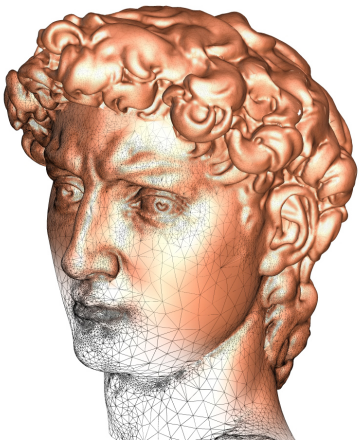
UNIVERSIDADE TÉCNICA DE LISBOA
INSTITUTO SUPERIOR TÉCNICO



Curvature Dependent Polygonization of Implicit Surfaces

Bruno Rodrigues De Araújo
(Licenciado)

Dissertation for the degree of Master of Science in
Information Systems and Computer Engineering

Adviser:	Doutor Joaquim Armando Pires Jorge
Chairman:	Doutor Joaquim Armando Pires Jorge
Members:	Doutor Abel João Padrão Gomes Doutor João António Madeiras Pereira

February 2008

Curvature Dependent Polygonization of Implicit Surfaces

Bruno Rodrigues De Araújo

A Thesis submitted to the Graduate School
for the degree of

Master of Science in
Information Systems and Computer Engineering

Department of

Information Systems and Computer Engineering

Instituto Superior Técnico

Adviser

Prof. Doutor Joaquim Armando Pires Jorge

Professor Auxiliar com Agregação from the
Department of Information Systems and Computer Engineering
Instituto Superior Técnico
Technical University of Lisbon

http://immi.inesc-id.pt/~brar

brar@immi.inesc-id.pt

Resumo

As superfícies implícitas oferecem uma representação de qualidade para dados médicos e nuvens de pontos obtidas por dispositivos de digitalização laser devido a sua flexibilidade de representar formas livres. No entanto, a sua visualização em tempo real é difícil e necessitam ser amostradas e convertidas para malhas poligonais para serem visualizadas interactivamente. Nesta dissertação, propomos uma nova abordagem de geração de malha adaptativa reproduzindo da melhor forma a curvatura de formas contínuas. O nosso algoritmo tenta acompanhar a superfície em vez de amostra-la por meio de decomposição espacial. A malha triangular é criada expandindo pontos próximo da superfície até cobrir toda a forma. Utilizando os valores da função implícita tais como os gradientes e as matrizes Hessianas, o tamanho dos triângulos produzidos pelo o algoritmo é afectado pela curvatura extraída da superfície. Uma aproximação poligonal de alta qualidade é produzida num único passo sem a necessidade de pós-processamento para seu embelezamento. Propomos soluções robustas e eficientes de forma a lidar com possíveis instabilidades numéricas e sobreposições indesejada de triângulos. A nossa abordagem foi testada com três representações implícitas usando conjuntos de dados não triviais de forma a gerar uma aproximação mais fidedigna que os algoritmos existentes sem sacrificar recursos computacionais excessivos.

Palavras Chave

Visualização de Superfícies Implícitas

Poligonização de Superfícies Implícitas

Triangulação baseada em Curvatura

Técnicas de Seguimento de Superfícies

Geração de Malhas Adaptativas

Processamento de Malhas e Geometria

Abstract

Implicit surfaces provide a solution of choice for imaging medical information or point sets arising from digitizing complex models thanks to their flexibility in manipulating data. However, fast visualization of implicit surfaces is difficult and they need to be sampled and converted to polygonal meshes in order to be visualized interactively. In this dissertation, we propose a novel polygonization approach which generates an adaptive polygonal mesh to better reproduce the curvature of continuous shapes. Our algorithm follows a surface tracking strategy instead of sampling the shape by space decomposition. The triangular mesh is created expanding points lying on the surface until all the shape is covered. Using implicit values, gradient vectors and Hessian matrixes extracted from the implicit function, our algorithm produces triangles which edges are heuristically dependent of the implicit surface curvature. Our high quality approximation is created on the fly without the need of any post-remeshing beautification step. We propose robust and efficient steps to overcome numerical instability and to avoid mesh overlapping. Our approach was tested with three implicit representations using non-trivial data sets generating a smoother and more feature sensitive approximation than existing polygonization algorithms without sacrificing computing resources.

Keywords

Implicit Surface Visualization

Polygonization of Implicit Surfaces

Curvature Dependent Triangulation

Surface Tracking Technique

Adaptive Meshing

Geometry Processing

Acknowledgements

I would like to thank my adviser Professor Joaquim Armando Pires Jorge for all the advises during the research related with this dissertation and deep discussions on all the technical aspects of this work sharing his valuable expertise in Computer Graphics.

In second place, I would like to thanks all the members of IMMI group and INESC-ID, to let me be part of dynamic research group always searching for new challenges . I would like also to show my gratitude to all the professors of *Computação Gráfica e Multimédia* of *Instituto Superior Técnico* and their teaching which had deeply influenced and motivated my research interest on Computer Graphics.

Last but not least, I would like to thanks the unconditional support of my Parents, my Godparents and Ana Luísa and their understanding of the importance of the achievement of this dissertation for my personal goals.

Sincèrement Merci à Tous.

Lisboa, February 2008

Bruno Rodrigues De Araujo

Contents

List of Figures

List of Tables

1

Introduction

Implicit Surfaces are a popular mathematical model used in Computer Graphics to represent shapes used for Modelling, Animation, Scientific Simulation and Visualization. Implicit surfaces provide a smoother and compact model requiring few high-level primitives to describe free-form surfaces becoming a suitable alternative to represent data gathered by 3D scan for re-inverse engineering or medical data scientific visualization. Compared to parametric representations, implicit surfaces are easiest to model and bend. They do not pose topological problems and can be described by low-degree functions. Furthermore, its mathematical definition makes implicit surfaces attractive to mix with physical based representation and to simulate natural phenomena such as fluids.

However the major drawback of implicit surfaces is that they are usually harder to display compared to other representations since all advanced graphic hardware is only able to display polygons, especially triangles. The mathematical representation used by implicit surfaces turns them easy to render using ray-tracing algorithms compared to other representations. However real-time ray-tracing with current hardware is still of the domain of super-computers which makes interactive visualization of implicit surfaces using ray tracing still inaccessible. For this purpose, it is important to provide controlled visualization techniques or algorithms to enable it conversion to other representations. Looking to existing graphic hardware, polygonal approximation are the more suited alternative representation to visualize implicit surfaces. However a special care need to be considered when generating a piecewise approximation of a surface since it cannot adequately describe and reproduce curvature information. Polygonization is the name given to the process of converting implicit surfaces into a

triangular piecewise representation.

The Marching Cubes algorithm introduced by Hilton [Hilton 96] and improved by Bloomenthal [Bloomenthal 94] is the most popular polygonization algorithm used by almost all the systems which use implicit surfaces for shape representation. However this method relies on sampling the surface using a cubical spatial subdivision. This strategy, even if efficient, generates unwanted triangular patterns related with the axis aligned subdivision. On the other hand, the density of the sampling is homogeneous which generates meshes with a large number of triangles when we want to approximate surfaces with regions of high curvature variation. Algorithms with the ability to adapt the density of the sampling with local characteristics of the surface are needed to present a more faithful polygonal approximation.

In this dissertation, we present a new polygonization algorithm for implicit surfaces generating an adaptive piecewise approximation. We proposes to use local shape characteristics such as curvature to produce more faithful approximation. Our algorithm tracks the surface expanding the polygonal approximation with new points and triangles on tangent planes of the surface instead of relying on a sampling method based on spatial subdivision. This strategy allows us to control and adapt the triangulation according to local characteristics of the shape extracted from the implicit surface mathematical formulation. The mesh is created continuously by the algorithm resulting in an adaptive triangulation generated on the fly in only one step without any post processing.

1.1 Contributions

The main contribution of our work is an algorithm to create controlled polygonal approximations of implicit surfaces to be used as a faithful representation for interactive visualization purpose.By using a surface tracking approach, we are able to present a polygonal approximation which is able to better approximate high curvature region than popular spatial subdivision approaches such as Marching Cubes and its variants. On the other hand, the good management of the triangle budget according to curvature of shape areas, open implicit surface

reconstruction as a good geometrical processing step to generate high quality meshing of points clouds compared to traditional post re-meshing techniques. The research described in this dissertation has yielded in the following contributions:

One step adaptive mesh generation Our algorithm is able to generate an adaptive mesh using larger triangles in low curvature regions and smaller triangles in high curvature areas on the fly in one step without requiring any post processing re-meshing step.

Support several implicit surfaces representations Our algorithm supports any implicit surface representations which mathematical definition enable to extract implicit values, gradient vectors and hessian matrixes

Robust surface tracking approach We propose a robust approach which is able to overcome the sensitiveness of surface tracking approaches to avoid mesh overlap or support implicit value un-definition. Doing so, we guarantee that the polygonization algorithm is finite in a given volume.

Heuristics using local implicit curvature We propose how local characteristics such as curvature information extracted from derivative analysis of the implicit function can be used to generate an adaptive polygonization. We present several heuristics which can be used to influence the triangle density regarding different type of curvatures.

Support local re-polygonization Our algorithm support local re-polygonization to adapt the mesh when local changes are applied to the implicit surface representation.

1.2 Publications

The research performed for this dissertation yielded in several original publications accepted in peer-reviewed scientific conferences and journal, which are listed reverse chronological order by date of publication.

1. De Araújo B. and Jorge J. *A Calligraphic interface for interactive free-form modeling with large datasets*. Proceedings of the XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2005), Natal, Brazil, Oct 2005

2. De Araújo B. and Jorge J. *Adaptive Polygonization of Implicit Surfaces*. In "State of the Art in Computer Graphics in Ibero-American Countries",Computers and Graphics (CG) journal, vol. 29, nº 5, 2005

3. De Araújo B. and Jorge J. *Curvature Dependent Polygonization of Implicit Surfaces*. Proceedings of the XVII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2004/SIACG 2004), Curitiba, Brazil, Oct 2004

4. De Araújo B., Jorge J., Sousa M., Samavati F. and Wyvill B. *MIBlob: A Tool for Medical Visualization and Modelling using Sketches*. Poster at the 31st International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH2004), Los Angeles, U.S.A., Aug 2004

5. De Araújo B. and Jorge J. *BlobMaker: Free-form modelling with variational implicit surfaces*. Proceedings of 12º Encontro Português de Computação Gráfica (12º EPCG), pages 17-26, Porto, Portugal, Oct 2003

1.3 Dissertation Outline

The dissertation is organized into five chapters. Following this introductory chapter, the outline of the dissertation is structured as following.

Chapter 2 presents as background the several implicit models created along the last two decades, to better understand the characteristic of its mathematical model and its reliability for surface reconstruction. Then we describe the local properties which can be extracted from the implicit surface representation and used as metrics of the shape to generate a good polygonal approximation.

Chapter 3 surveys the several techniques used for implicit surfaces visualization and discuss the several issues related with it visualization, i.e topology

correctness, sharpness and smoothness fidelity and visualization or conversion quality. We focus on the issues related with the improvement of polygonal approximation since it is the approach used by our algorithm. We define comparison criteria for their analysis and establish which are the best approaches depending of the quality researched and the purpose of the visualization.

Chapter 4 describes in detail our polygonization approach to generate the adaptive mesh. Starting from an overview of the algorithm, we describe all the steps followed by our approach to ensure its robustness as a surface tracking method.

Chapter 5 presents the results achieved by our approach considering several complex and large data-sets. A discussion of the several contributions of our work is presented showing its robustness and versatility to generate adaptive polygonal approximation for several implicit models.

Finally we present the conclusions and propose directions for future works to improve the quality of implicit surfaces polygonization and possible paths which can be followed to present faster methods.

2

Implicit Surfaces Mathematics

2.1 Implicit Mathematical Models

Along the last two decades, new implicit models have been created or refined in order to respond to the users needs of several areas such as Architecture, Physics, Botanic and Medicine. By definition, all the Implicit Models used to represent shapes are defined as the zero-set of a function $F : R^3 \rightarrow R$. Implicit Surfaces can be organized into three different classes, the first set is compound by the "blobby" like models such as soft objects and meta-balls. The second set of implicit models are the skeleton-based or globally defined implicit surfaces such as convolutions surfaces, these representations are more targeted for hierarchical and incremental modelling. Finally, the third set of implicit surfaces are the one that are used to approximate a set of data, points or constraints and are suitable to convert any kind of polygonal or volumetric based representation into an implicit surface. The following subsections present an overview of the several implicit models available in each sets, its advantages and limitation.

2.1.1 Basis Implicit Functions

The Blobby molecule [Blinn 82] introduced in 1982 by Blinn, is perhaps considered as the first implicit surface representation into Computer Graphics. Instead of using the traditional implicit quadric representation, Blinn proposes a new representation to model atoms and molecules based on a density function from Quantum mechanics representing the electron in an atom. This implicit model

named "Blobby Model" uses as basis an exponential function defining the density function of the spatial location of electrons such as presented by the following formula:

$$F(x, y, z) = \exp(-ar)$$

where $r = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}$ and (x_i, y_i, z_i) is the location of atom.

The generalized implicit function is a weighted sum of exponentials based on the distance of the electron to the center of the atom allowing to represent several electrons.

$$F(x, y, z) = \sum_i b_i \exp(-a_i r_i^2)$$

Blinn illustrates his model by visualizing 4000 atoms using a ray-tracing algorithm. On the other hand, he presents several simplifications in order to speed-up the ray-tracing visualization such as removing functions in ray intersection computation if their cores are located outside a given range. Finally he proposed hierarchical modelling issues for its implicit model in order to represent large polymerized molecules (like DNA) and extend the generalized implicit function with non-spherical primitives and negative volumes.

Figure 2.1 presents some of the results achieved by Blinn. The typical visualization was based on ray-tracing algorithms by computing the ray-intersection analytically or numerically depending on the number of atoms intersected. Numerical intersections were computed by an hybrid method which mixed Newton steps and Regula Falsi for each iteration based on interval analysis.

Blinn's implicit model was designed to represent molecules, however he presented other applications and discussed rendering issues that could transform implicit models as a reliable 3D shape representation. Following this idea, alternative implicit models for generic shape representation appeared such as the soft objects [Wyvill 86], the popular meta-ball [Nishimura 85] and the blobby model [Muraki 91]. All these implicit models rely on the same generalized implicit function which is a weighted sum of similar basis functions which try to mimic Blinn's exponential density function The formulas of these functions are the following depending of the model.

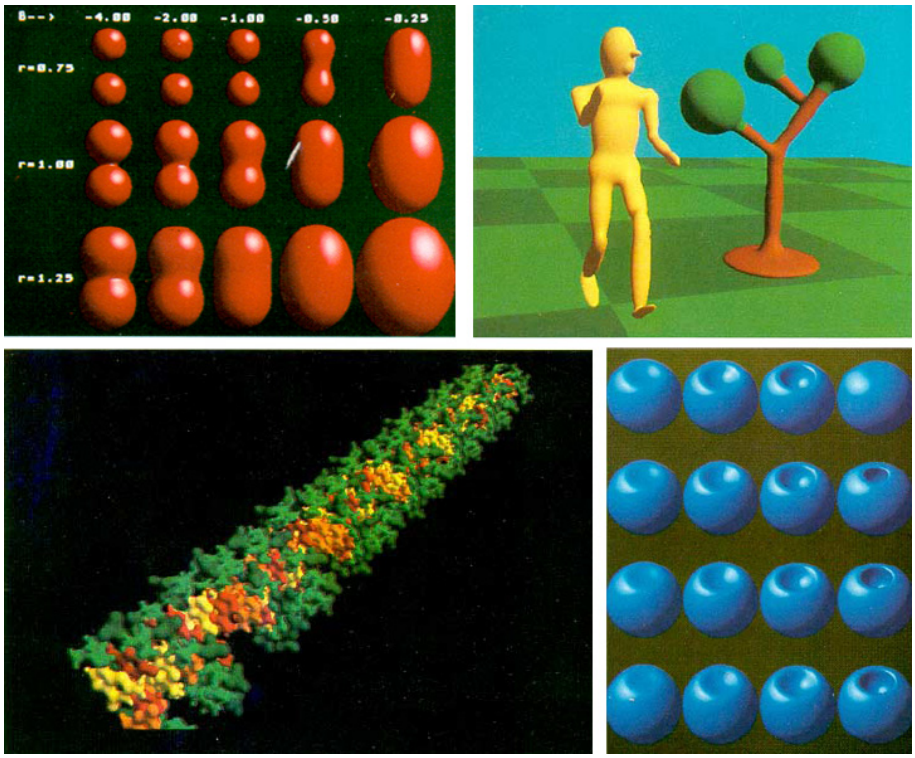


Figure 2.1: Ray-tracing renderings of Blinn implicit shapes

Metaballs:

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

Blobby Model:

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

Soft Object:

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

These models limit the influence of the field basis function which will decay depending of a radius of influence and relies on polynomial function instead of exponential function. These two options provide an easy way to control the shape definition by a finite set of constraint points and the influence of each constraint point. This is done by using basis function which are very cheap to compute compared to the Blinn implicit model avoiding the exponential. It is also possible

to note that the degree of the polynomial is chosen to be low (less than 6) and avoid the square root of the distance since all the powers are even.

2.1.2 Skeleton based implicit Functions

Implicit models, such as Blinn, Soft-Objects and Meta-balls focus on using only one type of basic primitive and on blending them in a general form. All these basic primitives are based on potential fields expressed around point locations in the 3D space using the exponential or polynomial form. Bloomenthal [Bloomenthal 91] presented a new implicit model named convolution surfaces. Instead of using only the potential field around one unique point, the potential field primitive is generalized over lines, curves and polygons. This new implicit model relies on blending the distance field to basic skeleton elements that are points, lines and polygons. The global continuous implicit form of convolution surfaces is the following equation:

$$F(S,X) = \sum_{s \epsilon S} \exp\left(\frac{-||s-X||^2}{2}\right) \tag{2.1}$$

where S is the characteristic function of the skeleton i.e $S(X) = 1$ if X is a point of the skeleton, otherwise 0. The continuous form of the formula 2.1, can be seen as the convolution operation over the skeleton function S and the function $h(X)$ such as :

$$h(X) = \exp\left(\frac{-||X||^2}{2}\right) \tag{2.2}$$

Using $\diamond$ to represent the convolution, we have:

$$F(S,X) = (h \diamond S)(X) = \int_{s \epsilon S} \exp\left(\frac{-||s-X||^2}{2}\right) ds \tag{2.3}$$

For isolated convex skeletons such as triangles, rectangles, or line segments convolution surfaces are smooth and well defined taking advantage of the superposition property of the convolution operator , ie. the union of two skeletons S_1 and S_2 will verify the formula 2.4.

$$h \diamond (S_1 + S_2) = (h \diamond S_1) + (h \diamond S_2) \tag{2.4}$$

There is a great number of advantages in convolution surfaces or skeleton based implicit model such as:

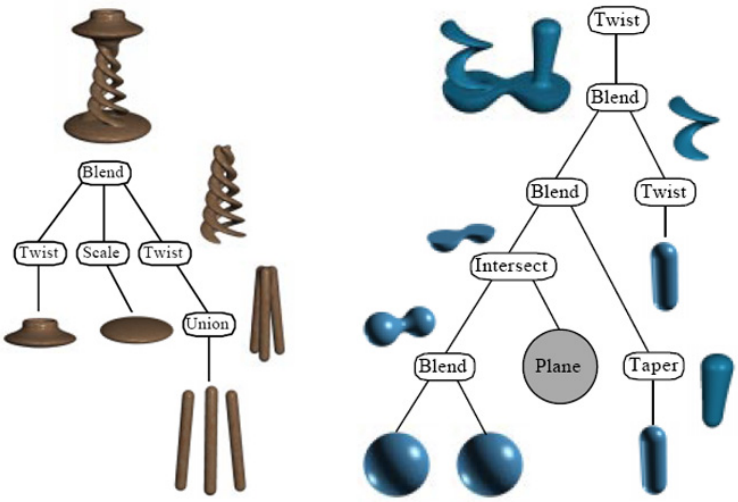


Figure 2.2: The Blobtree model: modelling hierarchy of two examples

- the shape of the skeleton suggests the shape of the surface
- the surfaces are smooth even if the skeleton is not
- topology can vary fluidly with skeleton changes
- blends are well behaved and implementation is simple

The modelling ability of skeleton based implicit surface originated the definition of new implicit models such as the SweepObject [Crespin 96] and the Blobtree [Wyvill 98] which provides an implicit model mixing skeleton based operators with modelling concepts such as blending, warping and boolean operations. The Blobtree is an implicit constructive hierarchy where the leaves are implicit skeletal primitives and the nodes are implicit operators such as blend, warp or boolean over children. The skeletal implicit primitives are points, lines, circles, polygons, polyhedra and planes. Over these primitives, boolean operators are formulated implicitly avoiding discontinuities:

$$F_{A\cup B} = \left(F_A + F_B + \sqrt{F_A^2 + F_B^2}\right)(F_A^2 + F_B^2)^{\frac{n}{2}} \quad (2.5)$$

$$F_{A\cap B} = \left(F_A + F_B - \sqrt{F_A^2 + F_B^2}\right)(F_A^2 + F_B^2)^{\frac{n}{2}} \quad (2.6)$$

Two different blending operators are proposed, the first denoted as $A + B$ representing the standard blending resulting from the sum F_A and F_B. The second

blending proposed is more controlled and is defined as:

$$F_{A\diamond B} = \left(F_A + F_B + \alpha\sqrt{F_A^2 + F_B^2}\right)(F_A^2 + F_B^2)^{\frac{n}{2}} \tag{2.7}$$

where α allows controlling the blending, note that if $\alpha\epsilon[-1,1]$ it creates interpolating blends between union and intersection operators. These two sets of implicit operation are complemented with a warping operator enabling to represent twist, taper and bend deformations. The general formulation of the warping is the following element:

$$F_{A_i}(X) = f_{A_i} \circ d_{A_i} \circ w_{A_i}(X) \tag{2.8}$$

where $f_{A_i}(r)$ is the potential function of the warp element, $d_{A_i}(X)$ the distance to its skeleton and its warp function $w_{A_i}(X)$.

2.1.3 Implicit functions for surface reconstruction

The skeleton based approaches are well adapted to constructive modelling or physically based animation, however, it is difficult to reconstruct existing 3D models from volumetric data such as range images or from a set of scattered point obtained by laser into an implicit representation in order to be modified. Taking in account the requirements of the conversion from discrete data to implicit formulation, several new implicit surface models were then defined extending the utility of implicit surfaces.

The "Blobby Model" [Muraki 91] was one of the first to handle the reconstruction problem by allowing the approximation of range images into an implicit

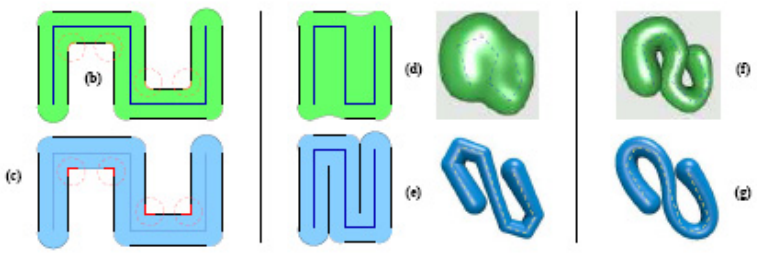


Figure 2.3: Convolution versus Blobtree

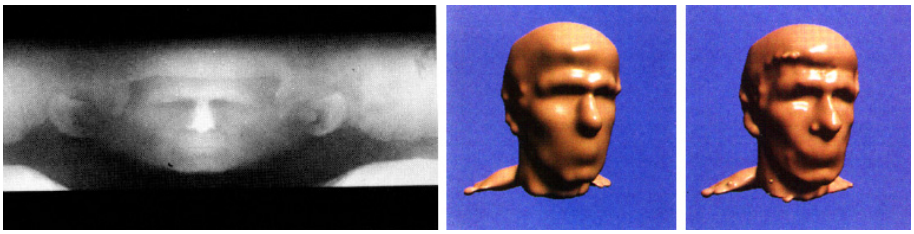


Figure 2.4: Blobby Model Reconstruction: range data image (left), implicit reconstruction with 120 primitives (center) and 451 primitives (right)

representation. The global implicit field uses the following definition:

$$F(X) = \sum_{i=1}^{N} b_i \exp\{-a_i f_i(X)\} \tag{2.9}$$

As shows, Figure 2.4, the range data images use color to represent depth and is converted into implicit surfaces by an incremental fitting procedure which was computationally expensive (several days to fit 256x256 images in 1991 giving 500 primitives). The fitting process starts with trying to approximate the data with one primitive f_i by solving the system which minimizes the energy equation 2.10:

$$E = \frac{1}{M}(E_{value} + \alpha E_{normal}) + \beta E_{shrink} \tag{2.10}$$

The weighting parameters α and β enable to control the strength of each sub-energy constraint. The E_{value} is the potential energy difference between the field value T to be approximate and the potential value F of a range data point X defined by 2.11,

$$E_{value} = \sum_{j=1}^{M} (F(X) - T)^2 \tag{2.11}$$

The E_{normal} is the energy to minimized the normal deviation between the implicit normal extracted from the gradient ∇F and the range data normal n_i computed using pixel data information defined by 2.12,

$$E_{normal} = \sum_{j=1}^{M} |n_j - \frac{\nabla F(X)}{|\nabla F(X)|}|^2 \tag{2.12}$$

Finally E_shrink is the energy to minimize the influence of the field of each primitive producing a shrinking effect which avoid to create infinitely far primitives for flat

surface areas,

$$E_{shrink} = (\sum_{i=1}^{N} a_i^{-\frac{3}{2}} |b_i|)^2 \tag{2.13}$$

At each step of the iterative fitting procedure, a primitive f_i is selected and subdivided defining two new primitives that are computed by energy minimization. The process is performed until the energy value becomes sufficiently small. The selection of the correct primitive to be subdivided is the key point to reduce the final number of primitives, however the process followed by Muraki is computationally expensive for large number of primitives, since the most influence primitive (energy point of view) is chosen.

