

# Shape Deformation for Objects of Greatly Dissimilar Shapes with Smooth Manifold

Yan Ke

School of Computing

National University of Singapore

A thesis submitted for the degree of

*Doctor of Philosophy*

February 2013

I would like to dedicate this thesis to my loving parents, my wife and  
my little daughter...

## Acknowledgements

I would like to thank Dr. Cheng Ho-lun, Alan, my supervisor, for his many suggestions and constant support over the past few years. He has been a great friend and patient supervisor to me, and I am very happy to work with him.

I would like to thank Shi Xinwei, Tony Tan, Chen Chao for their prior work in this research.

Many thanks to the  $G^3$  Lab mates, Ashwin, Guo Jiayan, Li Ruoru, Qi Meng and many others for the great time we spent together in this lab. I would like to thank Alvin Chia for his suggestions during my first year, and Prof. Tan Tiow Seng for providing us with a pleasant working environment and first class resources.

NUS, Graphics Laboratory

Yan Ke

July , 2012

## Abstract

Deformation animations between different computer-generated characters or objects have gained widespread attention in the recent years. In movie and gaming industries, deformation animations between different objects create breath-taking effects. In cartoon shows, computer-generated anthropomorphized characters are animated to tell a story.

Although many deformation techniques have been proposed in the recent years, fully automated computerized deformation animation generation is still seldom used in the movie industry. The reason for employing labor-intensive methods rather than utilizing a computer software is that there are two main limitations in deformation techniques that are currently available. First, most available deformation techniques rely on the close similarities between source and target shapes. Source and target objects of greatly dissimilar shapes create ambiguities in vertex correspondence mapping. Second, there are difficulties in handling topology changes automatically.

In the current work, a simple and efficient algorithm for deformation between objects of greatly dissimilar shapes, which does not require any form of similarity or vertex correspondence mapping, is presented.

This deformation algorithm is called general skin deformation algorithm, because all intermediate shapes are represented by a maximum curvature continuous surface type called skin surface. All intermediate skin surfaces share the same Voronoi complex, which is called the intermediate Voronoi complex. The Minkowski sum of the intermediate Voronoi complex and its dual Delaunay complex forms mixed cells which cut skin surfaces into patches. These skin patches are free to deform in their own mixed cells according to regular sphere or hyperboloid functions.

This solution has several advantages. First, no prior information, such as the similarity, is required. Second, topology changes are handled automatically. Third, prior work has been done on approximating real objects to skin meshes with homeomorphism, and the skin meshes generated are guaranteed to be in good quality. Fourth, each intermediate skin mesh is constructed more efficiently than existing programs, such as the online computational geometry library CGAL.

# Contents

<b>Contents</b>	<b>v</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>xii</b>
<b>Nomenclature</b>	<b>xii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Criteria for Good Surface Approximation in Deformation Algorithms	3
1.2 General Skin Deformation . . . . .	4
1.3 Related Work . . . . .	8
1.4 Framework . . . . .	10
1.5 New Issues . . . . .	11
1.6 Contribution . . . . .	12
1.7 Outline . . . . .	12
<b>2 Preliminaries</b>	<b>14</b>
2.1 The Skin Surface Representations . . . . .	14
2.1.1 Weighted Points . . . . .	15

2.1.2	Skin . . . . .	15
2.1.3	The Delaunay, Voronoi Complexes, General Position Assumption and Mixed Cells. . . . .	17
2.1.4	Skin Decomposition. . . . .	18
2.1.5	Quality Skin Surface Triangular Mesh . . . . .	21
2.2	The Overview of General Skin Deformation . . . . .	22
2.2.1	Growth Model . . . . .	23
2.2.2	General Skin Deformation . . . . .	25
<b>3</b>	<b>Intermediate Complexes in Skin Deformation</b>	<b>29</b>
3.1	Intermediate Voronoi Complexes . . . . .	30
3.2	Intermediate Delaunay Complexes . . . . .	33
3.2.1	Degeneracies in Intermediate Delaunay Triangulation . . . . .	33
3.2.2	Comparison of Delaunay Triangulation Updates and Dynamic Delaunay Triangulation . . . . .	35
3.3	Intermediate Mixed Cell . . . . .	37
3.3.1	Degeneracies in Intermediate Mixed Cells . . . . .	38
3.3.2	Mixed Cell Connections . . . . .	39
3.3.3	Allocating a Mesh Point in its Mixed Cell . . . . .	39
<b>4</b>	<b>General Skin Surface Mesh Deformation</b>	<b>42</b>
4.1	Surface Points Moving Trajectories . . . . .	43
4.1.1	Trajectory I: Scaling . . . . .	44
4.1.2	Trajectory II: Snapping . . . . .	46
4.1.3	Trajectory III: Sticking . . . . .	48
4.2	Topology Change Handling . . . . .	50

4.2.1	Hot Sphere Size . . . . .	54
4.2.2	Special Sampling in Hot Sphere . . . . .	55
4.3	Surface Point Scheduling for Changing Status . . . . .	56
4.3.1	Escaping Time Scheduling . . . . .	57
4.3.1.1	Trajectory I Escaping Time Calculation . . . . .	58
4.3.1.2	Trajectory II Escaping Time Calculation . . . . .	59
4.3.1.3	Trajectory III Escaping Time Calculation . . . . .	59
4.3.1.4	Special Situation: Escaping Degenerate Mixed Cells . . . . .	59
4.3.2	Metamorphosis Scheduling . . . . .	60
4.3.3	Make or Delete Sphere Scheduling . . . . .	60
4.4	Mesh Refinement Maintaining Triangle Quality . . . . .	61
4.4.1	Scheduling Edge Update . . . . .	63
4.4.2	Scheduling Triangle Update . . . . .	63
4.5	Combine Point Scheduling with Triangle Scheduling . . . . .	65
4.6	Summary . . . . .	66
<b>5</b>	<b>Simplified General Skin Deformation</b>	<b>68</b>
5.1	Simplified General Skin Deformation Algorithm . . . . .	71
5.2	Simplification of Weighted Point Set . . . . .	71
5.2.1	Volume and Volume Difference of Union of Balls . . . . .	74
5.2.2	Removal . . . . .	75
5.2.3	Recovering Volume . . . . .	76
5.3	Degeneracies in Intermediate Delaunay and Voronoi Complexes . . . . .	79
5.4	Escaping Time in New Degenerate Mixed Cells . . . . .	82
5.5	Complexity Analysis . . . . .	83

5.6	Conclusion . . . . .	84
<b>6</b>	<b>Special Deformation: Partial Movements</b>	<b>85</b>
6.1	More Degenerate Types of Intermediate Complexes . . . . .	86
6.2	Conclusion . . . . .	88
<b>7</b>	<b>Software Development and Experiment Results</b>	<b>90</b>
7.1	Software for General Skin Deformation . . . . .	90
7.2	Software for Simplified General Skin Deformation . . . . .	92
<b>8</b>	<b>Conclusions</b>	<b>97</b>
	<b>References</b>	<b>99</b>

# List of Figures

1.1	General skin deformation. . . . .	5
1.2	Deformation from a bunny to a torus. . . . .	5
1.3	Super-imposition of Voronoi complexes. . . . .	6
1.4	Local modification of skin surface. . . . .	7
1.5	Comparison of triangle quality. . . . .	7
2.1	Skin surface in $\mathbb{R}^2$ . . . . .	16
2.2	Skin surface in $\mathbb{R}^3$ . . . . .	17
2.3	Basic mixed cells. . . . .	19
2.4	Skin decomposition in $\mathbb{R}^2$ . . . . .	19
2.5	Skin decomposition in $\mathbb{R}^3$ . . . . .	20
2.6	Quality mesh. . . . .	22
2.7	The growth model. . . . .	23
2.8	Surface point trajectories. . . . .	24
3.1	Super-imposition of Voronoi complexes. . . . .	31
3.2	Triangle-prim Delaunay cell. . . . .	34
3.3	Degenerate Delaunay cells in regular Delaunay triangulation. . . .	34
3.4	Degenerate intermediate Delaunay triangles. . . . .	35

## LIST OF FIGURES

---

3.5	Weighted points movement in a higher dimension. . . . .	36
3.6	Connectivity chart. . . . .	40
3.7	Connectivity demonstration graph. . . . .	40
3.8	Mixed cell allocation. . . . .	41
4.1	Trajectory I. . . . .	45
4.2	Trajectory II. . . . .	47
4.3	Trajectory III. . . . .	48
4.4	Sticky point example. . . . .	49
4.5	Sticky tests I. . . . .	52
4.6	Sticky tests II. . . . .	53
4.7	Topology changes. . . . .	55
4.8	Status transfer. . . . .	58
4.9	Triangle refinement. . . . .	62
4.10	Edge contraction. . . . .	62
4.11	Triangle elements labeling for scheduling. . . . .	64
5.1	The new SGSD algorithm. . . . .	69
5.2	The simplification process. . . . .	70
5.3	Intermediate Voronoi diagram of simplification process. . . . .	77
5.4	Intermediate Voronoi diagram of simplification process (shifted). . . . .	77
5.5	Symmetric volume difference. . . . .	78
5.6	Simplified union of balls. . . . .	79
5.7	Intermediate Delaunay complex for the simplification process. . . . .	80
5.8	Intermediate Voronoi complex for the simplification process in $\mathbb{R}^3$ . . . . .	80
5.9	Intermediate Delaunay complex for the simplification process in $\mathbb{R}^3$ . . . . .	81

## LIST OF FIGURES

---

5.10	Triangle frustum Delaunay cell. . . . .	82
5.11	Degenerate mixed cell examples. . . . .	82
6.1	Degenerate super-imposed Voronoi cells. . . . .	87
6.2	An example of partial molecular movement. . . . .	88
6.3	Intermediate Delaunay complex for partial movement. . . . .	89
7.1	A zoomed-in snapshot of the general deformation. . . . .	92
7.2	Vertex correspondence. . . . .	93
7.3	Different simplification level break down for deformation between bunny and cow. . . . .	96
7.4	Different simplification level break down for deformation between mannequin and fist. . . . .	96

# List of Tables

2.1	Voronoi cells, Delaunay Cells, Mixed Cells and Skin Patches . . .	21
3.1	Voronoi Cells, Delaunay Cells, Mixed Cells and Skin Patches. . . .	32
5.1	Degenerate Voronoi and Delaunay cells. . . . .	81
6.1	Additional types of intermediate Voronoi, Delaunay and mixed cells.	86
7.1	Average time taken by different algorithms for different examples.	92
7.2	Input skin meshes. . . . .	94
7.3	Number of intermediate mixed cells for different deformation models.	94
7.4	Average number of schedules. . . . .	95
7.5	Average time taken in each frame. . . . .	95

# Chapter 1

## Introduction

Computer-generated shape deformation animation, where different shapes morph from one to another, has the potential to enrich human perception, education and entertainment in many scientific and industrial fields. Biological scientists test their hypothesis on the mechanisms of macromolecules in computer-simulated experiments and capture desired kinetics events, such as major conformational changes of macromolecules, binding properties, and response to mechanical forces [38; 54]. Realistic physical simulations for engineering purposes employ deforming surfaces as boundaries of spatial domains that vary with time. An example of this is the modeling of deforming boundaries in simulations, such as the isosurface of a certain threshold temperature in a dynamic heat system, whereby the model describes the surface of solidifying liquid formed by the isosurface at the ‘freezing point’ [51; 53]. Computer graphics professionals generate realistic models with meshes and human-like textures to mimic the role of human actors. As there is an increasing demand for time-evolving shape deformation animations, it is important to provide efficient and robust computational tools to handle these

---

forms of animated geometry.

The first difficulty of modeling shape deformation is in the handling of shape morphing between greatly dissimilar shapes automatically in real time. Most modern deformation techniques require similarities between the source and target shapes for the identification of the *feature correspondence* during deformation, i.e. computing the association of vertices and triangles in the same feature of both source and target shapes [1; 6; 10; 24; 27; 45; 55; 60; 71; 75]. For example, in the deformation process from a horse to a camel, feature correspondence of vertices and triangles is easily identified as the four legs of the horse morph to form the four legs of the camel. It is difficult for a computer program to figure out feature correspondence information between objects of greatly dissimilar shapes, for example, the deformation from a bunny to a torus (Figure 1.2). These examples are not unreasonable since many breath-taking movies and cartoons require deformation animations between objects of greatly dissimilar shapes (e.g. the robot deforms from a pool of liquid in the movie, Terminator 2). However, automatic correspondence mapping methods are inadequate in these cases, and usually require labor-intensive methods to handle the ambiguities in correspondence mapping. Therefore, there is great demand to build a fully automated deformation system for objects of greatly dissimilar shapes.

The second difficulty of modeling shape deformation is in selecting a suitable surface representation for both topology and local changes. On the one hand, it is difficult to handle topology change automatically in computer graphics; controlling the splitting, merger and creation of holes and tunnels is not trivial with explicit(parametric) surfaces [42; 70]. On the other hand, it is difficult to carry out local manipulations of implicit surfaces. For instance, a local change in a

---

small part of the surface may cause the whole surface to change unexpectedly [8; 65]. Furthermore, implicit surfaces are not easily understood, used and manipulated, especially by artists without formal training in Mathematics subjects such as Algebra. Thus, the search for a good representation of deforming objects between implicit and explicit surfaces is still on.

## 1.1 Criteria for Good Surface Approximation in Deformation Algorithms

There are a few criteria for selecting a suitable type of surface for performing shape deformation.

1. The surface must be capable of approximating any given object with promised Hausdorff distance.
2. The surface is adequate for the modeling of changes in shape, curvature and topology, of which topology is the most challenging aspect of modeling surface deformation.
3. The triangulation of the surface requires good triangle quality as an aid in the numerical analysis of physical simulations and homeomorphism to the surface during deformation. In order to maintain a triangulation with good quality, each angle of the triangle is bound with minimum and maximum degrees.
4. It is preferable for the surface to be able to deform the current mesh from the previous one instead of creating the current mesh from scratch. This

---

last requirement aids the *surface correspondence* between two meshes in two consecutive points of time, so that information that is attached to the surface, such as texture mappings or electric potential, deforms continuously over time.

The *skin surface* by Edelsbrunner [29], which satisfies all the requirements, is employed and presented in this thesis. The method presented in our previous work provides a way to approximate objects by skin surfaces [20] and this enables a fully automated process for the deformation of a given object into another without the need for manual assistance. Therefore, artists do not need to go through the tedious task of manipulating the deforming surface in order to accommodate topology changes.

## 1.2 General Skin Deformation

The *general skin deformation* (GSD) algorithm [21; 29] is a suitable surface modeling paradigm for the deformation of arbitrary shapes, including additional advantages, such as better mesh quality and automatic handling of topology changes with a certain general position assumption (examples are shown in Figure 1.1 and Figure 1.2). In the GSD algorithm, the real time visualization of deforming one shape into another can be obtained by super-imposing the Voronoi complexes of the two skin surfaces to produce the *intermediate Voronoi complex*. The intermediate Voronoi complex, which is different from regular Voronoi complexes, consists of degenerate Voronoi cells which are unavoidable even with the traditional general position assumption (Figure 1.3). Therefore, a new general position assumption for GSD algorithm is defined (Section 3.1) as *GGP Assumption*. An

---

example of the GGP Assumption is as follows: in  $\mathbb{R}^2$ , two Voronoi edges from the two Voronoi complexes are only allowed to intersect each other at their interior but not their endpoints in the intermediate Voronoi complex (Figure 1.3).



