing. *Computer Graphics Forum* 20, 3, 393–401.

Andrew P. Witkin and Paul S. Heckbert. 1994. Using particles to sample and control implicit surfaces. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'94)*. ACM, New York, NY, 269–277. DOI:http://dx.doi.org/10.1145/192161.192227

Zoe Wood, Hugues Hoppe, Mathieu Desbrun, and Peter Schroder. 2002. *Isosurface Topology Simplification*. Technical Report MSR-TR-2002-28. Microsoft Research. http://www.labri.fr/publications/is/2004/RJTBS04.

Brian Wyvill, Eric Galin, and Andrew Guy. 1998. The blob tree, warping, blending and Boolean operations in an implicit surface modeling system. In *Proceedings of Implicit Surfaces '98*.

Brian Wyvill and Kees van Overveld. 1996. Polygonization of implicit surfaces with constructive solid geometry. *Journal of Shape Modelling* 2, 4, 257–274.

Geoff Wyvill, Craig McPheeters, and Brian Wyvill. 1986. Data structure for soft objects. *Visual Computer* 2, 4, 227–234.

Yongjian Xi and Ye Duan. 2008. CAD and graphics: A novel region-growing based iso-surface extraction algorithm. *Computers and Graphics* 32, 6, 647–654. DOI: http://dx.doi.org/10.1016/j.cag.2008.09.007

Shuntaro Yamazaki, Kiwamu Kase, and Katsushi Ikeuchi. 2002. Non-manifold implicit surfaces based on discontinuous implicitization and polygonization. In *Proceedings of the Geometric Modeling and Processing Conference*. 138. DOI: http://dx.doi.org/10.1109/GMAP.2002.1027505

Yi Zhang, Xin Wang, and Xiao Jun Wu. 2006. Fast visualization algorithm for implicit surfaces. In *Proceedings of the 16th International Conference on Artificial Reality and Telexistence (ICAT'06).* 339–344. DOI: http://dx.doi.org/10.1109/ICAT.2006.62

Received December 2013; revised December 2014; accepted February 2015