Savchenko [Savchenko 95] proposed a different approach by using a volume spline based on Green's functions to offer implicit reconstructed surfaces. They start by using a "carrier" function as a sphere then approximate points by volumetric splines which will deform the "carrier" function to approximate the cloud of points. The process achieve a more reliable implicit representation than Muraki with less costs, however it is still too expensive to handle complex cloud of points with configurations that are difficult to achieve from a "carrier" function such as a sphere. Savchenko also explored mixing this approach with cross-sectional of the data fitting its contour and interpolating between implicit sections in order to offer a global implicit representation.

This idea was then generalized by both Carr [Carr 97] and Turk [Turk 99, Turk 99] who proposed the variational implicit surfaces (VIS) model (also known as Radial Basis Function RBF). VIS are smooth and respect a set of constraint points. VIS are always closed and can have arbitrary topology. They interpolate a cloud of points in a manner similar to thin-plate interpolation. To find the implicit function that approximate all constraint points with a minimum of curvature, $F(X)$ is defined such as to minimize 2.14:

$$E = \int_{\Omega} F''_{xx}(X) + 2F''_{xy}(X) + 2F''_{yy}(X) dX \tag{2.14}$$

The solution of this minimization problem is a variational interpolation which explains the name of variational implicit surfaces. There are several approaches

to solve this problem. Turk decomposes $F(X)$ into a linear combination of radial basis functions ϕ centered on constraint points, using $\phi(X) = X^3$.

The interpolating function, which satisfies the condition presented in 2.14 and defines the variational implicit surface, is presented in 2.15:

$$F(X) = \sum_{j=1}^{n} d_j \phi(X - c_j) + P(X) \tag{2.15}$$

Where c_j are the location of the constraints on the surface, d_j are the weights on each constraint and $P(X)$ is a one-degree polynomial which can be omitted if the number of constraints is greater than eight [Turk 01]. To compute the weight d_j values, we know the value h_j of the height field for each constraint such as $F(c_j) = h_j$. Based on Equation 2.15, the following linear system is defined as following:

$$M.D = H \tag{2.16}$$

In 2.16, $D = [d_j]$ are the unknowns, $H = [h_j]$ are the height field values and M, a matrix defined as a function of ϕ, P and c_j. While the linear system can be solved using LU decomposition in $O(k^3)$ steps where k is the number of constraints, iterative methods can solve large-scale systems in $O(k^2)$. To easy the VIS creation process, Turk proposes a method based on four different types of constraints:

- Boundary constraints c_j, placed on the boundary of the surface such that $F(c_j) = h_j$ with $h_j = 0$
- Normal constraints c_j, located outside the surface at a tiny distance to the boundary using the normal to the surface. These constraints verify $F(c_j) = h_j$ with $h_j = 1$
- Interior constraints c_j, located arbitrarily inside the surface such that $F(c_j) = h_j$ with $h_j < 0$
- Exterior constraints c_j, located arbitrary outside the surface such that $F(c_j) = h_j$ with $h_j >= 1$

It is possible to create variational implicit surfaces by specifying only boundary and normal constraints as shown in Figure 2.5. Yngve [Yngve 02] explored the

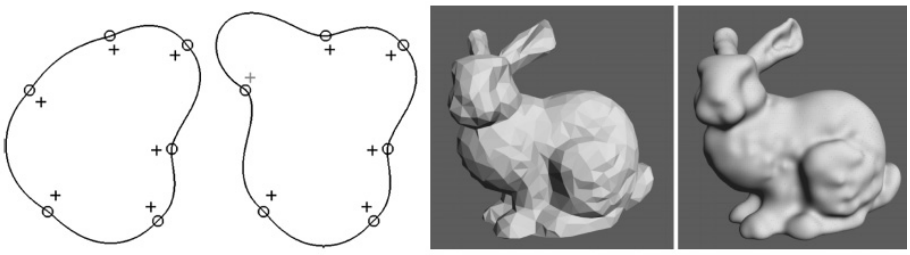


Figure 2.5: VIS example: boundary and interior constraints of contours (left), Bunny model implicit reconstruction with 1600 constraints (normal and boundary)

usage of VIS for surface reconstruction converting polygonal meshes using a global and iterative fitting process.

The flexibility of this representation allows defining complex and smooth models with arbitrary topology. It provides a compact and mathematically simple means of describing a surface using constraints, weights and a first order polynomial. It affords flexibility for computing normal and curvature information, while ensuring C^2 continuity. However, it requires the global evaluation of all the basis function and the energy minimization problem, from equation 2.14, is time consuming if the surface needs to approximate thousands of constraint points.

Morse [Morse 01] proposed a solution to extend the scalability of the variational approach, by using compactly supported radial basis functions (CS-RBF). The usage of CS-RBF enables to generate a sparse solution space reducing the computational complexity and making the technique practical for the reconstruction of large models. On their approach, they use RBF function from Wendland's family ie:

$$\begin{cases} \phi(r) = (1-r)^p P(r) & , 0 \leq r < 1 \\ \phi(r) = 0 & , r \geq 1 \end{cases}$$

These functions have a radius of support equal to 1 and by scaling the basis functions (i.e., $\phi(r/\alpha)$) it allows to define any radius of support α. Morse choose CS-RBFs as basis since they guarantee positive-definiteness and a sparse matrix for the reconstruction problem. By doing so, it provides a better scalability since the minimization can be computed by faster methods than the thin-plate matrix from Turk [Turk 99] minimization. These model allows also a better control since

the influence of each CS-RBF is local by its definition. The local behavior of CS-RBF allows to speedup the implicit function evaluation by selecting CS-RBF according to the distance to its center and radius of support.

Carr [Carr 01, Carr 03] proposed a solution following the previous approach by proposing fast radial basis functions (FastRBF). Compared to VIS, FastRBFs enable to reconstruct surfaces by allowing to approximate hundred of thousand points data-set thanks to two main mechanisms:

- Fast evaluation of RBF's performed via a fast multi-pole method
- RBF center reduction fitting for reconstruction

Instead of considering all constraint points as possible basis function centers for the resulting implicit surface such as VIS, the FastRBF uses an iterative greedy algorithm that adds new centers at each step depending of the fitting accuracy. The implicit FastRBF model allows the reconstruction of noisy or incomplete data creating a smooth surface. However discontinuities are still not supported since such as variational implicit surface the formulation is global and relies on C_2 functions.

Another approach was proposed by the HybridTree [Allègre 04, Allègre 06] which is an extension of the Blobtree [Wyvill 98] implicit model by allowing the mix between implicit and mesh representations and more complex operators were provided to this skeleton based implicit model. The mixing of both representations is achieved by creating an implicit view of the mesh using as basis the mesh triangles as skeleton elements. The global implicit approximation of the mesh is given by offsetting the implicit function computed from the mesh skeleton (ie mesh triangles) with the minimum distance between the offset surface and the mesh. Following this approach, the HybridTree is able to offer an implicit formulation for triangular meshes and supports discontinuities. However the representation is global and the evaluation time is strongly related with the number of points of the mesh.

2.1.4 Hierarchical/Local Implicit Representations

Othake et al. [Ohtake 05b] propose a hierarchical approach to 3D scattered data interpolation and approximation with compactly supported radial basis functions (CS-RBF). The implicit representation is a hierarchy of RBF and is faster than globally supported RBFs such as FastRBFs [Carr 01] and is also suitable to handle incomplete or noisy data. The hierarchy is created by creating several sets of points $P^1, P^1, ..., P^n = P$ from the point set definition of the surface P. Each simplification P^k is fitted by an implicit surface $f^k(x) = 0$ and P^k is obtained from P^{k+1} by clustering subsets of P^{k+1}. For each level of the hierarchy the single implicit surface is represented by:

$$f(x) = \sum_{p_i \epsilon P} [g_i(x) + c_i] \, \Phi_\sigma(\|x - P_i\|) \tag{2.17}$$

where $\Phi_\sigma(r) = \Phi(r$
$\sigma)$, $\Phi(r) = (1 - r)^4 + (4r + 1)$ is a Wendland's CS-RBF, σ is its support size, $g_i(x)$ a local quadratic approximation of P and coefficients c_i. This representation can be divided in two, ie. a rough partition of unity (PU) approximation by blending local quadratic approximation and a sum of CS-RBFs such as shows 2.18.

$$f(x) = \sum_{p_i \epsilon P} g_i(x) \Phi_\sigma(\|x - P_i\|) and f(x) = \sum_{p_i \epsilon P} c_i \Phi_\sigma \tag{2.18}$$

For each level , the implicit function $f^{k+1}(x)$ whose zero-level set approximates P^{k+1} is then obtained as the sum of $f^k(x)$, the PU approximation of P^{k+1}, and a sum of CS-RBFs. In order to construct the implicit surface, the hierarchy followed the point set hierarchy which is a space partition using an octree data structure. The octree subdivision is performed until each leaf clusters a maximum number of points. Each level of the octree data structure, is a level of the implicit hierarchy. Each leaf cell contains the points of P located inside, the centroid of these points and an average normal vector. Then the interpolating functions f are determined recursively for each level. Using this approach, the support size σ^k is defined recursively (ie $\sigma^{k+1} = \sigma^k/2$) starting from such $\sigma^1 = \alpha L$ where L is the length of the diagonal of the bounding box of P. In their implementation, they re-scale P such as $L = 20\sqrt{3}$ and use $\alpha = 0.75$. Finally the number of subdivision levels M can be

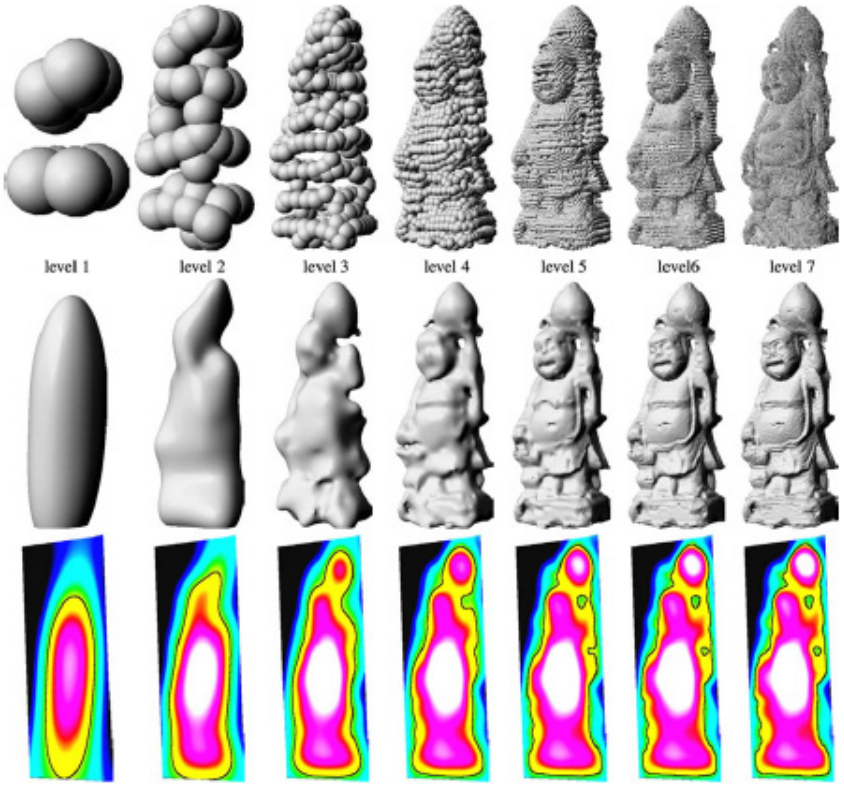


Figure 2.6: Multi level CS-RBF proposed by Ohtake [Ohtake 05b]

pre-computed since it is defined by σ^1 and σ^0 the support size for the single-level interpolation. Othake uses $M = [-\log_2(\sigma^0/(2\sigma^1))]$.

Compared to FastRBF [Carr 01], the quality of the Ohtake's approximation is similar but the approximation is computed four times faster (or two times faster if center reduction is used by FastRBF). Note that Ohtake´s approach does not use center reduction, which is an open speedup to be applied. This approach correctly handles noisy data and creates smooth surfaces, however when approximating thin surfaces extra zero level-sets can be created and sharp features are not supported.

The CS-RBFs based implicit surface is the basis of the Multi-Partition of Unity (MPU) [Ohtake 03], which presents a more suitable approach to approximate very large data-sets (ie. several millions of points) and be able to also better approximate sharp features which was not support by the previous model and the FastRBF. The implicit formulation of MPU relies also in an octree data-structure

and is the following:

$$f(x) = \sum_i \varphi_i Q_i(x) \tag{2.19}$$

such φ_i is the partition of unity function using nonnegative compactly supported functions $w_i(x)$ defined by

$$\varphi_i(x) = \frac{w_i(x)}{\sum_{j=1}^{n} w_i(x)} \tag{2.20}$$

and Q_i are the local shape functions of leaf cells. In their implementation, the weight functions are approximated by quadratic B-splines. This representation is simpler than the previous one and is able to better approach the sharp feature thanks to the local shape function used by the MPU implicit model. Ohtake et al. proposed three type of possible local approximations that can be used for a given leaf of the octree hierarchy:

- a general 3D implicit quadratic: to approximate larger parts of the surface,
- a bivariate quadratic polynomial in local coordinates: to approximate a local smooth patch,
- a piecewise quadric surface that fits an edge or a corner: to reconstruct sharp features.

Partition of unity provides fast and smooth shapes. On the other hand using different local shapes, the MPU is able to approach sharp features extending the idea presented in [Ohtake 05b]. However Ohtake shows that if a greater control of the smoothness is needed for a surface reconstruction, CS-RBF are better suited to represent local details and it is also possible to combine both representations on the same formulation as shown by equation 2.18. By combining both, Ohtake et al. [Ohtake 06] propose to apply the adaptive partition of unity for the base approximation and normalized RBF to better describe local details as illustrated by Figure 2.8.

A different approach was presented by Shen [Shen 04] to generate implicit surfaces from polygon meshes. Shen proposed the implicit moving least-squares surface commonly named IMLS surfaces. Instead of using a partition of unity, the

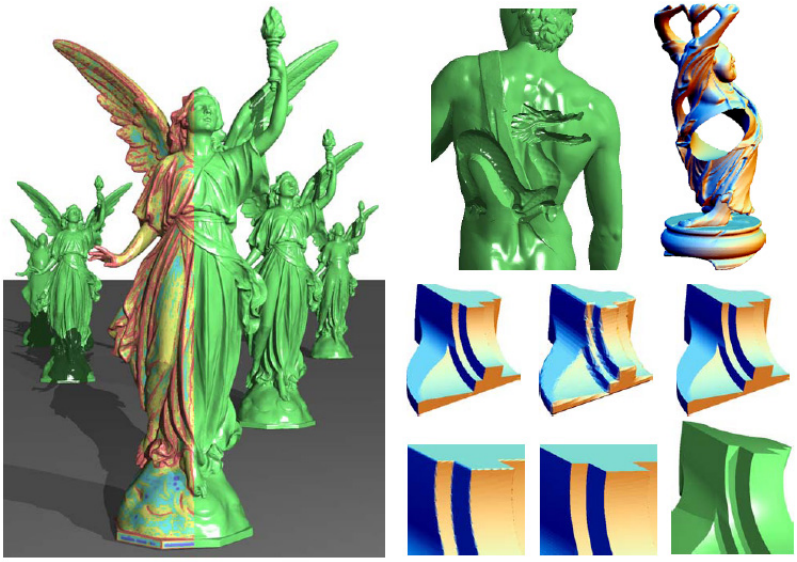


Figure 2.7: Multi Partition of Unity with sharp feature support [Ohtake 03]

definition relies on amoving least square method to interpolate scattered-data. The main difference between both methods is the usage of an improved normal and approximation procedure to enforce normal constraints. Shen demonstrates that the additional off-surface constraints from Turk [Turk 99] and Carr [Carr 01], introduces unwanted numerical instabilities in the field definition. This is solved by introducing a functional constraint based on neighbor points and their normal for off-surface point evaluation. On the other hand, IMLS rely on a different hierarchical evaluation scheme and an iterative method for generating useful approximation of the surface. The hierarchical scheme is based on a K-D tree where input triangles are stored and constant parts of the IMLS definition are stored to speedup calculations.

Reuter [Reuter 04] presented also a implicit surface representation that allows to represent sharp features and enable the reconstruction of complex models. Reuter chose to base its implicit representation on Enriched Reproducing Kernel Approximation (ERKPA) that are an extension of the Reproducing Kernel Particle Approximation method which is a generalization of the moving least squares method used by Shen [Shen 04]. The main advantage of ERKPA is to propose a coherent representation for discontinuities on its domain (support raduis) without using piecewise definitions such as MPU. However the computation time required

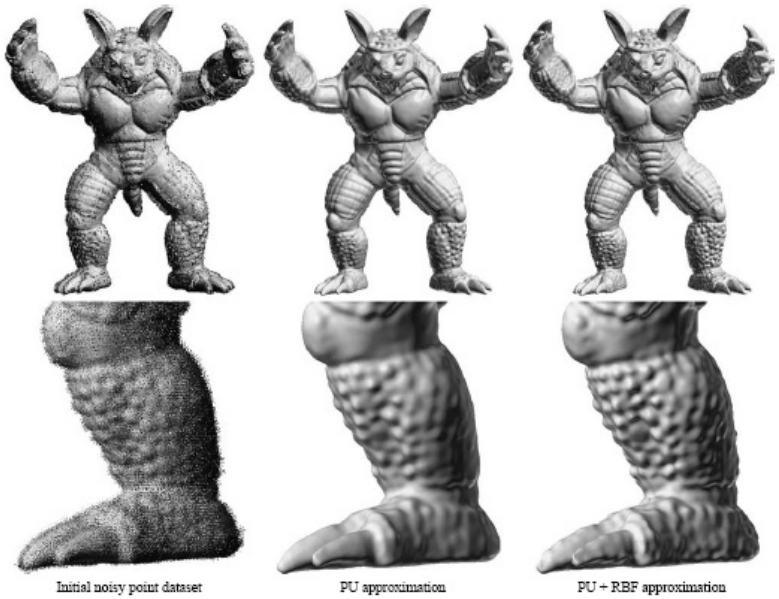


Figure 2.8: Mix implicit reconstruction approach using Partition of Unity and Radial Function Basis [Ohtake 06]

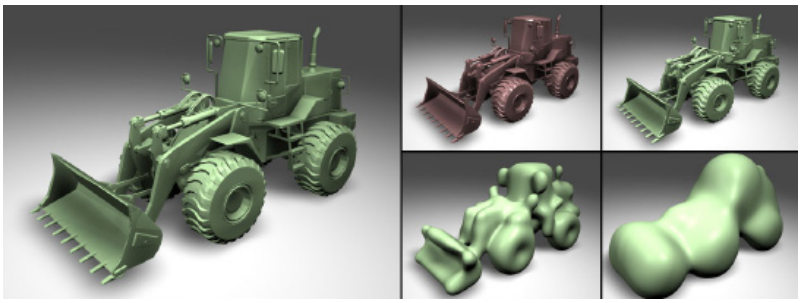


Figure 2.9: Implicit Moving Least Square surface representation [Shen 04]

by ERKPA reconstruction grows linearly with respect to the number of nodes in the radius of support which is too heavy compared to the previous model such as the MPU. More recently, Yang [Yang 06] proposed a method which combines Least Square Reproducing Kernel approximation(LSRKA) with Partition of Unity. LSRKA are an extension of RKPA by least square techniques, they are used such as Reuter's ERKPA [Reuter 04] to approximate local point set surfaces providing a consistent representation of sharp features. The global blending of the local function is done with an adaptive space partition verifying Partition of Unity such as the MPU [Ohtake 03]. They compare their method with MPU and also Moving Least Square method showing that sharp features can be efficiently represented and reconstructed without any computational cost penalty compared to other

methods.

Tobor [Tobor 04, Tobor 06] presents a new method that merges several advantages of the previous approaches. It mixes variational techniques using RBF, thinning algorithm and partition of unity (PU). This combination enables a robust multi-scale reconstruction of surfaces, on the other hand attributes from the input data, can be attached to the implicit representation (color/material information for example). This method can approximate and interpolate implicit surfaces from large unorganized point sets. The method relies on a binary tree domain decomposition which enables a multi-scale reconstruction of the surface. This allows to get several implicit reconstructions with different allowing adaptive evaluation. We should note that such as [Carr 01], this algorithm requires off-surface constraints for a correct reconstruction for each point. This constraint was relaxed by Wu [Wu 05] who proposed only a single off-surface point for each sub-domain reducing the quantity of data to be interpolated or approximated which leads to a faster reconstruction algorithm.

2.2 Differential Geometry over Implicit Surfaces

Curvature measures are widely used in Computer Graphics both to encode shape characteristics and to improve meshing of surfaces. Since implicit surfaces are represented by mathematical models, they make it possible to derive mathematic notions of curvature using differential geometry. As discussed in [Gray 96], methods to obtain these measures are well defined for parametric functions, however their application on implicit functions is not trivial and not so well documented. Hugues [Hughes 03] discusses differential geometry applied to implicit surfaces, providing interesting clues to retrieve shape characteristics from implicit function mathematics. Our approach uses these notions to extract relevant information from implicit functions as needed by our polygonization algorithm.

Given a point $X \in R^3$ such that $F(X) = 0$ where F is the implicit function that defines the surface, the following features can be extracted to gather properties from the surface.

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. Applied to all points lying over the surface, i.e, $X \in R^3$ such that $F(X) = 0$, they allow to get the normal vector of the surface to that point: $\vec{G} = \nabla F = \left[\begin{array}{ccc} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \end{array} \right]^T$ and $\vec{N} = \frac{\vec{G}}{\|\vec{G}\|}$

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

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

However, 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 study curvature values and directions on the plane tangent to $F = 0$ at X. In order to compute these, we need to use the matrix C defined by the partial derivatives of the normal $\vec{N}$ instead of the Hessian:

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

The matrix C can be defined using the Hessian matrix S 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 $dot_j = \vec{G}.\left[\begin{array}{ccc} H_{j0} & H_{j1} & H_{j2} \end{array} \right]^T$

We then compute curvature measures using the eigenvalues of C. 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 as depicted by Figure 2.10. Their respective eigenvectors are the principal directions defined to lie in the plane tangent to the surface at X. According to [Gray 96] and [Koenderink 90], the following curvature measures can be computed from k_1 and k_2:

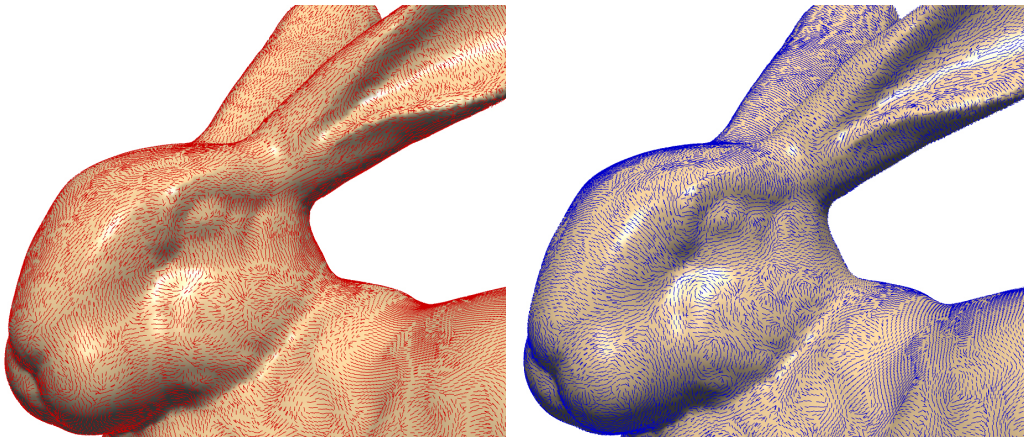


Figure 2.10: Principal curvature directions of an Implicit Reconstruction of the Stanford Bunny using MPU: left) k_{min}, right) k_{max}.