Figure 1.1: General skin deformation of a mannequin skin model from a question mark skin model.

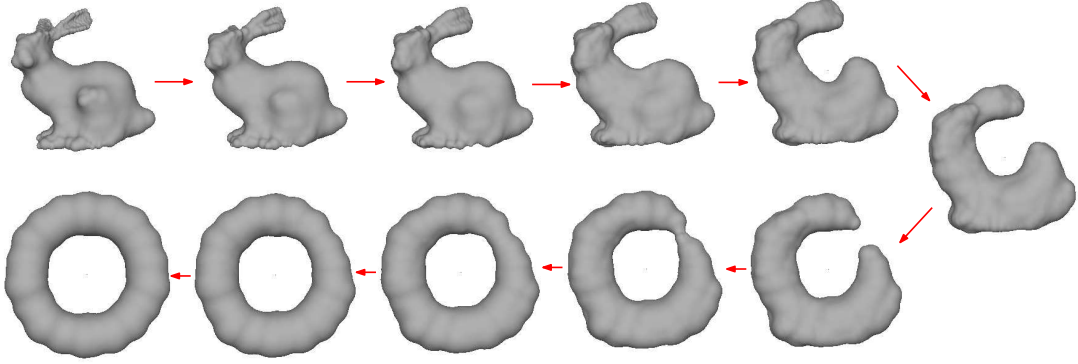


Figure 1.2: Deformation from a bunny skin surface mesh to a torus skin surface mesh.

The GSD algorithm has the following six advantages:

1. **Direct manipulation of deformation.** The Minkowski sum of the intermediate Voronoi complex and its dual Delaunay complex forms *mixed cells* which decompose skin surfaces into patches. Every skin patch is modeled by a sphere or hyperboloid function, which deforms freely in its mixed cell. No prior information about the similarities in shape between the objects is required. Therefore, the skin surface is a suitable representation for performing deformation between objects of greatly dissimilar shapes.

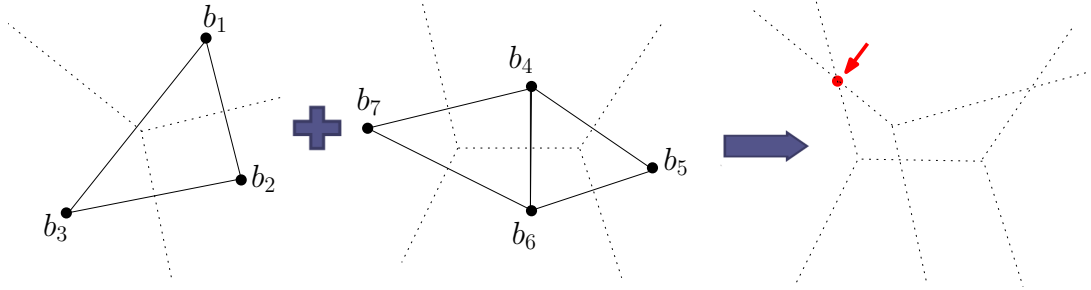


Figure 1.3: Super-imposition of two Voronoi complexes constructed by two weighted point set under GGP Assumption. The resultant intermediate Voronoi complex (right-most) has degenerate Voronoi cells such as the marked Voronoi vertex which is surrounded by four Voronoi regions.

2. **Handling topology change automatically.** When a surface is deforming, its topology changes when its components are split or merged, or when the surface creates or destroys voids and tunnels. Controlling these changes is not trivial for explicit surfaces [42; 66]. The GSD algorithm addresses this issue, and pre-schedules all topology change in the process of deformation. For example, in Figure 1.2, the creation of a tunnel and the destruction of a void are automatically handled in the process of deformation.
3. **Intuitive object representation and editing.** Explicit surfaces are not suitable for topology changes [46], whereas, implicit surfaces are not suitable for local manipulations [37; 50; 72]. In contrast, general skin surface deformation is able to handle both local changes (Figure 1.4) and topology changes (Figure 1.2) independently and automatically. Furthermore, skin surface approximates objects within an Hausdorff distance that is better than implicit surfaces [20].
4. **Quality triangulation.** Triangulation of the surface requires good triangle quality as an aid in the numerical analysis of physical simulations, while

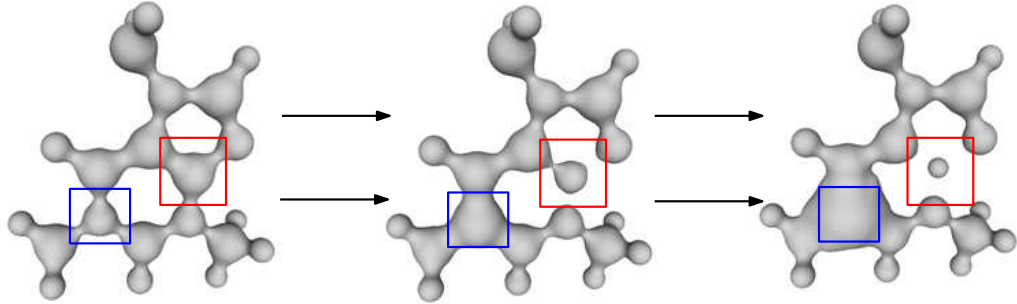


Figure 1.4: Local modification of a caffeine molecule. In the red box, one atom is shrunk to half its original size. In the blue box, another atom is enlarged to twice its original size. Topology changes are handled automatically.

maintaining homeomorphism to the surface during the deformation. In our previous work [16; 19], triangulation of the skin surface with good quality triangles was demonstrated. However, CGAL [14] (version 4.0.1) provides the user with triangles with very small angles, which leads to bad visualization and inaccurate computation (Figure 1.5).

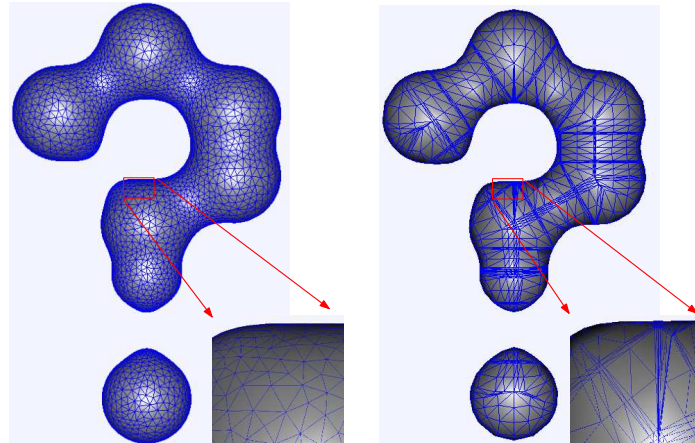


Figure 1.5: Comparison of the triangle quality of the question mark skin surface shown in Figure 1.1 by different programs. The triangular mesh that is generated by our program is shown on the left and the one that is generated by CGAL-4.0.1 is shown on the right.

- 
5. **Point-wise surface point correspondence.** During morphing, every surface point  $p(t) \in \mathcal{F}(t)$  moves to a new position  $p(t + \Delta t) \in \mathcal{F}(t + \Delta t)$ , and the pair  $p(t)$  and  $p(t + \Delta t)$  establishes correspondence for relating different portions between the two surfaces  $\mathcal{F}(t)$  and  $\mathcal{F}(t + \Delta t)$ . This relationship provides point-wise surface correspondence between the same surface at different times during deformation, which is a crucial feature for computation and visualization purposes in fields like medical imaging [58; 69], animation [43; 44] (e.g. texture and bump mappings) and physical simulations [7] (e.g. finite element analysis).
  6. **Efficiency Improvement.** Shape deformation requires visually continuous frames of meshes during morphing. Current static skin meshing algorithms [14; 17; 19; 23; 47] build each intermediate frame from scratch. For example, in Figure 1.2, the construction of each intermediate frame requires more than five minutes with any existing static skin mesh approach. The skin deformation algorithm has been improved in the current work, such that each intermediate skin mesh is obtained from the previous time frame. This improvement makes real time visualization of the skin surface deformation possible.

## 1.3 Related Work

The skin surface was first introduced as a maximum curvature continuous surface model for molecules by Edelsbrunner in 1999 [29]. It has several distinct properties such as smoothness, deformability and complementarity, which are desirable in biological studies such as protein docking and protein-protein interactions

---

[26; 32; 62].

However, there has been no proper algorithm for handling deformation between arbitrary skin surfaces for many years, after the initial idea was proposed by Edelsbrunner in 1999 [29]. In 2002, Cheng et al proposed the growth model as the first deformation framework of skin surfaces [16]. In the growth model, all weighted points increase or decrease by the same  $\alpha$  value [35], which is not useful for applications in the real world. The main difficulty of modeling skin surface deformation is all intermediate shapes have to maintain the skin surface properties with the correct topology. The intermediate skin surfaces are required to maintain the same homology groups as the alpha complexes [31; 34; 39], whereby most of the existing morphing theories failed to handle during the skin surface deformation. The only way to visualize the deformation between different skin surfaces is to generate each frame separately using static skin mesh generation methods, such as Kruithof’s algorithm [47] developed in CGAL [14] and quality skin mesh software developed by Cheng and Shi [18; 19]. These methods have several disadvantages, such as lack of efficiency, no surface point correspondence and discontinuity of homology group changes [21]. In 2006, Cheng and Chen found that the super-imposed Voronoi diagram of two or more skin surfaces remains unchanged during the deformation process [15]. This makes continuous skin deformation possible for any combination of skin surfaces. In 2010, the GSD algorithm was implemented in our work to perform deformation between any given skin surfaces under the general position assumption (GPA) [21].

The interest of skin surface deformation is no longer restricted to molecular studies, but applied to all forms of objects that are represented by sets of weighted points. With the use of algorithms converting polygonal objects into weighted

---

point sets [12; 20; 52; 68], the GSD algorithm can perform global deformation between real world objects which are approximated by skin surfaces. In contrast to many existing deformation algorithms, such as the shape-interpolation by Alexa et al. [1; 57], the skeleton-driven deformation works [49; 71] and the vector field based mesh editing [11; 41; 67; 74; 76], the GSD algorithm has the advantages of handling objects of greatly dissimilar shapes and automatic topology changes (Figure 1.1 and 1.2), real-time visualization and automatic surface point correspondence mapping (Chapter 7).

The GSD software is available at <http://www.comp.nus.edu.sg/~yanke2>.

## 1.4 Framework

The whole GSD algorithm is separated into three parts. First, both source and target shapes are converted into weighted point sets  $B_0$  and  $B_1$  by existing algorithms, such as the power crust [52] or the sphere-tree toolkit [12]. Second, the algorithm formulated from our previous work is used to convert  $B_0$  and  $B_1$  into skin surfaces  $\text{skin}(B_0)$  and  $\text{skin}(B_1)$  [16; 18; 19]. Finally, each skin patch is deformed in its mixed cell with a parameter  $t \in [0..1]$  as time. It is even possible to interpolate more than two objects for performing shape synthesis by their respective skin models [15].

The triangle quality of the mesh at any time  $t$ , is maintained at a certain quality that guarantees homeomorphism between the mesh and the skin surface. This is described in Section 2.1.5. It is assumed that at time  $t$ , a triangle  $\tau$  is in good quality. A time  $t + \Delta t$  in the future is scheduled into a priority queue, such that  $\tau$  may fail the quality check but is not beyond repair. Checks and

---

refinements (if necessary) are performed at time  $t + \Delta t$ , and all the involved triangles are rescheduled after refinement. At the same time, topological change operations are also scheduled into the priority queue in order to maintain the mesh.

## 1.5 New Issues

As compared to the old growth model [16; 30], the GSD algorithm faces new issues such as more sophisticated intermediate complex, surface point movements and scheduling. Solutions to these new issues are provided and the general skin surface deformation problem is solved and presented in this thesis.

First, the nature of the mixed cells in the general deformation is more complex than that of the growth model. The entire space is partitioned into a finite number of convex polytopes called *mixed cells*, and the intersection of each mixed cell and the skin surface is a part of a quadratic surface. In the growth model, there are only four types of mixed cells and they are fixed in space. However, there are seven types of mixed cells in the general deformation, and they move and deform with time. Thus, computation of the trajectory of each surface point within a mixed cell, and the *escaping time* when the surface point transfers from one mixed cell to another, are the new issues that have to be considered.

Second, movement of these new mixed cells complicate the trajectories of surface points. In the growth model, each surface point within a mixed cell moves in a straight line or a quadratic curve when one tracks the surface normals, and this enables the prediction of the triangle distortion. As the mixed cells in the general deformation undergo deformation, the surface points do not move in such

---

a simple manner. Tracking surface normals is not an easy task in the general skin deformation model. Therefore, a new way of surface point movement is proposed in the current work. Experiment results show that this new movement improves the efficiency of the GSD algorithm.

Finally, the topology changes are different. In the growth model, there is at most one topology change within one mixed cell but there are at most two in the general deformation.

## 1.6 Contribution

In the current work, a simple and efficient deformation solution, namely GSD, is presented for objects of greatly dissimilar source and target shapes with no similarity information provided. Our algorithm solves the general skin surface deformation problem based on the old growth model with new improvements. New types of mixed cells and their transformations are addressed. New surface point moving trajectories are proposed to deal with more complicated surface movement. New topology changes are handled by scheduling. The efficiency of the program is improved by moving the skin surface mesh from the previous time frame to the current time frame.

## 1.7 Outline

The main goal of this thesis is to present the investigation and implementation of the skin surface deformation algorithm, based on super-imposition of the Voronoi complexes of the source and target shapes [15].

---

Two versions of the GSD software are described in this thesis. The first version implements the general skin deformation as first proposed by Edelsbrunner [16; 29]. There is a total of six primary combinations of intermediate Delaunay and Voronoi complexes for different inputs under the GGP Assumption. The efficiency is improved as compared with the static skin algorithms [18; 19; 47], because each intermediate skin mesh is obtained from the previous time frame. A simplified version of GSD, namely SGSD, is proposed in the second version in order to further improve the efficiency [73]. In the SGSD algorithm, there are two more combinations of intermediate Delaunay and Voronoi complexes added due to the violation of the GGP Assumption. The new SGSD algorithm improves the overall efficiency of GSD from  $O(m^2n^2)$  to  $O(m^2 + n^2)$ . This improvement makes SGSD a real time software for any given morphing input.

In this thesis, the fundamentals of the skin surface deformation, such as the skin decomposition and super-imposition of the Voronoi complexes, are presented in Chapter 2. The definitions of the intermediate complexes are introduced in Chapter 3. The general skin surface deformation with quality triangular mesh is explained in Chapter 4. A simplified version of GSD is discussed in Chapter 5. More degenerate cases are discussed in Chapter 6 while focus is placed on a special deformation of partial movements. The description of the skin deformation software is in Chapter 7 and the conclusion is in Chapter 8.

