UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

LUÍS FERNANDO MAIA SANTOS SILVA

Merging Meshes Using Dynamic Regular Triangulation

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Prof. Dr. João Luiz Dihl Comba
Advisor

Porto Alegre, May 2010

CIP – CATALOGING-IN-PUBLICATION

Silva, Luís Fernando Maia Santos

Merging Meshes Using Dynamic Regular Triangulation / Luís Fernando Maia Santos Silva. – Porto Alegre: PPGC da UFRGS, 2010.

53 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2010. Advisor: João Luiz Dihl Comba.

1. Computer graphics. 2. Computational geometry. 3. Regular triangulations. 4. Mesh merging. 5. Object modeling. I. Comba, João Luiz Dihl. II. Title.

“Oh, so they have internet on computers now!”
— HOMER SIMPSON

ACKNOWLEDGMENTS

I have to first thank: João Luiz Dihl Comba, who give me the opportunity to join the UFRGS Computer Graphics Group, always watched my work closely, and helped me in all my five submissions; and Luis Gustavo Nonato for suggesting the idea of using Regular Triangulations to perform merging of meshes, and who helped me to push my limits while developing this work.

Thanks to my colleagues: Vitor Pamplona, Rafael Torchelsen, Denison Linus, Luiz Scheidegger and Tiago Etiene, with whom I had the pleasure to work; Leandro A. F. Fernandes and Francisco Pinto for interesting suggestions and discussions about mathematical concepts. I also thank all my lab colleagues for accepting my weird humor and standing my annoyances every day. I would like to mention the work of professors Carla M.D.S. Freitas, João D. Comba, Luciana P. Nedel and Manuel M. Oliveira in conducting the UFRGS Computer Graphics Group to a level of excellence, and thank CNPQ for sponsoring my research during these two years.

Thanks to all friends I have made in Porto Alegre. I will always remember Mesquita, Samantha, Pablo, Gustavo, Kao, and Márcio as friends for life.

Another special thanks goes to: Paulo Barbosa for always being a loyal and patient friend; Juan Marcus and Rafael Bezerra who also joined the challenge of masters along with me. To my family and all my friends, especially my parents and my brother André Luís for always being present in my life despite of the distance and for encouraging me along this journey. A very special thanks goes to Soraya who accepted my decision to join the masters, and was by my side in every situation I have went through. It would be harder without your help.

Thanks to Prof. Paulo Sérgio (UFPI), and Prof. Luiz Cláudio da Matta (UFPI) for the recommendation letters, and Prof. Francisco Vieira (UFPI) who always believed in my capacity and pushed me to the master level.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Simplicial meshes are used in many fields of Computer Graphics and Engineering, for instance, in visualization, simulation, prototyping, among other applications. This kind of mesh is often used as discrete approximations of continuous spaces, where they offer flexible and efficient representations.

Considerable effort is spent in generating good quality meshes, but in some applications the meshes can be modified over time. However, this kind of operation is often very expensive and inflexible, sometimes leading to results very different from the original meshes.

The ability to handle dynamic scenes reveals itself as one of the most challenging problems in computer graphics. This work proposes an alternative technique for updating simplicial meshes that undergo geometric and topological changes. It explores the property that a Weighted Delaunay Triangulation (WDT) can be used to implicitly define the connectivity of a mesh.

Instead of explicitly maintaining connectivity information, this approach simply keeps a collection of weights associated to each vertex. It consists of an algorithm to compute a WDT from any given triangulation, which relies on a breadth-first traversal to assign weights to vertices, and a subdivision strategy to ensure that the reconstructed triangulation conforms with the original one. This technique has many applications and, in particular, it allows for a very simple method of merging triangulations, which is illustrated with both 2D and 3d examples.

Keywords: Computer graphics, computational geometry, regular triangulations, mesh merging, object modeling.

Combinação de Malhas utilizando Triangulações Regulares Dinâmicas

RESUMO

Malhas simpliciais são utilizadas em várias áreas da Computação Gráfica e Engenharia, como por exemplo, em vizualização, simulação, prototipação, além de outras aplicações. Este tipo de malha é, geralmente, utilizada como aproximações discretas de espaços contínuos, onde eles oferecem representações flexíveis e eficientes.

Muito esforço é gasto visando gerar malhas de boa qualidade, porém, em alguns casos as malhas acabam sendo modificadas. Entretanto, este tipo de operação é geralmente custosa e inflexível, o que pode resultar na geraão de malhas bem diferentes das originais.

A habilidade de manipular cenas dinâmicas revela-se um dos problemas mais desafiadores da computação gráfica. Este trabalho propõe um método alternativo para atualizar malhas simpliciais que vai além de mudanças geométricas e topológicas. Tal método explora uma das propriedade das Tringulações de Delaunay com Pesos, que permite a usá-las para definir implicitamente as relações de conectividade de uma malha.

Ao contrário de manter as informações de conectividade explicitamente, a atual abordagem simplesmente armazena uma coleção de pesos associados a cada vértice. Além disso, criamos um algoritmo para calcular uma Tringulação de Delaunay com Pesos a partir de uma dada triangulação. O algoritmo consiste em uma busca em largura que atribui pesos aos vértices, e uma estratégia de de subdivisão para assegurar que a triangulação reconstruída será correspondente à original. Este método apresenta diversas aplicações e, em particular, permite a criação de um sistema simples de realizar combinação entre triangulações, que será ilustrada com exemplos em 2D e 3D.

Palavras-chave: computação gráfica, geometria computacional, triangulações regulares, combinação de malhas, modelagem.

1 INTRODUCTION

Simplicial meshes, also known as triangulations, are used in computer graphics, geometric computing, engineering, visualization, simulations, among other areas. They are preferred due to their ability to represent irregular and adaptive domains.

There are many papers dedicated to the optimization of mesh quality, and generation of good mesh triangulations (LUEBKE et al., 2002; GARLAND; HECKBERT, 1997; COHEN et al., 1996; TOURNOIS et al., 2009). Furthermore, several rendering aspects depend upon good quality meshes. For instance, texture mapping is degraded by topological changes, demanding continuous texture coordinates updates to ensure a correct mapping. Since much effort is spent in improving mesh triangulations, it is expected of applications to maintain such properties. In fact, when dealing with static scenes, these properties are kept. However, some applications demand the use of dynamic scenes, which can imply the loss of the original mesh quality. For instance, dynamic data structures for visibility ordering often require partial or total reconstruction of data structures (TORRES, 1990; CHRYSANTHOU; SLATER, 1992; NAYLOR, 1992; LUQUE; COMBA; FREITAS, 2005), and the capacity to handle these scenes reveals to be one of the most challenging problems in computer graphics.

A common problem that arises when handling dynamic scenes is that the result of a mesh processing operation can either require a costly mesh re-computation, thus impairing real-time usage, or it demands constant updates and additional storage to keep several information needed to perform this task. In particular, the connectivity information is hard to update, and it is often destroyed or modified during such operations. Often mesh operations such as merging, morphing, and simplification (among others) alter the number of vertices in the mesh (either by adding or deleting vertices), which requires that the connectivity must be changed accordingly.

While updating a mesh triangulation, it is necessary to attempt to maintain both consistence and correctness of adjacency and incidence relations between simplicial elements. There are works that focus on checking and repairing triangulations using data structures (e.g., kinetic data structures (GUIBAS, 2004)). Recently, these challenges have been driving forward development of point-based (hybrid) approaches (GROSS; PFISTER, 2007).

An alternative solution to this problem involves focusing only on triangulations whose connectivity can be implicitly derived from geometric constraints (TOURNOIS et al., 2009; CHENG et al., 1999; TOURNOIS; SRINIVASAN; ALLIEZ, 2009), such as Delaunay Triangulations (DT) (EDELSBRUNNER; SEIDEL, 1986). However, these approaches are rather limited, since in many cases the underlying meshes might not comply with the required geometric constraints, returning to the case where it is needed to manage mesh connectivity explicitly (CASTRO et al., 2009).

Another point to be considered is that the interplay of mesh operations might lead to different connectivity information depending on the order that operations are performed. Consider the situation where a set of vertices is removed from a mesh and later reinserted. The recovery of the exact original mesh is expected, since, as mentioned before, improving meshes triangulation demands great effort. While it is easy to enforce for the geometric coordinates of the vertices, preserving the connectivity is hard, and only a very restrictive class of surface meshes can be algorithmically reconstructed from the geometry of its vertices (i.e. disregarding connectivity information).

This work proposes an alternative technique for updating simplicial meshes that undergo geometric and topological changes and explores the property that a Weighted Delaunay Triangulation (WDT) can be used to implicitly define the connectivity of a mesh. The method naturally supports applications that require dynamic meshes, without the need for explicit connectivity management. A key contribution of this work is the extension of Delaunay-based implicit connectivity representations to general triangulations.

The presented proposal is based on a generalization of Delaunay triangulations called Weighted Delaunay triangulations (WDT), also known as Regular Triangulation (RT). WDTs are a generalization of DTs where a weight is associated with each vertex in the mesh. The weights can be modified in such a way to change the connectivity of the mesh, therefore providing to WDTs the ability to represent a wider collection of meshes. Still, not every triangulation can be represented as a WDT. A important feature of this work is to use a subdivision scheme to turn any mesh into a WDT through the addition of extra vertices (see Section 4.2).

This approach allows mesh operations to be performed by simply manipulating vertex weights. For instance, to remove a vertex requires only to set the weight to an appropriate value, which causes its automatic elimination from the hull, and therefore from the triangulation (EDELSBRUNNER; SEIDEL, 1986).

A previous work by Cignoni and De Floriani (Cignoni; De Floriani, 1998) proposed a technique for computing a WDT from a given mesh by modeling the weight constraints as a linear program. Balaven et al. (BALAVEN et al., 2002) also proposed a linear programming technique to solve the problem of transitional meshes. Unfortunately, this solution cannot be generalized, and the linear program might not have a solution in some cases. Our approach is based on a breadth-first traversal of the mesh that locally enforces convexity constraints on the lifting construction, together with a subdivision mechanism, which ensures that the resulting WDT conforms with the original mesh (see Chapter 4 for details).

The presented method offers a general way to deal with the lifting construction, allowing the definition of different merging techniques. These techniques can be constructed to produce results customized to particular applications. This work presents and discuss two ways to merge WDTs based on different ways of combining the lifted polyhedron of each mesh (see Chapter 5). It is also proposed a new adaptive scheme that maintains the geometric quality of simplices when merging meshes of distinct refinement levels (see Section 5.3). This approach generates an adaptive triangulation that varies smoothly between the meshes that will be merged.

The technique described in this dissertation was implemented and a number of experiments were performed to its validation. In particular, applications that demonstrate the merging of 2D and 3D meshes are presented at Chapter 6. Despite their simplicity, the proposed merging approaches automatically ensure that the connectivity between the meshes being merged remains consistent and well defined at all times, without the need

for complex data-structures.

In summary, the following contributions are introduced in this work:

- A new approach for computing a WDT from an arbitrary simplicial mesh;
- Two algorithms for merging regular triangulations;
- A simple fringe mechanism that establishes a smooth transition between merged meshes with different refinement levels;
- Validation of the proposed technique with examples in 2D and 3D

The weight computation and merging schemes are all based on very simple geometric constructions which do not demand complex data-structures. Besides, the method can automatically merge multiple triangulations with distinct mesh resolutions and arbitrary topology, while still ensuring consistent and well-defined connectivity. Those properties point out the method as simple and general. But it is also preservable, since mesh features tend to be preserved during the merging process, specifically the mesh anisotropy.

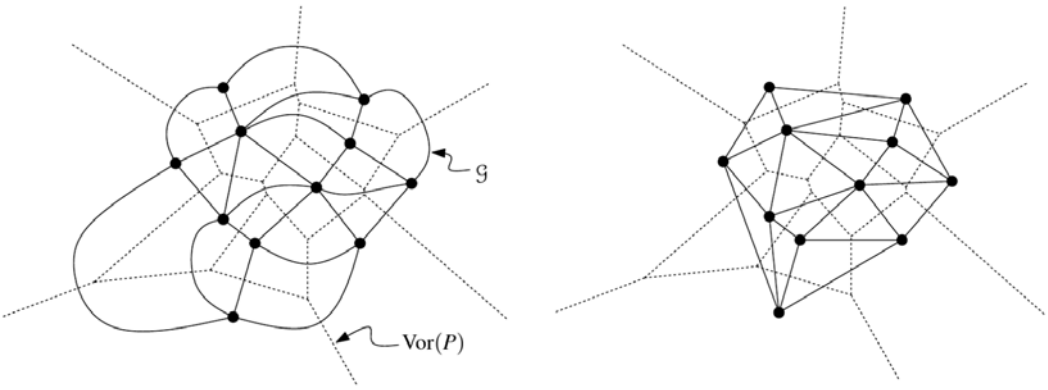
1.1 Structure of this Thesis

The remaining of this thesis is organized as follows: Chapter 2 introduces the WDTs and basic concepts that are fundamental to the understanding of this work. Chapter 3 reviews related work in comparison to the proposed approach. Chapter 4 describes the proposed technique to store the mesh triangulation on the vertices weights together with a subdivision mechanism, which ensures that the resulting WDT conforms with the original mesh. Chapter 5 presents the two merging methods developed using DRT. Chapter 6 illustrates the potential of the proposal in some applications. Chapter 7 summarizes relevant properties of this work and lists some avenues for future exploration.