- Mean Curvature = $\frac{k_1+k_2}{2}$
- Gaussian Curvature = $k_1 * k_2$
- Deviation from Flatness = $\sqrt{k_1^2 + k_2^2}$
- Maximum Curvature = $max(k_1, k_2)$
- Minimum Curvature = $min(k_1, k_2)$
- MAC = $max(|k_1|, |k_2|)$
- Shape Index = $-\frac{2}{\pi} \arctan \frac{k_{max}+k_{min}}{k_{max}-k_{min}}$

Figure 2.11 to Figure 2.13 present the curvature information of the Stanford Bunny using three different implicit surface representation: Multi Partition of Unity [Ohtake 03], Variational Implicit Surfaces [Turk 99] and Sparse Low Degree Implicit [Ohtake 05a]. In order to compare the curvature between the three model, we use the same color map proposed by Ohtake [Ohtake 05a] which map the curvature in a invariant way using the distance to the average value and considering the standard deviation. It is possible to observe the closeness of the different curvature representation, which easily identify high variation curvature areas from flat areas. We can notice that depending of the implicit model the variation of the curvature is not the same, since the values are different for each model. However all of them correctly depicts curvature variation and can be

used as local characteristic descriptor of the shape for the polygonization using different scale factors and thresholding the value to better adapt the variation of the implicit model. Both the gaussian and mean curvature values are widely used by re-meshing algorithms and several works [Alliez 03, Meyer 03] present estimations of these as applied to meshes which enable to compare the curvature of the implicit representation and the curvature estimation of the mesh.

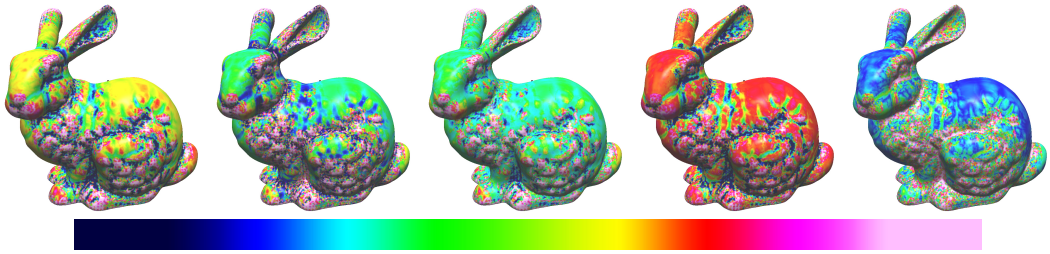


Figure 2.11: Colored Curvature of the Stanford Bunny using MPU [Ohtake 03]: Mean , Gauss , Maximum, Minimum, Deviation from Flatness (from left to right) full range color map used to represent the curvature (on the bottom)

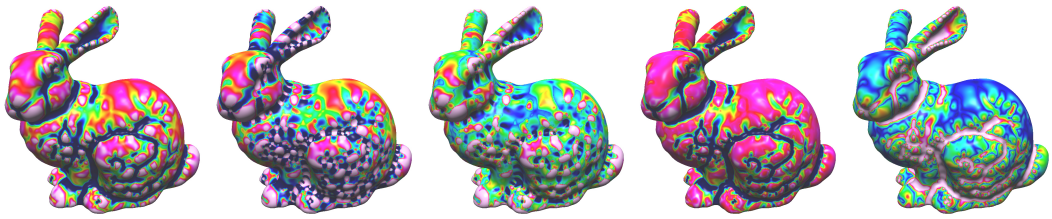


Figure 2.12: Colored Curvature Plot of the Stanford Bunny using VIS [Turk 99]: Mean , Gauss , Maximum, Minimum, Deviation from Flatness (from left to right)

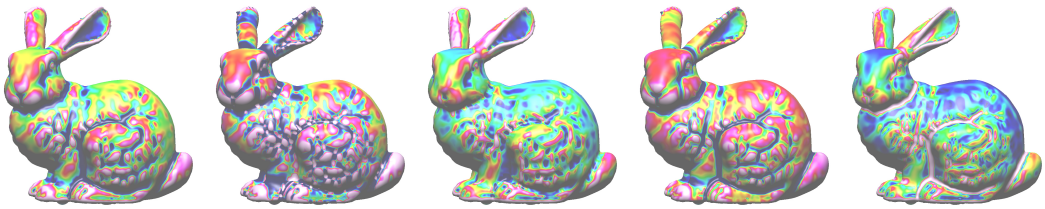


Figure 2.13: Colored Curvature Plot of the Stanford Bunny using SLIM [Ohtake 05a]: Mean , Gauss , Maximum, Minimum, Deviation from Flatness (from left to right)

Figure 2.14 depicts another curvature index which is the shape index of an MPU and VIS implicit surface. The shape index is a representation of curvature proposed by [Koenderink 90] which depicts both the convexity and con-

cavity features in a scale-independent manner. This measure was used by Masuda [Masuda 03] to visualize curvature estimations of signed distance fields. We can compare our results to Masuda's since the same data set is used for the visualization, albeit using a different representation. Visual inspection shows that curvature results are very similar, thus illustrating the validity of our measures. At the exception of Figure 2.13, all the rendering of the Stanford Bunny presented on this section were done using our algorithm to produce a constant polygonal approximation of the surface. Our approach will propose to adapt the length of the edge triangles using the mean, the gaussian and the maximum absolute of curvature (MAC). However we do not need to distinguish between negative and positive curvature values to compute edge length when creating polygons. Regarding the MAC value, it provide the guaranteed to be no smaller than any curvature measures on the plane tangent to $F(X) = 0$, regardless of direction. For all the curvature value, we will use their inverse estimating a radius value of curvature. For example regarding the MAC curvature, we can define the minimum radius r_{min} of curvature as follows:

$$r_{min} = \frac{1}{max(|k_1|, |k_2|)}$$

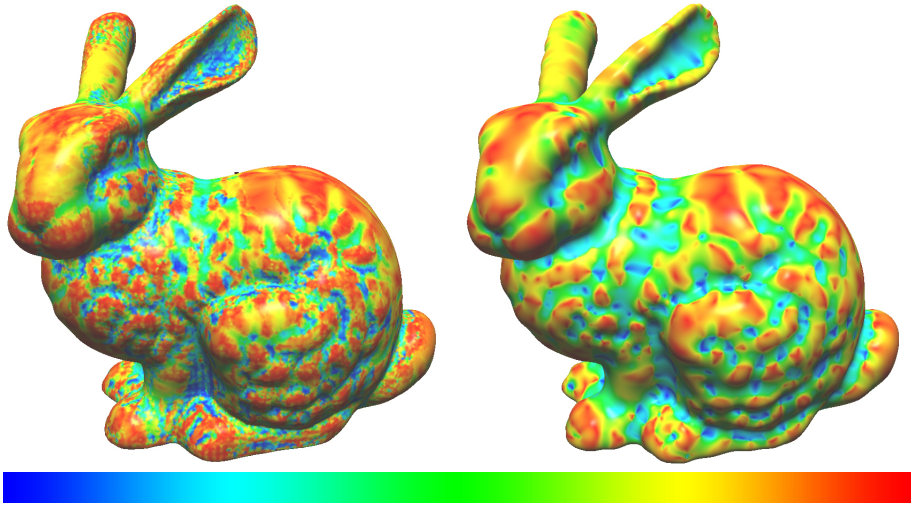


Figure 2.14: Colored Shape Index of the Stanford Bunny using MPU (on the left) and VIS (on the right)

2.3 Summary

This chapter presented a detailed survey of almost all the implicit surfaces representation for a deep understanding of the different implicit formulations and their flexibility as shape representations. Then we enumerated and described the meaning of the several mathematical property which can be extracted from the mathematical implicit formulation in order to gather shape characteristics such as curvature.

3

Related Work

This Chapter presents an overview of the work related with the topic of this dissertation.

3.1 Visualization of Implicit Surfaces

Since their introduction, implicit surfaces provide a solution of choice for imaging medical information or point sets arising from digitizing complex models thanks to their flexibility in manipulating data. However precise and fast visualization of implicit surfaces is difficult. The more obvious rendering approach of implicit surfaces is based on ray-tracing algorithms [Jevans 88, Kalra 89, Wyvill 90, Hart 93, Stolte 95, Gascuel 95, Sherstyuk 96, Sherstyuk 99]. Since implicit surfaces are well defined mathematically by the zero level of an implicit function, this first set of visualization approaches presents speedup techniques for both ray-surface intersection computation and implicit surface evaluation. However as it well known, it is at the cost of interactivity and real-time ray tracing rendering is still possible only for super computer. From the several speedup techniques, the spatial organization using octree is the preferred technique [Jevans 88, Wyvill 90, Stolte 95, Gascuel 95, Sherstyuk 96, Sherstyuk 99] and is usually combined with the Lipstchitz condition [Kalra 89, Wyvill 90, Hart 93, Stolte 95, Gascuel 95]. The Lipstchitz condition enables to rapidly discard non-intersecting rays or spatial cells and to faster the ray-surface intersection calculation. However, the computation of Lipstchitz constants is only acceptable for simple implicit models such as blobs, meta-balls, soft-objects, convolution surfaces and skeleton based representations where it can be estimated based on the skeleton primitives. As

result, the benefits are lost for other models such as global Radial Basis Functions with thousands of center nodes or reconstruction implicit representations. Sherstyuk [Sherstyuk 96, Sherstyuk 99] presents one of the faster ray-tracing but at the cost of a sampling of the surface which can lost details such as sharp features.

Non-photo-realistic rendering and particle based techniques are alternative visualization methods for implicit surface less accurate than ray tracing. However, these methods aims on presenting a partial view of the surface or focus on specific features of the surface. It is not the purpose to get an high fidelity visualization but to be accurate on the features that need to be highlighted. Silhouettes [Bremer 98, Foster 05] are more or less easy to extract from the implicit representation and can be tracked for an interactive visualization. However if a polygonal mesh is available, a faster hybrid method can be applied such as presented by [Schmidt 06]. Ridges and valleys are one of the main characteristics of non-photo-realistic-rendering, they usually are difficult to extract depending on accuracy needed. Methods such as [Belyaev 98, Bogaevski 03] present the more accurate extraction of features from implicit surfaces but at the cost of expensive calculations based on differential geometry and critical point theory. Faster methods such as [Belyaev 05, Ohtake 05b] present a good trade-off between computational cost and accuracy to extract ridge and valley. Regarding particle based techniques, existing methods are more targeted to be used as a complement of sampling techniques for existing rendering system than to be viewed as a complete method for implicit surface visualization as demonstrate several adaptive meshing methods [Rösch 97, Vorsatz 01, Kobbelt 03, Liu 05, Bouthors 07]. Most of the particle based methods [Witkin 94, dF92, Desbrun 95] are well designed to propose adaptive sampling of the implicit surface taking in account the smoothness and the existence of sharp features on surface. We should note that particle based system are very similar to the vertex relaxation mechanism and Laplacian smoothing used by several polygonal method to improve the quality of the triangulation [Neugebauer 97, Ohtake 01, Karkanis 01, Ohtake 02, Paiva 06, Peiró 07].

Ray Tracing

Approaches	Octree usage Lipchitz constant based intersection calculation
Advantages	high fidelity easy to implement and support almost all representation good for dynamic system
Issues	slow computation not suitable for realtime most of speedup dependent of IS representation

Particles Based

Approaches	octree based sampling uniform or random sampling skeleton based sampling
Advantages	sampling of the surface usually fast to render
Issues	partial visualization hard to capture sharp features number of iteration require to converge

Non Photo-realistic Rendering

Approaches	silhouette drawing ridge-valley drawing
Advantages	expressive rendering rendering of shape feature
Issues	partial visualization not always real-time

Table 3.1: Non Polygonal Rendering comparison

3.2 Polygonization Methods

Even with the usage of several speedup techniques to improve ray-tracing algorithms, they are still far away to be suited for an interactive visualization. Real time ray tracing is still of the domain of super-computers. Therefore, subsequent research has focused on approximating implicit surfaces using polygon meshes, more suitable for real-time visualization using commodity hardware. Moreover, polygonal meshes enable us to explore trade-offs between fidelity of representation and interactive performance. The following subsections present the main polygonal approaches which are cell partitioning, surface tracking and inflation/shrinkrap methods. Compare to the other techniques (Table 3.1, polygonal representations are easy to render for interactive visualization and generate an alternative representation with can be used in several domains of Computer Graphics.

3.2.1 Cell partitioning

Cell partitioning techniques are the most popular methods for rendering implicit surfaces, by creating a polygonal mesh that fits the surface based on space subdivision. Beyond introducing the soft objects, Wyvill [Wyvill 86] was one of the first to propose a polygonization process by subdividing the space into cubi-

cal cells. The approach starts with the identification of all cells intersecting the surface using the knowledge about the soft-object representation. Then each cell is classifies regarding their implicit value at the vertex i.e. hot vertex if the implicit function is positive and cold if negative. From this classification, non-intersecting cells are discarded and intersecting cells are polygonized based on the hot and cold vertex pattern at each face of the cubical cell. Wyvill identified seven different patterns to polygonize a face. This strategy is the basis of the well know Marching Cubes presented by Lorensen [Lorensen 87] which catapult implicit surfaces (even if the first marching cube was illustrate with discrete data) as a attractive and popular representation for medical data. The Marching Cubes applies a divide and conquer approach to generate the visualization representation by spatially subdividing the space into cubical cells. For each cell, edge's cell/surface intersection is calculated then it "marches" to the next cube. Polygonization of each cube is done by implicit classification of each cube's vertex (positive/negative ie inside/outside). Thanks to cube symmetry the 256 possible pattern are reduced to 14 cubical polygonization patterns.

Bloomenthal [Bloomenthal 88] presented an improved Marching Cubes version named Marching Tetrahedral. Since some of the cubical polygonization patterns proposed by Lorensen were ambiguous, the Marching Tetrahedral subdivides each cubical cell into 6 tetrahedral pyramids. This enables also a simpler table since the possible patterns are only based on the 4 vertices of the tetrahedra instead of the height vertex of the cubical cell. The implementation of this algorithm was later published in Graphic Gems [Bloomenthal 94] and became the most used algorithm to convert implicit surfaces into polygonal meshes. Ning [Ning 93] discussed and review the cell decomposition algorithm and concludes that tetrahedral decomposition produces a consistent topology however using twice the number of triangles than single-entry cubical lookup table.

The Marching Tetrahedra was optimized by Triquet [Triquet 01, Triquet 03] allowing almost interactive implicit polygonization. Triquet demonstrates that the Bloomenthal Marching Tetrahedra presented redundant calculations and neither optimal nor accelerate steps and introduced the following speedup techniques similar to the one proposed by Wyvill[Wyvill 86]:

- avoid re-evaluation at cube extremities since they are share by 8 cubes so calculation are reused
- Regarding new points located at the edge, redundancy is avoid also
- Time-stamps are applied to each value computation to avoid redundancy using a hash-map like mechanism

This work resulted in a new pattern table with 6 new patterns compared to the original Marching Tetrahedra table resulting in 20 patterns and reduce the polygonization time by a factor of 6. This work used compactly defined basis function such as meta-balls/soft objects and most of the speedup was achieved taking advantages of the field function structure and could not be applied to other implicit representation. The same was done by Cuno et al. [Cuno 04] which proposed a hierarchical adaptation of the Marching Tetrahedral to polygonize variational implicit surfaces trying to minimize the number of implicit evaluation and quickly prune the space. However this is achieved by simplifying the variational implicit model [Turk 99] i.e. by changing the pseudo-Euclidean distance computation based on the location of the normal and boundary constraints of VIS model. Zhang et al. [Zhang 06] proposed another simple speedup techniques which enable to cut to half the meshing time of Marching Cubes or Marching Tetrahedra. However the algorithm is only targeted to implicit surfaces which estimate a distance field.The time reduction is achieved by reducing the computation effort spent for empty cells. On the other hand, the algorithm simplifies the edge-surface intersection computation by using a estimation value based on the interpolation of vertex values instead of the usage of a convergence mechanism such as the Newton-Raphson method which use the implicit value and it gradient. This algorithm presents simple and effective speedups, however at the cost of several assumptions which does not guarantee that the resulting mesh correctly approximate the surface. This method is more adapted to get fast coarse representation of implicit surfaces.

MC/MT and its many variants, based on space subdivision provide a simple and effective approach made popular by the availability of source code. However, the resulting meshes are not homogeneous and they exhibit poor quality as we

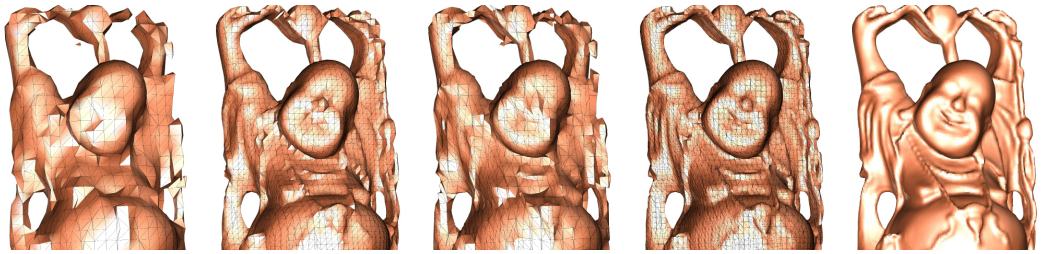


Figure 3.1: Polygonization mesh result using Marching Cubes and Marching Tetrahedra for the Happy Buddha Model: MC with 9320 triangles, MC with 40740, MT with 28164 , MT with 122516, MT with 508668

can see in Figure 3.1. The regular space subdivision of the space produces triangulation patterns not related with the characteristic of the implicit surface but with the discretization process i.e the spatial decomposition of the polygonization algorithm. In order to catch correctly all the topological features, high triangular mesh density is required since the algorithm use an homogeneous subdivision of the space.

3.2.2 Surface tracking

Surface tracking denotes a family of polygonization techniques including methods such as Marching Triangles [Hilton 96] which generates polygonal approximations by starting from a point lying on the surface and generating triangles by following the surface.

The Marching Triangle was proposed by Hilton [Hilton 96, Hilton 97] and allows to generate a 3D triangulation which verifies the Delaunay constraint, i.e. for each triangle of the mesh, the circumscribed sphere passing through each vertex does not contain other vertexes. Applying this constraints, it generates a mesh with a uniform but not regular vertex distribution over the surface. The Marching Triangle can be applied starting from a seed point/triangle or an incomplete mesh with holes to be refined. The algorithm generates new vertexes by expanding each triangle edge of the mesh generation boundary until it covers all the surface.

Hartman [Hartmann 98] proposes a similar surface tracking algorithm for implicit surface polygonization. The approach is similar to the Marching Triangle. However instead of expanding edges, the mesh is generated by expanding points

located at the mesh boundary organized into fronts. Starting from a seed point, the first expansion create a new front formed by boundary vertexes. During the mesh generation, it can append that due to the topological characteristic of the surface, fronts need to be split or merged with other to avoid mesh overlap. This requires a collision test inside the front and also with other fronts which can be costly. However more than one triangle can be generated by each point expansion compared to the edge expansion used by the Marching Triangle.

Cermak proposes another variant of the Marching Triangle named Edge Spinning [Cermák 02]. Such as the Marching triangle, the expansion is applied to edges by using an oriented rotation around one of the edge extremity. To avoid mesh overlapping, each new triangle is tested against existing active edges. After finding the closest point lying in an active edge (boundary), depending of it distance, they reuse existing points creating one new triangle or create three triangle if the closest point is closer than a given threshold using its adjacents. The edge expansion is made using a constant radius in order to create isoscele triangles. The projection of the new edge extremity on the surface is done by binary subdivision. This binary search does not require gradient computation such as Newton Step, however the convergence is slower. The Edge Spinning algorithm presents more almost equilateral triangles than Marching Triangle. The expansion usually create only one new triangle, however some scenarios can create two using the adjacent point or three using the nearest colliding point and its adjacent points.

Surface tracking approaches as depicted by Figure 3.2 can achieve good results

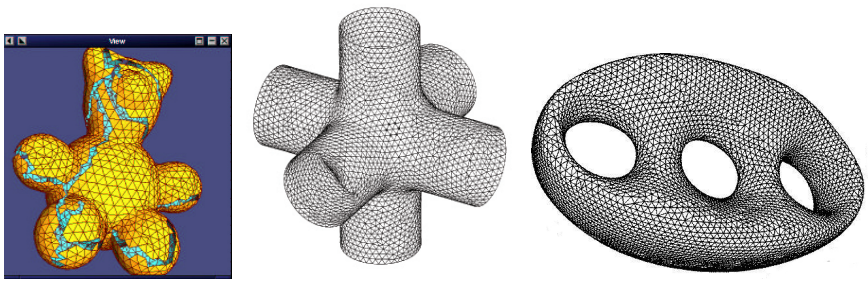


Figure 3.2: Polygonization meshes produced by surface tracking approaches: Marching Triangle implementation showing the need of overlap testing [Akkouche 01], Hartman algebraic surface [Hartmann 98], Edge Spinning result [Cermák 02]

by generating meshes of evenly-sized and quasi-equilateral triangles. However, the previous approaches are not adapted to local surface properties, requiring large numbers of small triangles to approximate surfaces with large variations in curvature instead of being adaptive. On the other hand, they require to avoid triangle overlapping during the mesh generation, which can be difficult to achieve by only consulting the mesh boundary. Furthermore, this method is more sensitive to numerical instability than space decomposition methods which might occur depending on the complexity of the implicit surface.

3.2.3 Inflation/Shrinkwrap Approaches

Inflation and Shrinkwrap are two opposites approaches to generate polygonal representations of implicit surface focusing on topological aspects instead of polygonizing cubical approximation of the shape or tracking the surface by marching near to the tangent plane of the surface at a given point location.

The inflation method was proposed by Stander [Stander 97]. This method was designed to guarantee the topological correctness of the polygonal approximation of the implicit surface. The approach relies on an inflation process based on the critical points of the implicit surface function. The topology is guaranteed by tracking the critical points of any C_2 continuous implicit surface. Stander proposes an interval analysis methods which searches for critical points starting from an initial bounding box which is subdivided as an octree data-structure. Then a zero Newton-based search enables to find critical points. These critical points are mathematical characteristic of the implicit function and they can be classified according to the eigenvalues of the Hessian matrix extracted from the implicit function at the critical location. This classification allows to identify the type of the critical point i.e if it is a maximum, minimum or a saddle point and also attached the sign of the change. The polygonization is performed by inflating a mesh at each critical point (maximum). Then the mesh is inflated and polygons are removed or added resulting in a reconnection procedure for re-polygonization.

The Shrinkwrap [vO04] algorithm was proposed by Van Overveld in its technical report in 1993. At the opposite of the inflation process, it starts with the

triangulation of a sphere surrounding the whole surface and applies successive deformation to the triangulation to approximate the surface. This algorithm only approximates iso-surface that are homeomorphic with a sphere, disconnected components or holes are not supported. The resulting mesh is adaptive to the local curvature of the surface. The algorithm is applied to skeleton based (point,lines, polygons) implicit surfaces. However it can be used by any implicit representation where the gradient value is available. The algorithm starts with the triangulation of a sphere surrounding the surface and at each iteration (predefine number of iterations) a Newton-Raphson step is applied to all mesh vertexes which results on a displacement along the implicit gradient direction improving the polygonal approximation. [Bottino 96] extended the Shrinkwrap algorithm to support implicit surfaces with arbitrary topological genus. This is done by adjusting the topological structure of the mesh with critical points.