# Chapter 2

## Preliminaries

In the first part of this chapter, all the necessary background definitions for the skin surface [29] are listed. The concepts of Delaunay and Voronoi complexes [5; 36], skin decomposition [22], and quality triangulation of the skin surface [18; 19] are introduced. In the second part, the original idea of general skin deformation proposed by Edelsbrunner in 2000 [17; 34], and our improvements made to the super-imposition proposed by Chen and Cheng [15; 21] are introduced. The notations that are presented in the later parts of this thesis are defined in this chapter. Readers may refer to the specific references for more details.

### 2.1 The Skin Surface Representations

In this section, the fundamentals of the skin surface representations are introduced, including weighted points, convex hull, the skin surface, skin decomposition by mixed cells and quality skin surface triangulation mesh [18; 19].

---

### 2.1.1 Weighted Points

A *weighted point* in  $\mathbb{R}^d$  can be written as  $b_i = (z_i, w_i) \in \mathbb{R}^d \times \mathbb{R}$  where  $z_i \in \mathbb{R}^d$  is the position and  $w_i \in \mathbb{R}$  is the weight. It is also viewed as a  $d$  dimensional sphere with center  $z_i$  and radius  $\frac{|w_i|}{w_i} \sqrt{|w_i|}$ . Negative radius is allowed since the weight  $w_i$  is possibly negative. Given a finite set  $B = \{b_1, b_2, \dots, b_n\}$ ,  $z(B) \subset \mathbb{R}^d$  is denoted as the set of the positions of the weighted points in  $B$ . The weighted distance of a point  $x \in \mathbb{R}^d$  from a weighted point  $b_i$  is defined as  $\pi_{b_i}(x) = \|xz_i\|^2 - w_i$ .

### 2.1.2 Skin

A skin surface is specified by a set of weighted points  $B = \{b_i \in \mathbb{R}^d \times \mathbb{R} \mid i = 1, 2, \dots, n\}$ . Three operations on weighted points are defined in the sphere algebra, namely *addition*, *scalar multiplication* and *square root* of weighted points for  $b_i, b_j \in B$  and  $\gamma \in \mathbb{R}$ , which are defined as,

$$\begin{aligned} b_i + b_j &= (z_i + z_j, w_i + w_j + 2\langle z_i, z_j \rangle), \\ \gamma b_i &= (\gamma z_i, \gamma w_i + (\gamma^2 - \gamma)\|z_i\|^2), \\ \sqrt{b_i} &= (z_i, w_i/2), \end{aligned}$$

where  $\langle z_i, z_j \rangle$  is the dot product of  $z_i$  and  $z_j$ .

The *convex hull* of  $B$  is defined as

$$\text{conv}(B) = \left\{ \sum \lambda_i b_i \mid \sum \lambda_i = 1 \text{ and } \lambda_i \geq 0, i = 1, \dots, n \right\}.$$

The skin body is the union of all shrunken balls in  $\text{conv}(B)$  and the skin surface is the boundary of the union of all shrunken balls in  $\text{conv}(B)$  (Figure 2.1), which

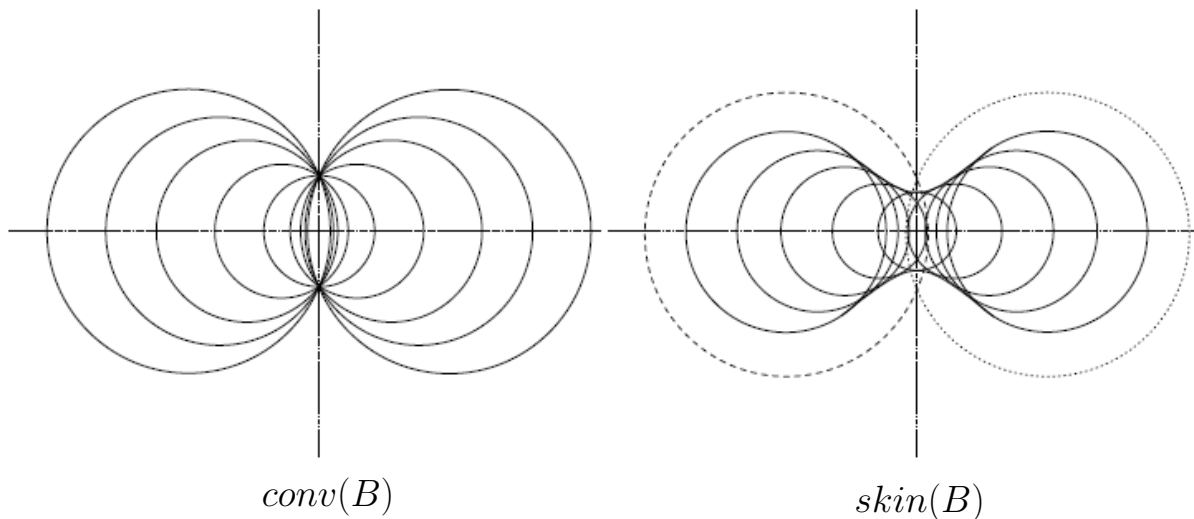


Figure 2.1: Assume  $B$  consists of only two weighted points, which are the two largest circles on the left sub-figure. The convex hull of  $B$  in  $\mathbb{R}^2$  is shown on left. The skin surface of all shrunk circles in the convex hull of  $B$  is shown on the right.

is formally expressed as

$$\text{skin}(B) = \partial\left(\bigcup \sqrt{\text{conv}(B)}\right).$$

In  $\mathbb{R}^3$ , the skin surface is a  $C^1$  continuous surface with continuous maximum normal curvature,  $\kappa$ , which is formed by a finite collection of weighted points or spheres (Figure 2.2). If the weighted points represent the atoms of a molecule, then the appearance of that surface is similar to the molecular surface used in structural biology [25; 48]. The skin surface and the molecular surface differ in a number of details, one such difference is that the former uses hyperboloids to blend sphere patches while the latter uses tori.

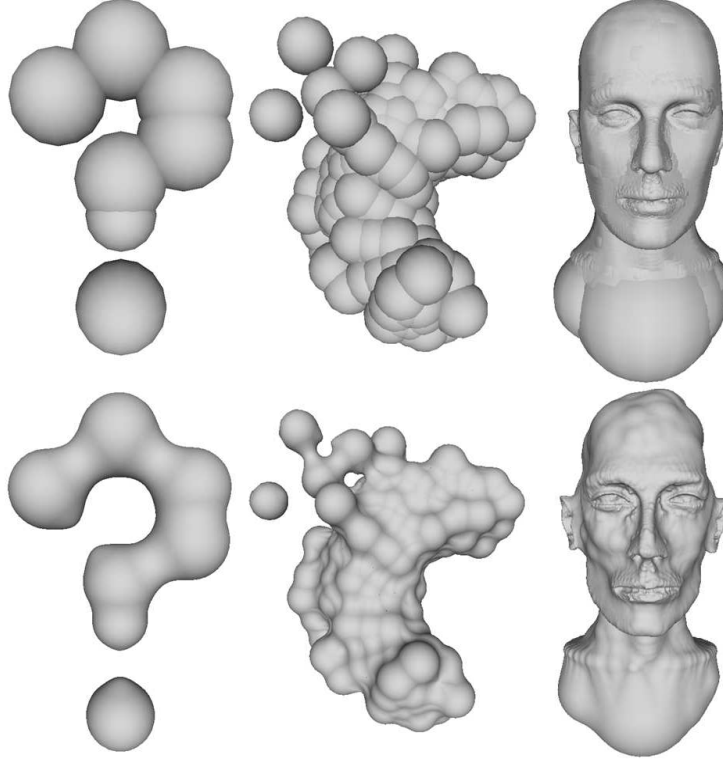


Figure 2.2: Demonstration of skin surfaces in  $R^3$ . Different sets of weighted points are shown in the first row. The corresponding skin surfaces for the weighted point sets in the first row are shown in the second row.

### 2.1.3 The Delaunay, Voronoi Complexes, General Position Assumption and Mixed Cells.

The *Voronoi region*  $\nu_i$  for each weighted point  $b_i \in B$  is defined as,

$$\nu_i = \{x \in \mathbb{R}^d \mid \pi_{b_i}(x) \leq \pi_{b_j}(x), b_j \in B\}.$$

For a set of weighted points  $X \subseteq B$ , the *Voronoi cell* of  $X$  is defined as  $\nu_X = \bigcap_{b_i \in X} \nu_i$ . The collection of all the non-empty Voronoi cells is called the *Voronoi*

---

complex of  $B$ , which is denoted by,  $V_B$ . For each  $\nu_X \in V_B$ , the corresponding *Delaunay cell*,  $\delta_X$ , is the convex hull of the set of centers of  $X$ , namely,  $\text{conv}(z(X))$ . The collection of all the Delaunay cells is called the *Delaunay complex* of  $B$ , which is denoted by  $D_B$ . The Delaunay complex is simplicial under the following general position assumption in  $\mathbb{R}^d$ :  $\forall \nu_X \in V_B, \text{card}(X) = \dim(\delta_X) + 1$ .

A general position assumption is usually made in order for the Delaunay complex to be simplicial. This is defined as the **Traditional General Position (TGP) Assumption**, namely,  $\forall \nu_X \in V_B, \text{card}(X) = \dim(\delta_X) + 1$ . Under this assumption, there are only four types of Delaunay cells in  $\mathbb{R}^3$ : vertices, edges, triangles and tetrahedra.

A mixed cell  $\mu_X$  is the Minkowski sum of a Delaunay cell and its corresponding Voronoi cell, formally expressed as  $\mu_X = (\delta_X + \nu_X) / 2$ . With  $\text{card}(X) = 1, 2, 3, 4$ , the four types of mixed cells are convex polyhedrons, prisms, triangular prisms and tetrahedra respectively (Figure 2.3). The center and size of a mixed cell are defined as

$$\begin{aligned} z_X &= \text{aff}(\delta_X) \cap \text{aff}(\nu_X), \text{ and} \\ w_X &= w_i - \|z_X z_i\|^2. \end{aligned}$$

where  $b_i = (z_i, w_i)$  is any weighted point in  $X$  and  $w_X$  is the negative square radius of the ball orthogonal to  $X$ .

#### 2.1.4 Skin Decomposition.

The skin surface  $\text{skin}(B)$  can be decomposed by mixed cells (Figure 2.4). All four types of mixed cells, which are constructed by the Delaunay and Voronoi

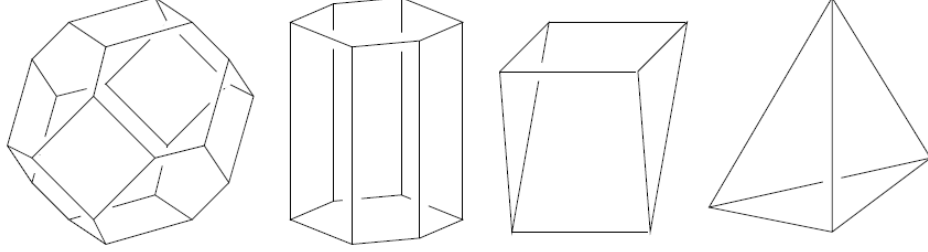


Figure 2.3: All possible mixed cell types under TGP Assumption.

complex of  $B$ , partition the whole space including the skin surface. Within each

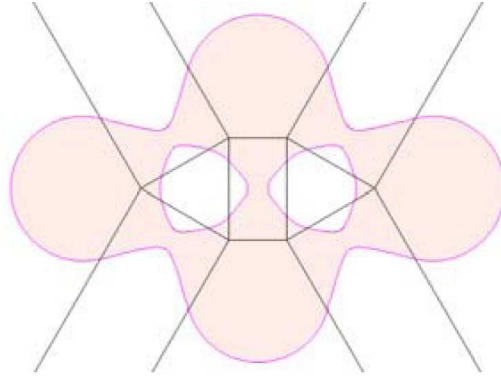


Figure 2.4: Skin decomposition in  $\mathbb{R}^2$ .

mixed cell  $\mu_X$ ,  $\text{skin}(B) \cap \mu_X$  is a quadratic surface. In  $\mathbb{R}^3$ , skin patches are pieces of spheres and hyperboloids of revolution (Figure 2.5), which can be expressed in the following standard forms

$$x_1^2 + x_2^2 + x_3^2 = R^2, \quad (2.1)$$

$$x_1^2 + x_2^2 - x_3^2 = \pm R^2, \quad (2.2)$$

after the translation of  $z_X$  to the origin and the oriented axis to the  $x_3$ -axis if they are hyperboloid patches. In Equation (2.2), the plus sign on the right hand

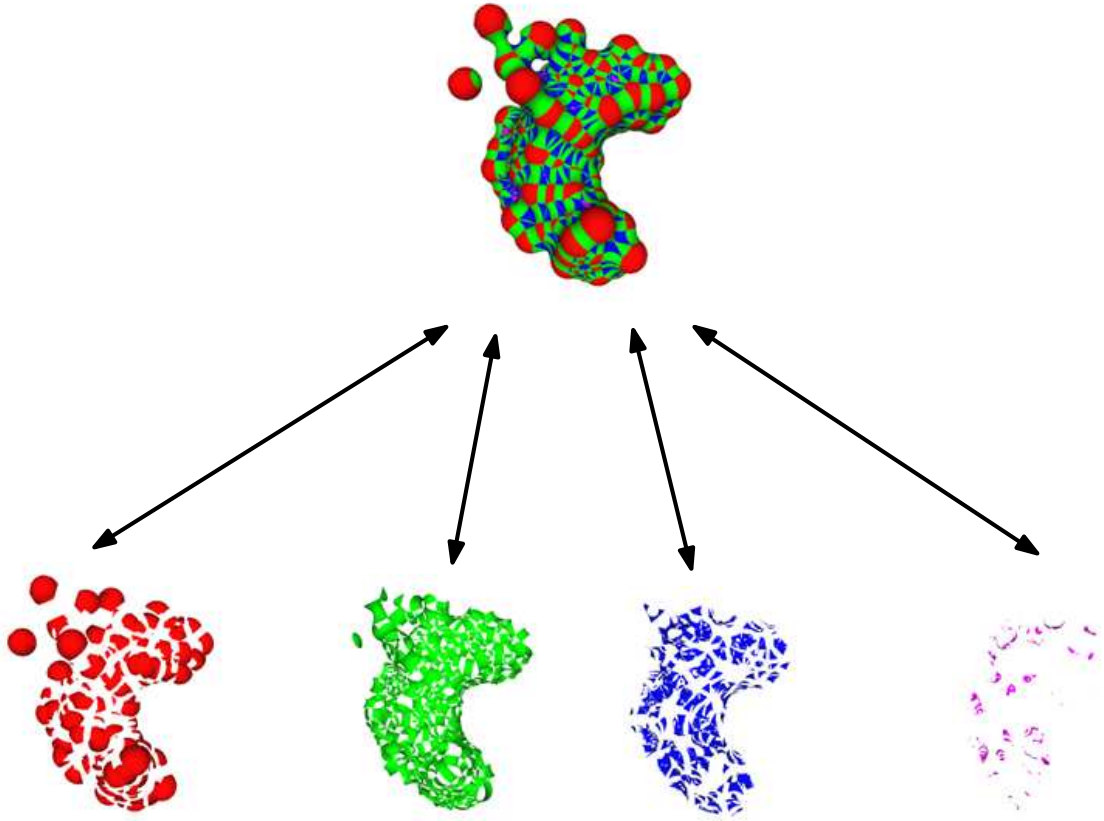


Figure 2.5: In  $\mathbb{R}^3$ , the skin surface is decomposed into sphere or hyperboloid patches bounded by four types of mixed cells.

side results in one-sheeted hyperboloid and the minus sign results in two-sheeted hyperboloid.