2 WEIGHTED DELAUNAY TRIANGULATION AND CONCEPTS

There is a set of mathematical concepts and properties that should be explained before diving into the proposed approach. The purpose of this chapter is, therefore, to address these concepts, since they are essential to the understanding of how WDTs can be employed towards representing any triangulation, and how the merging techniques were developed.

The Delaunay triangulation was created by Boris Delaunay in 1934 (DELAUNAY, 1934), and is built from the dual graph of the Voronoi Diagram (BERG et al., 2000; AURENHAMMER, 1991) as shown in Figure 2.1. An inherent property of Delaunay triangulation is that it maximize the minimum of all the angles of the triangles in the triangulation, thus tending to avoid sliver triangles. Another key property of DTs is the fact that all their vertices can be lifted onto a convex polyhedron in one extra dimension (also known as the *lifting property* (EDELSBRUNNER; SEIDEL, 1986; AURENHAMMER, 1987)). However, this property is not true for non-Delaunay meshes.



(a) (b)

Figure 2.1: Relation between the dual graph of the Voronoi tessellation and the Delaunay Triangulation. a) A Voronoi diagram and its corresponding dual graph; b) Delaunay triangulation and the Voronoi diagram of the same set of points.

A Weighted Delaunay Triangulations (WDT), also called Regular Triangulation (RT), is a generalization of the Delaunay triangulation, where each vertex has an associated weight. On a WDT, the weight modifies the height of the lifted vertices, thus moving them in or out of the lifted polyhedron. Later in this chapter we show how we use this property to save a mesh connectivity.

The same way the DT has a duality relationship with the Voronoi Diagram, the WDT has a corresponding duality with an geometric entity called Power Diagram (PD), also known as Weighted Voronoi diagram. In this chapter, at first, we show the basic elements that compose PDs and WDTs, and their relation to the arrangement of planes one dimension higher. Afterwards we focus on the weights to explain the role they play on the WDT, and how changing weights affects the mesh connectivity. This is a crucial concept, since the weights are the key elements of this work, used on both methods of saving connectivity and merging meshes.

For didactical reasons, this chapter firstly defines the WDT for two-dimensional Euclidean space $\mathbb{R}^2$, to afterwards generalize it to higher dimensions ($\mathbb{R}^d$).

2.1 Basic Elements of a WDT

A *weighted point* s $\mathbb{R}^2 \times \mathbb{R}$ is mathematically defined by its location p $\mathbb{R}^2$ and weight w $\mathbb{R}$, and can be interpreted as a circle with center p and radius $w^{\frac{1}{2}}$. In this work we will use the concept of *lifted vertice*, a point of $\mathbb{R}^3$ which can be seen as a specific weighted point, defined in $\mathbb{R}^d$ as:

$$v = (v_{x_1}, \ldots, v_{x_d}, v_{x_1}^2 + \cdots + v_{x_d}^2 - w) \tag{2.1}$$

where $v_{x_1}, ..., v_{x_d}$ are the Cartesian coordinates of v and $v_{x_1}^2 + \cdots + v_{x_d}^2 - w$ is its weight. Section 2.3 shows the role played by the weights and how they change the triangulation of a mesh. Since the vertices position remain unchanged, all the operations of our framework are performed over the weights.

Let $\mathcal{T}$ be a two-dimensional triangulation in $\mathbb{E}^2$, and $V = v_1, \ldots, v_n$, $T = t_1, \ldots, t_m$ be the set of vertices and triangles of $\mathcal{T}$. The triangulation $\mathcal{T}$ is said to be *regular* if, and only if, there exists a set of points $V^+ = v_1^+, \ldots, v_n^+$ $\mathbb{E}^3$ such that V^+ is a subset of the vertices of a convex polyhedron $\mathcal{P}$, where, for all i, v_i^+ projects orthogonally on v_i, and the projection of downward faces of $\mathcal{P}$ is T. In other words, a d-dimensional regular triangulation is the vertical projection of the "lower side" of some $(d+1)$-dimensional convex polyhedron in $\mathbb{E}^{d+1}$.

Given a set of weighted points $V = v_1, \ldots, v_n$ in $\mathbb{E}^d$ and, the *weighted Delaunay triangulation* (WDT) is obtained by projecting downward faces of the called *lifted polyhedron* (a polytope, to be more precise), defined as the convex hull of *lifted vertices*.

Now that the concept of a WDT was presented, we will introduce the elements of PDs, to further present how they are related. The *power distance* between two circles s_i and s_j is defined as $\texttt{pow}(s_i, s_j) = d^2(p_i, p_j) - w_i - w_j$, where $d(\cdot, \cdot)$ is the Euclidean distance. In particular, the power distance between s_i and a point x $\mathbb{R}^2$ is given by $\texttt{pow}(s_i, x) = d^2(p_i, x) - w_i$. A *chordale* of two non concentric circles s_i and s_j, is the set of points x such that $\texttt{pow}(s_i, x) = \texttt{pow}(s_j, x)$. It is well known that the three chordales defined by the circles s_i, s_j, and s_l (with non-collinear centers) intersect in a common point p that fulfills $w = \texttt{pow}(s_i, p) = \texttt{pow}(s_j, p) = \texttt{pow}(s_l, p)$. Furthermore, the circle s with center p and radius $w^{\frac{1}{2}}$ satisfies $\texttt{pow}(s_i, s) = \texttt{pow}(s_j, s) = \texttt{pow}(s_l, s) = 0$ and is called the *orthocircle* of s_i, s_j and s_l.

There is a straight relation between a chordale on the PD and an edge of the corresponding RT, since each chordale of two circles, for instance s_i and s_j, gives rise to an edge connecting the centers of s_i and s_j on the RT. We will show, later in this section, that this relation between PD and RT comprises many other elements and properties. Figure 2.2 shows both relations between the PD and the RT, and the *chordales* and edges.

Let $h_{s_i}(s_j)$ denote the closed half-space bounded by the chordale of s_i and s_j that contains the points with the smallest power distance to s_i. The *power cell* of s_i is given by

$$V(s_i) = \bigcap_{j,\, j \neq i} h_{s_i}(s_j) \tag{2.2}$$

Letting $S = \ s_1, \ldots, s_n \ $ be a set of non-concentric circles, the power cells of the circles in S comprise a partition of $\mathbb{R}^2$ in convex polygons, called *Power Diagram* ($PD(S)$). The Power Diagram coincides with the usual Voronoi Diagram if all circles in S have the same radius. In other words, in the same way the WDT is a generalization of the DT, the Power Diagram is an extension of the Voronoi Diagram.

However, the Power Diagram produced by circles with different radius may not satisfy the containment condition, that is, a circle s_i may not be contained in its power cell. Furthermore, depending on the weights, a circle can give rise to an empty power cell. In this case, such circle is called *redundant*. According to this property, in order to exclude a vertex from the PD, and therefore from the RT, it is only necessary to reduce its weight so that it gives rise to an empty cell. Under the same premise, a vertex can be inserted into both RT and PD by simply giving it a big enough weight to create a non empty power cell.

If we assume that the center of the circles in S are not collinear, then the intersection of three (or more) power cells is either empty or a vertex of $PD(S)$. It is not difficult to realize that a vertex of $PD(S)$ is the center of the orthocircle defined by at least three circles whose power cells have non-empty intersection. In a non-degenerate case, each vertex v of $PD(S)$ is defined by the intersection of exactly three power cells. In such case, the center of the three circles whose power cells intersect in v define a triangle and the union of all triangles makes up a simplicial complex, called the RT of the points.

In fact, such a correspondence leads to a duality relationship that associates circles to power cells, power cell edges to triangle edges and Power Diagram vertices to triangles. In this last case the Power Diagram vertex is called the orthocircle of the associated triangle. Note that redundant circles do not have dual power cells, and thus do not appear in the RT. From the duality property we can obtain the RT from the Power Diagram (and vice versa) in $O(n)$. In the two-dimensional case there are algorithms that compute both structures in $O(n \log n)$ (EDELSBRUNNER; SHAH, 1996) (expected time). In Chapter 4 we present ways to use the duality relation mentioned above in order to set weights to the vertices in such a way their corresponding planes maintain a given triangulation.

The concepts above mentioned have corresponding elements in the three-dimensional Euclidean space $\mathbb{R}^3$. A weighted point can be seen as a sphere with center p and radius $w^{\frac{1}{2}}$, while the chordales of a PD being a shared face between two adjacent points. In $\mathbb{R}^3$ the *orthocircle* is defined by four distinct weighted points and the *power cell* is a three-dimensional volume surrounding the point.

2.2 The Duality between PDs and RTs

An important fact that is deeply explored in this work is the strong relation between Power Diagrams in $\mathbb{R}^2$ and arrangement of planes in $\mathbb{R}^3$ (consequently, for a Power Diagram in $\mathbb{R}^3$ the arrangement of planes are in $\mathbb{R}^4$). Such a relationship can be obtained by associating each circle $s_i \quad S$ with a non vertical plane in $\mathbb{R}^3$, which is defined by the following function:

$$\pi_{s_i}(x) = 2 < x, p_i > - < p_i, p_i > + w_i \tag{2.3}$$

where $< \cdot, \cdot >$ is the usual dot product in $\mathbb{R}^2$. It can be shown that a point x $\mathbb{R}^2$ lies in $V(s_i)$ iff, at x, $\pi_{s_i} \geq \pi_{s_j}$, $i = j$. Thus, $PD(S)$ is the vertical projection of the Upper Envelope of the planes π_{s_i}, which is a convex polyhedron in $\mathbb{R}^3$ (for a detailed definition of Upper Envelope and its relation with the Power Diagram see (EDELSBRUNNER; SEIDEL, 1986)). Figure 2.2 illustrates the correspondence between Power Diagram, Regular Triangulation, and Upper Envelope.

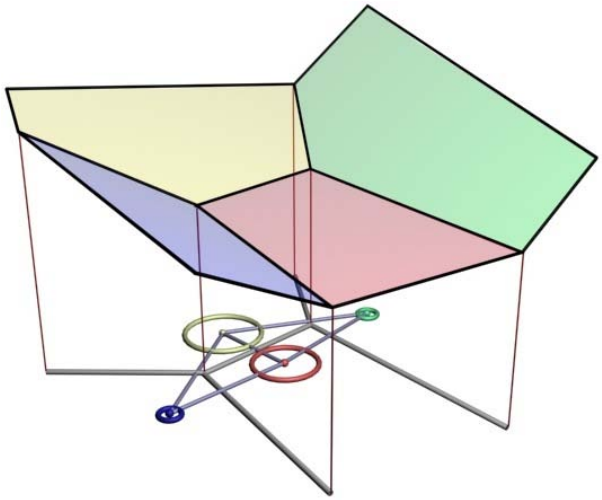


Figure 2.2: Relation between Power Diagram, Regular Triangulation, and Upper Envelope.

2.3 On the interpretation of the Weights

Since the weight are the basis of this work, the understanding of their properties is fundamental to follow the proposed methods. Our approach does not change the vertices s positions. All techniques developed were built upon changing the weights of the vertices. This section intends to explain the role played by the weights on both PD and RT as well as the interpretation of the consequences on their changing.

Equation (2.3) helps us to understand the behavior of $PD(S)$ when the radius (weight) of a circle changes. Suppose that both $PD(S)$ and $RT(S)$ have already been computed from a set of circles S and let s_i be a circle of S. When the radius of s_i increases the plane π_{s_i} moves up, leading to the center of the orthocircles that comprise the vertices of $V(s_i)$ to move away from p_i (center of s_i), as illustrated in Figure 2.3.