3.3 Polygonal Approximation Issues

Polygonization of implicit surfaces implies the conversion from a continuous mathematical model to a piecewise representation. This can be see as negative point compared to ray tracing, particle based and non-photo-realistic rendering techniques which are able to enable the visualization without needing to convert the surface. However these methods also need additional data-structure to complement the implicit representation to provide a faster visualization. If the polygonization can achieve a high quality approximation, it enables a perfect substitute of the implicit surface adapted to nowadays graphics hardware designed and optimized to render triangle soups. Of course, one of the major advantage is the real-time visualization available with current hardware even for models with billions of triangles which can be viewed interactively. However, converting from one representation to another is not a lossless process in particular from continuous to discrete representation. The accuracy and quality of the polygonal approximation is the major concern depending of the kind of targeted application for the implicit surface alternative. The following subsections review the related work considering the efforts made to improve polygonal approxima-

tion of implicit surfaces.To evaluate the polygonization, since its is a conversion from continuous to piecewise representation, the following concerns need to be handled.

Topology Such as illustrated by Figure 3.1, the polygonal result can present a different topology of the original surface. On the other hand, the polygonization should be able to reproduce any type of surface i.e non-manifold surfaces or disconnected elements, or it will not depict the surface correctly.

Sharp Feature Sensitive Beyond erroneous topological reproduction, sharp features are another concern, since the frequency of the sampling could not be enough to correctly reproduce and catch discontinuities that might occur on the surface.

Smoothness Piecewise representation are not adapted to correctly convey and reproduce curvature information. Sampling based on spatial subdivision does not seem to be the correct approach. The polygonization should be adaptive in order to better reproduce the curvature of the surface.

Quality of the piecewise representation The quality of the mesh representation, is also another concern in particular to use this representation for other applications such as simulation, modelling and deformations. As depicted in the Marching Cubes, unwished pattern are visible on spatial subdivision techniques. Methods to regularized and improve the mesh quality need to be combined with the polygonization algorithm. Meshing with an adaptive density of triangles compared to the curvature should be used, i.e. small triangles in high curvature regions and larger triangles in areas with low curvature.

3.3.1 Topological Guarantees

One of the main problem of the Marching Cubes is the topological inconsistency that may arise from the cubical cell classification and its corresponding triangulation. This point is very critical in particular when dealing with iso-surfaces extracted from Medical Scanned data. Most of the iso-surfaces implicit functions are trilinear piecewise interpolant functions using the scalar data

values at cubical vertexes. In order to improve the topological correctness of the polygonal approximation produced by the algorithm, several approaches [Montani 94b, Chernyaev 95, Hege 97, Lopes 03] present alternative lookup tables for the cubical cell classification instead of proceeding with the tetrahedral subdivision proposed by the Bloomenthal Marching Tetrahedra [Bloomenthal 94]. Nielson [Nielson 03] proposes an alternative method which also guarantee the topological correctness for iso-surfaces at the cost of a more complex lookup mechanism based on a three level classification i.e edges, faces and cell interior classification. More recently, a method based on the critical point analysis of the implicit function of the iso-surface was proposed by Renbo [Renbo 05]. Without using any look-up table, they triangulate each cell by classifying the nature of the critical points.

A different strategy was followed by Bloomenthal[Bloomenthal 95] which proposes a method to polygonize non-manifold implicit surfaces changing the tetrahedrons based classification as depicted by Figure 3.3. Yamazaki [Yamazaki 02] also proposes an approach for non-manifold implicit surfaces by extending the Marching Cubes algorithm to handle discontinuous fields correctly such as holes, boundaries and intersections by enhancing the distance field using bounding volumes to simplify the distance calculation between points. Features are correctly approximated, however the meshing quality depend of the marching subdivision which is defined by the user, i.e. polygonization with size similar to the original model will still give a poor mesh quality , if the cell size is smaller enough the triangulation is good and nicely adapted to sharp features.

Platinga [Plantinga 04] presents a marching cubes extension using an octree based space partitioning which produces an adaptive mesh and tries to guarantee

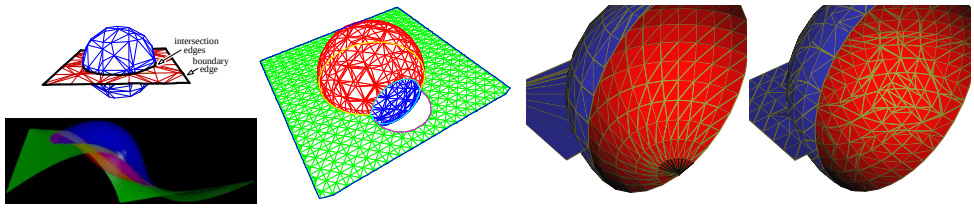


Figure 3.3: Polygonization of non manifold implicit surfaces: left) [Bloomenthal 95] and right) [Yamazaki 02].

its topological correctness. This is done by taking advantage of interval arithmetics to determine global properties of the implicit function and use it as criteria for the octree subdivision. In [Plantinga 07], they present the proof of the isotopy for the octree-based meshing and extend their approach applying tetrahedrization to Marching Cubes cells. However the results are limited to simple algebraic functions or meta-ball based models. The output mesh does not presents a smooth triangle size transition and reveals a jagged effect from adaptive subdivision when looking at triangle edges. The tetrahedrization proposed in [Plantinga 07] reduce but does not completely remove this problem. A similar approach was followed by [Alberti 05] to triangulate implicit algebraic surfaces. However, instead of using Marching Cubes, they propose a subdivision algorithm using a method similar to interval of analysis which partitions the space and isolates singularities and guarantee the topology in smooth parts. This algorithm results on a topological approximation producing meshes similar to the one obtained by Marching Cubes. However, even if the topology is only guaranteed until a given threshold, the triangulation is adaptive since the space partition is defined to catch any singularity.

3.3.2 Feature Sensitive

One of the main concerns regarding the Marching Cubes is its poor ability to approximate correctly sharp features and other discontinuities such as holes. Most of the artifact are related with the discrete sampling approach followed by the cubical cell subdivision of the space. Several approaches try to overcome this limitation by improving implicit function values. Both the Extended Marching Cubes [Kobbelt 01] and the Dual Contouring [Ju 02] propose implicit function adaptation relying on the spatial partitioning to better estimate the distance field to the surface and they refine the cell pattern classification according to the detection of sharp features. This approach was improved by Azernikov [Azernikov 05] which proposed an adaptive meshing of implicit surface by combining an anisotropic grid for sampling sharp features with Dual Contouring.

The main problem of both previous works is that they are only able to approx-

imate 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 [Varadhan 03] presents an alternative Extended Dual Contouring algorithm which mixes both approaches to generate an accurate polygonization of implicit surfaces from volumetric data. This new algorithm presents a more robust intersection test and applies adaptive subdivision techniques to better approximate sharp features. They also used additional criterions for the adaptive octree based sampling to guarantee one intersection per cells edge and they are able to catch any thin feature [Varadhan 04]. Dual Contouring based approaches and Extended Marching cubes requires that the iso-surface intersects the edges of the grid spatial subdivision and both algorithm are based on the uniform initial sampling grid.

Schaefer [Schaefer 04] proposed an alternative solution by introducing the Dual Marching Cubes (Figure 3.5 left). This algorithm relies on a dual grid representation used for Marching Cubes. Compared to Dual Contouring and Extended Marching Cubes, this approach requires a sparser underlying octree compared to other approaches to produce an equivalent contour . On the other hand, this algorithm provides a better approximation of sharp features when features are not axis aligned thanks to dual grid. Varadhan proposed a different solution in [Varadhan 06](Figure 3.5 right) to approximate implicit surfaces with an accurate topological and geometrical polygonal approximation using visibility mapping. The algorithm is based on the spatial subdivision presented in [Varadhan 04]to catch all topological features 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 is projected to generate the correct polygonal approximation of the surface patch within the cell. This method allows to polygonize high topological complex surfaces and is applicable to complex implicit models such as MPUs [Ohtake 03] or CSG based implicit models.

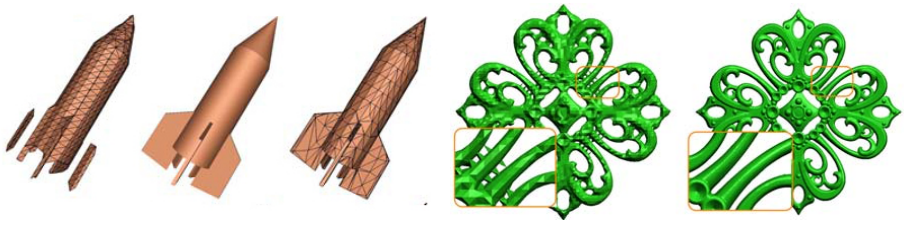


Figure 3.4: left) Dual Marching Cubes [Schaefer 04] result for sharp features approximation and right) spatial subdivision versus Varadhan [Varadhan 06] visibility maps for an MPU model

3.3.3 Regular Meshes

Controlled quality meshes are easily achieved using surface tracking approaches such as [Hilton 96, Hartmann 98, Cermák 02]. However, this is not true for polygonization based on space decomposition approaches. As studied by Chan [Chan 98], the tetrahedra decomposition enables to deal with the ambiguities of the Marching Cubes. However it introduces more subdivision related patterns on the resulting mesh which exhibits poor quality. Chan proposes a simpler subdivision of the cells than the one used by Bloomenthal. Results showed a small reduction of the number of triangles and produced more regular triangles by reducing the ratio of longer edges to shorter edges. However for scanning data, the method proposed may not be compatible since it requires a sampling inside the cube as opposite to Bloomenthal which does not introduce any new vertex during the tetrahedral subdivision. On the other hand, even if the edge length triangle ratio is improved, it still far to present a smooth triangulation without revealing any subdivision pattern.

This was partially solved by Treece [Treece 99] which presents a regularized Marching Tetrahedra for volume data extraction which produces a regular triangulation reducing the number of triangles. This approach combines Marching Tetrahedra for iso-surface extraction and vertex clustering to produce a regularized triangle set. By applying this additional step, the final triangle size of the mesh is reduced from 26 to 30 percent. Liu et al. [Liu 05] presents a regular meshing method for algebraic and skeleton based implicit surfaces designed to handle dynamic implicit descriptions creating a particle system similar to Witkin [Witkin 94] repulsion/fission mechanism to produce a uniform sam-

pling of the surface. Then a ball-pivoting algorithm is used which starting from a triangle created using three initial points travels along the surface by pivoting from sample to sample. This process creates new triangles and finish when all the samples are visited producing a regular mesh.

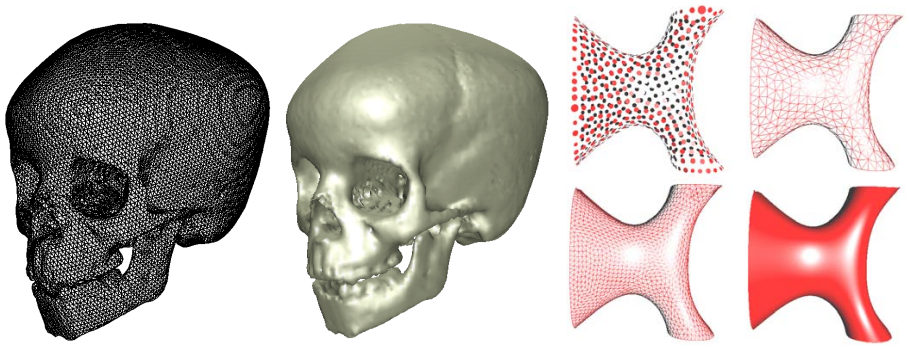


Figure 3.5: Regularized Marching Tetrahedra [Treece 99] on the left and Particle based optimization proposed by [Liu 05]

3.4 Adaptive Meshing

This section describes the several algorithms which are able to produce adaptive meshes. These approaches have the particularity to increase the number of the triangles in regions of high curvature or in presence of sharp feature to better approximate the surface. Adaptive meshes usually make a better usage of the budget of triangles using small triangles on high curvature regions or large triangles on low curvature regions.

3.4.1 Spatial Decomposition

Velho[Velho 99] presents a unified and general tessellation algorithm for parametric and implicit surfaces able to generate an adaptive mesh thanks to a controlled subdivision. The algorithm starts by creating a simple uniform decomposition similar to the Marching Cubes creating a coarse triangulation. Then a refinement step is performed sampling the edges and subdividing the triangulation to better approximate the shape. Galin [Galin 00] also proposes an incremental method leading to an adaptive mesh for skeleton based implicit surfaces.

The algorithm starts by creating an octree using a subdivision criteria based on the Lipschitz condition [Kalra 89]. The cell polygonization is performed using a lookup table such as Marching Tetrahedra [Bloomenthal 94]. However adjustments are proposed to correctly deal with ambiguities and adaptive cell sizes. When ambiguities are detected, it might require cell subdivision to avoid cracks on the final mesh.

Paiva [Paiva 06] presents an algorithm to generate an adaptive meshing which captures the exact topology of the implicit surface. The algorithm starts building an octree using three subdivision criteria based on the interval analysis of the implicit value and its gradient. The first criteria discards empty cells using interval of arithmetic, the second use the gradient value to catch topological features and the third criteria estimates the curvature from the gradient interval of analysis. This method is similar to [Azernikov 05] octree creation, however it relies on interval evaluation.Then the mesh is generated using an enhanced version of the Dual Marching Cubes [Schaefer 04], which use Lewiner [Lewiner 03] Marching Cubes implementation. Finally, mesh vertexes are shifted by vertex relaxation using the tangent plane defined by its normal and the barycenter formed by neighbor points. This method presents adaptive meshes with topology guarantee. However it does not approximate correctly singularities since it requires degree two of continuity. Bouthors [Bouthors 07] presents an adaptive meshing method for dynamic implicit models relying also on dual representation. This approach allows to generate up to four updated mesh per second of skeleton based implicit surfaces compound by a dozen of skeleton elements. The first representation is a mechanical mesh generated by Marching Cubes and which is optimized using a particle system technique to obtain a regular sampling of the surface and to better approximate sharp features using Ohtake [Ohtake 02] approach. Finally the visualization is offered through a second mesh named geometric mesh, which is based on the mechanical mesh and is generated by subdivision steps applied according to a subdivision criteria using mesh normal vector and implicit gradient analysis to achieve the final adaptive triangulation. This approach presents promising results as depicted by Figure 3.6, however only implicit models with few primitives (i.e meta-balls, convolution, skeleton based models) can be supported. Major topo-

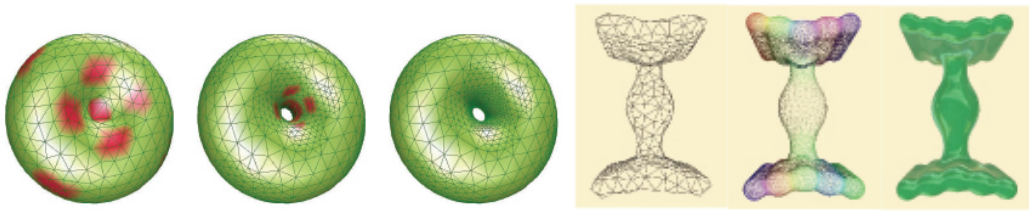


Figure 3.6: Dual Marching Cubes extension proposed by [Paiva 06] on left and adaptive mesh generated by [Bouthors 07] on right

logical changes of the implicit function cannot be approximated by this method such as the one leading to disjoint elements.

Tetrahedral cell decomposition was also used to achieve an adaptive mesh. Hall [Hall 90] polygonize algebraic implicit surfaces by using recursive adaptive subdivision of the space using tetrahedrons. Starting from a tetrahedra and its subdivision into several tetrahedrons, the shape is approximate by a set of tetrahedrons intersecting the surface. This approach was extended by Hui [Hui 99] which presents an adaptive marching tetrahedral algorithm for bounded implicit patches. The patch is initially enclosed by a tetrahedra which is subdivided into tetrahedrons according to the vertex value classification. Ambiguous tetrahedron/surface intersections are subdivided resulting in an adaptive polygonization algorithm. Crespin [Crespin 02] also proposes an algorithm based on tetrahedra. However, this method generates a dynamic triangulation for variational implicit surfaces [Turk 99] using incremental Delaunay Tetrahedralization. Using the constraint points of the implicit model, they create an extended bounding box which is subdivided into tetrahedrons instead of cubical cells. A refinement criterion based on the tetrahedron's circumscribe sphere is used to subdivide the tetrahedrons which are finally triangulated such as in Bloomenthal's approach.

These methods are able to produce adaptive meshes thanks to a non uniform spatial subdivision. However only Paiva [Paiva 06] method use local implicit characteristics such as curvature as criterion for the subdivision. The other methods only try to catch the correct topology of the surface by applying subdivision for a more robust surface/cell intersection classification instead of improving the curvature correctness of the approximation.

3.4.2 Surface tracking

Surface tracking algorithms such as Hilton Marching Triangle [Hilton 97] and Hartman [Hartmann 98] approach present more controlled and regular meshes than spatial decomposition methods since they try to sample the surface following the surface instead of sampling the space surrounding the implicit surface. However, their regularity is a problem when the edge step is not small enough to correctly approximate an high curvature surface area. On the other hand, on a low curvature area, such as Marching Cubes it may produce to many triangles. To better approximate implicit surfaces, the following approaches extended surface tracking methods to produce an adaptive meshing.

Akkouche [Akkouche 01] presents an extended version of Hilton Marching Triangle to produce an adaptive mesh and to support local re-polygonization when minor changes are applied to the implicit function. The algorithm is divided into two phases, first the growing phase adding triangles to the mesh if no intersection is detected when expanding an edge i.e. there is no triangle in the circumscribing sphere with the same orientation. Then the second step closes all the cracks that remains from the first steps. The adaptive mesh is achieved by constraining triangle edge lengths to local surface characteristics using both a mid-point projection heuristic and Delaunay triangulation properties, however it does not approximate correctly sharp features. Karkanis [Karkanis 01] presents a similar method than Akkouche Marching Triangle extension. However, instead of using the edge midpoint projection over the surface to define the edge length of the expansion, they choose to use a local curvature measure. The local curvature measure used is the radius of curvature which is computed using the minimum geodesic lying in the normal plane. The radius is computed using the angle between surface normal and point geodesic normal. Compared to Akkouche method, gap closing can create additional points to produce a smoother edge length transition of the mesh combining vertex relaxation, convex filling, edge flip and even subdivision.

Even with the usage of more reliable curvature estimator, both methods presented above are not able to correctly support sharp features. McCormick

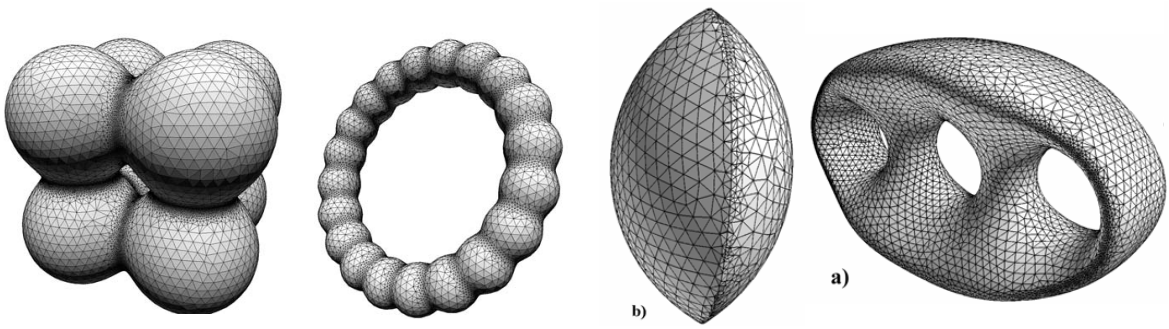


Figure 3.7: Adaptive meshes using surface tracking: right)Karkanis using geodesic based curvature estimation [Karkanis 01], left)Cermak adaptive edge-spinning [Cermák 04]

[McCormick 02] presents an improvement to the Marching Triangle more sensitive to sharp features such as edges and corners. This is done by constraining the Marching Triangle with the detection of C1 discontinuities. The detection of features creates a set of seed points for the algorithm to correctly approximate correctly the sharp feature (line fitting for edges). The resulting mesh, depicted in Figure 3.7 ,is able to better approximate sharp features than other surface tracking methods. Their improvements decrease the number of triangles with poor aspect ratio. However few results are presented by the author and it reliability to be applied over more complex implicit surface models is not illustrated.

Cermak [Cermák 04] also presents an adaptive extension of its edge-spinning algorithm. The logic of the algorithm remains the same as in the original version. However, the radius used to place the new vertex when the edge spin is now variable. This radius value is computed using a curvature estimation based on normal vectors. The radius is computed based on a constant factor over the angle between normal vectors, this allows to control the speed of the decrease/increase. A root finding adjustment is performed to better approximate sharp features computing the intersection of three planes (both from existing tangent, and the one from the circle, not clear) and the accurate location of the sharp point is made by binary subdivision.

3.4.3 Using Delaunay Triangulation

Several implicit surface meshing algorithms are based on Constrained Delaunay Triangulation following Chew[Chew 93] ideas. Chew presents a techniques to create high-quality triangular meshes for curved surfaces by generalizing Delaunay Triangulation to curved surfaces. By high-quality mesh, they refer to meshes with 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 over all the triangle and the mesh is updated by subdivision steps or edge-flipping until the mesh quality is reached and guaranteeing that the mesh exhibits Delaunay triangulation property. Using this approach Boissonat [Boissonnat 05] proposed an adaptive meshing algorithm for sampled surfaces which can be applied to implicit surfaces which are C^2-continuous and which gradient does not vanish. First, they search for the set of points of the surface that have horizontal tangent planes, i.e. 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 implicit surface is polynomial or use interval of analysis if the function is not polynomial. Following a surface tracking approach, triangles are added by finding the point with the smallest radius of curvature. At each step, Delaunay property is certified over the incremental mesh. This method presents a topological correct adaptive meshing. However, no global timing are presented in the article, only comparative relationships with the Delaunay triangulation and it does not approximate correctly sharp features.

Using the same sampling approach Cheng et al. [Cheng 04] propose an adaptive method relying on Delaunay Triangulation to triangulate smooth and compact surfaces without boundary. First, they starts by sampling the surface to define a set of seed points for the algorithm. However, points are added to the Delaunay Triangulation by a geometric sampling. First, the triangulation is improved in quality using Chew's approach [Chew 93]. Finally the adaptive triangulation is obtained by smoothing the triangulation according to a user's roughness threshold which uses vertex dihedral angles to represent a local estimation of the curvature. Following Cheng [Cheng 04] ideas, Dey [Dey 07] proposes an

adaptive meshing method for iso-surfaces. Starting from an initial 3D Delaunay Triangulation, they recover a ”rough” version of the surface applying Delaunay based criterions at each vertex insertion and proceed with ”poles” computation for each vertex to estimate the scale of the local features . Then a refine process is 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.

3.4.4 Post Remeshing

Another approach to generate adaptive meshes of implicit surfaces is to improve the polygonal approximation at the cost of an additional post-remeshing step. These methods use the result of an existing polygonization algorithm (most of the time using the Marching Tetrahedra) without any adaptation to the original algorithm. Then several steps are performed to the mesh to improve its quality, to be more sensitive to sharp feature, to become adaptive or to become more topologically correct. Rosh [Rösch 97] presents a post remeshing techniques based on particle based implicit surface rendering. First, an initial polygonization is performed using Marching Cubes. Then a particle system similar to [dF92, Witkin 94] is applied to mesh vertexes to optimized the mesh quality. Rosh also introduces a curvature estimator to create an additional spring mass which allows to improve the quality of the mesh and the approximation of cuts for meshes computed by the Marching Cubes algorithm. Neugebauer [Neugebauer 97] choose a different approaches by using subdivision over a coarse mesh obtained by Montani´s Marching Cubes [Montani 94a] to produce an adaptive triangulation which better approximates the implicit surface. The coarse mesh is improved by shifting the vertices of the mesh onto the center of gravity of surrounding polygons. Then, vertices located on a coplanar surface or along almost collinear edges are removed. Finally, a fix number of subdivision iteration is applied to mesh triangles as shows Figure 3.8.