All possible combinations of Delaunay cells, Voronoi cells, mixed cells and skin patches are listed in Table 2.1.4. Each type of Voronoi cell is classified by the dimension of that particular Voronoi cell, namely,  $\dim(\nu_X)$ . There are four types of combinations under the TGP Assumption. The table is expanded in Chapter 3 when the intermediate complexes are introduced.

---

Type	Voronoi Cell	Delaunay Cell	Mixed Cell	Skin Patch
3	Polyhedron	Vertex	Polyhedron	Sphere
2	Polygon	Edge	Prism	Hyperboloid
1	Edge	Triangle	Triangular Prism	Hyperboloid
0	Vertex	Tetrahedron	Tetrahedron	Sphere

Table 2.1: Possible combinations of Voronoi cells, Delaunay Cells, Mixed Cells and Skin Patches

### 2.1.5 Quality Skin Surface Triangular Mesh

Triangulation of the skin surface guarantees homeomorphism and good quality triangles if the following Conditions [U] and [L] are satisfied. At any point  $x$  on the skin surface,  $\kappa(x)$  is denoted as the maximum principal curvature at  $x$ , and the *local length scale* at  $x$  is expressed as  $\varrho(x) = 1/\kappa(x)$ . The half length of an edge  $ab$  is assigned as  $R_{ab} = \|a - b\|/2$ , and the circumcircle radius of a triangle  $abc$  is denoted by  $R_{abc}$ . The mesh is homeomorphic to the surface if it satisfies the Lower bound Condition [L] and the Upper bound Condition [U] below. Let  $\varrho_{ab}$  be the maximum of  $\varrho(a)$  and  $\varrho(b)$ , and  $\varrho_{abc}$  be the minimum of  $\varrho(a)$ ,  $\varrho(b)$  and  $\varrho(c)$ . Then the two conditions are,

$$[L] \quad R_{ab}/\varrho_{ab} > C/Q \text{ for every edge } ab, \text{ and}$$

$$[U] \quad R_{abc}/\varrho_{abc} < CQ \text{ for every triangle } abc,$$

where  $C$  and  $Q$  are positive constants obtained empirically. The triangle quality is guaranteed by the minimum angle of the triangles, namely  $\sin^{-1} \frac{1}{Q^2}$ . For  $C = 0.08$  and  $Q = 1.65$ , the minimal angle of the mesh is proven to be larger than  $21.54^\circ$  [16] (Figure 2.6).

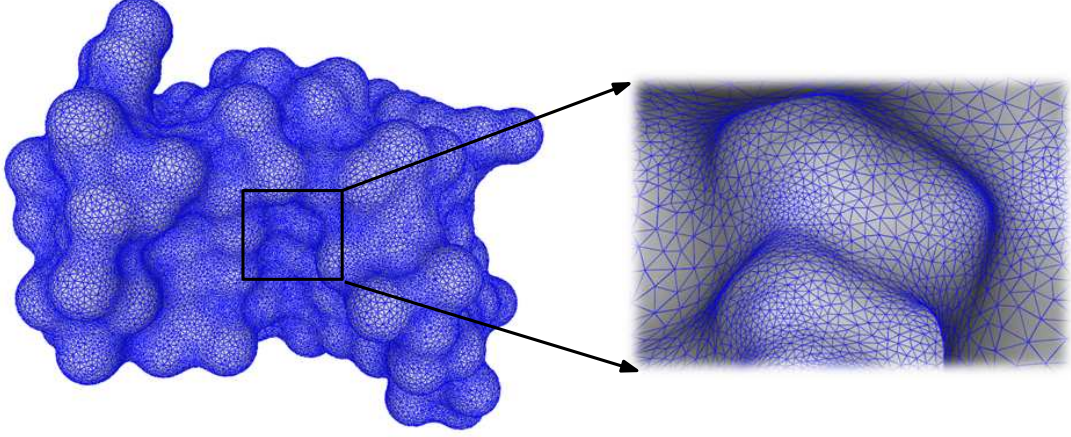


Figure 2.6: An example of quality triangular mesh satisfying [L] and [U] conditions.

## 2.2 The Overview of General Skin Deformation

The skin surface possesses the ability to deform from one shape into another freely in a general manner. However, there is no algorithm prior to our work which is able to perform the skin surface deformation with triangular meshes. If any two polygonal shapes are approximated by two skin surfaces, a robust and fully automatic deformation is possible without any constraint, such as similarity or topology of the two shapes [20]. Figure 1.1 and 1.2 show examples of such general skin surface deformation.

In this section, an overview of the GSD algorithm is provided. First, the growth model, which increases the weights of a set of weighted points without changing their positions, is introduced. The maintenance of the triangular mesh of the skin surface in the growth model is similar to that of the GSD algorithm. Therefore, the growth model serves as a base case of the GSD algorithm. Second, the construction of the intermediate weighted point sets, which define the

---

intermediate skin surfaces, is shown. The intermediate weighted point sets are obtained by interpolating the source and target weighted point sets, and they share the same Voronoi complex through out the deformation process.

### 2.2.1 Growth Model

The growth model is the first skin surface deformation model [16]. It begins with a set of weighted points, all with zero weights, and then slowly increasing all weights with the same  $\alpha$  value. Figure 2.7 illustrates the growth model using the example of increasing the radii of atoms in a DNA molecule. Since all weighted point positions remain unchanged and the weights are increased by the same  $\alpha$  value, all intermediate skin surfaces share the same Delaunay and Voronoi complexes.

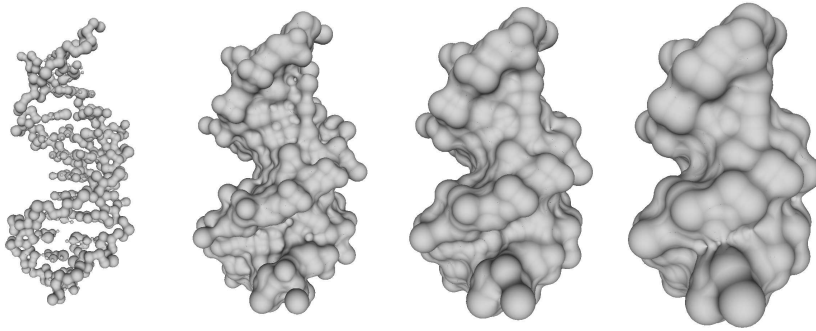


Figure 2.7: The growth model.

The deforming mesh algorithm presented by Edelsbrunner [30] updates the positions of points on the mesh surface for a growing skin from the previous time frame according to their trajectories. During the updates, *mesh maintaining operations* are scheduled to preserve the homeomorphism and maintain the quality of triangles. A point on the sphere patch, is moving towards or away from the

---

sphere center during the growth motion. The motion of a point on the hyperboloid follows a hyperbola (Figure 2.8).

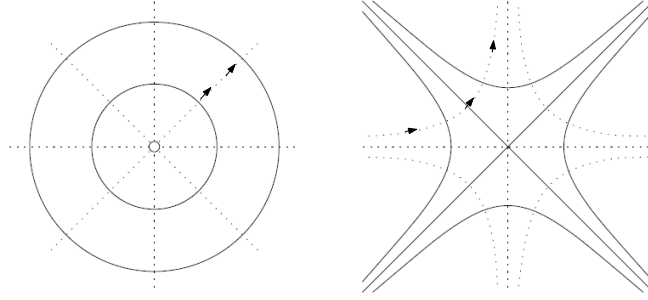


Figure 2.8: Trajectories of a point on the growing circle and growing hyperbola

In order to maintain the homeomorphism between the skin surface and the mesh during deformation, maintaining operations are applied to the mesh to ensure that it satisfies Conditions [U] and [L]. These operations are:

1. Coordinates update
2. Edge flipping
3. Vertex insertion
4. Edge contraction
5. Metamorphosis

Of these five operations, the coordinates of a point on the surface are updated according to the trajectory of the point. The other operations can be further decomposed into a sequence of sub-operations. For example, a vertex insertion relies on point additions and edge flips. The metamorphosis is used to handle

---

topology changes. For example, a two-sheeted hyperboloid becomes a one-sheeted hyperboloid. When the metamorphosis is scheduled, a special sampling is used to preserve the homeomorphism between the mesh and the skin surface [16].

### 2.2.2 General Skin Deformation

The growth model provides an idea of how to maintain a moving mesh without reconstructing each intermediate skin surface from scratch. However, the growth model is inadequate for handling general deformation that is more complicated and useful [15; 17; 23]. In general deformation, the weights of the points are not changing at the same increasing speed and their locations are not fixed.

Let  $B_0$  and  $B_1$  be the weighted point sets of source shape skin ( $B_0$ ) and target shape skin ( $B_1$ ) respectively. By denoting  $\mathbb{B} = \{B_0, B_1\}$ , the intermediate weighted points set at time  $t \in [0, 1]$  is defined as,

$$B(t) = \{b_{ij}(t) = (1 - t)b_i + tb_j \mid b_i \in B_0, b_j \in B_1\}, \text{ for } t \in [0, 1].$$

The intermediate shape is defined as skin ( $B(t)$ ) and it deforms smoothly as time  $t$  changes.

The *intermediate Voronoi complex*  $V$  is defined as the Voronoi complex of  $B(t)$  and is required for the generation of skin ( $B(t)$ ). Chen and Cheng [15] proved that all intermediate surfaces share the same intermediate Voronoi complex  $V$  which is the superimposition of all Voronoi complexes of  $\mathbb{B}$ .

Since the intermediate Voronoi complex remains unchanged, the general skin deformation moves the skin surface mesh in a manner similar to that of the growth model, but with additional freedom and different surface vertex movement. The

---

idea is to *schedule* updates on the surface triangles as the mesh morphs during the progression of time  $t$ . However, the growth model only allows a skin surface to ‘grow’ from nothing into a shape instead of deforming from one object into another freely. In general skin deformation, the following new issues are considered:

*The trajectory of vertices on the surface mesh.* A new way of moving the surface vertices on the mesh is employed. This new method is more efficient and ‘intuitive’ than the growth model, which moves vertices along their normal directions (Figure 2.8). First, the previous method creates a lot of ‘stretching’ or ‘compression’ near the tips of two-sheeted hyperboloids and the waists of one-sheeted hyperboloids. The large number of triangle modifications (creation and deletion) at these areas leads to unreasonable or undesirable texture distortion in animation or numerical instability in computations. Second, *all* surface triangles in Type 1 and 2 mixed cells require frequent checks on the triangles and edges to ensure their quality and the homeomorphism between the triangulation and the skin surface. Finally but most importantly, the model from the previous work has a simpler movement of surface vertices because the deformation is limited to the growth model and mixed cells do not move or deform. This enables the scheduling of checks and updates more easily than the general deformation presented in this thesis.

Therefore, a new type of vertex movement is proposed in order to improve and enable the implementation of general deformation. A decision is made to move vertices towards and away from their corresponding mixed cell centers in order to improve efficiency. It is faster and simpler to move a point along a straight line within a mixed cell. In the growth model, every point moves along a

---

hyperbola and solution of a quartic equation is required in order to compute the new location of the point.

***Escaping time scheduling.*** Every point on the surface is moving along a straight line within a mixed cell locally and the mixed cell is translating in a linear manner. However, the point does not move along a straight line after the combination of these two movements. Thus, there is still a need to handle and compute the time when a point reaches the boundary of its mixed cell and enters another one. This moment in time is called the *escaping time*, and it happens when a surface vertex  $p(t)$  in  $\mu_X(t)$  transfers to another nearby mixed cell  $\mu_Y(t)$  when  $p(t)$  is at the boundary of  $H_{XY}(t)$ .

***Scheduling of updates on the triangle quality.*** An element (a triangle or an edge) in a single mixed cell does not need checks and updates on its quality because it is only under scaling according to Conditions [U] and [L]. Thus, only the elements that span across more than one mixed cell require scheduling, which greatly reduces the priority queue size. Based on experimental results, the triangles that span across a few mixed cells occupy less than approximately 8% of the total triangle population.

***Topology changes of the surface.*** Topology changes in the GSD algorithm are different from that of the growth model because there may be at most two changes in each mixed cell. In the former case, there is at most one topology change for each patch in a mixed cell during the growth motion, but it is different in the

---

case of general deformation where each mixed cell size of  $w_X(t)$  is a quadratic function of  $t$ . Moreover, a topology change operation is scheduled only if the mixed cell center is in the mixed cell. Both the center and the cell are moving and deforming, which is different from that of the growth model. This indicates that there is a possibility that the center is not in the mixed cell when  $w_X(t) = 0$ .

## Chapter 3

# Intermediate Complexes in Skin Deformation

The intermediate Delaunay and Voronoi complexes are important underlying data structures for the construction of intermediate skin shapes in skin deformation. Let  $B_0$  and  $B_1$  be the weighted point sets of source shape  $\text{skin}(B_0)$  and target shape  $\text{skin}(B_1)$  respectively. At each time  $t$ , the intermediate Delaunay and Voronoi complexes are obtained by the intermediate weighted point set  $B(t) = \{b_{ij}(t) = (1 - t)b_i + tb_j \mid b_i \in B_0, b_j \in B_1\}$ .

Two approaches are used to generate intermediate skin shapes of  $B(t)$ . In the old approach, intermediate Delaunay and Voronoi complexes are constructed directly from  $B(t)$ , the skin patches are built separately in different mixed cells, and then the intermediate skin shapes are formed. This approach is slow and complicated. The intermediate skin shapes are all built separately. Therefore, it is difficult to deliver the surface information during deformation, such as textures. In contrast, in the current work, the intermediate skin shapes are generated di-

---

rectly from the previous time frame, which is much simpler and faster. In 2006, Chen and Cheng found that all intermediate skin surfaces built from  $B(t)$  share the same Voronoi complex, which is the superimposition of the Voronoi complexes of  $B_0$  and  $B_1$  (Figure 3.1). This unique Voronoi complex is called the intermediate Voronoi complex  $V$  relative to  $B(t)$  [15]. The dual intermediate Delaunay complex  $\delta$  and intermediate mixed cells move linearly with  $t$  because  $V$  remains unchanged. In the new general skin deformation approach, the skin surface mesh moves directly from the previous time frame according to the new positions and new weights of the intermediate mixed cells.

In this chapter, the unique intermediate Voronoi complex through-out the skin deformation process is first defined. Next, the intermediate Delaunay complex, which is the dual shape of the intermediate Voronoi complex, is introduced. A comparison between our Delaunay update system and the famous dynamic Delaunay triangulation [64] is done to highlight the advantages of our system. Finally, the intermediate mixed cells and their connections are defined. The mixed cell connection provides the mixed cell neighborhood information, which is important for the location of skin surface mesh points in their mixed cells during deformation.