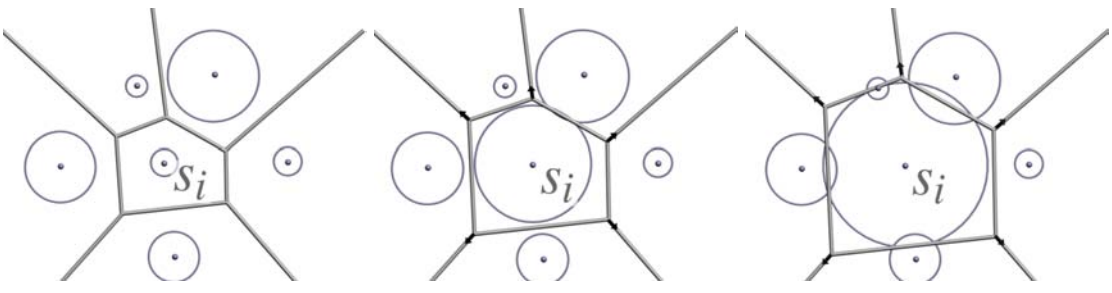
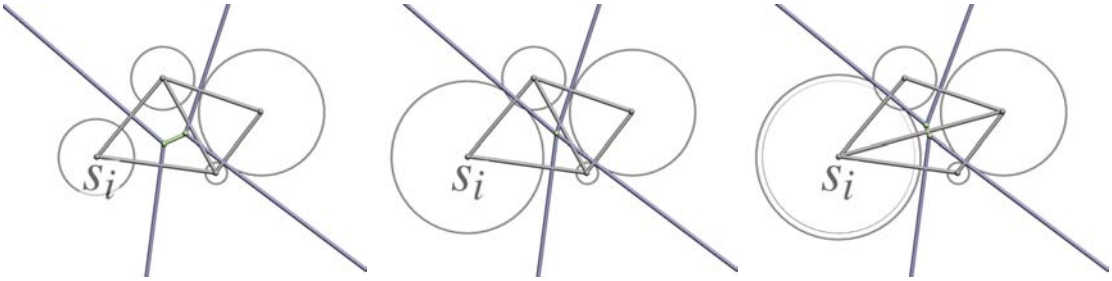


Figure 2.3: The center of the neighbor orthocircles move away from the center of s_i when w_i increases.

Due to the duality relationship, the RT remains unchanged until the increase of w_i gives rise to a degenerate case, that is, two diagram vertices are collapsed into a single one, causing the intersection of more than three power cells in such vertice. When a degenerate case occurs, any further increase of w_i gives rise to an edge flipping in the RT, as shown in Figure 2.4. This process has a corresponding event in planes defined by the circles onto the corresponding arrangement of planes. The inverse phenomenon occurs when the weight w_i is reduced.



(a) (b) (c)

Figure 2.4: Edge flipping after a degenerate case. a) original configuration; b) the increase in the weight of s_i gives rise to a degenerate case; c) further increase of w_i causing an edge flip.

The dynamic process described above allows us to control the weights so as to avoid edges flipping. Furthermore, the understanding of such a dynamics is the main key to convert any given triangulation into a WDT, as we shall see in Chapter 4.

2.4 Local Convexity

In this section we present the concept of local convexity, which (in Chapter 4) will be used to assign weights to the vertices of a mesh in order to encode its original triangulation. Again, for didactical purposes, we show the concepts in $\mathbb{R}^2$.

Let t_i and t_j be two adjacent triangles sharing a common edge r_{ij} (in $\mathbb{E}^3$ r_{ij} will be a triangle shared by two adjacet tetrahedra) and t_i^+, t_j^+ be the lifted simplices of t_i and t_j. The edge r_{ij}^+ is said to be *locally convex* if the $(d+1)$-simplex generated from the union of vertices in t_i^+ and t_j^+ lies on the upper half-space of each hyperplane containing t_i^+ and t_j^+ (Figure 2.5 shows the lifting process in $\mathbb{R}^2$). A polyhedron in $\mathbb{E}^{d+1}$ is called locally convex if all edges shared by two triangles are locally convex.

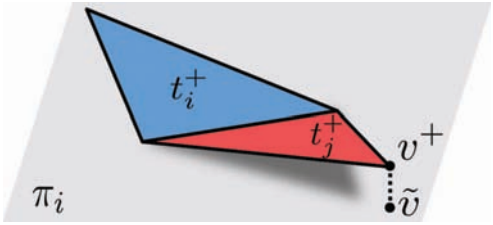


Figure 2.5: Simplices are lifted to ensure local convexity.

Local convexity is a necessary, but not sufficient, condition to ensure global convexity of a polyhedron (MEHLHORN et al., 1996). However, in the particular case of a polyhedron generated by lifting a convex triangulation, local convexity leads to global convexity

(see (DEVILLERS et al., 1998) for details) and, as remarked by Balaven et al. (BALAVEN et al., 2002), this property has widely been used as a mechanism to verify whether a given triangulation is regular. As we show in the next section, the association between lifting and local convexity comprises the core of our approach.

It is important to highlight that not every triangulation is regular. See, for example, the classic seven triangle mesh shown in (EDELSBRUNNER; SHAH, 1996). Because of this, it may be impossible to build a convex polyhedron whose projection matches a given triangulation, and therefore any lifting of this triangulation will contain at least one non-locally convex $(d-1)$-simplex.

2.5 Summary

This chapter presented a brief overview of the elements that compose a WDT as well as its duality relation with the arrangement of planes. Such description was intended to provide just enough information to familiarize the reader with some concepts that will be used in the next chapters. An in-depth description of these subjects is beyond the scope of this thesis and can be found in the works mentioned on Chapter 3.

3 RELATED WORK ON REGULAR TRIANGULATIONS

This chapter summarizes the most important academic efforts concerning Delaunay Triangulations and connectivity oblivious methodologies in computer graphics. Since the proposed technique is validated using dynamic meshes, it also reviews the main issues regarding connectivity reconstruction in this context.

Regular Triangulations and Power Diagrams (EDELSBRUNNER; SEIDEL, 1986; AURENHAMMER, 1987) are basic geometrical tools that have been often used for problems related with surface (AMENTA; CHOI; KOLLURI, 2001), quality mesh generation for numerical simulations (CHENG; DEY; RAMOS, 2007; CHENG; DEY; RAY, 2005; CUADROS-VARGAS et al., 2005), and depth sorting (EDELSBRUNNER, 1989; Cignoni; De Floriani, 1998). The essential task of assigning weights to points makes RT and Power Diagram an extremely flexible and dynamic geometrical structure, and its usefulness goes beyond mesh generation. Several theoretical and computational frameworks have been developed towards understanding and exploring such potentiality.

In regard to techniques that handles connectivity automatically, their use for manipulation of simplicial meshes can be divided into two main categories: methods that reproduce an existing triangulation from geometric information, and methods that build a triangulation based on the constraints given by curves and/or surfaces defined in the domain of interest. In both cases, the vast majority of approaches relies on the properties of DTs and WDTs to avoid explicitly handling the mesh connectivity. These methods have solid foundations on both theoretical (AURENHAMMER, 1987; AURENHAMMER; IMAI, 1987; EDELSBRUNNER; SEIDEL, 1986) and computational (VIGO; PLA; COTRINA, 2002; EDELSBRUNNER; SHAH, 1996; EDELSBRUNNER, 2001) sides.

3.1 Theoretical works

Theoretical techniques construct a WDT corresponding to a given mesh by computing a valid set of weights for its vertices. The advantage of this approach is that, once the weights are computed, connectivity information can be discarded. The original triangulation can then be rebuilt algorithmically from vertex coordinates and weights. This technique has great advantages over connectivity dependent approaches, since many mesh processing operations have to create versions of input meshes with different connectivity information. For instance, morphing between meshes requires generating an intermediate mesh that combines the geometric and topological aspects of the input meshes.

Common to all methods discussed in the literature is the definition of a correspondence between vertices and connectivity information. This is often done by first creating a mapping into an unified space for the geometry and connectivity of both meshes, and by a synthesis algorithm that generates an intermediate mesh from this space.

Several papers discuss aspects that range from how and when such mappings are defined (FLOATER; C.GOTSMAN, 1999; SURAZHSKY; GOTSMAN, 2001; DANCIGER; DEVADOSS; SHEEHY, 2006), to how mesh information can be mapped to a parametric space, such as the inter-surface mapping described in (SCHREINER et al., 2004). There are several variations on how the connectivity information is generated for intermediate meshes (AHN; LEE, 2002; AHN; LEE; SEIDEL, 2004; PARUS; KOLINGEROVá, 2004; LEE et al., 1999; ZORIN; SCHRöDER; SWELDENS, 1996; BREEN et al., 2001). This work offers an implicit way to reconstruct connectivity information that can be used in conjunction with several of the methods proposed.

A well explored way for computing a valid set of weights is to reduce the problem to a linear programming problem, looking for an optimal weight distribution (PIRES, 2008). Edelsbrunner (EDELSBRUNNER, 1999) explored the equivalence between RTs (Power Diagrams) and the boundary of convex polyhedra one dimension higher (also called the lifting property (EDELSBRUNNER; SEIDEL, 1986; AURENHAMMER, 1987)) as shown in Figure 3.1. From this relations, Edelsbrunner derived an algebra of spheres to formulate a new paradigm for designing smooth surfaces.

The relationship between RT and convex polyhedra has also been investigated by Massada et al (MASADA; IMAI; IMAI, 1996), who proposed an output-size sensitive algorithm to enumerate all RTs. From a theoretical point of view, it has been shown a close connection to well established algebraic concepts, such as the relation of RTs and convex polyhedra to Gröbner bases (DELOERA; STURMFELS; THOMAS, 1995).

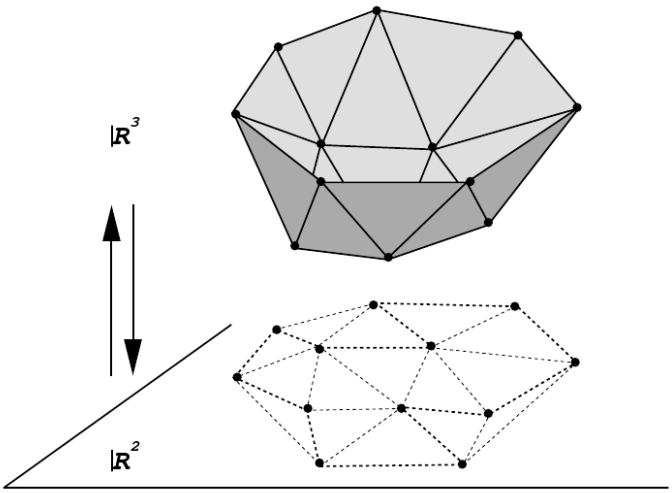


Figure 3.1: Relation between vertices and its projections onto the upper envelope as presented in (BALAVEN et al., 2002).

Cignoni and De Floriani (Cignoni; De Floriani, 1998) use this formulation to perform depth sorting of unstructured simplicial meshes, without having to store connectivity information. They explore the lifting property of RTs to compute a convex polyhedron in a $(d+1)$-dimensional space whose projection on the d-dimensional space is the simplicial complex one wished to depth sorting.

Balaven et al. (BALAVEN et al., 2002) used linear programming to generate transitional meshes for oil reservoir simulations. However, their solution has the limitation of only dealing with meshes that can be represented as WDTs. When this is not the case, they have to fall back to explicit management of connectivity information. As far as we

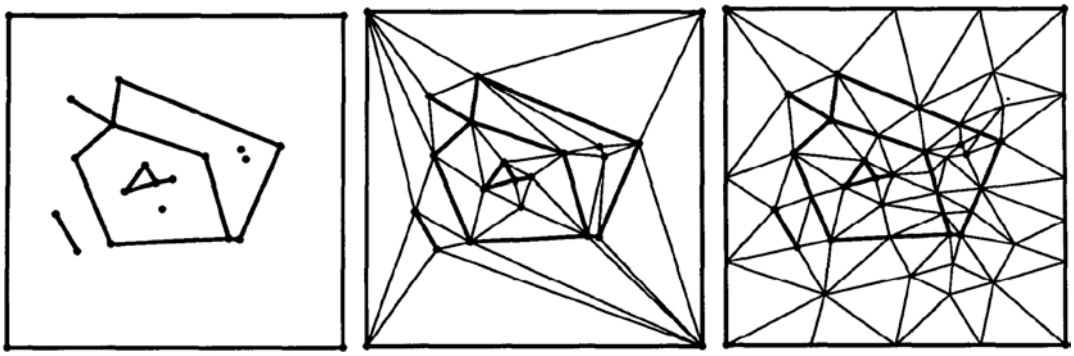
know, no satisfactory alternative has been proposed to overcome these limitations, and our approach addresses the problem of weight computation in a different way, avoiding the linear programming formulation altogether.

3.2 Computational works

Computational techniques are much more numerous, and in general these methods are concerned with either object modeling or domain decomposition for numerical simulation. While for object modeling, the focus is on generating a triangulation that approximates a given model, using geometric certificates to filter undesirable simplices, for domain decomposition methods, the focus is on ensuring that output meshes comply with constraints given by curves and/or surfaces defined inside the domain of interest. Despite their different focus, both branch of approaches aim at enforcing the quality for simplicial elements.

Amenta et al. (AMENTA; CHOI; KOLLURI, 2001) created a Delaunay-based surface reconstruction from point clouds. They first approximate the medial axis transformation of the object, and from it they apply an inverse transformation to produce the surface. Their approach is based on (AMENTA; BERN, 1998; AMENTA; BERN; KAMVYSSELIS, 1998) and uses the vertices farthest of its Voronoi cell of a surface, which were defined as poles, to be selected as an approximation of the medial axis.

Delaunay triangulations were also used on isosurfacing methods, for instance Cheng et al. (CHENG et al., 2004) maintained the restricted Delaunay of a set of points sampled on the surface by using a height function on the surface to generate more sample points dictated by certain conditions while updating the triangulation. Boissonnat and Oudot (BOISSONNAT; OUDOT, 2005) introduced the concept of Lipschitz radius which was used to determine a sampling condition to be used on surface reconstruction.