A mix of these techniques was followed by both Ohtake works [Ohtake 01, Ohtake 02] on implicit surface polygonization in order to produce good approximations of surface sharp features (illustrated in Figure 3.9. In [Ohtake 01], an

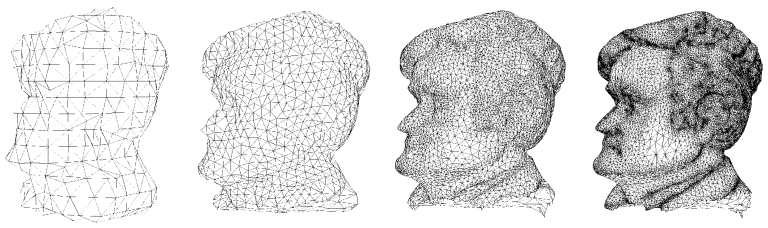


Figure 3.8: Adaptive meshing using subdivision over Marching cubes triangles [Neugebauer 97]

adaptive triangulation is produced by improving the mesh obtained by Marching Cubes to better approximate sharp features. The optimization is made on mesh vertexes combining the usage of three forces such as a 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 flow. At the cost of several iterations of the local operator, the resulting mesh better approximate sharp features and became of higher quality regarding regularity. Ohtake, also tested a different approach in [Ohtake 02] based on Dual Primal mesh concept which consists into two steps. The first creates the dual mesh based on triangle centroids and by projecting these vertices over the surface instead of directly using the vertexes obtained by Marching Cubes as it was done by Neugebauer approach [Neugebauer 97]. Then the second step optimizes the modified mesh vertexes by minimizing a quadratic energy function using Garland-Heckbert error metric. During these two steps, curvature-weighted vertex re-sampling (similar to the Laplacian smoothing) and adaptive mesh subdivision procedures are performed depending of the normal deviation. This method produces good results, however it requires a fine initial mesh to retrieve all shape measures and results in many calls to the implicit function during the post re-meshing process. A similar approach is followed by Peiro [Peiró 07] mixing also Laplacian smoothing with local modifications such as side swapping and an optimization step using a curvature estimation of the surface for triangle side collapsing .

The previous methods require additional information given by the implicit surface during the post-remeshing, which can become prohibitive for very complex implicit surfaces or very large triangular approximations. Several methods such as [Vorsatz 01, Wood 02, Attene 03] present feature sensitive re-meshing tech-

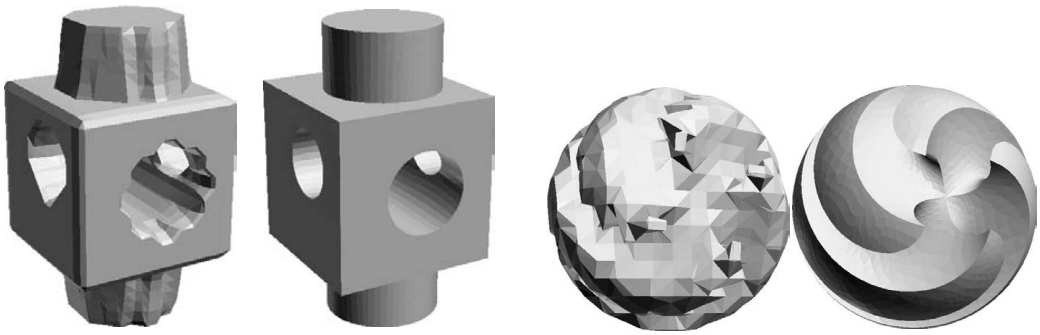


Figure 3.9: Ohtake remeshing technique over Marching Cubes: left) [Ohtake 01], right) [Ohtake 02]

niques applicable to meshes generated by Marching Cubes without using any implicit surface information. In [Vorsatz 01], several triangle operators are applied to improve the mesh. This approach was also used by other re-meshing techniques such as [Kobbelt 03] to generalized sensitive re-meshing techniques as a re-inverse engineering tool and complementing its previous work over iso-surface i.e. the Extended Marching Cubes [Kobbelt 01]. Attene [Attene 03] presents a re-meshing method sensitive to sharp features which is based on simple filter named Edge-Sharpener. The filter starts by classifying mesh edges, vertexes and triangles. Using the filter, smooth area are clustered which allows to identify the remaining triangles located near or over sharp features. Chamfer triangles are subdivided and their vertexes are optimized using a process similar to Ohtake normal error minimization [Ohtake 02] to better approximate sharp features. This method only re-mesh areas with sharp features . However, this method works correctly mostly over regular or almost regular meshes so it cannot be used over adaptive triangulations and it is influence by the initial mesh sampling to correctly catch all sharp features. Marching cubes such as any space decomposition algorithm can introduce non-existing topological features such as unwanted handles on the polygonal approximation of the surface. This problem can be solved using Wood re-meshing technique [Wood 02].First, handles are found using a reeb graph representation, then their relevance is evaluated. Finally insignificant handles are removed, which allows to remove extra erroneous topological handle. Thanks to reducing and removing unnecessary topological feature, it allows a better approximation for other meshing processing techniques in particular mesh compression

algorithms.

3.5 Polygonal Meshing Discussion

Taking in account the several concerns related with representation conversion, Table 3.2 presents the classification of existing polygonal approaches comparing their speed, how topology is reproduced, the quality of the sharp feature approximation, if the resulting mesh is able to approximate the curvature of the surface and the quality of the mesh. Looking at this table and Table 3.3, we can highlight two classes of techniques that produce the best polygonal approximations in term of mesh quality which are surface tracking algorithms [Karkanis 01, Cermák 04, Akkouche 01, McCormick 02] and re-meshing techniques over Marching Tetrahedra or Marching Cubes triangulations [Ohtake 02, Vorsatz 01, Kobbelt 03, Rösch 97, Neugebauer 97, Ohtake 01, Peiró 07, Attene 03]. These algorithms present better adaptive meshes and does not exhibit the triangulation pattern created by cubical space decomposition. Surface tracking techniques produce well defined adaptive meshes and are the more suited to use local shape information such as curvature during the generation of the polygonal approximation. On the other hand, re-meshing techniques show to be the more complete method allowing to not only produce an adaptive mesh but also to improve the sharp feature approximation using methods such as [Vorsatz 01, Ohtake 02, Kobbelt 03, Attene 03]. From the surface tracking approach, only [Cermák 04] proposes an adaptation to improve sharp features approximation. The sharp feature approximation of surface tracking can be performed easily by mixing interval of analysis techniques and optimization techniques such as [Ohtake 01, Ohtake 02]. Using the curvature information a robust detection method can be applied during the mesh expansion. Another advantage of surface tracking, compared to post re-meshing, is that existing algorithm could be improved to achieve similar results on the fly.

Regarding the algorithm speed and its implementation, both techniques are easy to implement and depend on the complexity of the mesh for re-meshing techniques and the area to be covered for surface tracking approaches. The main

		Speed	Topology	Features	Smoothness	Mesh Type	Mesh Quality	Techniques	Notes
S.D.	[Wyvill 86]	☆	✕✕		✕✕	P	✕	S.D.	
	[Velho 99]	≃	☆☆	☆	☆☆	A	☆	S.D.	
	[Hall 90, Hui 99]	≃	☆☆	✕	☆	PA	✕	S.D. Tet	
	[Galin 00]	≃	✕	✕	☆	A	☆	S.D.	For Skeleton based IS
	[Crespin 02]	≃			☆	A	☆☆	S.D. Delaunay	For VIS
	[Alberti 05]		☆☆		☆	PA	✕	S.D.	
based on Marching Cubes	[Lorensen 87]	MC.	≃	✕✕		✕	P	✕	
	[Montani 94b]	☆	☆☆		✕	P	✕	MC. Lookup	For Iso-surface data
	[Hege 97]	☆	☆☆		✕	P	✕	MC. Lookup	For Iso-surface data
	[Yamazaki 02]	☆	☆☆	☆☆☆	✕	P	✕✕	MC. Lookup	Non-Manifold
	[Chernyaev 95, Lewiner 03, Lopes 03]	☆	☆☆☆		✕	P	✕	MC. Lookup	For Iso-surface data
	[Nielson 04]	≃	☆☆☆		☆☆	R	☆☆	MC. Lookup	For Iso-surface data
	[Nielson 03]	≃	☆☆☆		✕	P	✕	MC. NoLookup	For Iso-surface data
	[Renbo 05]	☆	☆☆☆	☆	✕	P	✕	MC. NoLookup	
	[Bloomenthal 95]	☆	☆☆	☆☆☆	✕	P	✕✕	MC. NoLookup	Non-Manifold
	[Rösch 97]	☆☆		☆☆	☆☆	A	☆☆☆	MC. Remesh.	Particle
	[Neugebauer 97]	☆☆		☆☆	☆☆	A	☆☆☆	MC. Remesh.	Dual Mesh
	[Bouthors 07]	≃	✕	☆☆	☆☆	A	☆☆	MC. Remesh.	For Skeleton based IS
based on Marching Tetrahedras	[Bloomenthal 88, Bloomenthal 94]	☆	✕		✕	P	✕✕	MTet.	
	[Velho 96, Siqueira 98]	☆	☆		☆	PA	✕	MTet.	
	[Treece 99]	≃	✕	☆	☆	R	☆☆	MTet.	
	[Ning 93]	☆	✕		✕	PR	✕	MTet. Subdiv.	
	[Chan 98]	☆☆	✕		☆	P	✕	MTet. Subdiv.	Lesser triangles
	[Triquet 01, Triquet 03]	☆☆			✕	P	✕	MTet. Speedup	For Blobs-like IS
	[Cuno 04]	☆☆			✕	P	✕	MTet. Speedup	For VIS
	[Zhang 06]	☆☆			✕	P	✕	MTet. Speedup	For Distance Field IS
	[Schmidt 05]	☆☆☆			✕	P	✕	MTet. Speedup	For BlobTree
	[Plantinga 04, Plantinga 07]	☆	☆☆	✕	✕✕	PA	✕	MTet. Topo.	
	[Ohtake 01]	☆☆		☆☆	☆☆	A	☆☆☆	MTet. ReMesh.	Like particle system
	[Ohtake 02]	☆☆		☆☆☆	☆☆☆	A	☆☆☆	MTet. ReMesh.	Dual Mesh
	[Peiró 07]	☆☆		☆☆	☆☆	A	☆☆☆	MTet. ReMesh.	Laplacian
	[Kobbelt 01]	≃	☆	☆☆	☆	P	✕	Extend. MC	Distance Field Simpl.
	[Ju 02]	≃	☆	☆☆	☆	PA	☆	DualContour	
	[Azernikov 05]	≃	☆	☆	☆	PA	☆	DualContour	
	[Varadhan 03, Varadhan 04]	≃	☆☆	☆☆☆	☆	PA	☆	ExtDualContour	N feat. per cells
	[Schaefer 04]	≃	☆	☆☆	☆	P	✕	DualMC.	N feat. per cells
	[Varadhan 06]	≃	☆☆☆	☆☆☆	☆	PA	☆	Visibility Map	N feat. per cells
	[Liu 05]	≃	✕	✕	☆	R	☆☆	Particles	For Skeleton based IS
	[Paiva 06]	≃	☆☆	☆☆	☆☆	A	☆☆	DualMC.	Requires C2 function
Alternative Approaches	[Hilton 96]	VA			☆	R	☆☆	MTri.	
	[Hartmann 98]	VA			☆	R	☆☆	MTri. like	
	[Cermák 02]	VA			☆	R	☆☆	MTri. like	
	[Akkouche 01]	VA			☆☆☆	A	☆☆	MTri. like	Use Curv. estimation
	[Karkanis 01]	VA			☆☆☆	A	☆☆☆	MTri. like	Curv info: geodisc
	[McCormick 02]	VA		☆☆☆	☆☆☆	A	☆☆	MTri. like	Use Curv. estimation
	[Cermák 04]	VA		☆☆	☆☆☆	A	☆☆☆	MTri. like	Use Curv. estimation
	[Boissonnat 05]	✕✕	☆☆☆	☆	☆	A	☆☆	Delaunay	Use Curv. estimation
	[Cheng 04]	✕✕	☆☆	☆	☆	A	☆☆	Delaunay	
	[Dey 07]	≃	☆☆	☆	☆☆	A	☆☆	Delaunay	Curv info: Amenta poles
	[Stander 97]		☆☆☆	☆	☆	A	☆☆	Inflation	Requires C2 function
	[vO04]		☆	✕	☆	A	☆☆	Shrinkwrap	For Skeleton based IS
	[Bottino 96]		☆☆	☆	☆	A	☆☆	Shrinkwrap	
	[Vorsatz 01, Kobbelt 03]	☆☆		☆☆☆	☆☆☆	A	☆☆☆	Full ReMesh.	Particle
	[Wood 02]	≃	☆	☆	☆☆		☆☆☆	Full ReMesh.	Reeb graph
	[Attene 03]	☆☆		☆☆☆	☆☆	A	☆☆	Partial ReMesh.	Filter Based

Table 3.2: Polygonal approaches comparison: worst ✕ to more adequate ☆ classification for each category. The mesh type column describes the mesh: Adaptive (A), Regular (R), exhibiting pattern from spatial decomposition (P). On the speed column, 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. Re-Meshing, Subdiv. Subdivision, S.D. Spatial Decomposition, Topo. Topology.

difference regarding the generation time, is that usually re-meshing algorithms depends of the number of triangles or vertexes of the mesh and the quality of the initial mesh. The initial mesh due to be extracted from spacial decomposition,

its generation time can be more controlled for a given precision due to the finite aspect of the method. Regarding surface tracking methods, the generation time is not deterministic and ends when all the surface is covered by the mesh. However, the several surface tracking algorithms showed that they can be as fast as spatial decomposition algorithms and they need less implicit surface evaluation to produce equivalent mesh with a better triangle distribution. The main concern, regarding surface tracking algorithm, is to guarantee that mesh overlap is avoided during the mesh expansion which can be easily done using fast collision detection tests during the mesh creation.

Topology	Features	Smoothness	Mesh Quality
[Stander 97]	[Ohtake 02]	[Ohtake 02]	[Karkanis 01]
[Varadhan 06]	[Vorsatz 01, Kobbelt 03]	[Vorsatz 01, Kobbelt 03]	[Ohtake 02]
[Boissonnat 05]	[Attene 03]	[Cermák 04]	[Vorsatz 01, Kobbelt 03]
[Renbo 05]	[Varadhan 04, Varadhan 06]	[Akkouche 01]	[Cermák 04]
[Nielson 03, Nielson 04]	[Bloomenthal 95]	[Attene 03]	[Rösch 97]

Table 3.3: Best five methods for each category

From both techniques mentioned above the major drawback is the topological correctness of the approximation, since none of these methods can guarantee by itself the isotopy between the polygonal representation and the implicit surface. Regarding re-meshing techniques, none of them improve the topology beyond Woods [Wood 02] techniques. However this technique does not provide any guarantee over the topology. Regarding surface tracking techniques, they can produce correct results to approximate surfaces with non-disconnect elements but require additional steps to support disconnected elements. The topological basis used by the delaunay based methods such as [Boissonnat 05, Cheng 04, Dey 07] could be used to overcome this issues. Topology guarantee is mostly offered by methods studying the critical points of the implicit surfaces such as [Stander 97, Bottino 96]. Regarding mesh generation for iso-surface, this problem is overcome using space decomposition improvements to deal with the topological ambiguities [Lewiner 03, Nielson 03, Nielson 04, Renbo 05]. From these methods, we should highlight Nielson[Nielson 04] approach which produces a smoother polygonal approximation at the cost of an iterative step.

3.5.1 Curvature usage for Adaptive meshing

All the methods generating adaptive meshes try to capture local shape characteristic through curvature estimation. Akkouche [Akkouche 01] expands edges by projecting the midpoint of the active edge on the surface using the gradient of the implicit function. Then the new point is placed according to the gradient deviation analysis which give clues about the curvature of the shape. The new point are placed a variable distance constrains to produce equilateral triangle. The method does not use a direct curvature estimation but is able to capture part of the smoothness of the shape. Cermak [Cermák 02, Cermák 04] approach is very similar to this method. It generates the adaptive mesh checking the angle between two normal vectors. The normal vector at the middle point of the active edge to be expanded and the normal vector located at a fixed distance form the middle point on the mesh triangle plane. Then, depending of a threshold value over the angular measure, a smaller radius is computed to define the new location of the point generated by the edge spinning algorithm. The Cermak [Cermák 04] algorithm seems to present more attractive results than Akkouche and tries to approximate sharp features, however both of them are illustrated with simple models. Karkanis [Karkanis 01] proposes a more direct usage of the curvature by trying to estimate the radius of curvature. This is done using geodiscs which also use normal vector deviation, however several measures are used and the method provides a more reliable curvature estimation than both previous method. The several estimations enable to define a minimum radius of curvature value which is used to place point on the tangent place. However this is done at the cost of many implicit function evaluation, which can be prohibitive depending of the implicit surface model used. Karkanis mentioned that between 27 to 50 function evaluations are need by triangle.Using a different polygonal approach, Paiva [Paiva 06] generates an adaptive mesh using local subdivision where the curvature is high. The curvature is estimated by studying the variation of the gradient at the proximity of the point (bounded volume) and using a threshold. This approach is very similar to the one used by surface tracking algorithm. More recently, Bouthor [Bouthors 07] also compared normal vectors at vertices on edges and between adjacent triangles, and compute the angle between them as criteria

for the subdivision. Depending of the angle, the triangle is subdivided until the angle is less than a given threshold.

None of the previous methods use real curvature measures extracted from the hessian matrix of the implicit function. However the hessian matrix can produce robust curvature values, at the cost of differential geometry which can be computed with a cost similar to the implicit evaluation itself. Most of correct topological approximation of implicit surfaces relies on critical point analysis as shows [Stander 97, vO04, Boissonnat 05] works. On the other hand, several approaches have already used hessian based curvature analysis for other purpose such as feature line extraction [Pasko 88, Bogaevski 03, Ohtake 04].

3.5.2 Recommendations

Even if direct visualization methods such as ray-tracing produce the most fidel visualization of implicit surface, conversion algorithms to polygonal representation demonstrate to be not only the more popular due to their support by nowadays graphics cards but also the more controlled visualization method. A real-time representation is the most attractive approach in particular if its quality can be guaranteed. Existing polygonization algorithms of implicit surfaces have focused in improving the main important concerns of any conversion mechanism into a piecewise representation. Polygonal approaches are able to guarantee topological correctness, high quality approximation of surface features such as edge and corners, an adaptive representation to local shape features such as normals and curvature and a good mesh quality for this alternative representation to be used by several application i.e in the scope of computer graphics or scientific simulation. From the existing methods surface tracking and post re-meshing techniques propose the more adaptive polygonization of implicit surfaces. Surface tracking is the more adapted strategy to use local implicit information during the mesh generation such as curvature information. As opposite to re-meshing techniques, surface tracking are able to generate an adaptive representation in one step.

However several issues should and could be improved to present a more ro-

bust method for the polygonization of implicit surface. Topological correctness and sharp features reconstruction is still only achievable at the cost of complex approaches or are still too approximative. A better usage of the extraction of the local mathematical information of the implicit surface can be made to offer more guarantee on the reconstruction of sharp features in order to produce results so robust than re-meshing techniques in the sharp reconstruction field. On the other hand, gradient and curvature information can be more useful in surface tracking approach to guarantee the topological correctness of the polygonal approximation. We note that existing approaches still under used curvature information considering only estimative approaches rather than heuristics based on real curvature measures. Regarding the topology, interval of analysis as proved to be a robust basis for topological correctness in space decomposition algorithms and could be adapted to be used by surface tracking algorithms.

3.6 Summary

In this chapter, we had surveyed most of existing techniques and their issues for the visualization of implicit surface. We introduced the several techniques to visualize implicit surface and discussed the issues related with the type of visualization, i.e topology correctness, sharpness and smoothness fidelity and visualization or conversion quality. Then we have focus on the issues related with the improvement of polygonal approximation since it is the more suited for existing graphics hardware and the strategy followed by our algorithm.

4

Polygonization Algorithm

We present a new surface tracking polygonization algorithm in order to generate meshes adapted to local feature of the shape on the fly. Given any surface defined implicitly such as the surface is represented by the zero value set of an implicit function, we generate an adaptive polygonal approximation based on expanding points near to the surface. This is done by evaluating the implicit function extracting implicit values, gradient vectors and hessian matrixes. By expanding points, new triangles are generated and add to the mesh in order to create a faithful piecewise approximation of the surface.

Our algorithm starts from a seed point which is projected on the surface, then we expand it to generate a set of triangles which will be the starting triangulation of our meshing algorithm as depicted by the first step illustrated by Figure 4.1. Beyond the new triangles, a set of unexpanded points is created as result of the first expansion. This set of points is stored in a bucket which will be used by the algorithm as the border of the triangular approximation. An iterative process is applied over the set on unexpanded point trying to expand each points in order to extend the triangular mesh. Each expansion can result in the generation of both new triangles and new unexpanded points. After its expansion, the processing of a point is finished and it is removed from the unexpanded point bucket. The algorithm tries to expand all the points of the bucket until it is empty which is achieved when the mesh is closed by itself i.e when all the surface area of the implicit surface is covered by triangles.

By using a surface tracking approach, we are able to control the density of the triangles and their edge length during the tessellation algorithm to better approximate the surface. This is achieved thanks to an heuristic which defines the more

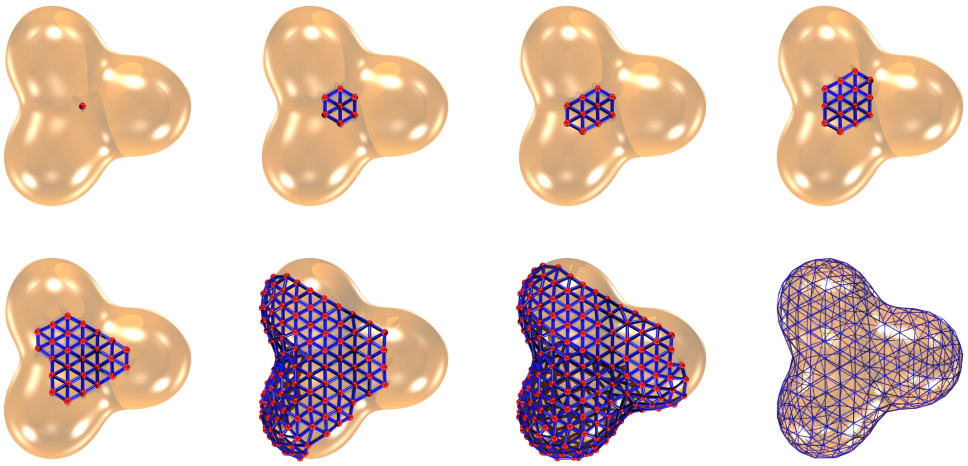


Figure 4.1: Step Sequence of our approach in order to polygonized a simple meta-ball implicit model. Starting from the initial triangulation of the seed point the mesh is completed by successive point expansions until covering the whole surface

suited edge length for the new triangles based on the mathematical characteristic which can be extracted from the implicit representation such as its curvature. The edge length determination using the curvature allows to generate small triangles in areas of high curvature and larger triangles in areas of low curvature. However such as other surface tracking algorithms, we need to guarantee that new triangles will not overlap existing triangles. This is done thanks to a collision test which is performed before the point expansion. Depending of the result of this test, a point can be attached to a colliding point instead of being expanded. Since the mesh expansion does not rely on homogeneous edge length, collisions might occur. However this can be caused due to other features, such as the genus characteristic of the shape or even due to unstable values of the implicit function evaluation. Our polygonization is robust enough to overcome these problems thanks to an efficient test which guarantees that mesh overlapping is avoided leading our approach to the generation of a triangular approximation in a finite time. We would like to highlight that the lack of support regarding collisions would make our algorithm to generate new triangles indefinitely. The following sections present a more detailed description of each step of our polygonization approach highlighting its robustness and its versatility to approximate shapes described by implicit surfaces.

4.1 Data structures

Points are the main entity for our polygonization and can represent any expanded or unexpanded points of the polygonal approximation generated by the surface tracking process. Unexpanded points are located on the boundary of the mesh and its set represents all the incomplete part of the mesh. In order to support expansion for the generation of new triangles, each new point is identifiable and it is possible to easily detected its neighbors. Beyond its location and connectivity, we also attach the following information:

- position and normal vector: information for the mesh definition
- principal curvature values and vectors: used to define tangent plane of expansion
- adjacent points and front angle: defines the status of the polygonization (unexpanded part)

This information will enable us to perform the expansion of any point to generate new triangles for the mesh. The mesh data is just the output of the algorithm storing indexes of triangle vertices which are the identifier used by our point representation. In fact only two data structures are used during the mesh generation which are the list of unexpanded points and a spatial hierarchy storing all the points used to avoid mesh overlapping.