### 3.1 Intermediate Voronoi Complexes

Let the Voronoi complex of  $B_0$  be  $V_0$ , and the Voronoi complex of  $B_1$  be  $V_1$ . The intermediate Voronoi complex is the super-imposition of  $V_0$  and  $V_1$ ,

$$V(t) = \{\nu_{XY} | \nu_X \in V_0, \nu_Y \in V_1, \nu_X \cap \nu_Y \neq \emptyset\},$$

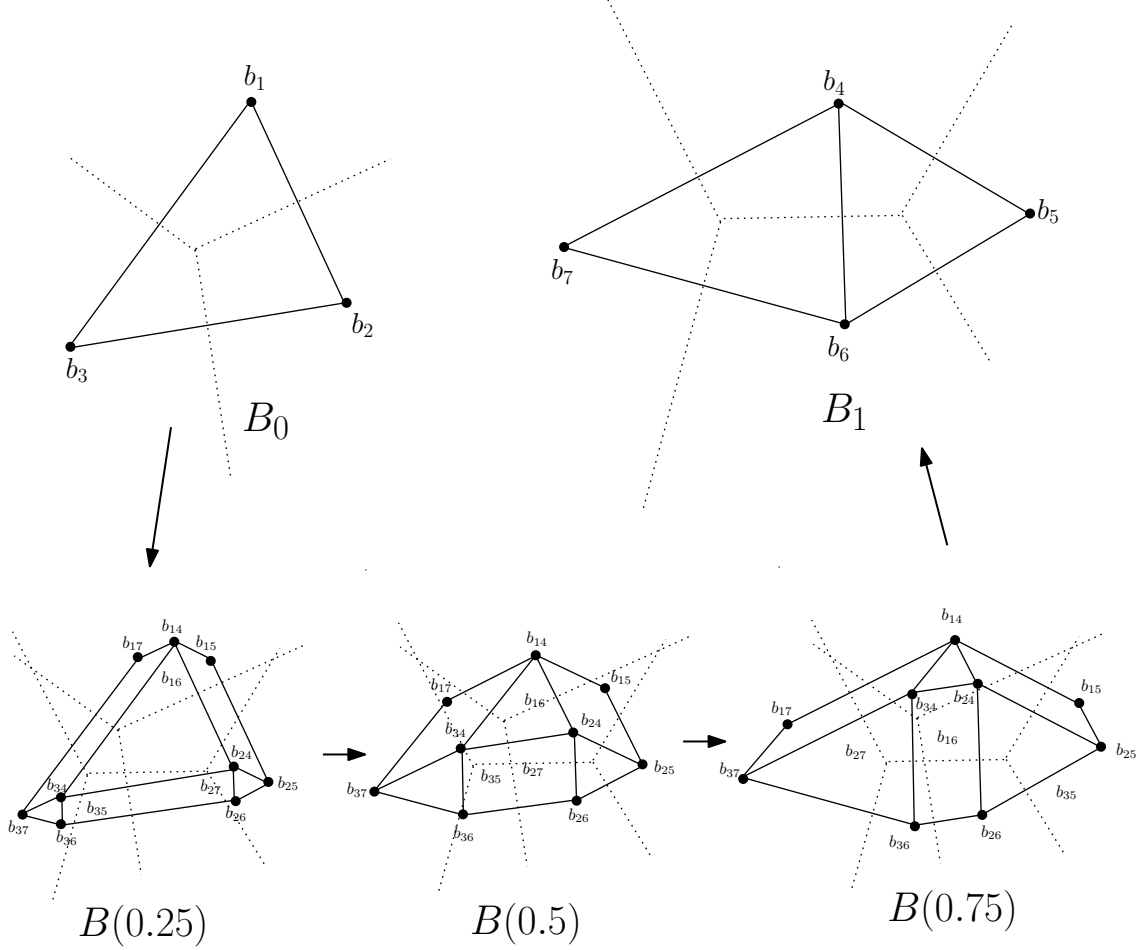


Figure 3.1: Demonstration of deformation from a three weighted point model  $B_0$  to a four weighted point model  $B_1$  in  $\mathbb{R}^2$ . All intermediate weighted point sets  $B(0.25)$ ,  $B(0.5)$  and  $B(0.75)$  share the same Voronoi complex.

where each  $\nu_{XY}$  is called an intermediate Voronoi cell of  $V(t)$ . For  $t \in [0, 1]$ , with the invariance of the intermediate Voronoi complex, the type of intermediate Voronoi cells in the process of deformation is determined. It is assumed that the two weighted point sets are under the TGP Assumption individually. A new general position assumption is proposed for the super-imposition of two different Voronoi complexes.

---

**GSD General Position Assumption (GGP Assumption).** In  $\mathbb{R}^d$ , the dimension of  $\nu_{XY} = \nu_X \cap \nu_Y$ ,  $\nu_X \in V_0$  and  $\nu_Y \in V_1$ , is

$$\dim(\nu_{XY}) = \dim(\nu_X) + \dim(\nu_Y) - d.$$

The equation above violates the TGP Assumption as it allows new types of Voronoi cells. There are six possible intermediate Voronoi cell types under GGP Assumption after super-imposing Voronoi complexes in  $\mathbb{R}^3$  (refer to Table 2.1.4). Each type of intermediate Voronoi cells is classified by a tuple, namely,  $(\dim(\nu_X), \dim(\nu_Y), \dim(\nu_{XY}))$ . It is assumed that  $\dim(\nu_X) > \dim(\nu_Y)$  and all possible tuples are (3,3,3), (3,2,2), (3,1,1), (3,0,0), (2,2,1) and (2,1,0) (Table 3.1). The first four types of tuples represent the simplicial types, whereby their corresponding Delaunay cells are simplicial. The types, (2,2,1) and (2,1,0) represent two non-simplicial types. For example, Type (2,1,0) represents a Voronoi face (shared by two Voronoi regions) intersecting a Voronoi edge (shared by three Voronoi regions) at a vertex. This intermediate Voronoi vertex is a non-empty intersection of six Voronoi regions in  $V(t)$  under the GGP Assumption.

Table 3.1: Possible combinations of intermediate Voronoi Cells, Delaunay Cells, Mixed Cells and Skin Patches

Index	Type	Voronoi	Delaunay	Mixed Cells	Patch
0	(3,3,3)	Polyhedron	Vertex	Polyhedron	Sphere
1	(3,2,2)	Polygon	Edge	Right Prism	Hyperboloid
2	(3,1,1)	Edge	Triangle	Right Tri. Prism	Hyperboloid
3	(3,0,0)	Vertex	Tetrahedron	Tetrahedron	Sphere
4	(2,2,1)	Edge	Parallelogram	Parall. Prism	Hyperboloid
5	(2,1,0)	Vertex	Tri. Prism	Sheared Tri. Prism	Sphere

---

## 3.2 Intermediate Delaunay Complexes

The intermediate Delaunay complex,  $D(t)$ , which is not a simplicial complex, is defined as the Delaunay complex of  $B(t)$ . Apart from the regular Delaunay triangulation, the intermediate Delaunay complex is defined as,

$$D(t) = \{conv(z(\nu^{-1}(\nu_{XY}))) | \nu_{XY} \in V(t)\}, \quad (3.1)$$

where

$$\begin{aligned} z(X) &= \{z_i | b_i \in X\}, \text{ and} \\ \nu^{-1}(\nu_{XY}) &= \{b_{ij}(t) | \nu_X \in V_0, \nu_Y \in V_1, \nu_X \cap \nu_Y \neq \emptyset\}. \end{aligned}$$

Table 3.1 is verified with an example of Type (2,1,0). Type (2,1,0) is formed by an intersection of a Voronoi face and a Voronoi edge. This is the case when a Delaunay edge  $ab$  deforms to form a Delaunay triangle  $cde$  (see Figure 3.2). It was mentioned in the previous subsection that there are a total of six Voronoi regions sharing this type of Voronoi vertex. Hence, there are six intermediate weighted points sharing this Voronoi vertex. The intermediate Delaunay cell is the convex hull of the six weighted point centers, which is a triangular prism.

### 3.2.1 Degeneracies in Intermediate Delaunay Triangulation

Non-simplicial Delaunay cells exist in Table 3.1 because of degenerate weighted points violating the TGP Assumption in the intermediate Delaunay triangulation. For example, in regular Delaunay triangulation, the Delaunay cell in Type (2,1,0), which is a triangle prism, is decomposed into three tetrahedra (Figure

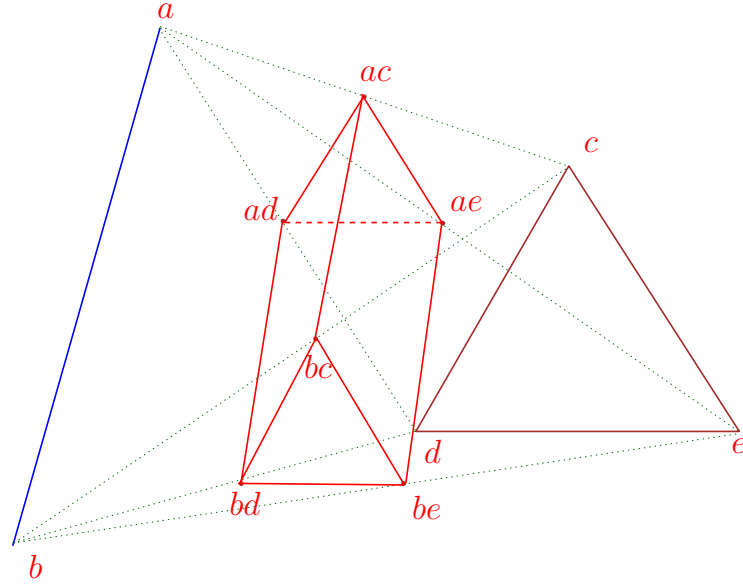


Figure 3.2: An Delaunay edge  $ab$  deforms to a Delaunay face  $cde$ . The intermediate Delaunay cell is non-simplicial and is a triangular prism.

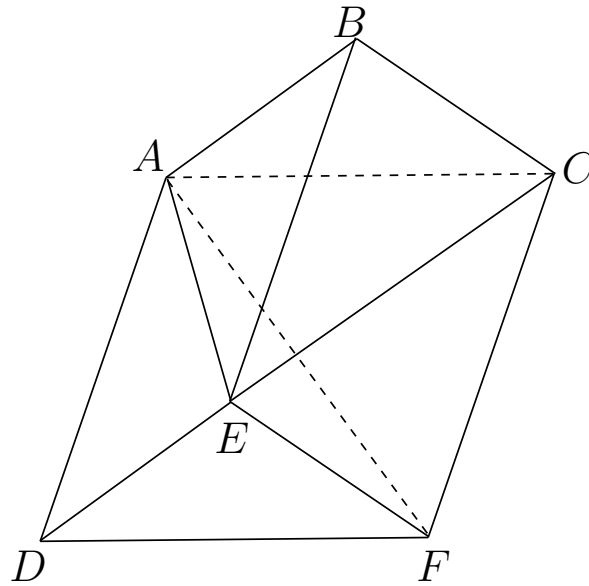


Figure 3.3: In a regular Delaunay triangulation, non-simplicial intermediate Delaunay cells are decomposed into simplices sharing the same Voronoi center.

3.3). However, these three tetrahedra share the same Voronoi center. In the intermediate Delaunay triangulation, the three degenerate tetrahedra are merged

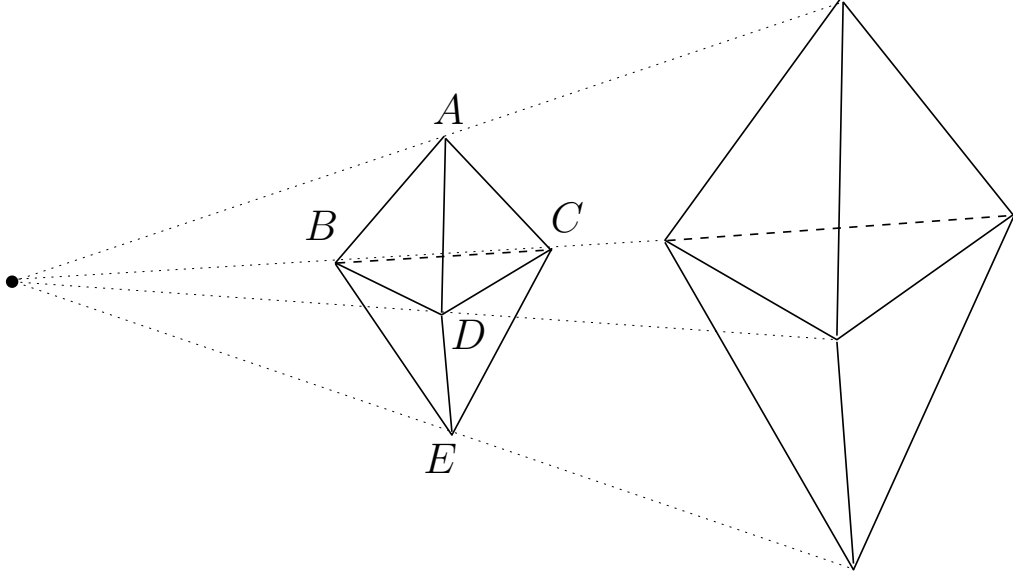


Figure 3.4: Five points  $A$ ,  $B$ ,  $C$ ,  $D$ ,  $E$  share the same Voronoi center. The degenerate triangle  $BCD$  exists in the intermediate Delaunay triangulation.

to provide a clearer Delaunay complex structure.

It is noted that the intermediate Delaunay triangulation does not prevent all degenerate cases violating the TGP Assumption. Two tetrahedra sharing the same Voronoi center are not combined, leaving a degenerate Delaunay triangle (Figure 3.4). The degenerate Delaunay triangle creates a zero volume mixed cell, which is discussed in Section 3.3.1.

### 3.2.2 Comparison of Delaunay Triangulation Updates and Dynamic Delaunay Triangulation

The complexity of a Delaunay triangulation update is always an interesting topic in computational geometry. The most famous algorithm to handle this problem is the dynamic Delaunay triangulation (DDT) [64].

The GSD algorithm is generally faster than DDT as it takes advantage of the

---

unchanging intermediate Voronoi complex. Although DDT performs the same function as the GSD algorithm of modeling weighted points in motion, DDT has to update the changes in topology of its Delaunay and Voronoi complexes [28; 56]. The GSD algorithm maintains the same topology for its Voronoi complex throughout deformation.