(a) (b) (c)

Figure 3.2: Delaunay refinement comparison (RUPPERT, 1993). a) Sample input planar straightline graph (PSLG); b) typical (non-Steiner) triangulation of PSLG and bounding box; c) Delaunay refinement algorithm s output.

Many studies were conducted on mesh refinement making use of Delaunay Triangulations. For example, Chew (CHEW, 1987) proposed an $O(n \log n)$ time algorithm to construct a constrained Delaunay Triangulation using a divide-and-conqueror approach. Thereafter, Ruppert (RUPPERT, 1993) created a technique based upon Chew s work which consisted in a successive refinement algorithm (see Figure 3.2). Later on, Shewchuk (SHEWCHUK, 2002) improved both Chew s and Ruppert s techniques and helped to

solve a weakness their approaches presented when handling meshing of non-manifold domains with small angles.

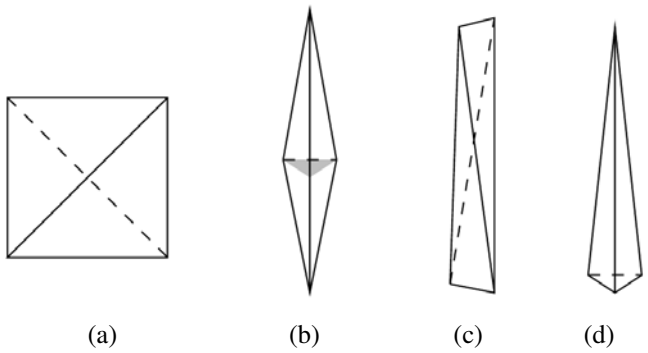


Figure 3.3: Examples of poor quality tetrahedra (LABELLE, 2006) eliminated by Sliver Exudation.

Another mesh refinement technique using Delaunay Triangulations is the work of Cheng et al. (CHENG et al., 1999) named Sliver Exudation. Slivers can be defined as tetrahedra whose four vertices lie close to a plane and whose projection to that plane is a convex quadrilateral with no edge (CHENG et al., 1999). In his work, Cheng explored Weighted Delaunay Triangulations properties and proposed an algorithm to compute weights for vertices of a triangulation in such a way to eliminate slivers (see Figure 3.3). Furthermore Edelsbrunner and Guoy (EDELSBRUNNER; GUOY, 2001) conducted an experimental study of sliver exudation to prove its effectiveness. Section 5.3 of this work introduces the fringe, which is an approach used to adapt meshes of different granularity during the merging process. It also has the advantage of avoiding the insertion of slivers when reconstructing the triangulation.

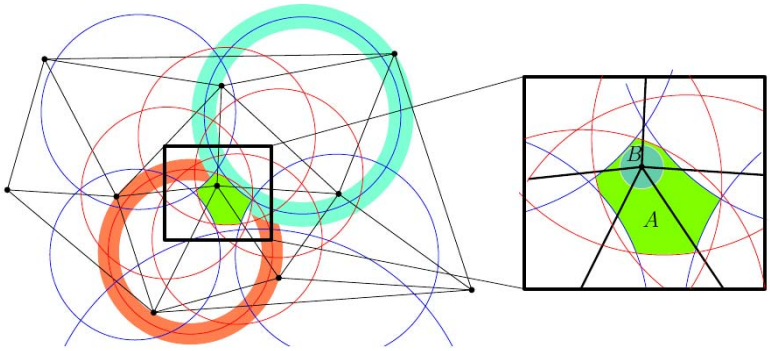


Figure 3.4: The region A is the safe region of the point placed on the center of B, while B is its tolerance region (CASTRO et al., 2009).

Castro et al. (CASTRO et al., 2009) explored the problem of updating Delaunay Triangulations when its vertices move. They developed a set of filters based upon the concept of vertex tolerance to determine whether it is better to update the whole triangulation from scratch or only reallocate the vertices. From these filters, Castro was able to determine a safe region and a tolerance region within a vertex could move without the need of updating the triangulation. Figure 3.4 illustrates the concepts of safe region and tolerance region.

The majority of the works referred before are examples of connectivity oblivious object modeling. Within this context, the proposed methodology can be seen as an alternative mechanism to encode connectivity information on vertices weights and a WDT based merging approach. The proposed merging technique proves that many object modeling methods can be implemented by extending this framework.

4 COMPUTING WEIGHTS

In the previous chapter we presented the main concepts upon which RT and PD stand, as well as the notion of local convexity and the importance of weights on changing the structure of the triangulation on a WDT. This chapter introduces the approach we developed to create a conformal mesh from a original triangulation by computing and assigning weights to vertices. We also present a method to handle special cases when the weight assignment procedure does not ensure a complete match between the original and the conformal mesh.

We start from a given triangulation $\mathcal{T}$ in $\mathbb{E}^d$, and our goal is to compute a set of weights for its vertices to represent $\mathcal{T}$ as a weighted Delaunay Triangulation (see Figure 4.1). We rely on a geometric construction in $\mathbb{E}^{d+1}$ that explores the lifting and local convexity properties previously mentioned in Section 2.4 to compute the weights. A triangulation conforming to the original mesh can be reconstructed by an algorithm that computes a weighted Delaunay triangulation from vertex coordinates and associated weights (see, for example, (EDELSBRUNNER; SHAH, 1996)). Hence, incidence and adjacency relations between simplices can be recovered from weighted vertices, and therefore we can disregard the mesh connectivity.

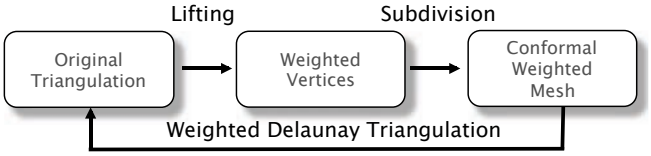


Figure 4.1: WDT computation: the original triangulation is traversed in a breadth-first manner to assign weights to vertices using the lifting construction. A following subdivision step ensures that the resulting weighted mesh conforms with the original triangulation.

4.1 The Proposed Algorithm

The proposed algorithm starts by an elected d-simplex (a triangle in $\mathbb{R}^2$ or a tetrahedron in $\mathbb{R}^3$) which is called *seed*. If no *seed* is specified we chose the d-simplex closest to the centroid, thus the lifted polyhedron is prone to be balanced.

For a better understanding of the method we will start to explain the algorithm using the two-dimensional case and further move to higher dimensional cases. Take t_i and t_j as triangles of a triangulation $\mathcal{T}$ sharing a common edge r_{ij}. Suppose that weights have already been assigned to vertices of t_i, thus it becomes t_i^+. We can compute the hyperplane

π_i containing t_i^+, that is, π_i is the *support hyperplane* of t_i^+. Let v be the vertex of t_j opposite to r_{ij} and $\tilde{v}$ be the vertical projection of v onto π_i, as illustrated in Figure 2.5. Notice that a small vertical perturbation of $\tilde{v}$ toward the positive half-space of π_i, denoted as v^+, makes r_{ij}^+ locally convex with respect to t_i^+, and $t_j^+ = r_{ij}^+ \quad v^+$. Therefore, moving to the generic case (in $\mathbb{E}^d$) the coordinates of $\tilde{v}$ are $(v_{x_1}, \ldots, v_{x_d}, \tilde{v}_{x_{d+1}})$, and we can use equation (2.1) to define the weight w_v of the vertex v as:

$$w_v = v_{x_1}^2 + \ldots + v_{x_d}^2 - (\tilde{v}_{x_{d+1}} + \epsilon) \tag{4.1}$$

where ϵ is a small vertical perturbation of $\tilde{v}$ that makes r_{ij}^+ locally convex with regard to t_i^+ and t_j^+.

The same formulation was used for the three-dimensional case where two adjacent tetrahedra share a common face and the computed hyperplane π_i is in $\mathbb{E}^4$. In the Chapter 6 of this work we show results of the proposed technique along with the merging approach presented in Chapter 5 illustrated with examples in 2D and 3D.

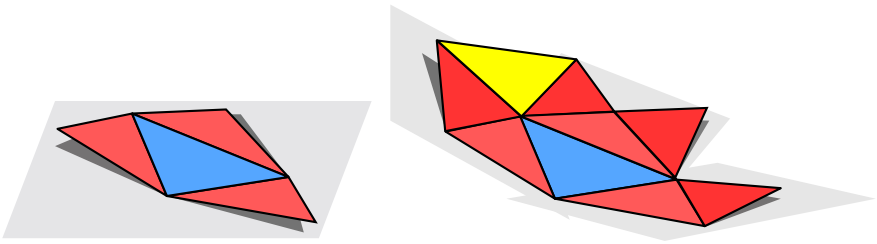


Figure 4.2: Simplices are lifted using a breadth first scheme, red simplices are lifted ensuring local convexity, but local convexity can not be ensured for edges of the yellow triangle.

By repeating the construction above for all neighbor simplices of each lifted d-simplex (red triangles in Figure 4.2), we can ensure that each lifted $(d-1)$-simplex is locally convex. Although weights are assigned consistently to vertices of $\mathcal{T}$, some $(d-1)$-simplices (an edge in $\mathbb{R}^2$ or a face in $\mathbb{R}^3$) of the original triangulation are not handled during the weight computation process described above (see the yellow triangle in Figure 4.2). However, most of these cases are locally convex naturally and do appear in the reconstructed triangulation.

In Section 4.2 we describe how to handle the simplices that do not automatically fit this criterion by employing a subdivision process to handle these simplices, thus ensuring that a convex polyhedron is generated as output. As a result, the original triangulation can be rebuilt from the weighted vertices, even though a few simplices of $\mathcal{T}$ may appear subdivided after the reconstruction.

4.2 Enforcing Conformal Meshes

Absent simplices can be forced to appear in the weighted triangulation by using a subdivision process, which is usually employed in many Delaunay-based mesh generation and modeling techniques (NONATO et al., 2005). These subdivision strategies ensure that the reconstructed weighted triangulation conforms to the original triangulation. We will present the formulation used to fix this issue on the n-dimensional case, to further exhibit a two-dimensional example of the method.

Formally, given a triangulation $\mathcal{T}$ in $\mathbb{E}^d$, the subdivision process ensures that for each k-simplex r of $\mathcal{T}$ there exists a set of k-simplices $r_{w_1}, \ldots, r_{w_s}$, $s \geq 1$ in $\mathcal{T}_w$ (the reconstructed weighted triangulation) such that $r = r_{w_1} \cdots r_{w_s}$, where r is the underlying space of r.

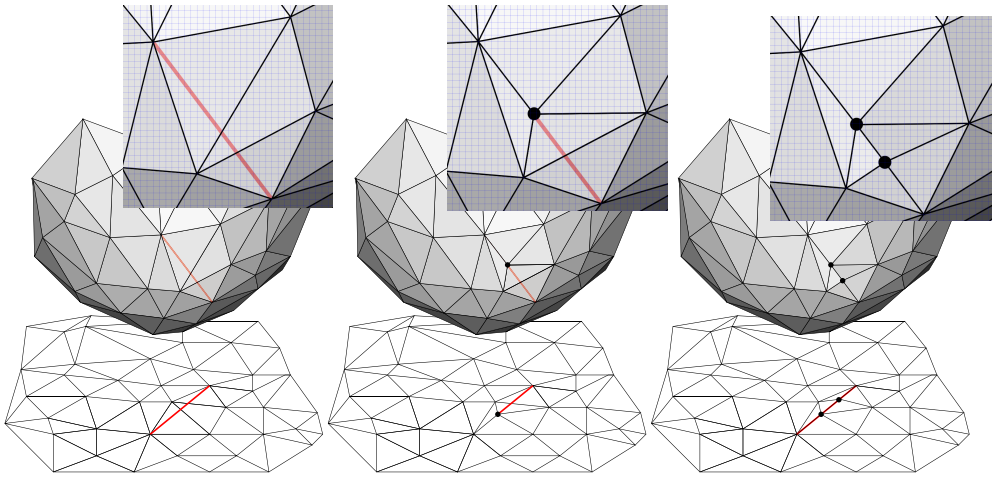


Figure 4.3: A red edge is absent from the final triangulation (first). Absent edges are included by a recursive subdivision process that generate new vertices with proper weight assignments (second and third).

Suppose that an edge $e \in \mathcal{T}$ is absent from $\mathcal{T}_w$, where $\mathcal{T}_w$ is the weighted triangulation obtained as described before. The subdivision step consists of splitting e at its midpoint v_e, and assigning a weight to v_e to ensure that it appears in the weighted triangulation. As illustrated on the left of Figure 4.3, the absent edge e intersects a set T_{w_e} of d-simplices in $\mathcal{T}_w$. To keep changes as local as possible, we ensure that the weight of v_e only alters the simplices of T_{w_e}, and maintains the remaining triangulation unchanged. This property can be satisfied by computing the weight of v_e as follows: let t_{v_e} be the d-simplex of $\mathcal{T}_w$ containing v_e and $t^+_{v_e}$ be the lifting of t_{v_e} in $\mathbb{E}^{d+1}$. We call l_{v_e} the vertical line through v_e, and compute the intersection point $\tilde{v}_{e_1}$ between l_{v_e} and the support hyperplane of $t^+_{v_e}$, as well as the highest intersection $\tilde{v}_{e_2}$ of l_{v_e} with the hyperplanes supporting the lifted d-simplices in $\tilde{\mathcal{T}}_w - T_{w_e}$.