4.2 Polygonization Cycle

The polygonization starts with the seed point which is projected over the surface. The projection is performed applying successive Newton steps over the point until the implicit value is less than a predefined threshold. The Newton's method is a well known iterative process to solve polynomial equations which is the basis of most of implicit surface models. The method only requires the computation of both the implicit value and the gradient at each iteration and is known to converge quadratically. During the polygonization, each new point generated

```
Polygonization( implicit-surface )
 1  pt0 ← Get-Seed-Point( implicit-surface )
 2  L ← Create-Empty-UnExpanded-Point-List()
 3  M ← Create-Empty-Mesh()
 4  Initial-Expansion(pt0, L, M)
 5  while Is-Not-Empty(L)
 6  do
 7     pt ← Get-Point-Minimal-Angle(L)
 8     Remove-Point-From-List(pt, L)
 9     ptI ← Check-Collision(pt, L, M)
10     if ptI = nil /* no collision */
11        then Expand(pt, L, M)
12        else Link-points(pt, ptI) /* duplicates pt and ptI */
13             Expand-Duplicate(pt, L, M)
14             Expand-Duplicate(ptI, L, M)
15
16  return M
```

Figure 4.2: Algorithm pseudo-code

by the expansion is also projected on the surface using the same approach which usually only requires few iteration since the expansion generates new points located on the tangent plane of the surface at the unexpanded point location. By expanding the projected seed point on the surface, our algorithm starts to create six new points and six triangle generating an hexagon. Each new point is projected using the Newton method near to the surface and the local characteristic of the implicit function are extracted and store in the unexpanded point list. Then we expand points iteratively until the list is empty. Each expansion results in the generation of new points and processed points are automatically removed from the list.

As mentioned before, expansions can only proceed if it is guaranteed that the triangles will not overlap existing triangles of previous expansions that are now part of the resulting mesh. This can be controlled by selecting the point to be expanded according to the angle form with its neighbors on the border of the mesh. This approach was already followed by Hartman [Hartmann 98] who choose as candidate point for the expansion, the point with the minor front angle. Hartman organized unexpanded points into fronts which were the boundary representation

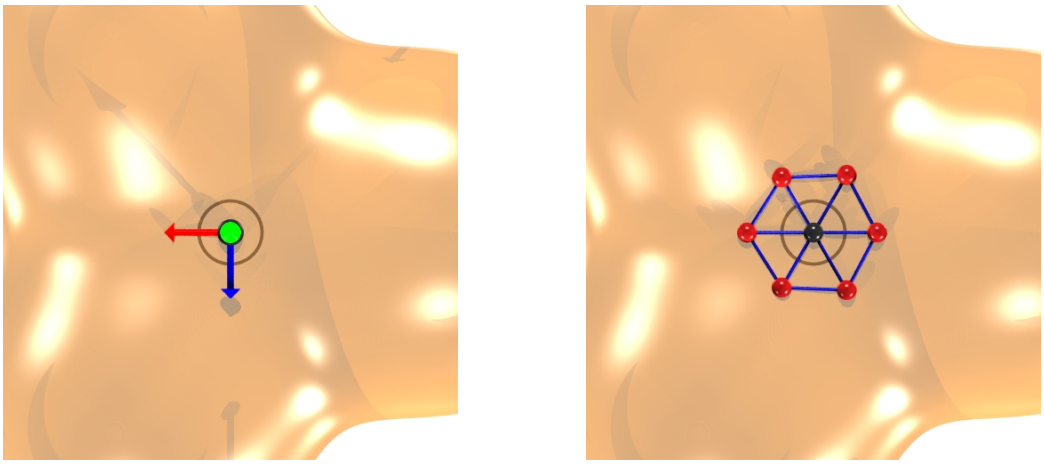


Figure 4.3: Expansion applied to the initial point, since there is no neighbors six new triangles and points are created over the tangent plane

of the mesh. However this representation requires an additional management since fronts might split or merge during the mesh expansion. Our approach does not require an explicit representation of the mesh boundary, however we face the same problem regarding deciding which unexpanded point might be more suited to be expanded. We tested several strategies by applying several ordering policy to the expansion list from non ordered such as FIFO and LIFO to ordered by minor or major angle. Due to the nature of the expansion which tries to generate most equilateral triangles, the ordering of the unexpanded point list by minor angle showed to generate less collision than other approaches.

After selecting the candidate point, we estimate the distance which will be verified between the current point and new points computed from its expansion. This estimation is computed based on the implicit derivatives analysis or a constant value is used. Then the collision test is performed searching all the points located at a distance minor than the estimated value. The collision test return the nearest colliding point when the expansion is unsafe. In this scenario a connection is create between the unexpanded point and the colliding point. Local expansion is performed using this connection. When the point is safe, the expansion step is applied generating up to four points and up to five triangles. However it is possible that the expansion does not create any new point and simply create a single triangle using the unexpanded and both neighbors. Then this process is applied to the remaining point of the expansion list until it become empty. Figure 4.2

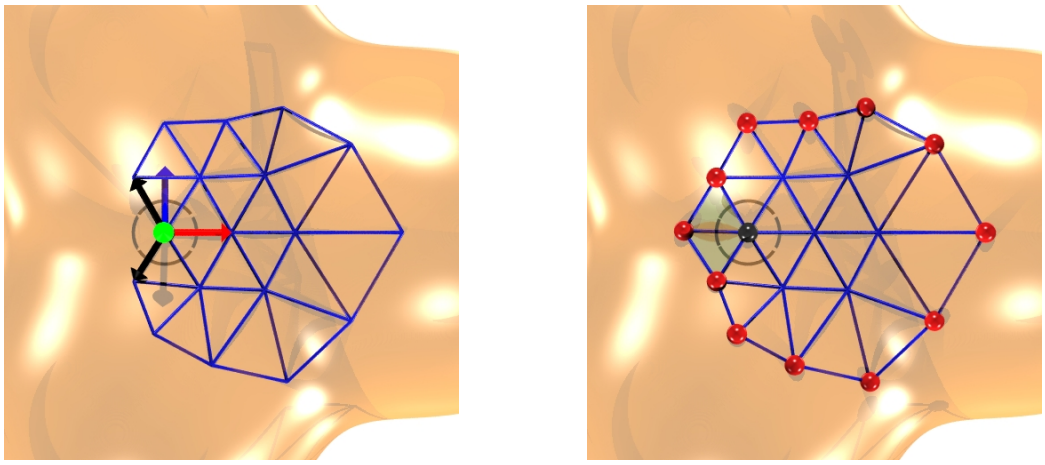


Figure 4.4: Expansion of the candidate point using the angle formed with both neighbors, resulting in two new triangles and one new point to the unexpanded point list.

presents the pseudo code of our algorithm.

4.3 Expanding points

For each candidate point, the expansion generates new triangles to be added to the mesh approximating the implicit surface and new unexpanded points to the expansion list. The number and the position of new points and resulting triangles is defined using the front angle θ and the estimated expansion distance d. As depicted by Figure 4.4, the front angle θ is computed projecting both adjacent point over the tangent plane defined by the principal curvature directions. The number n of triangles needed for the complete expansion of the candidate point and its incident angle θ is computed as follows: $n = floor(\frac{3\theta}{\pi})$ and $\theta_{out} = \frac{theta}{n}$

The expansion tries to create quasi-equilateral triangles. When the front angle is smaller than $\pi/3$, we do not create any new points. Rather, we extend the mesh with a single triangle formed by the candidate and adjacent points in the front. In all other cases, to correct extreme case due to truncated function, we apply the adjustments proposed by Hartmann's [Hartmann 98] using our heuristic edge length for distance checking with adjacent points to the candidate point.

According to the number $(n - 1)$ of new edges required, we place new points

on the tangent plane at a distance to the candidate point given by the heuristic. These are then projected on the surface using a Newton Step. This procedure requires few iterations depending on the desired precision. Finally, we extend the mesh with triangles joining the new and candidate points. The candidate point is then removed from the expansion list and the adjacency information of both neighbors of the point are updated considering the new triangulation layout as shown in Figure 4.4.

4.4 Controlling edge length

The length of all triangle edges is controlled by heuristic used by the point expansion. Several heuristics were tested in order to produce an adaptive mesh as output for our algorithm. To produce a constant triangulation of the implicit surface, we just need to apply a constant value instead of the heuristic based on implicit values. Different strategies were followed to produce and control the quality of the adaptive mesh. We used implicit curvature value extracted from the implicit function, we mix the heuristic from a candidate point with its neighbors (weighting adjacent heuristic values depending of the relative position of the neighbors). We also used additional information from the implicit surface representation such as the octree level when dealing with Ohtake's MPU representation.

By extracting the curvature information of the implicit function, we are able to adapt the triangulation to local features, generating larger triangles in flat region than in high curvature areas of the surface. As presented in Chapter 2, several curvature measure can be extracted from the implicit function using the gradient vector and the hessian matrix. In our algorithm, we tested the following three curvature measures as basis for the heuristic edge length:

- Mean Curvature
- Gaussian Curvature
- Absolute Maximum of Curvature

- Shape Index

The edge length heuristic will scale the curvature measure according a scalar value defined by the user. Beyond the scaling, since the implicit function can generate an infinite interval of curvature values depending of the implicit surface representation, we complement the scale with two thresholds to bound the minimum and the maximum accepted edge length heuristic. By bounding the value, we avoid generating too tiny or too large triangles which allow us to control the triangle budget used by the mesh and the quality of the approximation.

The curvature usage cannot guarantee a fixed error measure of the polygonal approximation since we do not study its derivative. For this reason, the curvature can be only used as heuristic and not as a direct method to define the best edge triangle size. To provide a more robust edge length definition more aware of the possible curvature variation, our algorithm also evaluate the heuristic on the neighbors of the candidate point and use as final edge length for the new points a weight sum based on the candidate point value ,its neighbors and the distance between them. Thanks to this approach, our algorithm is less sensible to high curvature variation and provides a mesh with a smooth triangle size transition.

4.5 Avoiding Mesh overlap

As mentioned before, point expansion can only proceed if it is guaranteed that the new triangle resulting from the expansion will not overlap existing triangle. This is a common problem to all surface tracking algorithms which requires that we need to prevent any collision before performing the expansion by classifying the safeness of point expansions depending of their location and the estimated edge length provided by the heuristics. The collision detection consists on identifying all visible points located near to the candidate point. These points might be unexpanded points located on an opposite boundary or the mesh i.e with a greater front angle. However due to imprecision that might occur on the implicit definition, they can also be already expanded points that are now points of the mesh. Since, the test need to be call for each expansion, the method to gather

neighbor points of any points should be fast. Our approach proposes a spatial hierarchy which is constructed and updated along all the polygonization process. We use a octree data structure to store all the points generated by our algorithm. The root of the octree is established at the beginning of the polygonization as an input parameter which bounds the implicit surface on the three axis. Then each new unexpanded point generated by the algorithm is stored in the octree. We specified a maximum number of points N_{max} per cell which is used to dynamically subdivide and construct the octree.

The test to avoid mesh overlap is performed searching all the points of the octree which are located in a sphere centered in the candidate point and with a radius equal to the estimated edge length. The list of point is fast to computed due to the spatial partitioning of the octree which enables to find intersecting cells trough interval analysis on the three axis. For each cell which intersect the sphere , we verify the distance from the candidate point to the points located within the cell. The index of these points are stored in a list as possible colliding points. Then the list is processed and a visibility test is performed using two planes. Each plane is defined by the candidate point and the perpendicular vector to adjacent edges as normal vector. The remaining points are colliding points which are classified to define the nearest point located on a boundary of the mesh and the nearest point located within the mesh. If the nearest point is located on the border, a connection is created on the mesh, otherwise the expansion is cancelled. Since part of the efficiency of our algorithm depends on the collision test and the underlying octree structure, we complement it with a simple cache mechanism which store the last M visited cells. This allows to take advantage of the spatial coherence of the mesh propagation and speedup the access to the octree data-structure. The cache is used to quickly located the correct cell to store new unexpanded points and reduce the cost of traversing the tree from the root node.

Collisions with border points of the mesh can be easily overcome by creating a new edge between the candidate point and the colliding point instead of proceeding with the normal expansion. Since our algorithm does not support edges and unexpanded points cannot be duplicated, we perform two local expansion one for the duplicate candidate point create by the edge and another

for the colliding point. The local expansion ensure that it is not possible to find any unexpanded points which does not delimit the triangulated area from the non-triangulated area with the edge formed with its neighbors. Regarding the collision scenario with points located within the mesh, the expansion is aborted and the point remains processed but unexpanded. This scenario is rare, however it might happen due to unstable values of the implicit surface representation. The algorithm is not able to prevent these errors, inconsistent value are checked during the convergence method and might also occur during the polygonization. However these scenarios are considered and support robustly by our approach generating possible cracks that can be repaired after the polygonization.

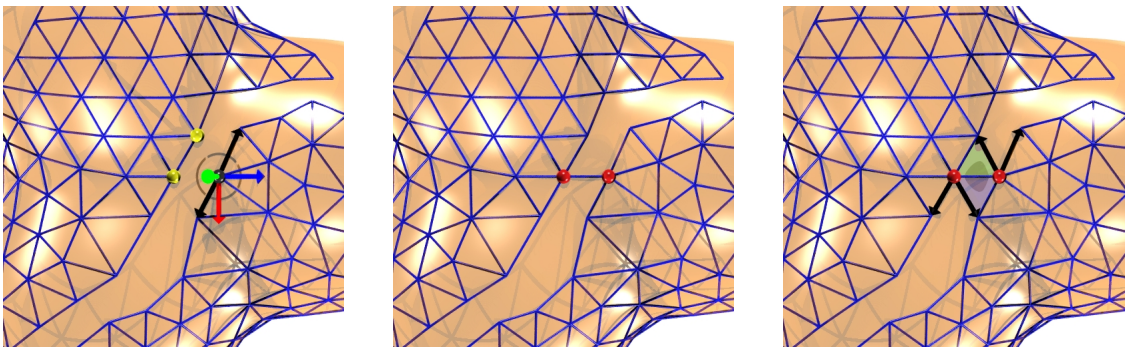


Figure 4.5: Mesh overlapping avoidance: the collision test detect colliding points, then a new link is created between the candidate point and the closest colliding point, finally two local expansions are applied avoiding point duplication on the unexpanded point list

4.6 Summary

This chapter has presented our surface tracking polygonization approach which is able to generate a adaptive mesh in a single step. Starting from a seed point projected on the surface by the successive Newton's step, a initial hexagonal triangulation is created expanding the point on the tangent plane to the surface. Then the border points of the mesh are considered as unprocessed points which need to be expanded in order to create new point and new triangles to cover all the surface. The set of points and triangles forms the output of our algorithm providing a polygonal approximation of the implicit surface. The local control

of the polygonization is achieved thanks to an heuristics which try to estimate the most suited triangle edge length. This is done using as basis of the heuristic, local characteristic of the shape extracted from the implicit function such as the curvature values which can be computed using both the hessian matrix and the gradient vector at the unexpanded point location. Due to the surface tracking approach, we need to guarantee that mesh overlapping is avoided during the expansion. This is done thanks to an efficient collision test which relies on an octree data-structure to find possible colliding neighbors. The octree data-structure is complemented with a simple cache mechanism is order to take advantage of the spatial coherence of the mesh propagation.

5

Experimental Results

Our algorithm was tested using three different classes of implicit surfaces. For each class the following interface was used to gather the implicit values needed by our algorithm:

- *Get_Value*(X) : Returns the value of the implicit function for a given point $X \in \mathcal{R}^3$.
- *Get_Gradient_Vector*(X) : Returns the gradient vector for a given point $X \in \mathcal{R}^3$.
- *Get_Normal_Vector*(X) : Returns the normal vector for a given point $X \in \mathcal{R}^3$.
- *Get_Hessian_Matrix*(X) : Returns the Hessian matrix for a given point $X \in \mathcal{R}^3$.

These functions are straightforward to code and only depend on the surface representation. To speed some calculations up we also implemented composite functions such as *Get_Value_and_Gradient*() for the Newton Step and *Get_Gradient_and_Hessian*(), since for each curvature evaluation we compute the curvature at a point, we also need its normal vector. Since the Hessian matrix reuse the calculations performed to compute the normal vector, we calculate them efficiently at once to avoid doing extra work.

We tested simple analytical implicit surfaces such as spheres which present constant curvatures, toruses and general quadric implicit surfaces. We choose Multi Partition of Unity [Ohtake 03] and Variational Implicit Surfaces [Turk 99] since they can be constructed from data point sets which allows to test a big variety of shapes. On the other hand using these two models, we are able to illustrate the versatility of the curvature based polygonization using two different

Model	*Points*	*Characteristic*	*Origin*
Foot	263542	Sharp Feature plus local high curvature regions near to the fingers	NA
Horse	96966	Smooth shape with variable size parts	Greg Turk Archives
Bunny	69451	Smooth shape with incomplete data on the bottom	Stanford [Sta07]
Venus	268686	Smooth model used in [dA04]	Cyberware [Cyb]
Dragon	871414	Well defined different curvature areas	Stanford [Sta07]
Armadillo	172974	Complex model with local details	Stanford [Sta07]
Buddha	543652	Complex data set with non-zero genus	Stanford [Sta07]
David head	827181	Complex model with well defined curvature areas and difficult curly hair topology	Michelangelo Project
Lucy	14027872	Very Large data-set	Stanford [Sta07]

Table 5.1: Characteristic of the point data set used for the evaluation of our algorithm

classes of implicit surfaces which are globally defined radial basis functions and hierarchical implicit functions based on the blending of local shape functions. Most of the testes were made reconstructing well known large data sets using MPUs to generate complex implicit surfaces. All the times generated during our evaluation were measured using a laptop computer with a Intel Centrino processor at 1.6 GHz, with 768Mb of RAM and running Microsoft Windows XP Tablet PC Edition.

The characteristics of the data sets used for the evaluation are the presented in Table 5.1. All the models were resized in order to fit in a cubical cell of 40 units to apply similar curvature heuristics. The configuration used by the MPU implicit reconstruction was the following: approximation error of 0.005 with a support size of factor 0.75 allowing the usage until 20 levels, each cell local shape was configured to use a minimum of 15 points with λ equal to 0.1. The heuristic based sharp feature reconstruction was activated with the default values (corner at 0.7 and edges at 0.9). For more details about the meaning of each parameter, the reader should refer to author's article [Ohtake 03]. The source code of the MPU reconstruction used to test our algorithm is available at [MPI07] (the code was

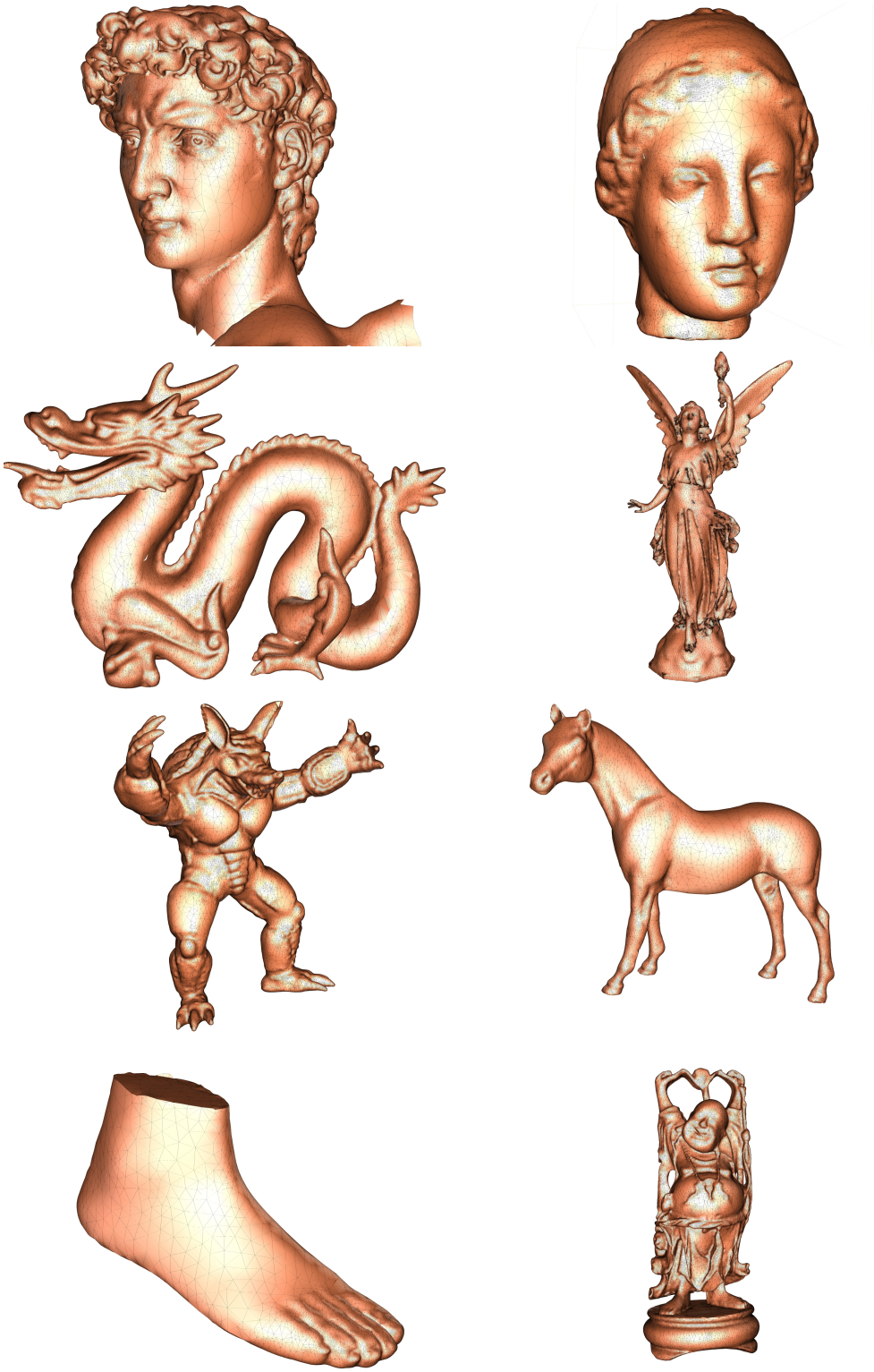


Figure 5.1: Polygonization results of several MPU models

adapted in order to generate the implicit functions mentioned above). Regarding the variational implicit surfaces, we followed [Yngve 02] approach to generate a version of the Stanford Bunny with 4004 constraint points, which is the same as the one used by our previous polygonization approach based on front expansion presented in [dA04].

5.1 Computational analysis

Since our algorithm uses a surface tracking approach, it is difficult to measure its complexity. The best criterion for performance comparisons is the time of polygonization. Table 5.2 presents results for different models; as we can see meshing times depend on the complexity of surfaces, not only on the curvature calculation but also on their area as related to our heuristics limits on the radius of curvature. The column with the number of triangles generated per second shows that meshing rate depends on the complexity of evaluating the implicit surface properties at each sample point.

Model	*Time*	*Triangles*	*Triangles*	*Triangles*	*Ratio L/s*	
	second		*Per second*	*Per expansion*	Avg	St.Dev
Foot	2.459	17711	7202.52	2.028	1.497	0.790
Horse	8.017	57860	7217.16	2.003	1.490	0.526
Bunny	12.805	75762	5916.6	2.007	1.588	0.694
Venus	18.559	93352	5030.01	2.011	1.533	0.628
Dragon	33.795	206027	6096.38	2.003	1.417	0.562
Armadillo	33.408	214475	6419.87	2.003	1.440	0.445
Buddha	33.674	254269	7550.9	2.006	1.380	0.445
David head	85.909	436765	5084.04	2.005	1.403	0.440
Lucy	78.922	414813	5255.99	2.012	1.541	0.639

Table 5.2: Performance Results for several MPU models

Regarding the usage of memory resources, our approach seems to have the same cost as Marching Tetrahedra which is not excessive. This is because for each point we only need to maintain the simplest set of complementary information: curvature, the tangent plane , front angle and adjacent points. Unlike MT which requires maintaining edge and corner lists plus the cube structure of the space

Model	*Time second*	*Triangles*	*Triangles Per second*	*Triangles Per expansion*	*Ratio L/s*	
					Avg	St.Dev
Our Approach	32.056	31085	969.709	2.004	1.517	0.5201
[dA04]	280.633	34640	123	NA	1.328	0.266

Table 5.3: Performance Comparison with VIS Bunny model

subdivision, we only need to store the point list and the octree data structure to speedup the collision test. Thus it is easy to see that our method is more conservative regarding memory resources. This is particularly important for large complex surfaces.