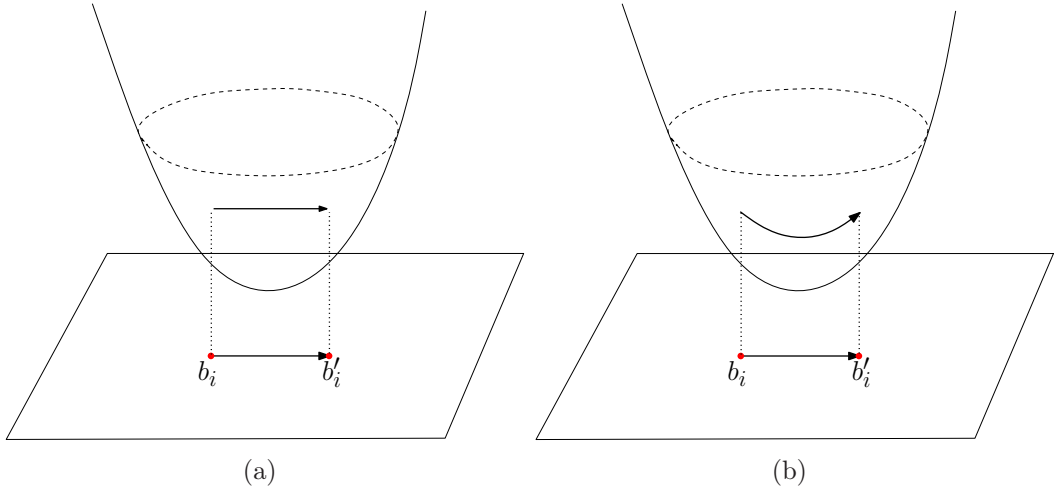


Figure 3.5: Weighted points movement in a higher dimension. On the left, a weighted point moves in a straight line in a higher dimension with GSD algorithm. On the right, a weighted point moves along the paraboloid in a higher dimension with DDT.

The difference lies in the fact that, the weighted points move linearly in a higher dimension after lifting in the GSD algorithm. A lifting map projects the weighted points in  $\mathbb{R}^d$  into unweighted points in  $\mathbb{R}^{d+1}$ . The Delaunay triangulation of the weighted points is the inverse projection of the convex hull of the lifted unweighted points. The GSD algorithm moves a lifted unweighted point in  $\mathbb{R}^{d+1}$  linearly, such that its Voronoi complex remains the same through-out the movements (Figure 3.5(a)). In contrast, DDT moves the lifted unweighted points along a paraboloid in a non-linear manner. This causes the Voronoi complex to

---

change (Figure 3.5(b)). Thus, frequent updates are necessary during deformation.

### 3.3 Intermediate Mixed Cell

Intermediate mixed cells are the Minkowski sum of their corresponding Delaunay and Voronoi cells. There are two more types for intermediate mixed cells, namely, parallelogram and Sheared Triangular Prism in Table 3.1 as compared to Table 2.1.4.

The skin patch within a mixed cell  $\mu_X(t)$  is determined by the position, the size and the shape of  $\mu_X(t)$ . The position and size are determined by the center  $z_X(t)$  and the size  $w_X(t)$ , and the latter defines  $R = \sqrt{\frac{w_X(t)}{2}}$  in Equations (2.1) and (2.2).

The intersection of the affine hulls of  $\nu_X(t)$  and  $\delta_X(t)$ , namely,  $z_X(t)$ , is first computed. As  $\nu_X(t)$  remains unchanged and all the centers of  $X(t)$  move linearly with  $t$ ,  $\delta_X(t)$  moves linearly, and thus, the intersection  $z_X(t)$  moves linearly along  $\nu_X(t)$ . The two positions  $z_X(0)$  and  $z_X(1)$  are computed as

$$z_X(t) = (1 - t) \cdot z_X(0) + t \cdot z_X(1), \quad (3.2)$$

$$w_X(t) = w_i(t) - \|z_i(t)z_X(t)\|^2, \quad (3.3)$$

where  $b_i(t) = (z_i(t), w_i(t)) \in X(t)$ .

The shape of  $\mu_X(t)$  is a convex polytope formed by the intersection of a finite number of halfspaces. Each halfspace  $H_{XY}(t)$  is bounded by a plane  $h_{XY}(t)$ , which separates the two mixed cells  $\mu_X(t)$  and  $\mu_Y(t)$  with  $Y(t) \subset B(t)$ , and  $\delta_Y(t)$

---

is a face or coface of  $\delta_X(t)$ . The symmetric difference  $X(t) \ominus Y(t)$  is a singleton set. The halfspace  $H_{XY}(t)$  is defined as

$$H_{XY}(t) = \{x \in \mathbb{R}^3 \mid \langle x, n_{XY}(t) \rangle \leq \langle m_{XY}(t), n_{XY}(t) \rangle\},$$

where the normal of  $h_{XY}(t)$  and a point on  $h_{XY}(t)$  are

$$\begin{aligned} n_{XY}(t) &= z_Y(t) - z_X(t), \\ m_{XY}(t) &= \frac{z_X(t) + z_Y(t)}{2}. \end{aligned}$$

### 3.3.1 Degeneracies in Intermediate Mixed Cells

In the current work, all zero-volume mixed cells are referred as degenerate mixed cells. These degenerate mixed cells are created by degenerate simplices violating TGP Assumption. The degenerate mixed cells are important in deformation of skin surface and are required for special considerations. For example, in Figure 3.4, the degenerate Delaunay triangle  $BCD$  creates a zero volume mixed cell of Type (3,1,1). For the intermediate mixed cell structure under GGP Assumption (violating TGP Assumption), only mixed cells of Type (3,1,1), (3,2,2) and (2,2,1) are possibly degenerate. More degenerate cases, by violating the GGP Assumption, are discovered in simplified general skin deformation in Chapter 5 and partial movement in Chapter 6.

It is impossible for surface points lie inside the degenerate mixed cells. Any point hits the boundary of this mixed cell will be transferred immediately into another mixed cell (Section 4.3.1.4).

---

### 3.3.2 Mixed Cell Connections

A mixed cell connection provides the information about any given mixed cell to all possible neighboring mixed cells. This is important in many implementation details, such as locating the mixed cell for newly inserted points, escaping time computation (Section 4.3.1) and sticky point testing (Section 4.1.3). Promptly finding neighboring mixed cells for the current mixed cell increases the program efficiency.

The mixed cell connectivity chart is shown in Figure 3.6. The chart is verified, beginning with mixed cell Type (3,3,3). In mixed cell Type (3,3,3), the Delaunay cell is a vertex, which connects to Delaunay edge cells. Therefore, mixed cell Type (3,3,3) connects to Type (3,2,2). The mixed cell Type (3,2,2) connects to Type (3,1,1) and (2,2,1), since the Delaunay edge cell possibly belongs to a triangle cell or a parallelogram cell. The parallelogram cell must be part of the sheared triangle prism, which connects Type (2,1,0) and (2,2,1). The triangle cell is possibly belongs to a tetrahedron or a sheared triangle prism. The mixed cell Type (3,1,1) connects to both (2,1,0) and (3,0,0).

In Figure 3.7, a simple example of the mixed cell connection graph is drawn with all mixed cell types included.

### 3.3.3 Allocating a Mesh Point in its Mixed Cell

The most direct application of the mixed cell connection (Section 3.3.2) is to allocate a mesh point into its mixed cell. There are many reasons to find the mixed cell  $\mu$  containing the mesh point  $p$ . For example, during the initialization of skin deformation, all given mesh points have to be allocated into different

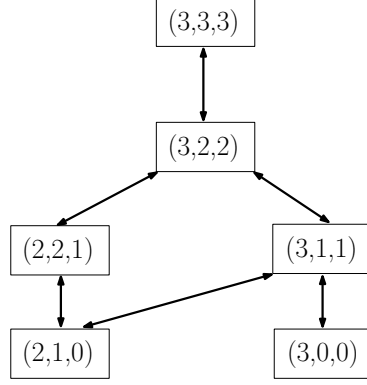


Figure 3.6: The connectivity chart of different mixed cell types.

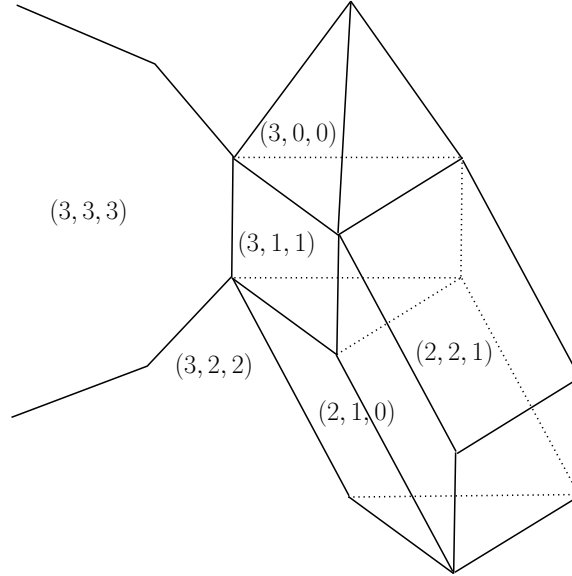


Figure 3.7: The connectivity graph of all mixed cell types.

mixed cells in order to obtain their initial moving trajectories. The newly inserted circum-center of a large triangle is required to find its mixed cell during triangle refinement, while the mixed cell information of the three vertices of the large triangle is provided. In escaping time calculation, an escaping vertex is allocated into a new mixed cell before the calculation of its new escaping time. Therefore, the allocation of mixed cells is an important task in the GSD algorithm.

---

Given a mesh point  $p_v$  and its mixed cell  $\mu_v$ , the task is to find the mixed cell  $\mu_X$  for a requested point  $p_X$  (Figure 3.8). Since all mixed cells are convex polyhedrons, the problem is reduced to the calculation of the intersection between a line segment  $p_v p_X$  and mixed cell boundaries (polygons), which is an easy task in computational geometry [59; 63]. Together with the mixed cell connection information given in the previous section, the required mixed cell  $\mu_X$  is found.

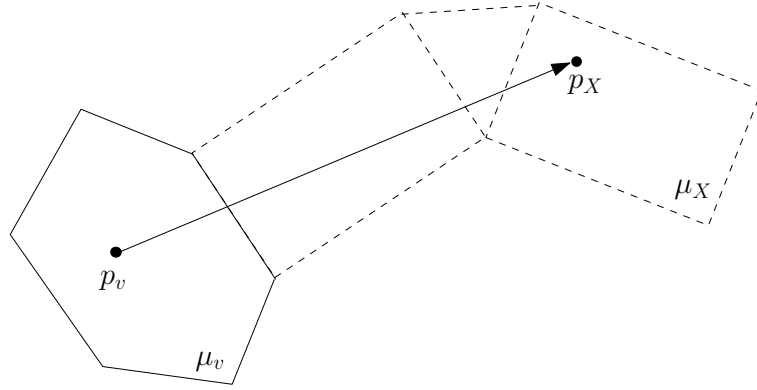


Figure 3.8: Finding the mixed cell  $\mu_X$  for a point  $p_X$  with another given point  $p_v$ , with known mixed cell  $\mu_v$ .

## Chapter 4

# General Skin Surface Mesh Deformation

General skin deformation in mesh is described in this chapter. As concluded from the previous chapters, the intermediate Voronoi complex remains unchanged during deformation, and the intermediate mixed cells move linearly. The intermediate mixed cells decompose the skin surface into sphere and hyperboloid patches. All these patches are free to deform according to the standard sphere and hyperboloid functions (Equation 2.1 and 2.2) with time  $t \in (0, 1)$ .

In this chapter, the details of deforming skin surface meshes is described. First, the different trajectories of the skin surface mesh points are introduced. Skin surface mesh points flow in their own mixed cells according to the dedicated trajectories. Second, there is a chance that a surface point hits the boundary of the mixed cell and escapes from its own mixed cell. The time  $t_p$  for each surface point  $p$  that hits its own mixed cell boundary is denoted by the escaping time. The deformation process is paused at time  $t_p$  in order to transfer  $p$  from

---

its own mixed cell to a new mixed cell. A new moving trajectory is assigned to  $p$  with a new  $t_p$  as the time when it reaches another mixed cell boundary. Third, the surface triangles might be twisted and the [U] and [L] conditions violated when the surface point moves. In addition to the original  $C$  and  $Q$  (Section 2.1.5), a loose  $Q_1$  which defines a refinable lower quality triangulation, is assigned. The triangulation is refined before it violates the [U] and [L] conditions with  $C$  and  $Q_1$  during deformation, in order to maintain the surface triangle quality. Finally, a solution to handle all possible topology changes during the skin surface deformation is given.

## 4.1 Surface Points Moving Trajectories

A point  $p$  on the skin surface at time  $t$  is denoted by  $p(t)$ . From a given starting point  $p(t_0)$ ,  $p(t)$  follows a direction and moves to a new position  $p(t_1)$ . The line connecting  $p(t_0)$  and  $p(t_1)$  is defined as the trajectory of  $p(t)$ . There are in total three types of trajectories: scaling, snapping and sticking. Under different situations, surface points following one trajectory may change to another trajectory as they cross the mixed cells or their mixed cells' weights approach zero. The time that  $p$  may change its trajectory before the deformation starts is calculated and inserted as schedules into the program. When the scheduled time is reached in the program, the deformation process is paused, the surface point trajectory is changed and the deformation process is resumed subsequently.