It is easy to see that any choice of weight assignment that places v^+_e strictly between $\tilde{v}_{e_1}$ and $\tilde{v}_{e_2}$ ensures that v_e will appear in the weighted triangulation (v^+_e will be underneath the lifted convex hull). This approach only modifies the simplices in T_{w_e}, since v^+_e is above the hyperplanes supporting the simplices in $\tilde{\mathcal{T}}_w - T_{w_e}$. In our current implementation, we set a weight that places v^+_e halfway between $\tilde{v}_{e_1}$ and $\tilde{v}_{e_2}$. Although this procedure adds v_e to the weighted triangulation, it does not ensure that each sub-segment of e will appear in the triangulation. If a sub-segment still does not appear, we recursively apply the same procedure. Since the lifting of each new vertex is placed below existing simplices, the subdivision process finishes in a finite number of steps. In practice, very few absent simplices demand more than one or two iterations.

The procedure described above for recovering absent edges can be extended to enforce triangular faces in three-dimensional triangulations. The centroid of each missing triangular face is inserted and positioned exactly as was done for absent edges. In fact, in the three-dimensional case, we first recover all missing edges before splitting absent faces, since most missing triangles naturally appear after the edges are recovered.

4.3 Summary

Conventional approaches to weight computation try to solve a linear programming problem (BALAVEN et al., 2002), which may not have a solution. On the other hand, we use a deterministic procedure that builds an approximation of a locally convex polyhedron in $\mathbb{E}^{d+1}$. However, this step generates a preliminary weight assignment that might not enforce local convexity to some $(d-1)$-simplices. The proposed subdivision method solves this issue.

A major advantage of using WDT to process meshes is that it automatically ensures that the connectivity of the meshes remains consistent without the need to handle complex data-structures. Besides, there are numerous possible applications that can be built on top of this approach by using the weights to manipulate the conformal mesh. Chapter 3 showed many works developed using linear systems, and some of those applications could make use of this method. Mesh simplification, morphing, and merging, as we will show in Chapter 5, are some examples of applications that can be performed by simple manipulation of the lifted polyhedron created by our algorithm. In this work we focused on developing two merging approaches which testify the effectiveness of the developed method.

5 MERGING WEIGHTED DELAUNAY TRIANGULATIONS

Now that we presented how weights can be used on a WDT to create a conformal triangulation from a given mesh, we will introduce two merging methods developed exploring the properties of WDTs. The procedure of assigning weights to vertices of a mesh ends up building a lifted polyhedron, and the triangulation of the mesh can be recovered from the lower side of its convex hull.

Based on the property mentioned above, we realized that the merging of meshes could be performed by simple manipulation of their lifted polyhedra. For instance, Figure 5.1 shows where the smaller lifted polyhedron will be projected (at left), and how it can be translated in relation to the other one to generate the merged triangulation (at right). Notice that, after the merging process (at right), the overlapped mesh (the big one) has some edges added in order to create a consistent triangulation with the small one.

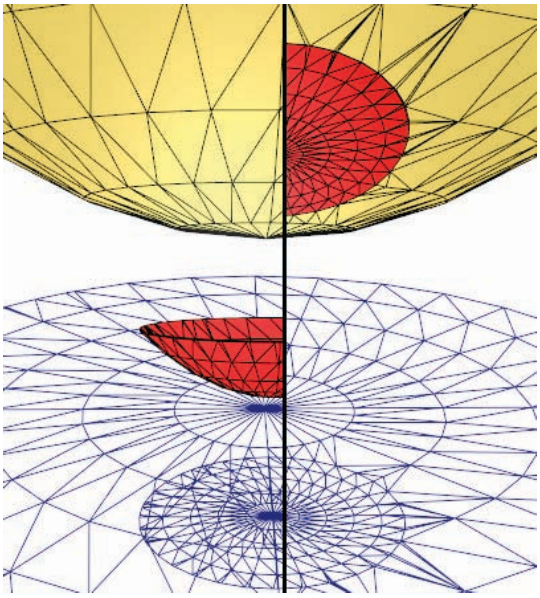


Figure 5.1: The merging of meshes can be performed by simple manipulation of the lifted polyhedra. At left is shown a preview of where the small mesh will be merged, and at right the resulting mesh is created by translating and joining the lifted polyhedron.

This chapter describes two techniques that use a lifted polyhedron construction for merging of WDTs. Those approaches were designed to keep changes as local as possible and only modify a small neighborhood of the merged region. Since meshes are often represented with different levels of detail or refinement, merging results are prone to generate badly shaped simplicial elements. We address this issue by introducing the concept

of fringe, which will be explained in Section 5.3.

One of the main advantages of a WDT is the fact that incidence and adjacency relations among simplices are automatically resolved, thus avoiding the need for connectivity manipulation. This benefit can be greatly explored in applications involving the interaction between two or more meshes, where the guarantee of consistent connectivity between the meshes is usually a complex and painful task.

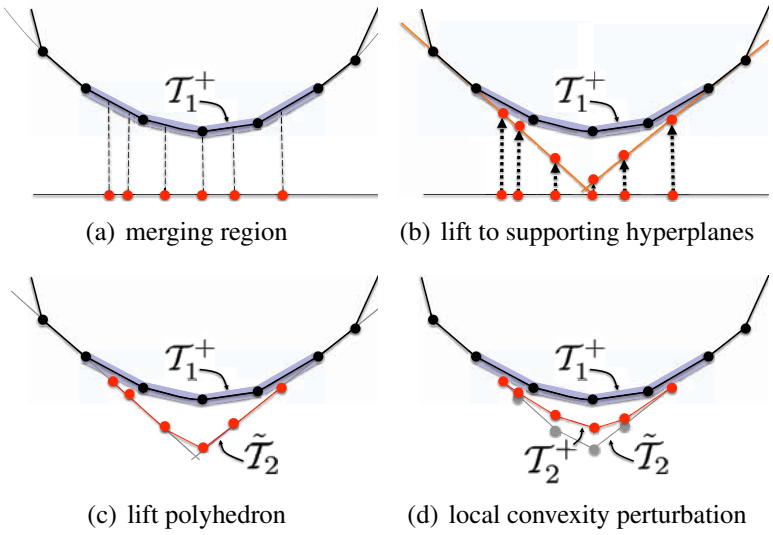
In summary, the two merging approaches differs in the number of flips on the merged mesh and the number of local changes they cause on the merged region. The convexification-based merging is introduced in Section 5.1, and it works by changing the lifted polyhedron construction as to fit it into the merged area, thus tempting some flips at expense of affecting a small area of the merged region. In Section 5.2 the translation-based merging is introduced as an alternative method that keeps the mesh triangulation unchanged, and woks by adjusting the vertical position of the lifted polyhedron construction. Since the lifted polyhedron construction is not changed, it may cause some vertices to be overlapped, thus affecting more neighbors of the merged area.

5.1 Convexi cation-based Merging

The first proposal is called *convexification-based* merging, and is illustrated in Figure 5.2. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two triangulations and suppose that $\mathcal{T}_2$ must be merged with $\mathcal{T}_1$ while keeping $\mathcal{T}_1$ as unchanged as possible (i.e. we only want to affect a neighborhood surrounding $\mathcal{T}_2$). We position $\mathcal{T}_2$ inside $\mathcal{T}_1$ and lift the polyhedron $\mathcal{T}_2$ in such a way that $\mathcal{T}_2^+$ is squeezed up between the lifted polyhedron $\mathcal{T}_1^+$ and the supporting hyperplanes for those simplices of $\mathcal{T}_1^+$ are not affected by the merging.

The squeezing process is composed of three steps. First, the d-simplices of $S \subset \mathcal{T}_1$ that do not intersect $\mathcal{T}_2$ are identified (Figure 5.2.a) and their supporting hyperplanes computed (Figure 5.2.b). In the second step, we set the weight of each vertex v of $\mathcal{T}_2$ in such way that the vertex is placed at the highest intersection $\tilde{v}$ between the vertical line passing through v and the supporting hyperplanes of the d-simplices in S (Figure 5.2.b). This construction results in a polyhedron in $\mathbb{E}^{d+1}$ that we denote by $\tilde{\mathcal{T}}_2$ (Figure 5.2.c). The final step makes the $(d-1)$-simplices of $\tilde{\mathcal{T}}_2$ locally convex, using a subdivision procedure as before. Starting from the d-simplices $\tilde{t}_i^+$ $\tilde{\mathcal{T}}_2$ with at least one face on the boundary of $\mathcal{T}_2$, we apply a small vertical perturbation on each vertex opposite to $\tilde{t}_i^+$, to position it above the supporting hyperplane of $\tilde{t}_i^+$ and bellow $\mathcal{T}_1^+$. We repeat this procedure recursively, moving vertices towards the interior of $\tilde{\mathcal{T}}_2$ (Figure 5.2.d). Finally, we subdivide absent simplices to force them to appear. Observe that the vertices of $\mathcal{T}_1$ inside $\mathcal{T}_2$ become internal to the convex hull of $\mathcal{T}_1^+$ $\mathcal{T}_2^+$, hence they become redundant and do not appear in the merged triangulation.

Although this approach ensures that only the simplices of $\mathcal{T}_1$ affected by the merging are the ones in S, it is also prone to discard more elements of $\mathcal{T}_2$. As opposed to the breadth-first scheme described in Section 4.1 that propagates a single front while computing weights, the squeezing process lifts $\mathcal{T}_2$ by propagating different fronts simultaneously from the elements at the boundary of the merging region. Therefore, no guarantee can be given with respect to the local convexity for simplices at the union of two or more fronts, and more non-locally convex faces can appear.



(a) merging region (b) lift to supporting hyperplanes

(c) lift polyhedron (d) local convexity perturbation

Figure 5.2: Steps taken to perform the convexification-based merging.

5.2 Translation-based Merging

While the *convexification-based* merging does not ensure the local convexity for all simplices, the second method was developed aiming to fix this issue and therefore reduce the number of missing elements in $\mathcal{T}_2$. The *translation-based* merging works by translating vertically the lifted polyhedron of the mesh in such a way that it overlaps the merged region (see Figure 5.3).

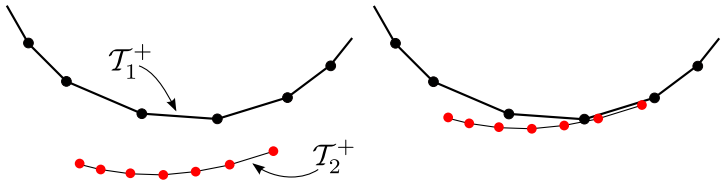


Figure 5.3: The translation-based merging translates $\mathcal{T}_2$ vertically so as to position it below $\mathcal{T}_1$.

The first step of the method is to apply the lifting scheme described in Section 4.1 to $\mathcal{T}_1$ and $\mathcal{T}_2$ independently, but using a vertical displacement for $\mathcal{T}_1$ twice as large as the one used for $\mathcal{T}_2$. More specifically, the value of ϵ in equation (4.1) is halved when lifting $\mathcal{T}_2$, thus making the lifted polyhedron $\mathcal{T}_2^+$ less curved than $\mathcal{T}_1^+$.

The reason why $\mathcal{T}_2^+$ must not be less curved than $\mathcal{T}_1^+$ is that the convex hull of $\mathcal{T}_1^+$ $\mathcal{T}_2^+$ can overlap vertices of $\mathcal{T}_2^+$, therefore removing them from the resulting triangulation. Figure 5.4 illustrates this issue by showing the removed vertices (in green).

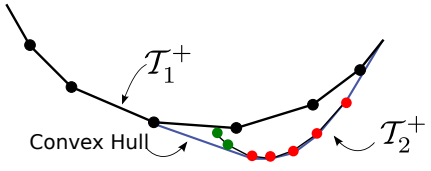


Figure 5.4: Vertices removed (in green) by the convex hull when $\mathcal{T}_2^+$ is more curved than $\mathcal{T}_1^+$.

Once weights have been assigned to vertices in $\mathcal{T}_1$ and $\mathcal{T}_2$, we displace $\mathcal{T}_2^+$ vertically (adding a constant to all weights) until it approaches $\mathcal{T}_1^+$ from below, as illustrated in Figure 5.3. As $\mathcal{T}_2^+$ is less curved than $\mathcal{T}_1^+$, their support hyperplanes tend to be below $\mathcal{T}_1^+$, thus keeping $\mathcal{T}_2$ unchanged after the merging process (including simplices subdivided during the weight assignment step).