Figure 5.2 and Table 5.3 presents the comparison of our algorithm, with our first approach presented in [dA04]. In order to perform a coherent comparison, the same variational implicit model was used by our algorithm with the previous one. Both results were measured on computers with the same characteristics and the time penalty of evaluating the global implicit representation is the same. The curvature heuristic was configured in order to produce a mesh with a similar number of triangles. As we can see, we have increase the performance of the polygonization algorithm by a factor of almost 9. This is due to two main factors followed by our algorithm. First this new version, even if it also relies on point expansion do not need to manage a explicit representation of the borders such as the fronts on our previous approach. On the other hand, the new collision test to avoid the mesh overlap is faster thanks to the octree data structure to find closest points. We should highlight also, that the current algorithm is more robust since the collision not only handle the border (unexpanded points) but also mesh points. The previous version was also tested using MPUs and it was not robust enough to handle complex models such as the Dragon data-set.

5.2 Mesh quality analysis

Good meshes tend to have twice as many points as triangles. Our approach tries to respect other factors that characterize a good meshing. Meshes resulting from Marching Tetrahedra show patterns that arise from the space subdivision

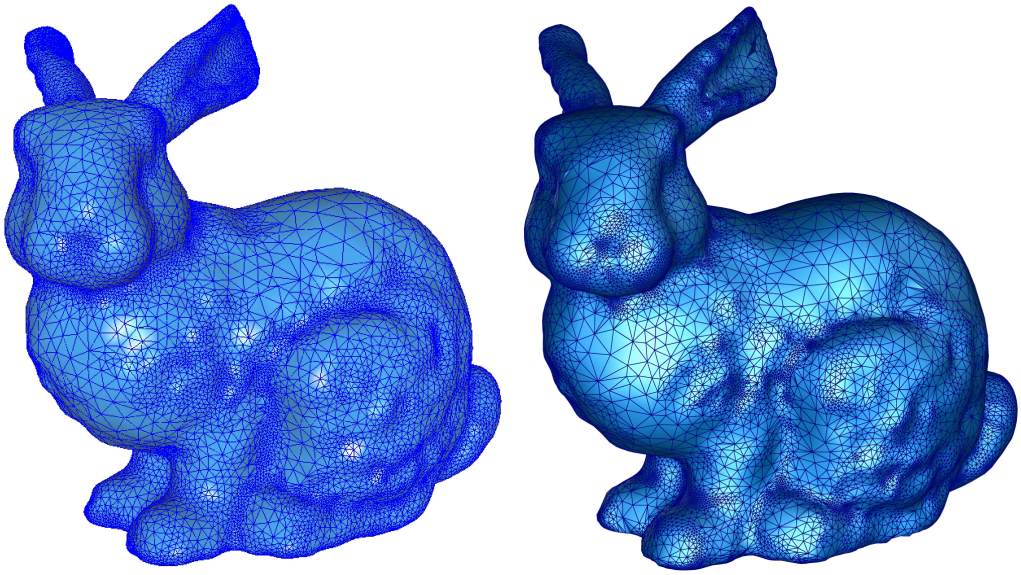


Figure 5.2: Bunny using Variational Implicit Surface a)old result b)new result

approach. Looking at Table 5.2, our approach produces triangles with edge ratios not above 1.5 and with small standard deviations .For each model, we present the standard deviation to ascertain that this is hold along the entire surface. Figure 5.3 presents a wire-frame rendering of two models with well defined curvature areas. As we can see, the adaptiveness of the triangulation enables to generate smaller triangles in high curvature regions and larger triangles in low curvature areas with a smooth transition of the triangle size between both areas. On the other hand, Figure 5.4 depicts the result achieved for the same model using our approach and two popular decimation techniques which are [Schroeder 92] and [Garland 97]. The three model presents meshing of the same model with the same amount of

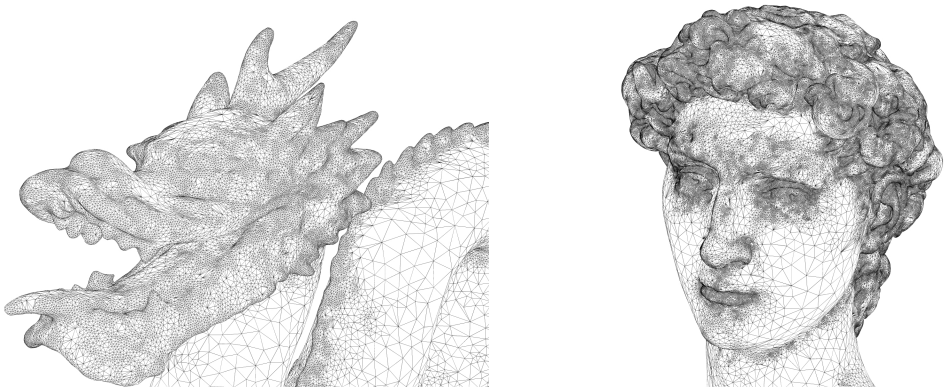


Figure 5.3: Adaptive Meshes of the Dragon and the David's head models

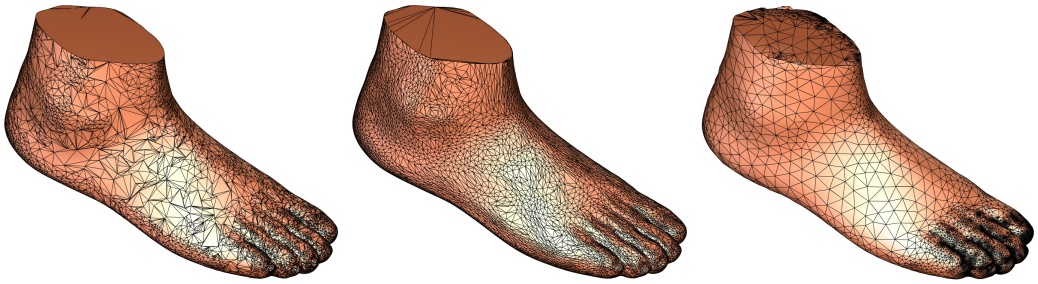


Figure 5.4: Foot Adaptive Meshes with similar number of triangles (18000) obtained by decimation [Schroeder 92] and [Garland 97] and our approach (from left to right)

triangles which is around 18000 triangles. As we can see, even if [Garland 97] uses quadric errors to decimate the mesh, our approach produces better results using the curvature information. Using the curvature, we are able to produce a better piecewise approximation and better manage the triangle density. The high curvature dependent triangulation produced by our algorithm shows that converting unorganized point data sets into implicit surface, then applying our polygonization, is a high quality meshing algorithm to decimate data-sets loosing less details than existing methods based on mesh processing.

5.3 Meshing accuracy analysis

To evaluate the accuracy of our polygonization algorithm and to perform a comparison with Marching Cubes and Marching Tetrahedra, we have defined two error measures. The average implicit value deviation which, for each triangle of the resulting mesh, computes the centroid and evaluate the implicit function at this location as a distance estimation to the real surface. We have tested other distance functions such as the Taubin distance [Taubin 94] dividing the implicit value by the gradient norm. However we verified that the metric was similar to the one proposed above. Thanks to this distance measure, we can evaluate how close we are from the real implicit surface. To analyze the curvature reproduction of the approximation, we have defined an angular normal based error. This error measure the angular deviation of triangle normals with the normal gathered at the centroid location of the triangle. Using this measure, we can verified the accuracy

of the curvature reproduction using a simple dot product between both vectors. For this measure, values closer to one are better, since the triangle normal mimics the normal of the surface.

The first accuracy comparison is a global comparison between our surface tracking approach and space decomposition approaches such the Marching Cubes and the Marching Tetrahedra. The results of this comparison are presented in Table 5.4 where we used a constant length approximation for a more fair comparison since both Marching Cubes and Marching Tetrahedra does not generated adaptive meshes. The heuristic scalar value correspond to the constant edge length used by our algorithm and to the cell size of the spatial decomposition used by both Marching cubes and Marching Tetrahedra. Looking to the average implicit deviation and the normal deviation, we can see that our surface tracking even with coarse approximation generates approximation with less errors than spatial decomposition techniques, and that the surface tracking is able to better approach the curvature of the shape. We can also highlight that the generation of coarse approximations is cheaper to create using our algorithm than using spatial decomposition, since the octree based sampling needs more implicit calls to converge to the surface. This is why the number of triangle per second is lower in Marching Cubes and Tetrahedra. We can see that our approach is able to maintain good triangles rates even for coarse representations. On the other hand when both approaches generate high detail approximations, the speed is similar showing that surface tracking is so efficient that space decomposition algorithms.

Method	MC.	MC.	MT.	MT	Cst	Cst	Cst	Cst
Heur. value	0.2	1.0	0.2	1.0	0.2	0.3	0.5	1.0
Triangles	69676	2684	211724	8148	55352	24716	9020	2198
Time(ms)	13189	2243	35391	3805	9224	4186	1542	431
Triangle/sec	5283	1197	5982	2141	6001	5904	5850	5100
Avg. Deviation	0.003	0.044	0.001	0.023	0.002	0.005	0.014	0.042
Std Avg Dev.	0.004	0.050	0.003	0.034	0.004	0.007	0.016	0.047
Norm. Deviation	0.998	0.974	0.997	0.981	0.999	0.998	0.994	0.981
Std Norm. Dev.	0.013	0.071	0.022	0.034	0.005	0.008	0.020	0.041

Table 5.4: Comparison between several approximations using Marching Cubes, Marching Tetrahedra and our constant expansion

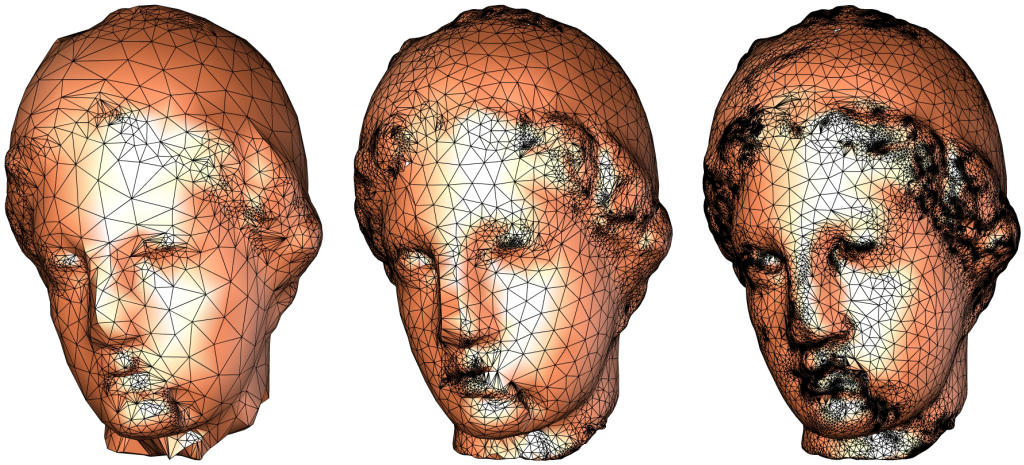


Figure 5.5: Adaptive Meshes of the Igea model using Mean Curvature Heuristics: 6908, 25596, 82511 triangles from left to right

Regarding the generation of adaptive meshes, Table 5.5 presents the results of the usage of different heuristic configurations to define the length of triangle edges. In this Table, we show the accuracy results for two different heuristics, one using the mean curvature value and the other using gaussian curvature. As it is possible to see in Figure 5.5 and Figure 5.6, both heuristics are able to reproduce without problem an adaptive approximation of the implicit surface. Looking to Table 5.5, Gauss curvature is more adequate to reproduce the curvature of the shape, however for coarse representation the Mean curvature is more sensitive to the shape as we can see in Figure 5.5. This is due to the variation of the Gauss curvature which is greater than the Mean curvature. However to generate high detailed approximations, the Gaussian curvature can catch more details than the heuristics based on Mean curvature. With the same heuristic bound parameters, we can verify that polygonization using Gaussian based heuristics are faster considering the number of triangles per second.

5.4 Discussion

As we have seen, our algorithm follow a surface tracking to approximate a given implicit surface using "true" curvature information. Starting from a seed point on the surface and following it while varying mesh properties in

Curv. Method	Mean.	Gauss.	Mean.	Gauss.	Mean.	Gauss.
Heur. value	0.1	0.1	0.2	0.2	0.3	0.3
Heur. minimum	0.05	0.05	0.05	0.05	0.1	0.1
Heur. maximum	1	1	1	1	2	2
Triangles	82511	93352	25596	28396	6908	7629
Time(ms)	14791	16704	4767	5147	1472	1412
Triangle/sec	5578.46	5588.6	5369.41	5517	4692.93	5401.97
Avg. Deviation	0.002	0.002	0.007	0.006	0.021	0.018
Std Avg Dev.	0.004	0.003	0.010	0.008	0.036	0.027
Norm. Deviation	0.9983	0.9989	0.9958	0.9967	0.9871	0.9884
Std Norm. Dev.	0.0182	0.0120	0.0215	0.0199	0.0497	0.0464

Table 5.5: Comparison between several adaptive approximations generated by our algorithm using different heuristics. For each column, the scalar value factor and the limit of the heuristic are given.

close agreement to local surface curvature. This allows our approach to better match surface features, especially connectivity, in "tight spots", than cell-sampling techniques such as Marching Cubes [Hilton 97] and Tetrahedra [Bloomenthal 94] which subdivide the space in cells of fixed precision. In doing so, we can avoid topological inconsistencies that often arise in such methods when the cells are not small enough to match local shape details.

Another advantage of our approach is that it uses the local information extracted from the surface to compute triangle expansion. Marching Tetrahedra generates useless triangles in a pattern that respects the spatial subdivision, rather than local features of the surface. Surface tracking algorithms avoid such degenerate meshes by construction, thus avoiding an expensive post-remeshing step as is done by Treece [Treece 99] or Igarashi [Igarashi 03]. Similar to Marching Triangles, we can generate a mesh with quasi-uniform triangles on-the-fly using a constant edge length. Other techniques such as ShinkWrap [vO04] achieve similar results but at the cost of an iterative method that can be too slow to obtain high fidelity meshes. Other iterative techniques such as Akkouche's [Akkouche 01] based on MT and Cermak's Edge Spinning [Cermák 04] can produce similar results. However our method use curvature measures extracted from implicit functions, we would like to make a formal comparison of both measures, but it was not possible since the data set used by Akkouche is not available and Cermak adaptive

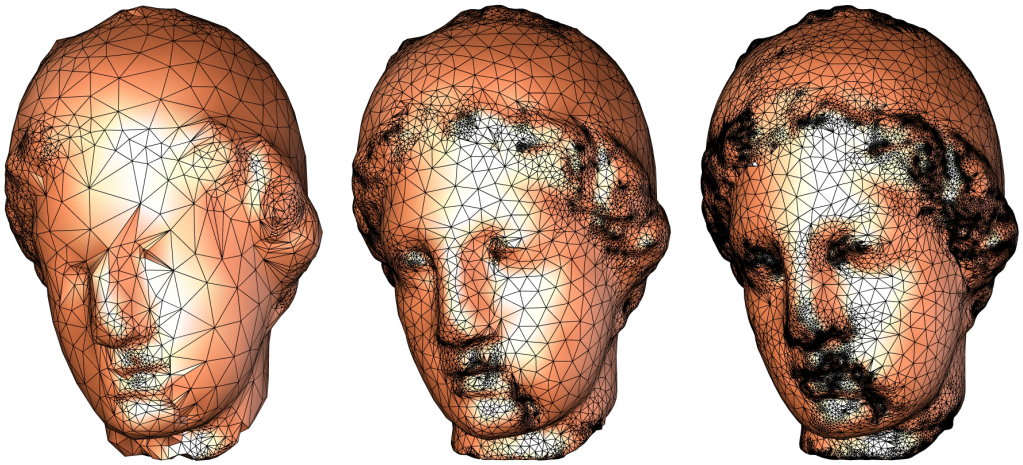


Figure 5.6: Adaptive Meshes of the Igea model using Gauss Curvature Heuristics: 7629, 28396, 93352 triangles from left to right

approach was only illustrated with a simple 3 genus shape.

Unlike Edge Spinning, MT and its variants, which are directed by edges, our algorithm works by expanding points. Akkouche's [Akkouche 01], Cermak's [Cermák 04] and Karkanis's [Karkanis 01] method are good examples of the former strategy. These approaches are only able to generate one extra point for each edge considered which may require a high number of steps in areas of rapid change. Our point expansion guided by curvature based heuristics is better suited to such cases, since edge length is recomputed at each point and generate up to three triangles. The main advantage of our approach regarding both methods, is that we do not require any post step to solve cracks since our expansion creates edge links on the mesh, overcoming cracks on the fly. Finally, Karkanis et al. [Karkanis 01] estimate curvature as the minimum radius of different geodiscs, this procedure can become very expensive since it requires repeated evaluations of the implicit function at each expansion point. Moreover, it is not clear that this method yields higher accuracy than ours.

6

Conclusions and Future Work

This chapter summarizes the dissertation and presents the final conclusions, identifying the most relevant contributions of my work and discussing the strengths and weaknesses of algorithms and methodologies presented in this thesis. In addition I point possible directions for future work in each field.

6.1 Summary of the Dissertation

In this dissertation, we have tackled the problem of visualizing implicit surfaces using a polygonal approximation. Considering nowadays application of implicit surface representation and the several visualization methods applied to this model, we choose to produce a novel approach to generate a piecewise approximation of this continuous representation which is able to provide a reliable shape description for interactive visualization trying to best reproduce the curvature of the original implicit shape. Nowadays graphic hardware is designed to deal and render non-continuous representation such as points and polygons which are not appropriate to correctly convey curvature information. However polygons are cheap to display and current graphic boards are able to render hundred millions of triangles per second providing interactive visualization of high detailed meshes.

By reviewing the several implicit surface representations, we have surveyed the different implicit formulations and demonstrate their flexibility as shape representations. By studying the implicit function, we can gather important shape characteristics such as curvature using differential geometry which can be used by

polygonization algorithms to propose a high quality alternative representation. However several issues need to be studied when dealing with a conversion process from a continuous to a non-continuous representation. In this dissertation we have contribute not only to identify existing techniques to visualize implicit surfaces, but we have classified them and proposed criterions for comparisons which are topology correctness, sharp feature sensitivity, smoothness reproduction and quality of the polygonal representation.

Considering these criterions, we have proposed a novel polygonization algorithm for implicit surfaces generating a curvature dependent piecewise approximation. Our algorithm takes advantage of the curvature which can be extracted from the implicit function to generate a faithful mesh approximation. This is achieved following a surface tracking approach generating the polygonal approximation by expanding points tangentially to the surface instead of relying on a sampling method based on spatial subdivision. This strategy allows us to control and adapt the triangulation according to local characteristics of the shape extracted from the implicit surface mathematical formulation. The mesh is created continuously by the algorithm resulting in an adaptive triangulation generated on the fly in only one step without any post processing.

6.2 Our Approach Benefits

The results demonstrated several benefits of our approach compared to existing polygonization algorithms. Following the surface tracking approach, we are able to present a polygonal approximation which is able to better approximate high curvature region than popular spatial subdivision approaches such as Marching Cubes and its variants. We have validated the usage of a real curvature measure for the adaptive polygonization of several implicit models without a performance lost compared to existing techniques. Using the curvature, we are able to offer a good management of the triangle budget needed to polygonize an implicit surface.

Our approach generated an adaptive mesh to better reproduce the curvature of the shape, using a single step algorithm. Most of existing work, is only able to

achieve such result by applying post-remeshing step to a high density triangulation of an implicit surface. Our algorithm achieve an high quality triangulation on the fly.

On the other hand, the algorithm can be applied to any implicit surfaces which implicit evaluation can retrieve implicit values, gradient vectors and hessian matrixes. Using only these three functions, we are able to produce our mesh, using them to evaluate the surface, project points on the surface and control the expansion process. We demonstrate our approach with several classes of implicit surfaces: Variational Implicit Surfaces, Multi-Partition of Unity, Meta-balls and traditional algebraic or quadric surfaces. Our algorithm demonstrates that a better usage can be made from the implicit evaluation. Using differential geometry curvature measures can be extracted and used as heuristics to be enable to capture the maximum of features of the surface. Doing so, we are able to catch even thin local features and better approximate sharp features than traditional spacial based decomposition techniques. We present several heuristics which can be used to influence the triangle density regarding different type of curvatures.

The polygonization process is robust and efficient to avoid mesh overlap, implicit value un-definition or open implicit surfaces. This guarantees that the polygonization is finite even if the implicit surfaces model is not robust over all its domain. Regarding the mesh overlapping avoidance, the process it not only able to deal with collision with the border of the expansion but also with already expanded points that are part of the resulting mesh. This is done efficiently using a spatial re-organization of the points based on an octree-data structure to speedup the search of neighbor points. Our algorithm was applied to well known large data-sets with millions of points, illustrating that our algorithm combined with implicit surface reconstruction is a good alternative re-meshing approach to generate adaptive meshes of unorganized point clouds obtained by scanning particularly important for reverse engineering.

Using point expansion, we reduce the complexity of the polygonization algorithm. Existing surface tracking approaches used more constrained expansions, such as edge based expansion, which produce only on triangle by expansion such as the Marching Triangles or the Edge-Spinning. We also avoid an explicit

representation of the border of the polygonization such as Hartmann approach. Doing so, there is no front management needed resulting on complex mechanism such as front splitting and merging. In our approach, we just maintain, for each unexpanded point, its neighbors to define the angle of the missing triangulation.

6.3 Limitations and Future Work

Even if the result of our algorithm is a well defined adaptive mesh, there are still some limitations and issues that can be improved. We verified that the current expansion can lead to points with more than height adjacent points on the mesh due to high curvature variation. Even using the heuristic edge length definition considering neighbor points, very high curvature variations can lead to an increase of possible collisions. To overcome these collision, we cannot create new points on the expansion resulting in a closing triangle using the unexpanded point and its neighbors. This limitation could be overcome using several techniques such as applying a local vertex optimization or triangle clustering after a point being expanded.

Regarding the sharp features, the result achieved are better than spatial decomposition techniques, however the border reproduction could be improved, since we do not consider a different angle calculation when expanding a point located over an edge. This can be overcome combining the polygonization with a robust sharp feature detection, to propose a triangulation that follows sharp features avoiding jagged effects.

Our collision detection is robust to guarantee that the polygonization process will avoid to enter in a infinite mesh overlapping. However since the collision test only uses points, collision between a small expansion and a large triangle cannot be prevent and the algorithm is only able to recover when it detects points. This failure results on partial incomplete meshing because new triangles are created then the expansion stopped due to mesh overlaps but these useless triangles are not avoided and can remain under the mesh. A post-re-meshing of open points could be applied or the collision test should be improved such as using sphere hierarchy to handle mesh point versus point collision (currently supported using

the octree), large edge inter-collision and large triangle inter-collision. Using spheres which circumscribe the triangles of the mesh, a more robust collision test can be created to detect full point triangle collision.

As future work, our algorithm can be integrated on graphic processor unit (GPU) to propose a tessellation approach of implicit surface so common as parametric surface tessellation, taking advantages of the geometry shader stage on existing GPU architecture. By parallelizing the collision test, our algorithm could be parallelized, and several expansion could be generated concurrently. The global sorting based on the front angle value is not a requirement, efficient polygonization could be presented using a partial sorting more adapted to parallelization. As mentioned before, the quality of the mesh could be improved combining our approach with a local vertex optimization step after the expansion of each point. Particle based optimization or Laplacian smoothing filters have proved to be efficient by several post-remeshing algorithms and could be used to improve this point. Finally regarding efficiency,the usage of the curvature for adaptive meshing can be faster and classified. Such as the lookup table used by Marching Tetrahedra and Marching Cubes after cell classification, curvature measures could be used to provide a lookup mechanism to propose the more adapted local triangulation. Research has proposed methods for an optimal triangulation of parametric and implicit quadrics based on both principal curvatures and directions presenting novel issues to offer a perfect conversion of implicit surfaces into a polygonal representation.

Bibliography

[Akkouche 01] Samir Akkouche and Eric Galin. Adaptive implicit surface polygonization using marching triangles. *Comput. Graph. Forum*, 20(2):67–80, 2001.

[Alberti 05] L. Alberti, G. Comte, and B. Mourrain. Meshing implicit algebraic surfaces: the smooth case. *Mathematical Methods for Curves and Surfaces: Tromso'04*, pages 11–26, 2005.

[Allègre 04] Rémi Allègre, Aurélien Barbier, Eric Galin, and Samir Akkouche. A hybrid shape representation for free-form modelling. In *2004 International Conference on Shape Modeling and Applications (SMI 2004), 7-9 June 2004, Genova, Italy*, pages 7–18. IEEE Computer Society, 2004.