---

### 4.1.1 Trajectory I: Scaling

The first trajectory that scales  $p(t)$  towards or away from the mixed cell center  $z(t)$  is shown in Figure 4.1. According to the local coordinate system of a mixed cell, its surface patch within at time  $t_1$  is a scaling with a factor of  $\sqrt{\frac{w(t_1)}{w(t_0)}}$  of the patch at time  $t_0$ . Therefore, surface vertices only shrink or expand towards mixed cell centers with that factor. This implies that the benefit of the good quality of a triangle or an edge does not change if all its vertices belong to the same mixed cell. When a given surface vertex  $p(t_0)$  lies in a mixed cell  $\mu(t_0)$  at time  $t_0$ ,  $p(t)$  moves with a scaling factor within its mixed cell and a translation of its mixed cell center. The location of  $p(t_1)$  is

$$p(t_1) = z(t_1) + (p(t_0) - z_X(t_0)) \cdot \sqrt{\frac{w(t_1)}{w(t_0)}}. \quad (4.1)$$

The velocity vector at point  $p(t)$  is calculated, as it is crucial in the implementation of sticking trajectory and triangle quality maintenance. Given that  $p(t) = z(t_1) + \sqrt{\frac{w(t_1)}{w(t_0)}} * (p(t_0) - z_X(t_0))$ , the instant velocity of  $p(t)$  is divided into two parts. First, the mixed cell center  $z(t)$  moves with a velocity  $\frac{d}{dt}z(t) = z_X(1) - z_X(0)$ , which is a constant (Equation 3.2). The point  $p(t)$  moves together with its mixed cell. Second,  $p(t)' = p(t) - z(t)$  moves with a velocity of  $\frac{d}{dt}\sqrt{w(t)}$  in the local coordinate system of the mixed cell. By expanding  $w(t)$  in

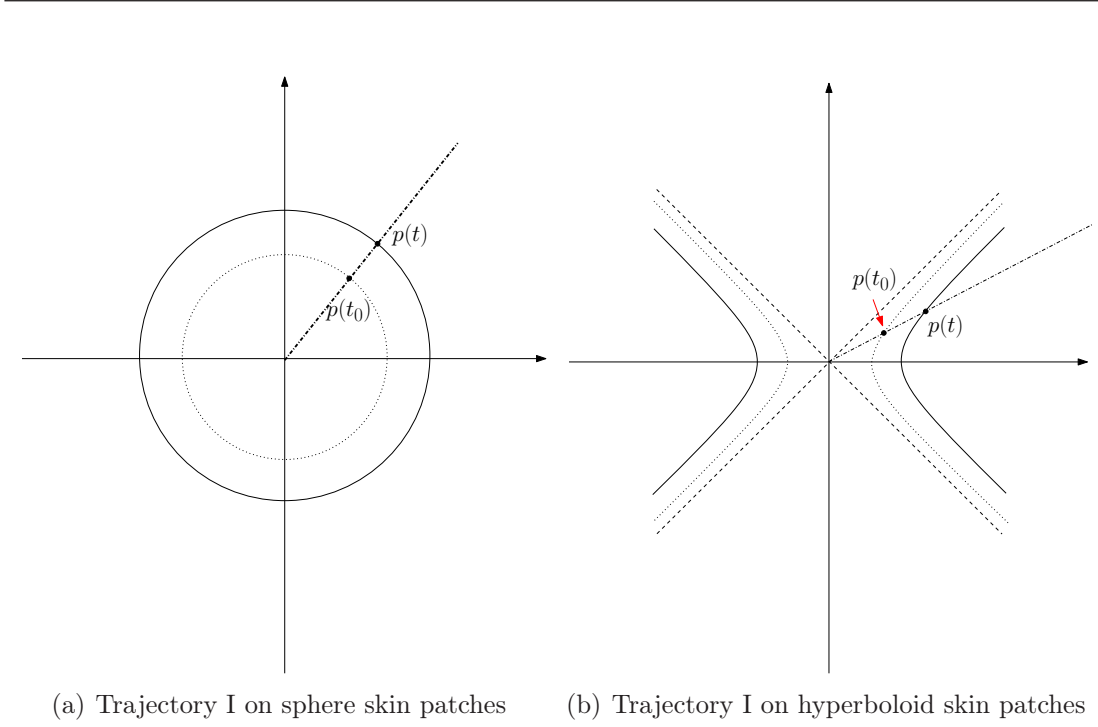


Figure 4.1: Trajectory I: moving away or towards the mixed cell centers.

terms of  $t$  with Equation 3.2 and 3.3

$$\begin{aligned}
w(t) = & (\|z(0) - z(1)\|^2 - \|z(1) - z(0) - z_i(1) + z_i(0)\|^2)t^2 + \\
& (w_i(1) - w_i(0) - \|z(0) - z(1)\|^2 - \\
& 2\langle z(1) - z(0) - z_i(1) + z_i(0), z(0) - z_i(0) \rangle)t + \\
& w_i(0) - \|z(0) - z_i(0)\|^2.
\end{aligned}$$

The velocity for  $p(t)$  in the local coordinate system of its mixed cell is

$$-\frac{\frac{d}{dt}w(t)}{2\sqrt{w(t)}} = -\frac{L(t)}{2\|p(t)\|},$$

where  $L(t) = \frac{d}{dt}w(t)$  is a linear equation in terms of  $t$ .

The velocity of  $p(t)$  in trajectory I is denoted by  $v_I(t)$ . The maximum speed

---

of  $p(t)$  is  $\|v_I(t)\| \leq \|z_X(1) - z_X(0)\| + \frac{\|L(t)\|}{2\|p(t)\|}$  and the two parts of  $v_I(t)$  are in the same direction. This loose upper bound of  $\|v_I(t)\|$  is useful in the triangle quality maintenance, which will be presented in the later section. The local velocity of  $p(t)$  within its mixed cell is denoted by  $v_I(t)' = -\frac{L(t)}{2\|p(t)\|}$ , which is important in trajectory III implementation.

#### 4.1.2 Trajectory II: Snapping

In hyperboloid skin patch mixed cells, such as Type (3,2,2), (3,1,1) and (2,2,1), surface points require another moving trajectory perpendicular to the asymptote in order to cross the asymptote, which is called snapping (Figure 4.2). Surface points on hyperboloid skin patches following trajectory I cannot cross the asymptote. Therefore, as the hyperboloid enters its hot ball, i.e.  $-2H^2 < w(t) < 2H^2$ , surface points on hyperboloid skin patches have to change to snapping trajectory, which is orthogonal to the asymptote boundary.

Assuming that the snapping direction is a normalized vector  $\hat{n}$ , and when  $p(t_0)$  is given, the position for  $p(t)$  is calculated as  $p(t) = (p(t_0) - z(t_0)) + \hat{n} \cdot l + z(t)$ . The local coordinates of  $p(t)$  is denoted by  $p'(t) = p(t) - z(t)$  in its mixed cell. The unknown  $l$  is the Euclidian distance from  $p(t_0)'$  to  $p(t)'$ . By recalling that surface points on hyperboloid patches satisfy Equation 2.2 with  $w(t)$ , the equation for  $l$  is expressed as:

$$(p(t_0)'.x + \hat{n}.x \cdot l)^2 - (p(t_0)'.y + \hat{n}.y \cdot l)^2 + (p(t_0)'.z + \hat{n}.z \cdot l)^2 = \pm w(t).$$

By solving the above quadratic equation, the representation of  $l$  by  $t$ , namely,

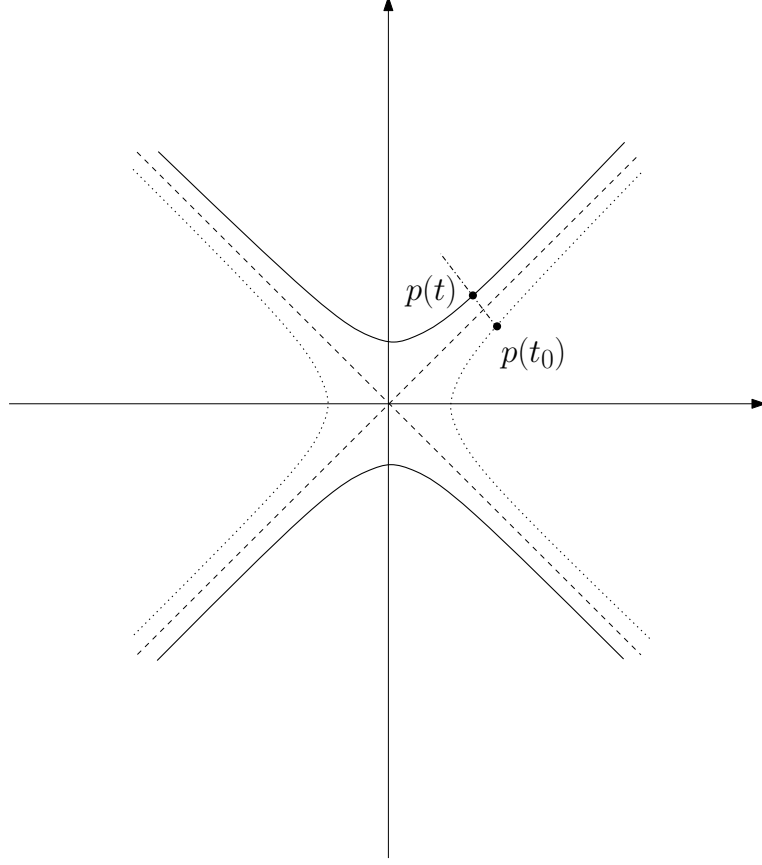


Figure 4.2: Surface points changing to snapping trajectory in order to cross the asymptote in hyperboloid skin patch mixed cells.

$l(t)$ , is obtained. The new position of  $p(t_1)$  is given by  $p(t_1) = (p(t_0) - z(t_0)) + \hat{n} \cdot l(t_1) + z(t_1)$ .

The local velocity relative to the mixed cell center for  $p(t)$  with trajectory II is  $v_{II}(t)' = \frac{d}{dt}l(t)$ . Although it is complicated to solve  $l(t)$ , it is straight forward to show that  $l(t) \leq \sqrt{w(t)}$ . Therefore  $\|v_{II}(t)'\| = \left\| \frac{d}{dt}l(t) \right\| \leq \left\| \frac{d}{dt}\sqrt{w(t)} \right\| = \|v_I(t)'\|$ . The upper bound of  $\|v_I(t)'\|$ , which is  $\frac{\|L(t)\|}{2\|p(t)\|}$ , is also a loose upper bound for  $\|v_{II}(t)'\|$ .

---

### 4.1.3 Trajectory III: Sticking

Surface points may be forced to ‘stick’ on the boundary of their mixed cells as they are unable to escape to a new mixed cell. This new trajectory is called sticking and is denoted by trajectory III. The sticking trajectory connects  $p(t_0)$  on the mixed cell boundary to  $m_{XY}(t_0)$  (Figure 4.3). Assuming that the surface point  $p$  from mixed cell  $\mu_X$  moves according to trajectory I or II to the boundary of  $\mu_X$  and another mixed cell  $\mu_Y$ , there are situations whereby  $p$  cannot enter either  $\mu_X$  or  $\mu_Y$ . These surface points are called *sticky points*. These sticky points then follow the sticking trajectory and move along on the boundary of  $\mu_X$  and  $\mu_Y$ .

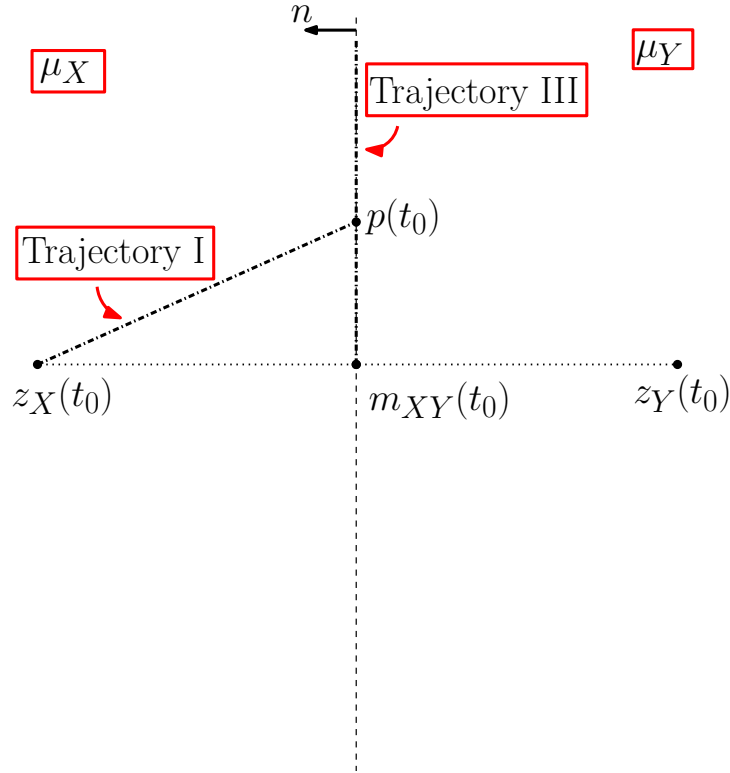


Figure 4.3: Sticky points move along the line connecting  $m_{XY}(t)$  and  $p(t_0)$ .

The algorithm to determine whether a transit point  $p$  from  $\mu_X$  to  $\mu_Y$  is a

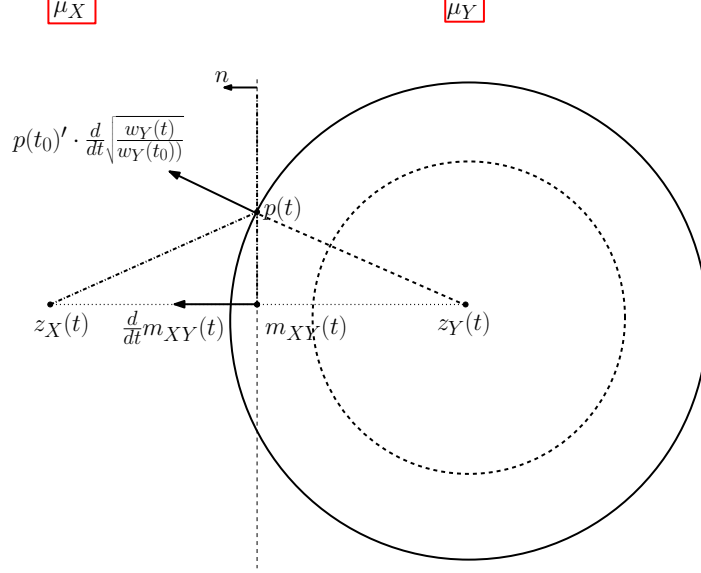


Figure 4.4: Sticky point  $p(t_0)$  in case 9.

sticky point is listed in Algorithm 1. Several notations are introduced before the algorithm is presented. The normal of the mixed cell boundary is denoted by  $n = z_X(t) - z_Y(t)$ , which does not change with varying  $t$ . The local position of  $p(t)$  in  $\mu_Y$  is  $p(t)' = p(t) - z_Y(t)$ . Similarly,  $m_{XY}(t)' = m_{XY}(t) - z_Y(t)$  is the local position of  $m_{XY}(t)$  in  $\mu_Y$ . The differentiation of  $m_{XY}(t)'$  is the local velocity of  $m_{XY}(t)$ , namely  $v'_{m_{XY}} = \frac{d}{dt}m_{XY}(t)'$ .

All cases in Algorithm 1 are shown as flow charts in Figure 4.5 and 4.6. First, the surface patch type of the destination mixed cell  $\mu_Y$  is tested. It is a simpler case if  $\mu_Y$  is a sphere patch mixed cell. There is a need to switch between trajectory I and trajectory II for hyperboloid mixed cells, depending on the metamorphosing status of  $\mu_Y$ . Second, the location of  $z_Y$  in the two halfspaces of  $h_{XY}(t_0)$ , which is distinguished by  $n$ , is checked (see the definition of halfspace in Section 3.3). Third, differentiation of  $w_Y(t)$  is performed in order to obtain the expansion (to determine whether it is expanding or shrinking) of  $\mu_Y$ . Finally,

---

the direction of  $v'_{m_{XY}}$  with respect to  $n$  is checked. For example, in Figure 4.4, it is assumed that  $\mu_Y$  is a sphere patch mixed cell,  $z_Y$  is in the opposite direction of  $n$  and  $v'_{m_{XY}}$  is in the same direction as  $n$ . If the sphere patch equation in  $\mu_Y$  is shrinking, it is clearly a sticky case for  $p(t)$ . Otherwise, a comparison of the magnitude of  $v_I(t_0)'$  and  $v'_{m_{XY}}$  in the direction of  $n$  is made. It is still a sticky case if the instant speed of  $p(t_0)'$  is faster than the instant speed of  $m'_{XY}$  in the direction of  $n$ .