The main drawback of the translation-based method is that it affects more the neighborhood of the merged mesh in comparison to the convexification-based method. While in the first approach the lifted polyhedron is changed to not affect the neighbors of the merged area, the second one only translates the mesh aiming to overlap the merged area, therefore the convex hull of $\mathcal{T}_1^+ \quad \mathcal{T}_2^+$ end up removing more vertices of $\mathcal{T}_1^+$. Despite this issue, the vertical translation of $\mathcal{T}_2^+$ behaves well in practice, as we show in Chapter 6.

5.3 Fringe

We observed that when merging meshes with different refinement levels the resulting triangulation often presented badly shaped simplicial elements on the neighborhood of the merged area. Those simplices are automatically inserted by the RT to create a consistent connectivity on the triangulation. In this section we address this issue by introducing the concept of *fringe*, an intermediate triangulation that makes a smoother transition between meshes to be merged together. A fringe is mostly necessary when merging meshes with distinct refinement levels, since they are prone to generate badly shaped simplices.

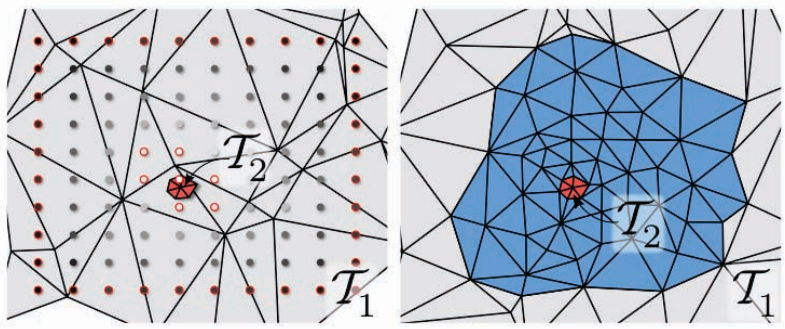


Figure 5.5: Fringing process. At left, we show the two meshes that will be merged, along with the regular grid used to compose the local density distribution. The red circles mark boundary conditions given to the Laplacian solver. On the right, the smaller mesh is merged to a fringe (in blue) that makes a graceful transition between the resolution of the internal and external meshes.

The fringing process works as follows: we initially compute an estimate of local density. We assign an average edge length to each vertice, and normalize these values to make the largest average equal to 1. We then define a regular grid whose resolution is given by the smallest edge present in the scene, and place it around the inner mesh. The density estimated for each vertex serve as boundary conditions on this grid, and we run a Laplacian solver to smooth the transition between densities on the inside and on the border of the grid.

The solution given by the Laplacian solver can be seen as the probability that a vertex on the regular grid should present in a transition mesh. Therefore, we randomly include vertices of the grid based on their density estimates. Finally, we construct the fringe by generating a Delaunay triangulation of the points that were maintained. We also apply

one iteration of Laplacian smoothing to the fringe to make its simplices better shaped. Once the fringe is constructed, we accomplish the merging in two steps: first, we merge the internal mesh with the fringe, and then merge this result with the outer mesh, as shown in Figure 5.5.

As we will show in Chapter 6 the fringe improves the overall quality of simplices generated by the merging methods, therefore solving the problem of badly shaped simplices when merging meshes with different refinement level.

5.4 Summary

This chapter presented two merging methods developed using WDT. The convexification-based merging that changes the original lifted polyhedron, but affects the neighborhood of the merged area to a lesser extent, and the translation-based merging that keeps the original polyhedron unchanged while modifies the neighborhood of the merged area more than the first one. We also addressed the problem of merging meshes with different refinement level by introducing an intermediate triangulation called fringe.

In Chapter 6 we show applications of the merging methods with both 2D and 3D meshes, as well the time required to perform both approaches. The fringe examples will also be presented, and we show the average quality of triangles to demonstrate its effectiveness.

6 RESULTS AND APPLICATIONS

In order to demonstrate the potential use of the methods proposed in this work, we will present 2D and 3D examples of the merging approaches mentioned before, and also show how the fringe solves issues related to badly shaped simplices.

The applications were developed using CGAL s WDT algorithm (BOISSONNAT et al., 2000; TEILLAUD, 1999; FABRI, 2007; CGAL, 2010) to generate triangulations from vertex coordinates and weights computed by our code. Below we show a series of merging examples containing several triangulations of different characteristics.

The value of ϵ (see equation 4.1 for details) defines how high a vertex will be raised in order to create a local convexity of a simplice and one of its neighbors. It is clear that high values of ϵ will generate a lifted polyhedron with high concavity. The concavity of the lifted polyhedron affects the merging method since it has a direct influence over the convex hull of the meshes. For instance, a highly concave mesh is prone to generate more cases of redundant vertices in the neighborhood of the merged area. However, if ϵ is too small it may cause numerical instabilities.

In our experiments, values for ϵ ranging from 10^{-4} to 10^{-8} presented numerical stability and the merging methods yielded fine results. In this chapter, all experiments were conducted using $\epsilon = 10^{-5}$.

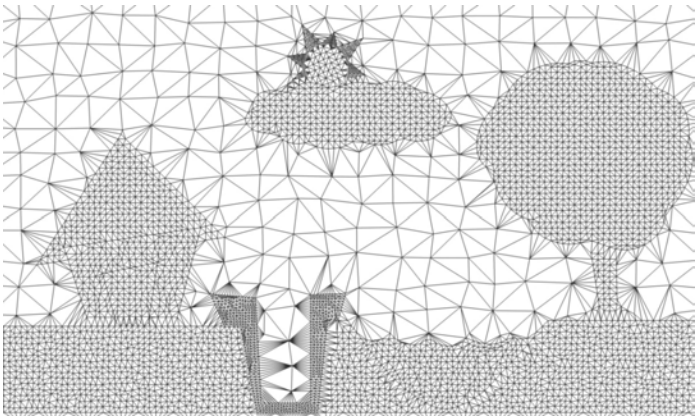
6.1 Merging 2D meshes

We illustrate the flexibility and capability of merging triangulations with our scheme with a physics-based cartoon animation of 2D meshes. The animation consists of a scene with several meshes that move inside a background triangulation. The background mesh performs collision detection and simplifies object placement, since we can compute the adjacency between distinct objects by analyzing whether edges of the merged triangulation connect them. In our implementation the elastic deformation of each object was pre-computed. Figures 6.1(a) and 6.1(b) show the meshes used in the animation and the merging result using the translation-based scheme.

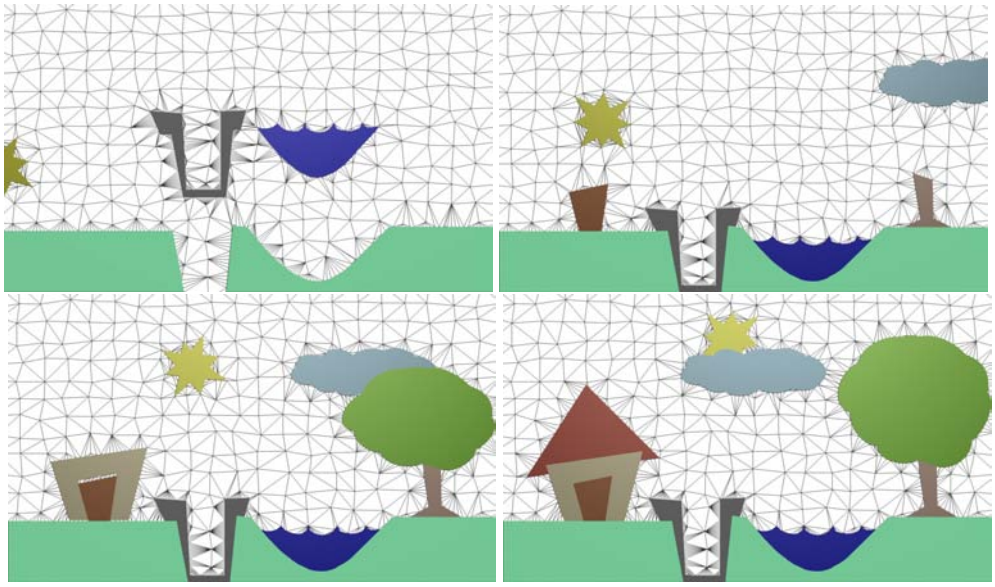
Figure 6.2 shows frames of a simple animation developed to demonstrate the capability of our technique to handle several meshes simultaneously. To perform an example with multiple meshes is firstly necessary to define a preference order, since the method applies the merging for two meshes on each interaction, then the resulting mesh is used to merge with the next, following the preference order. The order with which objects are merged naturally solves the problem of front-to-back alignment. Our merging algorithm priorizes the mesh of the second mesh merged, thus the last object that is merged ends up in front of all the others.

As algorithms to compute the WDT always generate a triangulation for the convex

(a) 2D meshes of the cartoon animation



(b) Merging of all meshes in a given frame



(c) Four snapshots of the animation sequence

Figure 6.1: 2D Cartoon Animation illustrating the merging results with several dynamic meshes.

hull of the weighted points, we have to handle concavities explicitly. Vertices in the background grid that lie inside a concavity of a merged mesh can be forced to appear by adjusting their weights. This adjustment makes the vertices lifted counterparts lie below the convex polyhedron of all merged meshes, above their supporting hyperplanes.

In this example we only handle missing edges that occur at the boundary of objects, while not forcing missing edges in the interior of objects to appear. Therefore, some flips

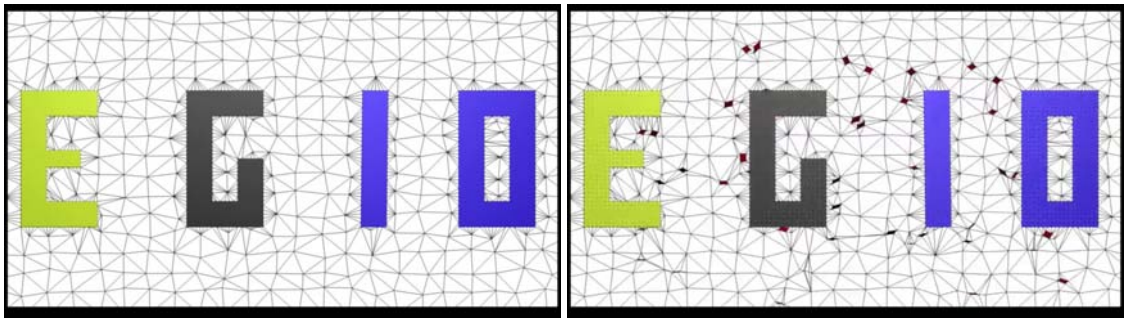
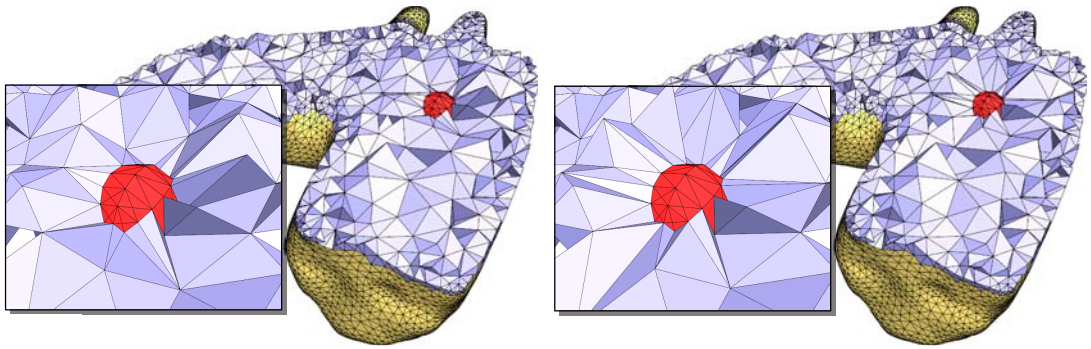


Figure 6.2: Figures showing several meshes being merged.

can be observed in the resultant merge. Nevertheless, this only happened in very few cases.

6.2 Merging 3D meshes



(a) Convexification-based merging (b) Translation-based merging

Figure 6.3: Merging the aorta artery tetrahedral mesh with a tetrahedral decomposition of a sphere. a) Convexification-based merging, 4.63% missing edges; b) Translation-based merging, no absent edges.

This second set of experiments demonstrates the interaction between tetrahedral meshes. Due to occlusion, screenshots of the merging result in 3D hardly convey the strength of our connectivity oblivious technique. Therefore, we present renderings of cutaway sections of the models. In addition, we validate merging results initially without the fringe strategy, since we want to show the affected tetrahedra closer to the merging region.

The first experiment is composed of a moving tetrahedral sphere passing through a mesh of the aorta artery. Figures 6.3 show the sphere merged inside the aorta using both the convexification and translation-based merging schemes. Even though the convexification-based approach impacts a smaller neighborhood around the merging region, more absent simplices tend to be generated with this technique.

In Figure 6.3(b) we observe that some vertices in the neighborhood of the sphere were removed (became redundant) when using the translation-based merging scheme. However, no missing simplices were identified in the translation-based approach, while 4.63% of missing simplices were found and subdivided using the convexification-based merging (all inside the sphere).