[Allègre 06] Rémi Allègre, Eric Galin, Raphaëlle Chaine, and Samir Akkouche. The hybridtree: Mixing skeletal implicit surfaces, triangle meshes and point sets in a free-form modeling system. *Graphical Models, SMI'04 special issue*, 68(1):42–64, January 2006.

[Alliez 03] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Levy, and Mathieu Desbrun. Anisotropic polygonal remeshing. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, pages 485–493, 2003.

[Attene 03] Marco Attene, Bianca Falcidieno, Jarek Rossignac, and Michela Spagnuolo. Edge-sharpener: recovering sharp features in triangulations of non-adaptively re-meshed surfaces. In *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 62–69, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[Azernikov 05] Sergei Azernikov and Anath Fischer. Anisotropic meshing of implicit surfaces. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, pages 94–103, Washington, DC, USA, 2005. IEEE Computer Society.

[Belyaev 98] A.G. Belyaev, A.A. Pasko, and T.L. Kunii. Ridges and ravines on implicit surfaces. *cgi*, 00:530, 1998.

[Belyaev 05] Alexander G. Belyaev and Elena V. Anoshkina. Detection of surface creases in range data. In *IMA Conference on the Mathematics of Surfaces*, volume 3604, pages 50–61. Springer, 2005.

[Blinn 82] James F. Blinn. A generalization of algebraic surface drawing. *ACM Trans. Graph.*, 1(3):235–256, 1982.

[Bloomenthal 88] J. Bloomenthal. Polygonization of implicit surfaces. *Comput. Aided Geom. Des.*, 5(4):341–355, 1988.

[Bloomenthal 91] Jules Bloomenthal and Ken Shoemake. Convolution surfaces. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 251–256, New York, NY, USA, 1991. ACM Press.

[Bloomenthal 94] Jules Bloomenthal. An implicit surface polygonizer. In Paul Heckbert, editor, *Graphics Gems IV*, pages 324–349. Academic Press, Boston, 1994.

[Bloomenthal 95] Jules Bloomenthal and Keith Ferguson. Polygonization of non-manifold implicit surfaces. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 309–316, New York, NY, USA, 1995. ACM Press.

[Bogaevski 03] Ilia Bogaevski, Veronique Lang, Alexander Belyaev, and Tosiyasu L. Kunii. Color ridges on implicit polynomial surfaces. In *Proceedings GraphiCon 2003*, pages 161–164, Moscow, Russia, September 2003. Moscow State University.

[Boissonnat 05] Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graph. Models*, 67(5):405–451, 2005.

[Bottino 96] A. Bottino, W. Nuij, and K. Van Overveld. How to shrinkwrap through a critical point: an algorithm for the adaptive triangulation of iso-surfaces with arbitrary topology. In *Implicit Surfaces '96*, pages 53–72, October 1996.

[Bouthors 07] Antoine Bouthors and Matthieu Nesme. Twinned meshes for dynamic triangulation of implicit surfaces. In *GI '07: Proceedings of Graphics Interface 2007*, pages 3–9, New York, NY, USA, 2007. ACM Press.

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

[Carr 97] Jonathan C. Carr, W. Richard Fright, and Richard K. Beatson. Surface interpolation with radial basis functions for medical imaging. *IEEE Transactions on Medical Imaging*, 16(1):96–107, February 1997.

[Carr 01] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76, New York, NY, USA, 2001. ACM Press.

[Carr 03] J. C. Carr, R. K. Beatson, B. C. McCallum, W. R. Fright, T. J. McLennan, and T. J. Mitchell. Smooth surface reconstruction from noisy range data. In *GRAPHITE '03: Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 119–ff, New York, NY, USA, 2003. ACM Press.

[Cermák 04] Martin Cermák and Václav Skala. Adaptive edge spinning algorithm for poligonization of implicit surfaces. In *2004 Computer Graphics International (CGI 2004), 16-19 June 2004, Crete, Greece*, pages 36–43. IEEE Computer Society, 2004.

[Cermák 02] M. Cermák and V. Skala. Polygonization by the edge spinning. In *16th Conference on Scientific Computing, Algoritmy 2002*, Slovakia, 2002.

[Chan 98] Shek Ling Chan and Enrico O. Purisima. A new tetrahedral tesselation scheme for isosurface generation. *Computers & Graphics*, 22(1):83–90, 1998.

[Cheng 04] Siu-Wing Cheng, Tamal K. Dey, Edgar A. Ramos, and Tathagata Ray. Sampling and meshing a surface with guaranteed topology and geometry. In *SCG '04: Proceedings of the twentieth annual symposium on Computational geometry*, pages 280–289, New York, NY, USA, 2004. ACM Press.

[Chernyaev 95] E. Chernyaev. Marching cubes 33: Construction of topologically correct isosurfaces. Technical Report Technical Report CN/95-17, CERN, Geneva, Switzerland, 1995.

[Chew 93] L. Paul Chew. Guaranteed-quality mesh generation for curved surfaces. In *SCG '93: Proceedings of the ninth annual symposium on Computational geometry*, pages 274–280, New York, NY, USA, 1993. ACM Press.

[Crespin 96] Benoît Crespin, Carole Blanc, and Christophe Schlick. Implicit sweep objects. *Comput. Graph. Forum*, 15(3):165–174, 1996.

[Crespin 02] B. Crespin. Dynamic triangulation of variational implicit surfaces using incremental delaunay tetrahedralization. In *VVS '02: Proceedings of the 2002 IEEE symposium on Volume visualization and graphics*, pages 73–80, Piscataway, NJ, USA, 2002. IEEE Press.

[Cuno 04] Alvaro Cuno, Claudio Esperanca, Antonio Oliveira, and Paulo Roma Cavalcanti. Fast polygonization of variational implicit surfaces. In *XVII Brazilian Symposium on Computer Graphics and Image Processing, (SIBGRAPI 2004) 17-20 October 2004, Curitiba, PR, Brazil*, pages 258–265. IEEE Computer Society, 2004.

[Cyb] Sample model cyberware page.

[dA04] Bruno de Araújo and Joaquim Jorge. Curvature dependent polygonization of implicit surfaces. In *XVII Brazilian Symposium on Computer Graphics and Image Processing, (SIBGRAPI 2004) 17-20 October 2004, Curitiba, PR, Brazil*, pages 266–273. IEEE Computer Society, 2004.

[Desbrun 95] Mathieu Desbrun, Nicolas Tsingos, and Marie-Paule Gascuel. Adaptive sampling of implicit surfaces for interactive modeling and animation. In *proceedings of Implicit Surfaces '95, Grenoble, France*, Mars 1995.

[Dey 07] Tamal K. Dey and Joshua A. Levine. Delaunay meshing of isosurfaces. In *Shape Modeling International*, pages 241–250, 2007.

[dF92] Luiz Henrique de Figueiredo, Jonas de Miranda Gomes, Demetri Terzopoulos, and Luiz Velho. Physically-based methods for polygonization of implicit surfaces. In *Proceedings of the conference on Graphics interface '92*, pages 250–257, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[Foster 05] Kevin Foster, Pauline Jepp, Brian Wyvill, Mario Sousa, Callum Galbraith, and Joaquim Jorge. Pen-and-ink for blobtree implicit models. *Computer Graphics Forum (Proc. of Eurographics '05)*, 24(3):267–276, 2005.

[Galin 00] Eric Galin and Samir Akkouche. Incremental polygonization of implicit surfaces. *Graphical Models*, 62(1):19–39, 2000.

[Garland 97] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[Gascuel 95] Jean-Dominique Gascuel. Implicit patches: An optimised and powerful ray intersection algorithm. In *Implicit Surfaces'95*, pages 143–160, Grenoble, France, April 1995.

[Gray 96] Alfred Gray. *Modern Differential Geometry of Curves and Surfaces with Mathematica*. CRC Press, Inc., Boca Raton, FL, USA, 1996.

[Hall 90] Mark Hall and Joe Warren. Adaptive polygonalization of implicitly defined surfaces. *IEEE Comput. Graph. Appl.*, 10(6):33–42, 1990.

[Hart 93] John C. Hart. Ray tracing implicit surfaces. In *SIGGRAPH 93 Modeling, Visualizing, and Animating Implicit Surfaces course notes*, pages 13–1 to 13–15, 1993.

[Hartmann 98] Erich Hartmann. A marching method for the triangulation of surfaces. *The Visual Computer*, 14(2):95–108, 1998.

[Hege 97] Hans-Christian Hege, Martin Seebas, Detlev Stalling, and Malte Zockler. A generalized marching cubes algorithm based on non-binary classifications. Technical report, 1997.

[Hilton 96] A. Hilton, A. J. Stoddart, J. Illingworth, and T. Windeatt. Marching triangles: Range image fusion for complex object modelling. *International Conf. on Image Processing*, B:381–384, 1996.

[Hilton 97] A. Hilton and J. Illingworth. Marching triangles: Delaunay implicit surface triangulation. Technical Report CVSSP Technical Report 01, Univ. of Surrey, Guidford, CVSSP, January 1997.

[Hughes 03] John Hughes. Differential geometry of implicit surfaces – a primer. Technical report, 2003.

[Hui 99] Kin Chuen Hui and Z. H. Jiang. Tetrahedra based adaptive polygonization of implicit surface patches. *Comput. Graph. Forum*, 18(1):57–68, 1999.

[Igarashi 03] Takeo Igarashi and John F. Hughes. Smooth meshes for sketch-based freeform modeling. In *SI3D '03: Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 139–142. ACM Press, 2003.

[Jevans 88] David Jevans and Brian Wyvill. Ray tracing implicit surfaces. Technical report, University of Calgary, Dept. of Computer Science, 1988.

[Ju 02] Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 339–346, New York, NY, USA, 2002. ACM Press.

[Kalra 89] D. Kalra and A. H. Barr. Guaranteed ray intersections with implicit surfaces. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pages 297–306, New York, NY, USA, 1989. ACM Press.

[Karkanis 01] Tasso Karkanis and A. James Stewart. Curvature-dependent triangulation of implicit surfaces. *IEEE Comput. Graph. Appl.*, 21(2):60–69, 2001.

[Kobbelt 01] Leif P. Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. Feature sensitive surface extraction from volume data. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 57–66, New York, NY, USA, 2001. ACM Press.

[Kobbelt 03] Leif Kobbelt and Mario Botsch. Feature sensitive mesh processing. In *SCCG '03: Proceedings of the 19th spring conference on Computer graphics*, pages 17–22, New York, NY, USA, 2003. ACM Press.

[Koenderink 90] Jan J. Koenderink. *Solid Shape*. Artificial Intelligence Series. MIT Press, Cambridge, MA, USA, 1990.

[Lewiner 03] Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of Graphics Tools*, 8:1–15, 2003.

[Liu 05] Shengjun Liu, Xuehui Yin, Xiaogang Jin, and Jieqing Feng. High quality triangulation of implicit surfaces. In *CAD-CG '05: Proceedings of the Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, pages 133–138, Washington, DC, USA, 2005. IEEE Computer Society.

[Lopes 03] Adriano Lopes and Ken Brodlie. Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):16–29, 2003.

[Lorensen 87] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, NY, USA, 1987. ACM Press.

[Masuda 03] Takeshi Masuda. Surface curvature estimation from the signed distance field. pages 361–368, 2003.

[McCormick 02] Neil H. McCormick and Robert B. Fisher. Edge-constrained marching triangles. Technical Report EDI-INF-RR-0188, Univ. of Edinburgh, Division of Informatics, June 2002.

[Meyer 03] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In Hans-Christian Hege and Konrad Polthier, editors, *Visualization and Mathematics III*, pages 35–57. Springer-Verlag, Heidelberg, 2003.

[Montani 94a] C. Montani, R. Scateni, and R. Scopigno. Discretized marching cubes. In *VIS '94: Proceedings of the conference on Visualization '94*, pages 281–287, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

[Montani 94b] Claudio Montani, Riccardo Scateni, and Roberto Scopigno. A modified look-up table for implicit disambiguation of marching cubes. *The Visual Computer*, 10(6):353–355, December 1994.

[Morse 01] Bryan S. Morse, Terry S. Yoo, David T. Chen, Penny Rheingans, and K.R. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. *smi*, 00:0089, 2001.

[MPI07] Max plant institute mpu site, 2007.

[Muraki 91] Shigeru Muraki. Volumetric shape description of range data using ̈blobby model. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 227–235, New York, NY, USA, 1991. ACM Press.

[Neugebauer 97] P. Neugebauer and K. Klein. Adaptive triangulation of objects reconstructed from multiple range images. *IEEE Visualization '97*, October 1997.

[Nielson 03] Gregory M. Nielson. On marching cubes. *IEEE Transactions on Visualization and Computer Graphics*, 09(3):283–297, 2003.

[Nielson 04] Gregory M. Nielson. Dual marching cubes. In *VIS '04: Proceedings of the conference on Visualization '04*, pages 489–496, Washington, DC, USA, 2004. IEEE Computer Society.

[Ning 93] Paul Ning and Jules Bloomenthal. An evaluation of implicit surface tilers. *IEEE Comput. Graph. Appl.*, 13(6):33–41, 1993.

[Nishimura 85] H. Nishimura, M. Hirai, T. Kawai, T. Kawata, I. Shirakawa, and K. Omura. Object modelling by distribution function and a method of image generation. J68-D:718–725, 1985. in Japanese,translated into English by Takao Fujuwara.

[Ohtake 01] Yutaka Ohtake, Alexander Belyaev, and Alexander Pasko. Dynamic meshes for accurate polygonization of implicit surfaces with sharp features. *SMI 2001 International Conference on Shape Modeling and Applications*, 00:0074, 2001.

[Ohtake 02] Yutaka Ohtake and Alexander G. Belyaev. Dual/primal mesh optimization for polygonized implicit surfaces. In *SMA '02: Proceedings of the seventh ACM symposium on Solid modeling and applications*, pages 171–178, New York, NY, USA, 2002. ACM Press.

[Ohtake 03] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 463–470, New York, NY, USA, 2003. ACM Press.

[Ohtake 04] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph.*, 23(3):609–612, 2004.

[Ohtake 05a] Yutaka Ohtake, Alexander Belyaev, and Marc Alexa. Sparse low-degree implicit surfaces with applications to high quality rendering, feature extraction, and smoothing. In *Eurographics Symposium on Geometry Processing*, pages 463–470. The Eurographics Association, 2005.

[Ohtake 05b] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. 3d scattered data interpolation and approximation with multilevel compactly supported rbfs. *Graph. Models*, 67(3):150–165, 2005.

[Ohtake 06] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Sparse surface reconstruction with adaptive partition of unity and radial basis functions. *Graphical Models*, 68(1):15–24, January 2006.

[Paiva 06] Afonso Paiva, Hélio Lopes, Thomas Lewiner, and Luiz Henrique de Figueiredo. Robust adaptive meshes for implicit surfaces. In *19th Brazilian Symposium on Computer Graphics and Image Processing*, pages 205–212, Manaus, AM, october 2006.

[Pasko 88] Alexander A. Pasko, V. V. Pilyugin, and V. N. Pokrovskiy. Geometric modeling in the analysis of trivariate functions. *Computers & Graphics*, 12(3-4):457–465, 1988.

[Peiró 07] J. Peiró, L. Formaggia, M. Gazzola, A. Radaelli, and V. Rigamonti. Shape reconstruction from medical images and quality mesh generation via implicit surfaces. *International Journal for Numerical Methods in Fluids*, 53(8):1339–1360, 2007.

[Plantinga 04] Simon Plantinga and Gert Vegter. Isotopic approximation of implicit curves and surfaces. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 245–254, New York, NY, USA, 2004. ACM Press.

[Plantinga 07] Simon Plantinga and Gert Vegter. Isotopic meshing of implicit surfaces. *The Visual Computer*, 23(1):45–58, January 2007.

[Renbo 05] Xia Renbo, Liu Weijun, and Wang Yuechao. A robust and topological correct marching cube algorithm without look-up table. In *CIT '05: Proceedings of the The Fifth International Conference on Computer and Information Technology*, pages 565–569, Washington, DC, USA, 2005. IEEE Computer Society.

[Reuter 04] Patrick Reuter, Pierre Joyot, Jean Trunzler, Tamy Boubekeur, and Christophe Schlick. Reconstructing implicit surfaces with sharp edges via enriched reproducing kernel particle approximation. Technical Report RR-133404, Laboratoire Bordelais de Recherche en Informatique, Bordeaux, France, 2004.

[Rösch 97] Angela Rösch, Matthias Ruhl, and Dietmar Saupe. Interactive visualization of implicit surfaces with singularities. *Eurographics Computer Graphics Forum*, 16(5):295–306, 1997.

[Savchenko 95] V. V. Savchenko, A. A. Pasko, O. G. Okunev, and T. L. Kunii. Function representation of solids reconstructed from scattered surface points and contours. *Computer Graphics Forum*, 14(4):181–188, October 1995.

[Schaefer 04] Scott Schaefer and Joe Warren. Dual marching cubes: Primal contouring of dual grids. In *PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference on (PG'04)*, pages 70–76, Washington, DC, USA, 2004. IEEE Computer Society.

[Schmidt 05] Ryan Schmidt, Brian Wyvill, and Eric Galin. Interactive implicit modeling with hierarchical spatial caching. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, pages 104–113, Washington, DC, USA, 2005. IEEE Computer Society.

[Schmidt 06] Ryan Schmidt, Tobias Isenberg, and Brian Wyvill. Interactive pen-and-ink rendering for implicit surfaces. In *ACM SIGGRAPH 2006 Conference Abstracts and Applications (SIGGRAPH Sketch)*, July 2006.

[Schroeder 92] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and*

interactive techniques, pages 65–70, New York, NY, USA, 1992. ACM.

[Shen 04] Chen Shen, James F. O'Brien, and Jonathan R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Trans. Graph.*, 23(3):896–904, 2004.

[Sherstyuk 96] Andrei Sherstyuk. Ray-tracing implicit surfaces: a generalized approach. Technical report, Department of Computer Science ,Monash University,Australia, 1996.

[Sherstyuk 99] Andrei Sherstyuk. Fast ray tracing of implicit surfaces. *Comput. Graph. Forum*, 18(2):139–147, 1999.

[Siqueira 98] M.F. Siqueira, S.R. Freitas, A.C. Filho, and G. Tavares. Speeding up adaptive polygonization. In *West Side Computer Graphics 1998 (WSCG'98), Poster*, pages 11–12, University of West Bohemia, Plzen, Czech Republic, February 1998.

[Sta07] Standford 3d scan repository, 2007.

[Stander 97] Barton T. Stander and John C. Hart. Guaranteeing the topology of an implicit surface polygonization for interactive modeling. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 279–286, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[Stolte 95] N. Stolte and R. Caubet. Fast high definition discrete ray tracing implicit surfaces. In *5th DGCI - Discrete Geometry for Computer Imagery*, pages 61–70, Clermont-Ferrand, France, 1995.

[Taubin 94] Gabriel Taubin. Distance approximations for rasterizing implicit curves. *ACM Trans. Graph.*, 13(1):3–42, 1994.

[Tobor 04] Ireneusz Tobor, Patrick Reuter, and Christophe Schlick. Multiresolution reconstruction of implicit surfaces with attributes from large unorganized point sets. In *Proceedings of Shape Modeling International (SMI 2004)*, pages 19–30, 2004.

[Tobor 06] Ireneusz Tobor, Patrick Reuter, and Christophe Schlick. Reconstructing multi-scale variational partition of unity implicit surfaces with attributes. *Graph. Models*, 68(1):25–41, 2006.

[Treece 99] G. M. Treece, R. W. Prager, and A. H. Gee. Regularised marching tetrahedra: improved iso-surface extraction. *Computers and Graphics*, 23(4):583–598, 1999.

[Triquet 01] Frederic Triquet, Philippe Meseure, and Christophe Chaillou. Fast polygonization of implicit surfaces. In *WSCG'2001 (Plzen, Czech Republic)*, volume 2, pages 283–290, February 2001. http://wscg.zcu.cz.

[Triquet 03] Frederic Triquet, Laurent Grisoni, Philippe Meseure, and Christophe Chaillou. Realtime visualization of implicit objects with contact control. In *GRAPHITE'2003, International Conference on Computer Graphics and Interactive Techniques in Australia and South East Asia, Sponsored by ACM Siggraph (Melbourne, Australia)*, volume 1, February 2003. http://www.anzgraph.org/graphite2003.

[Turk 99] Greg Turk and James F. O'Brien. Shape transformation using variational implicit functions. In *SIGGRAPH '99 Conference Proceedings*, pages 335–342. ACM Press, 1999.

[Turk 01] Greg Turk, Huong Quynh Dinh, James F. O'Brien, and Gary Yngve. Implicit surfaces that interpolate. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, page 62, Washington, DC, USA, 2001. IEEE Computer Society.

[Varadhan 03] Gokul Varadhan, Shankar Krishnan, Young J. Kim, and Dinesh Manocha. Feature-sensitive subdivision and isosurface reconstruction. In *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, page 14, Washington, DC, USA, 2003. IEEE Computer Society.

[Varadhan 04] Gokul Varadhan, Shankar Krishnan, TVN Sriram, and Dinesh Manocha. Topology preserving surface extraction using adaptive subdivision. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 235–244, New York, NY, USA, 2004. ACM Press.

[Varadhan 06] Gokul Varadhan, Shankar Krishnan, Liangjun Zhang, and Dinesh Manocha. Reliable Implicit Surface Polygonization using Visibility Mapping. In *Proc. Eurographics Symposium on Geometry Processing*, pages 211–221. Eurographics Association, 2006.

[Velho 96] Luiz Velho. Simple and efficient polygonization of implicit surfaces. *Jornal Graphic Tools*, 1(2):5–24, 1996.

[Velho 99] Luiz Velho, Luiz Henrique de Figueiredo, and Jonas Gomes. A unified approach for hierarchical adaptive tesselation of surfaces. *ACM Trans. Graph.*, 18(4):329–360, 1999.

[vO04] Kees van Overveld and B. Wyvill. Shrinkwrap: An efficient adaptive algorithm for triangulating an iso-surface. *Vis. Comput.*, 20(6):362–379, 2004.

[Vorsatz 01] Jens Vorsatz, Christian Rössl, Leif Kobbelt, and Hans-Peter Seidel. Feature sensitive remeshing. *Computer Graphics Forum, Proceedings of Eurographics 2001*, 20(3):393–401, 2001.

[Witkin 94] Andrew P. Witkin and Paul S. Heckbert. Using particles to sample and control implicit surfaces. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 269–277, New York, NY, USA, 1994. ACM Press.

[Wood 02] Zoe Wood, Hugues Hoppe, Mathieu Desbrun, and Peter Schroder. Isosurface topology simplification. Technical Report MSR-TR-2002-28, Microsoft Research January 2002, 2002.

[Wu 05] Xiaojun Wu, Michael Yu, and Wang Qi Xia. Implicit fitting and smoothing using radial basis functions with partition of unity. In *CAD-CG '05: Proceedings of the Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, pages 139–148, Washington, DC, USA, 2005. IEEE Computer Society.

[Wyvill 86] Geoff Wyvill, Craig McPheeters, and Brian Wyvill. Data structure for soft objects. *The Visual Computer*, 2(4):227–234, February 1986.

[Wyvill 90] G. Wyvill and A. Trotman. Ray-tracing soft objects. In *CG International '90: Proceedings of the eighth international conference of the Computer Graphics Society on CG International '90: computer graphics around the world*, pages 469–476, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[Wyvill 98] Brian Wyvill, Eric Galin, and Andrew Guy. The Blob Tree, Warping, Blending and Boolean Operations in an Implicit Surface Modeling System. *Implicit Surfaces*, 3, June 1998. Chosen for inclusion in a special issue of Computer Graphics Forum.

[Yamazaki 02] Shuntaro Yamazaki, Kiwamu Kase, and Katsushi Ikeuchi. Non-manifold implicit surfaces based on discontinuous implicitization and polygonization. *Geometric Modeling and Processing*, 00:138, 2002.

[Yang 06] Jun Yang, Changqian Zhu, and Hua Zhang. Surface reconstruction with least square reproducing kernel and partition of unity. *icat*, 0:375–380, 2006.

[Yngve 02] Gary Yngve and Greg Turk. Robust creation of implicit surfaces from polygonal meshes. *IEEE Transactions on Visualization and Computer Graphics*, 8(4):346–359, 2002.

[Zhang 06] Yi Zhang, Xin Wang, and Xiao Jun Wu. Fast visualization algorithm for implicit surfaces. *16th International Conference on Artificial Reality and Telexistence–Workshops (ICAT'06)*, 0:339–344, 2006.