Table 6.1 lists the number of vertices in each mesh, as well as the time to perform

Dataset	Vertices	Weight Computation	Triangulation
Aorta	9529	45.05	1259.52
Sphere	58	0.094	12.28
Grid 1	50480	305.15	6633.47
Rocker	10713	53.24	1662.97
Rod	3098	10.24	870.40
Valve	3857	16.38	882.68
Grid 2	8371	30.72	744.14
Buddha	27975	178.17	5230.59

Table 6.1: Mesh data and WDT computation (all times in ms)

Merging Meshes	Convexification-based	Translation-based
Aorta-Sphere	53.24	63.48
Grid 1-Rocker	33062.91	38082.17
Rocker-Arm	528.38	503.80
Valve-Rod	223.23	196.69
Grid 2-Buddha	61526.01	665647.87

Table 6.2: Merging Times (all times in ms)

weight computation and the triangulation. Table 6.2 shows the merging time using both approaches (observe that in the second example we use three merges to compute the final result).

Figure 6.4 illustrates the capability of our technique to handle large and complex tetrahedral meshes with arbitrary topology. The topmost left image in Figure 6.4 shows the tetrahedral meshes (only boundary faces are shown) of three mechanical pieces merged inside a cubic background grid, as illustrated in the topmost right image. Notice from the close-ups that the anisotropy of the gray meshes has been preserved after merging. In fact, no missing simplices (thus no new vertices) have appeared after merging these models using the translation-based scheme.

Another example with complex tetrahedra shows that our approach is able to handle meshes with distinct levels of refinement. Figure 6.5 shows the merge of a Buddha dataset with a background grid, notice that the internal tetrahedra of the buddha have not been refined, therefore the WDT fills the inner region.

6.3 Fringe

The fringe was developed to ensure a smoother transition between the resolution of the two meshes merged together, thus minimizing the occurrence of elongated, low-quality tetrahedra. Besides, in practice it turned out to improve the overall quality of the merged area. Figure 6.6 illustrates the same merge shown in Figure 6.3, only this time using a fringe to smooth the transition between the aorta and the sphere.

In Table 6.3 we show the quality of tetrahedra generated by both convexification-based and translation-based merging method with and without the fringe. The quality measurement indicates an equilateral tetrahedron at value 1, and a hypothetical tetrahedron with an infinity face at value 0. Observe that the convexification-based merging presents tetrahedra with better quality than the translating-based merging, which is expected since the last method affects more neighbors of the merged area.

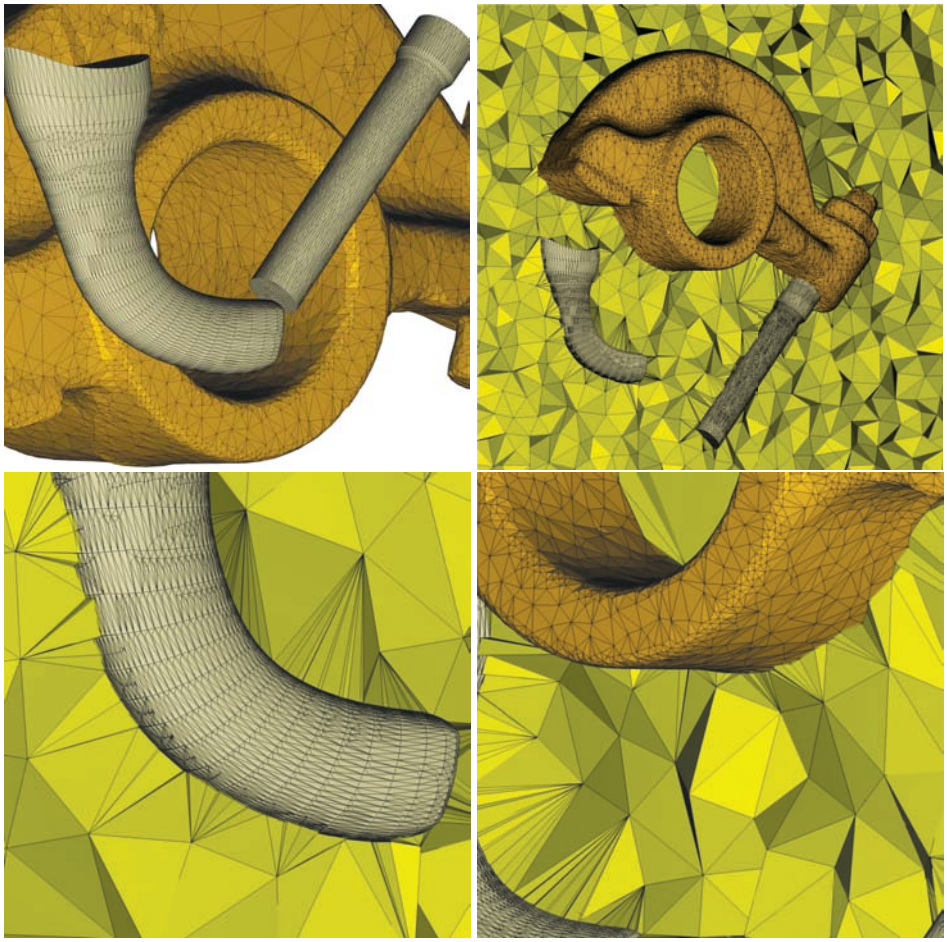


Figure 6.4: Top left: tetrahedral meshes to be merged (boundary surface is shown); Top Right: Mechanical pieces merged in the background grid; Bottom: Zoomed views showing that the original meshes of the pieces have been preserved after merging.

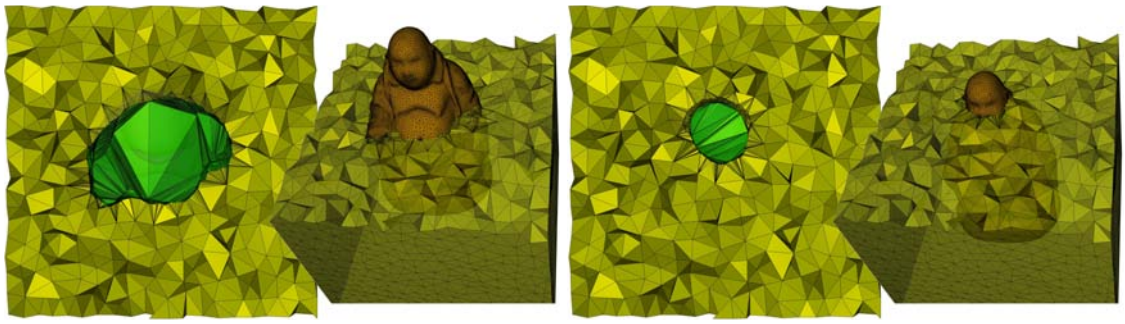


Figure 6.5: Merging a buddha tetrahedral mesh with a background grid.

Merge Method	Without fringe	With fringe
Convexification-based	0.48	0.74
Translation-based	0.44	0.72

Table 6.3: Comparison showing the tetrahedra quality of merging methods with and without fringe.

6.4 Summary

In this chapter we applied the proposed applications on several 2D and 3D examples. The applications revealed the effectiveness of the methods even when handling complex

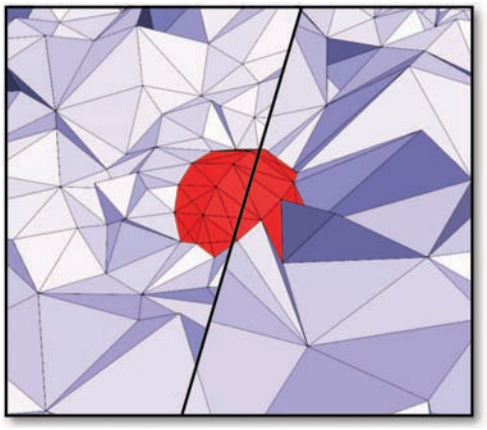


Figure 6.6: Comparison between merging with a fringe (a) and the convexification-based merging (b).

meshes. The weight computation method revealed to be fast, although the time CGAL spends to triangulate large datasets prevents real-time execution for large meshes. As expected, both merging approaches present similar execution times, and for complex meshes they revealed to be slower than CGAL triangulation algorithm.

Both 2D and 3D merging examples show that the designed approaches are suitable to handle meshes with different refinement levels, and that the fringe solves the problem of generating sliver tetrahedra.

7 CONCLUSIONS AND FUTURE WORK

This dissertation proposed a novel way of calculating weights of vertices to build a conformal WDT. While numerous approaches use a linear problem (Cignoni; De Floriani, 1998; BALAVEN et al., 2002) to determine the weights, we developed an algorithmical solution based on a breadth-first traversal of the mesh, and using fundamental mathematical concepts of WDTs. This solution proved to be fast and suitable for several applications, and opens a field to be further explored.

A remarkable aspect of our approach is that, despite its simplicity, it is able to handle and merge complex simplicial meshes without managing connectivity explicitly, as it is usually done with conventional Delaunay methods. However, it is important to point out that there are significant differences between our approach and Delaunay-based schemes. Our technique starts from a set of triangulations, whereas Delaunay-based methods build a triangulation from curves and surfaces constraining the domain of interest. The difference between the two methodologies becomes more evident when dealing with anisotropic meshes.

Anisotropic versions of Delaunay triangulations and constrained Delaunay triangulations have been used to handle anisotropic meshes (BOROUCHAKI et al., 1997). However, such techniques demand either tensor estimation to define the anisotropy or an explicit manipulation of constraints. This can become as complex as the connectivity management during merging operations. In fact, our approach can be seen as a first step toward a new framework for connectivity oblivious mesh representation using the idea of regular triangulations.

In this work, we also proposed different methods to merge simplicial complexes without the need to explicitly manage connectivity information. The merging schemes presented gave very satisfactory results, especially regarding algorithm stability and the locality of mesh updates. While the translation-based merging works by translating vertically the lifted polyhedron of the mesh in such a way that it overlaps the merged region, the convexification-based merging squeezes up the lifted polyhedron of the mesh between the lifted polyhedron of the merged area and the supporting hyperplanes of the simplices on the border.

In some applications, the translation-based merging scheme may affect a wide neighborhood around the merging region, making many vertices redundant. Although one can always reinstate missing vertices by vertically displacing then beneath the convex hull of lifted points, checking for redundancy can become costly. Plane projection plus convexification can indeed ensure local mesh updates for the background grid. However, the number of internal missing simplices exceeded 10% in one of our experiments (our worst case), which can be unacceptable in applications where internal meshes must be preserved.

The large number of absent elements, in many cases, is due to numerical instabilities, as the cavity between the convex hull and the support planes can become too narrow. This makes it numerically difficult to convexify the lifted polyhedron with vertical perturbations. If an excessively vertical perturbation is applied to a vertice, numerical predicates tend to consider adjacent simplices as coplanar, violating what is known as *the general position hypothesis*. This makes it difficult to predict which vertices will be seen as coplanar.

The convexification-based could be improved with some effort on creating a new way to squeeze the lifted polyhedra to reduce this issue could reduce the occurrence of this issue. While the translation-based merging ensures the maintenance of the simplices at the cost of having a heavier effect on the the neighborhood of the merged area. A study conducted to define a better convexity of the lifted polyhedron could improve the results, since when the polyhedron is more concave than the merged area, some of its simplices become redundant, and when the polyhedron is more convex it impacts on the neighborhood.

We solved the issue with badly shaped simplices inserted when merging meshes with different refinement levels by introducing the fringe. The fringe worked by improving the quality of tetrahedra created by both merging approaches. Nevertheless, in practice to use the fringe requires that the merged area has enough space so that vertices of the fringe can smooth the transitions, otherwise it will not present satisfactory results and can even give rise to badly shaped tetrahedra. Therefore, the size a fringe will depend on both refinement level and size of the meshes being merged.

The examples developed in this work validated the proposed techniques. They clearly show the merging process working well for several kinds of meshes. Additionally, they demonstrated that the fringe, in turn, solves problems of meshes with different refinement levels and improves the quality of simplices. Finally, they made possible to observe that the computing weights method is both fast and usable with other applications.

Since WDTs generates triangulations with their convex hull, the merging methods are better suitable to convex meshes. While for convex meshes the result presents triangulations that match the originals, for concave datasets it is necessary a post processing step to remove simplices created due to the convex hull. In some of our examples we had to deal with this issue, and depending on the mesh surface it can be a painful task.

Regarding execution time, the weight computation algorithm turned out to be efficient even with large datasets. It is necessary to use CGAL s WDT method, which prevents real-time executions. The merging methods proved to be faster than the triangulation for small datasets, however, for large meshes they become slower. The method used to find the merged area explains this behavior, since after the area is found our approach consists of a linear procedure.

In the future, we would like to further explore our weight assignment technique, to better enforce local convexity of the lifted polyhedron. Also, an interesting avenue for future research is to combine our framework with numerical simulation applications. Our merging schemes generate high quality simplices, and we believe it should be possible to couple it with existing numerical models.

An important study could be conducted to analyze the influence of ϵ on the method to assign weights and on the merging techniques. Since there is a trade-off between numerical stability and better results, this study could point an optimal value of ϵ that avoids missing simplices while keeps numerical stability.

REFERENCES

AHN, M.; LEE, S. Mesh Metamorphosis with Topology Transformations. **pg**, Los Alamitos, CA, USA, v.00, p.481, 2002.

AHN, M.; LEE, S.; SEIDEL, H.-P. Connectivity transformation for mesh metamorphosis. In: SGP 04: PROCEEDINGS OF THE 2004 EUROGRAPHICS/ACM SIGGRAPH SYMPOSIUM ON GEOMETRY PROCESSING, 2004, New York, NY, USA. **...** ACM Press, 2004. p.75–82.

AMENTA, N.; BERN, M. Surface reconstruction by Voronoi filtering. In: SCG 98: PROCEEDINGS OF THE FOURTEENTH ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 1998, New York, NY, USA. **...** ACM, 1998. p.39–48.

AMENTA, N.; BERN, M.; KAMVYSSELIS, M. A new Voronoi-based surface reconstruction algorithm. In: SIGGRAPH 98: PROCEEDINGS OF THE 25TH ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 1998, New York, NY, USA. **...** ACM, 1998. p.415–421.

AMENTA, N.; CHOI, S.; KOLLURI, R. The Power Crust. **Computatinal Geometry**, [S.l.], v.19, p.127–153, 2001.

AURENHAMMER, F. Power Diagrams: properties, algorithms and applications. **SIAM J. Comput.**, [S.l.], v.16, n.1, p.78–96, 1987.

AURENHAMMER, F. Voronoi diagrams—a survey of a fundamental geometric data structure. **ACM Comput. Surv.**, New York, NY, USA, v.23, n.3, p.345–405, 1991.

AURENHAMMER, F.; IMAI, H. Geometric relations among Voronoi diagrams. In: ANNUAL SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCES ON STACS 87, 4., 1987, London, UK. **...** Springer-Verlag, 1987. p.53–65.

BALAVEN, S. et al. **Conforming Orthogonal Meshes**. [S.l.]: INRIA - Sophia Antipolis, 2002. (4404).

BERG, M. de et al. **Computational Geometry**: algorithms and applications. 2 ed. [S.l.]: Springer-Verlag, 2000. 367p.

BOISSONNAT, J.-D. et al. Triangulations in CGAL (extended abstract). In: SCG 00: PROCEEDINGS OF THE SIXTEENTH ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 2000, New York, NY, USA. **...** ACM, 2000. p.11–18.

BOISSONNAT, J.-D.; OUDOT, S. Provably good sampling and meshing of surfaces. **Graphical Models**, [S.l.], v.67, p.405–451, 2005.

BOROUCHAKI, H. et al. Delaunay mesh generation governed by metric specifications. Part I. Algorithms. **Finite Elements in Analysis and Design**, [S.l.], v.25, p.61–83, 1997.

BREEN, D. et al. 3D Metamorphosis Between Different Types of Geometric Models. In: CHALMERS, A.; RHYNE, T.-M. (Ed.). **EG 2001 Proceedings**. [S.l.]: Blackwell Publishing, 2001. v.20(3), p.36–48.

CASTRO, P. M. M. de et al. Filtering Relocations on a Delaunay Triangulation. **Computer Graphics Forum**, [S.l.], v.28, n.5, p.1465–1474, 2009.

CGAL. **Computational Geometry Algorithms Library**. [Online; accessed 21-January-2007].

CHENG, S.-W.; DEY, T. K.; RAMOS, E. A. Delaunay refinement for piecewise smooth complexes. In: SODA 07: PROCEEDINGS OF THE EIGHTEENTH ANNUAL ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS, 2007, Philadelphia, PA, USA. **...** Society for Industrial and Applied Mathematics, 2007. p.1096–1105.

CHENG, S.-W.; DEY, T.; RAY, T. Weighted Delaunay Refinement for Polyhedra with Small Angles. In: INTERNATIONAL MESHING ROUNDTABLE, 14., 2005. **...** Springer-Verlag:, 2005. p.11–14.

CHENG, S.-W. et al. Sliver exudation. In: SCG 99: PROCEEDINGS OF THE FIFTEENTH ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 1999, New York, NY, USA. **...** ACM Press, 1999. p.1–13.

CHENG, S.-W. et al. Sampling and meshing a surface with guaranteed topology and geometry. In: SCG 04: PROCEEDINGS OF THE TWENTIETH ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 2004, New York, NY, USA. **...** ACM, 2004. p.280–289.

CHEW, L. P. Constrained Delaunay triangulations. In: SCG 87: PROCEEDINGS OF THE THIRD ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 1987, New York, NY, USA. **...** ACM, 1987. p.215–222.

CHRYSANTHOU, Y.; SLATER, M. Computing dynamic changes to BSP trees. **Computer Graphics Forum**, [S.l.], v.11, n.3, p.C321–C332, September 1992.

Cignoni, P.; De Floriani, L. Power Diagram Depth Sorting. In: CANADIAN CONFERENCE ON COMPUTATIONAL GEOMETRY, 10., 1998, Montréal, Québec, Canada. **Proceedings...** School of Computer Science: McGill University, 1998. p.88–89.

COHEN, J. et al. Simplification envelopes. In: SIGGRAPH 96: PROCEEDINGS OF THE 23RD ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 1996, New York, NY, USA. **...** ACM, 1996. p.119–128.

CUADROS-VARGAS, A. J. et al. Imesh: an image based quality mesh generation technique. **Computer Graphics and Image Processing, Brazilian Symposium on**, Los Alamitos, CA, USA, v.0, p.341–348, 2005.

DANCIGER, J.; DEVADOSS, S. L.; SHEEHY, D. Compatible triangulations and point partitions by series-triangular graphs. **Comput. Geom. Theory Appl.**, Amsterdam, The Netherlands, The Netherlands, v.34, n.3, p.195–202, 2006.

DELAUNAY, B. N. **Sur la sphère vide**. 1934.

DELOERA, J.; STURMFELS, J.; THOMAS, R. Gröbner Bases and Triangulations of the Second Hypersimplex. **Combinatorica**, [S.l.], v.15, n.3, p.409–424, 1995.

DEVILLERS, O. et al. Checking the convexity of polytopes and the planarity of subdivisions. **Computational Geometry**, [S.l.], v.11, p.187–208, 1998.

EDELSBRUNNER, H. An acyclicity theorem for cell complexes in d dimensions. In: SCG 89: PROCEEDINGS OF THE FIFTH ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 1989, New York, NY, USA. **...** ACM, 1989. p.145–151.

EDELSBRUNNER, H. Deformable Smooth Surface Design. **Discrete & Computational Geometry**, [S.l.], v.21, n.1, p.87–115, 1999.

EDELSBRUNNER, H. **Geometry and Topology for Mesh Generation**. [S.l.]: Cambridge University Press, 2001.

EDELSBRUNNER, H.; GUOY, D. **An Experimental Study of Sliver Exudation**. 2001.

EDELSBRUNNER, H.; SEIDEL, R. Voronoi diagrams and arrangements. **Discrete Comput. Geom.**, [S.l.], v.1, p.25–44, 1986.

EDELSBRUNNER, H.; SHAH, S. Incremental Topological Flipping Works for Regular Triangulations. **Algorithmica**, [S.l.], v.15, p.223–241, 1996.

FABRI, A. Voronoi Diagrams in CGAL, the Computational Geometry Algorithms Library. In: ISVD 07: PROCEEDINGS OF THE 4TH INTERNATIONAL SYMPOSIUM ON VORONOI DIAGRAMS IN SCIENCE AND ENGINEERING, 2007, Washington, DC, USA. **...** IEEE Computer Society, 2007. p.8–14.

FLOATER, M.; C.GOTSMAN. How to morph tilings injectively. **J. Comput. Appl. Math.**, Amsterdam, The Netherlands, The Netherlands, v.101, n.1-2, p.117–129, 1999.

GROSS, M.; PFISTER, H. (Ed.). **Point-Based Graphics**. [S.l.]: Morgan Kaufmann, 2007.

GUIBAS, L. Kinetic data structures. In: HANDBOOK OF DATA STRUCTURES AND APPLICATIONS, 2004. **...** CRC Press, 2004.

LABELLE, F. Sliver removal by lattice refinement. In: SCG 06: PROCEEDINGS OF THE TWENTY-SECOND ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 2006, New York, NY, USA. **...** ACM, 2006. p.347–356.

LEE, A. W. F. et al. Multiresolution mesh morphing. In: SIGGRAPH 99: PROCEEDINGS OF THE 26TH ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 1999, New York, NY, USA. **...** ACM Press/Addison-Wesley Publishing Co., 1999. p.343–350.

LUEBKE, D. et al. **Level of Detail for 3D Graphics**. New York, NY, USA: Elsevier Science Inc., 2002.

LUQUE, R.; COMBA, J. L. D.; FREITAS, C. M. D. Broad-Phase Colision Detection Using Semi-Adjusting BSP-Trees. In: ACM SIGGRAPH INTERACTIVE 3D GRAPHICS AND GAMES - I3D 2005, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.179–186. (1595930132).

MASADA, T.; IMAI, H.; IMAI, K. Enumeration of regular triangulations. In: SCG 96: PROCEEDINGS OF THE TWELFTH ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 1996, New York, NY, USA. **...** ACM Press, 1996. p.224–233.

MEHLHORN, K. et al. Checking geometric programs or verification of geometric structures. In: ANNU. ACM SYMPOS. COMPUT. GEOM., 12., 1996. **Proceedings...** [S.l.: s.n.], 1996. p.159–165.

NAYLOR, B. F. Interactive solid geometry via partitioning trees. In: GRAPHICS INTERFACE 92, 1992. **Proceedings...** [S.l.: s.n.], 1992. p.11–18.

NONATO, L. G. et al. Beta-Connection: generating a family of models from planar cross sections. **ACM Transactions on Graphics**, [S.l.], v.24, n.4, p.1239–1258, 2005.

PARUS, J.; KOLINGEROVá, I. Morphing of meshes with attributes. In: SCCG 04: PROCEEDINGS OF THE 20TH SPRING CONFERENCE ON COMPUTER GRAPHICS, 2004, New York, NY, USA. **...** ACM Press, 2004. p.73–81.

PIRES, F. B. **Triangulações regulares e aplicações**. 2008. Dissertação (Mestrado em Ciência da Computação) — Instituto de Ciências Matemáticas e de Computação - USP.

RUPPERT, J. A new and simple algorithm for quality 2-dimensional mesh generation. In: SODA 93: PROCEEDINGS OF THE FOURTH ANNUAL ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS, 1993, Philadelphia, PA, USA. **...** Society for Industrial and Applied Mathematics, 1993. p.83–92.

SCHREINER, J. et al. Inter-surface mapping. In: SIGGRAPH 04: ACM SIGGRAPH 2004 PAPERS, 2004, New York, NY, USA. **...** ACM Press, 2004. p.870–877.

SHEWCHUK, J. Delaunay Refinement Algorithms for Triangular Mesh Generation. **Computational Geometry**: theory and applications, [S.l.], v.22, n.2-3, p.21–74, 2002.

SURAZHSKY, V.; GOTSMAN, C. Morphing Stick Figures Using Optimized Compatible Triangulations. In: PG 01: PROCEEDINGS OF THE 9TH PACIFIC CONFERENCE ON COMPUTER GRAPHICS AND APPLICATIONS, 2001, Washington, DC, USA. **...** IEEE Computer Society, 2001. p.40.

SURFACE SIMPLIFICATION USING QUADRIC ERROR METRICS, 1997, New York, NY, USA. **...** ACM Press/Addison-Wesley Publishing Co., 1997. p.209–216.

TEILLAUD, M. **Three Dimensional Triangulations in CGAL**. 1999.

TORRES, E. Optimization of the Binary Space Partition Algorithm (BSP) for the Visualization of Dynamic Scenes. In: EUROGRAPHICS 90, 1990. **...** North-Holland, 1990. p.507–518.

TOURNOIS, J. et al. Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation. **ACM Trans. Graph.**, New York, NY, USA, v.28, n.3, p.1–9, 2009.

TOURNOIS, J.; SRINIVASAN, R.; ALLIEZ, P. **Perturbing Slivers in 3D Delaunay Meshes**. 2009.

VIGO, M.; PLA, N.; COTRINA, J. Regular triangulations of dynamic sets of points. **Comput. Aided Geom. Des.**, Amsterdam, The Netherlands, The Netherlands, v.19, n.2, p.127–149, 2002.

ZORIN, D.; SCHRöDER, P.; SWELDENS, W. Interpolating Subdivision for meshes with arbitrary topology. In: SIGGRAPH 96: PROCEEDINGS OF THE 23RD ANNUAL CONFERENCE ON COMPUTER GRAPHICS AND INTERACTIVE TECHNIQUES, 1996, New York, NY, USA. ... ACM Press, 1996. p.189–192.