The calculation of the position of  $p(t_1)$  in trajectory III is similar to that of trajectory II. However,  $p(t_0)'$  is replaced with  $m_{XY}(t_0)'$ , because  $p(t)$  has to stick on the boundary of  $m_{XY}(t)$  at all times. Hence,  $p(t) = m_{XY}(t_0)' + l \cdot \hat{n} + z_Y(t)$ , where  $\hat{n}$  is the normalized vector perpendicular to  $z_X(t) - z_Y(t)$ . The unknown  $l(t)$  is calculated in a similar manner as the calculation in trajectory II with Equation 2.1 and 2.2. The new position of  $p(t_1)$  is given by  $p(t_1) = m_{XY}(t_0)' + \hat{n} \cdot l(t_1) + z_Y(t_1)$ .

## 4.2 Topology Change Handling

Topology changes are important for the preservation of the homeomorphism between the mesh and the skin. In GSD, there are only two types of topology changes, namely

- creating or destroying a sphere,
- changing from a one-sheeted to a two-sheeted hyperboloid, or vice versa,

which are handled automatically.

---

**Algorithm 1** TestSticky( $p(t_0)$ )

---

```
1: if  $\langle m_{XY}(t_0)', n \rangle < 0$ . then
2:   if sphere/hyperboloid patch expands then
3:     if  $\langle v_{m_{XY}}(t_0)', n \rangle < 0$  and  $|\langle v(t_0)', n \rangle| < |\langle v_{m_{XY}}(t_0)', n \rangle|$  then
4:       return sticky.
5:     end if
6:   else
7:     if  $(\langle v_{m_{XY}}(t_0)', n \rangle < 0)$  or  $(\langle v_{m_{XY}}(t_0)', n \rangle > 0$  and  $|\langle v(t_0)', n \rangle| >$ 
       $|\langle v_{m_{XY}}(t_0)', n \rangle|)$  then
8:       return sticky.
9:     end if
10:  end if
11: else
12:  if sphere/hyperboloid patch expands then
13:    if  $(\langle v_{m_{XY}}(t_0)', n \rangle < 0)$  or  $(\langle v_{m_{XY}}(t_0)', n \rangle > 0$  and  $|\langle v(t_0)', n \rangle| >$ 
       $|\langle v_{m_{XY}}(t_0)', n \rangle|)$  then
14:      return sticky.
15:    end if
16:  else
17:    if  $\langle v_{m_{XY}}(t_0)', n \rangle < 0$  and  $|\langle v(t_0)', n \rangle| < |\langle v_{m_{XY}}(t_0)', n \rangle|$  then
18:      return sticky.
19:    end if
20:  end if
21: end if
22: return non-sticky.
```

---

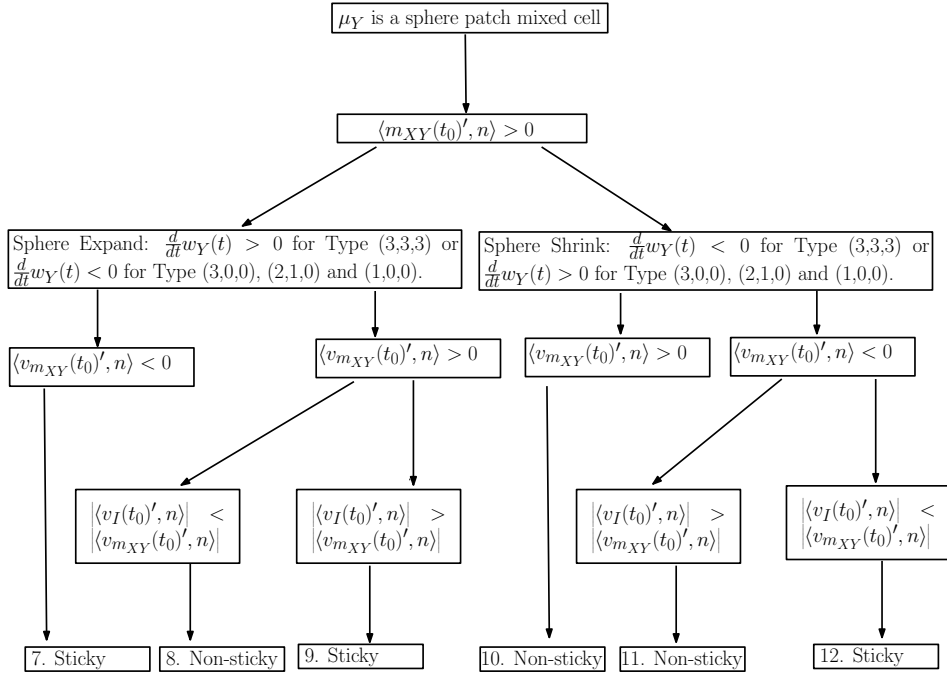
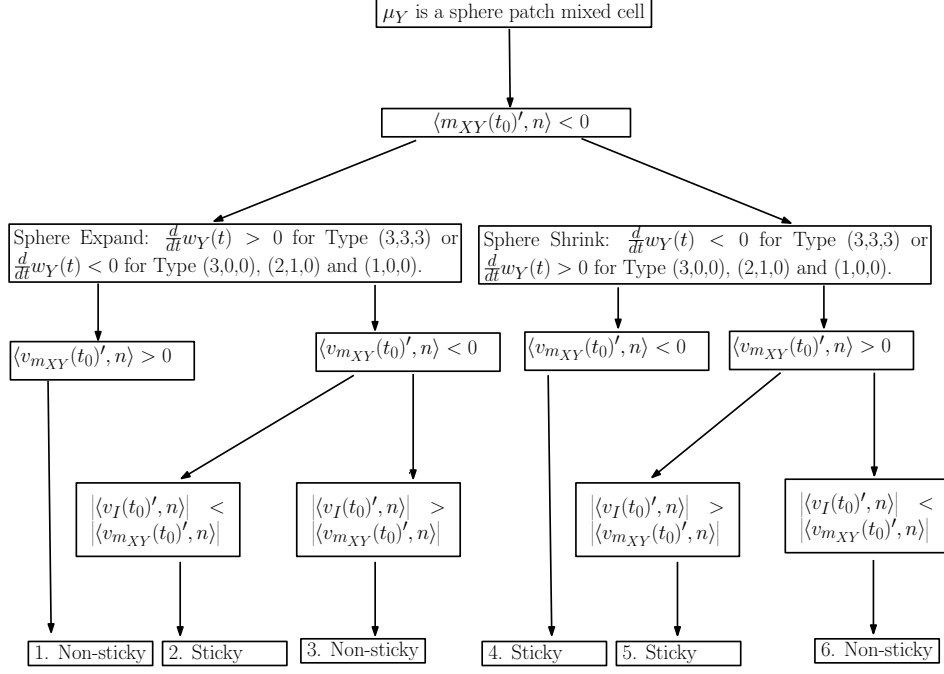


Figure 4.5: Cases for sticky test on  $p(t_0)$  crossing from mixed cell  $\mu_X$  to mixed cell  $\mu_Y$ , assuming that  $\mu_Y$  is a sphere skin patch mixed cell.

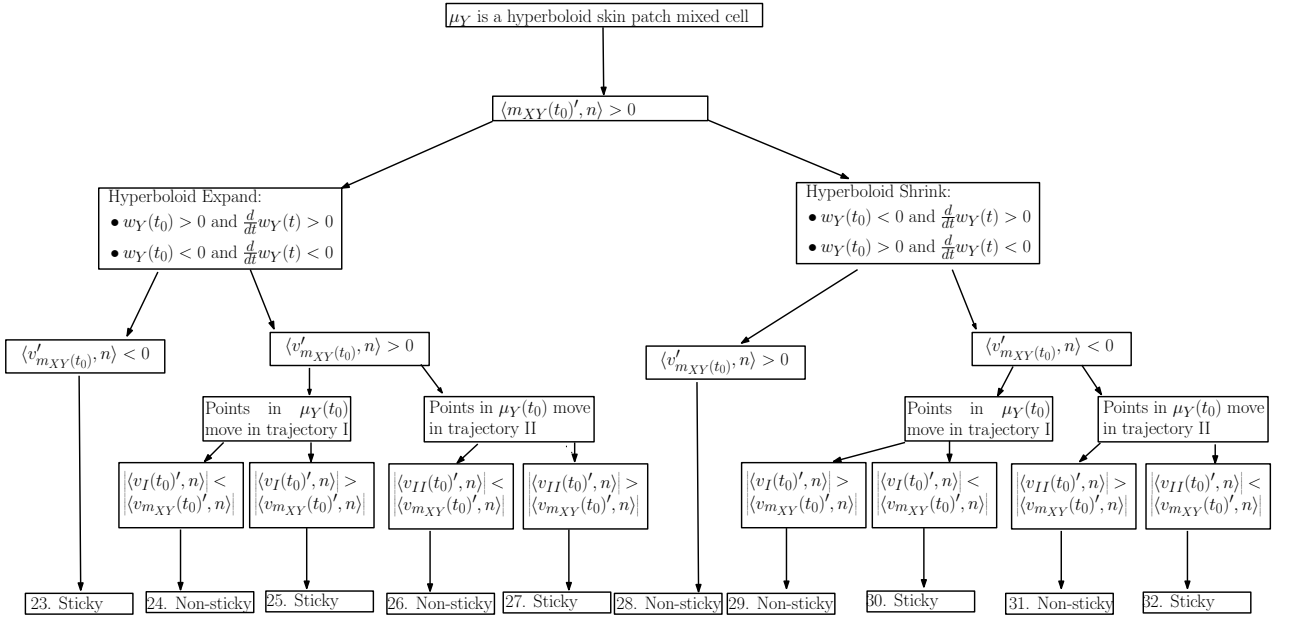
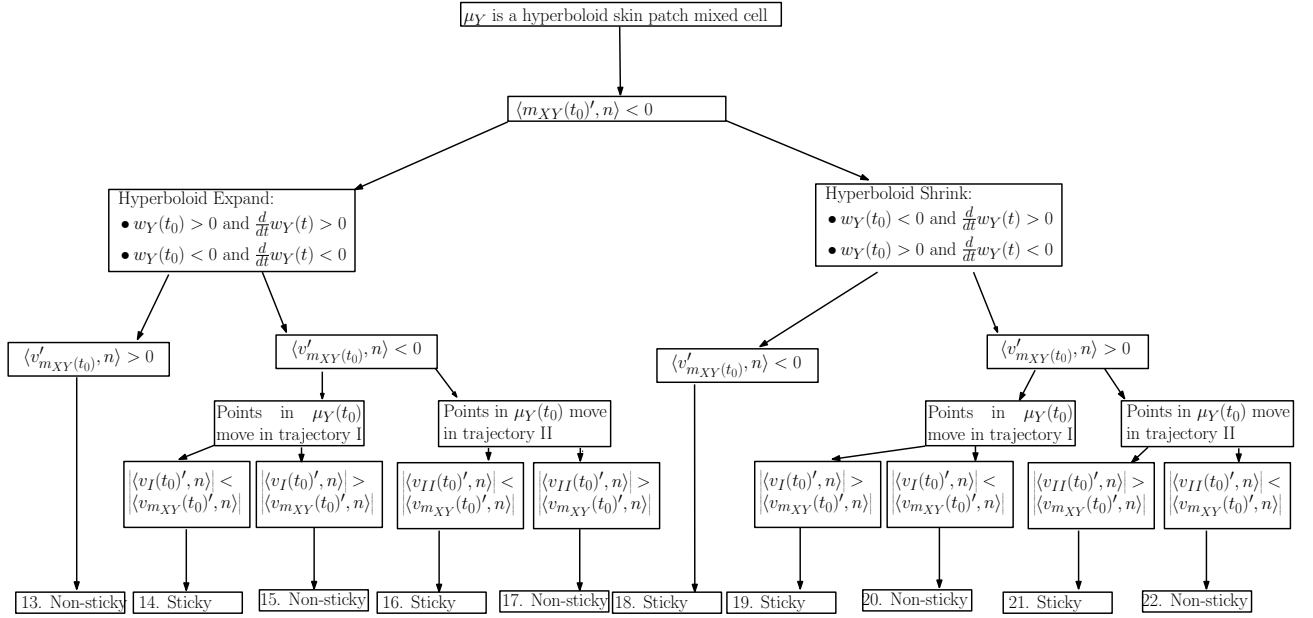


Figure 4.6: Cases for sticky test on  $p(t_0)$  crossing from mixed cell  $\mu_X$  to mixed cell  $\mu_Y$ , assuming that  $\mu_Y$  is a hyperboloid skin patch mixed cell.

---

The mixed cell weight  $w_X(t)$  is a quadratic function of  $t$ . Therefore, there are at most two topology changes that happen in one mixed cell. Moreover, a topology change operation is scheduled only if the mixed cell center is in the mixed cell. Both the center and the cell are moving and deforming, which is different from that of the growth model. This indicates that there is a possibility that the center is not in the mixed cell when  $w_X(t) = 0$ .

These topology changes in a mixed cell occur if and only if the size of a mixed cell reaches a value of zero and the metamorphosis,  $w_X(t') = 0$ , is scheduled at the time when this occurs. However, a special sampling is used within a *hot sphere* [16] at the time slightly before and after the time  $t'$ . The triangulation within the hot sphere is specially restructured and controlled from the time  $t' - \varepsilon_X$  to  $t' + \varepsilon_X$  in  $\mu_X(t)$  for a small value of  $\varepsilon_X$ .

#### 4.2.1 Hot Sphere Size

For every mixed cell  $\mu_X(t)$  that approaches a topology change, the absolute of the weight  $\|w_X(t)\|$  always decreases. The skin surface  $\mathcal{F}(t)$  approaches the mixed cell center  $z_X(t)$  and enters a hot sphere  $\mathbb{R}_H^3$  with center at  $z_X(t)$  and radius  $H$ . The skin patches in  $\mu_X(t)$  is denoted by  $\mathcal{F}_{\mu_X}(t)$ . For each topology change, the hot sphere size  $H$  is computed in advance to guarantee

$$\mathcal{F}_{\mu_X}(t) \cap \mathbb{R}_H^3 = \mathcal{F} \cap \mathbb{R}_H^3.$$

The skin surface inside the hot sphere is called the hot portion of  $\mathcal{F}$ , which is denoted by  $\mathcal{F}_H$ , is usually difficult to triangulate. Although it is straight-forward to create or delete mesh spheres in the hot spheres, it is not so for hyperboloids.

---

$\epsilon$ -sampling has to be given up for the construction of special structures in hot portions on hyperboloid patches (Section 4.2.2), in order to change from two-sheeted hyperboloids to one sheeted hyperboloids, and vice versa.

### 4.2.2 Special Sampling in Hot Sphere

Special sampling in the hot sphere aids in the topology change from two-sheeted hyperboloids to one-sheeted hyperboloids, and vice versa. When the case of a two-sheeted hyperboloid in the mixed cell  $\mu_X(t)$  entering the hot sphere at time  $t_0$  is considered, the weight of  $\mu_X(t)$ ,  $w_X(t_0)$ , is  $-2H^2$ . The topology change happens at time  $t_1$ , i.e.  $w_X(t_1) = 0$ . A double-cap is constructed at time  $t$  in between  $t_0$  and  $t_1$  in order for the crossing at the asymptote to happen, with  $w_X(t) = -2H^2h^2$  (Figure 4.7). The constant  $0 < h < 1$  has a reasonable value of 0.98 with  $C = 0.08$  and  $Q = 1.65$  [16].

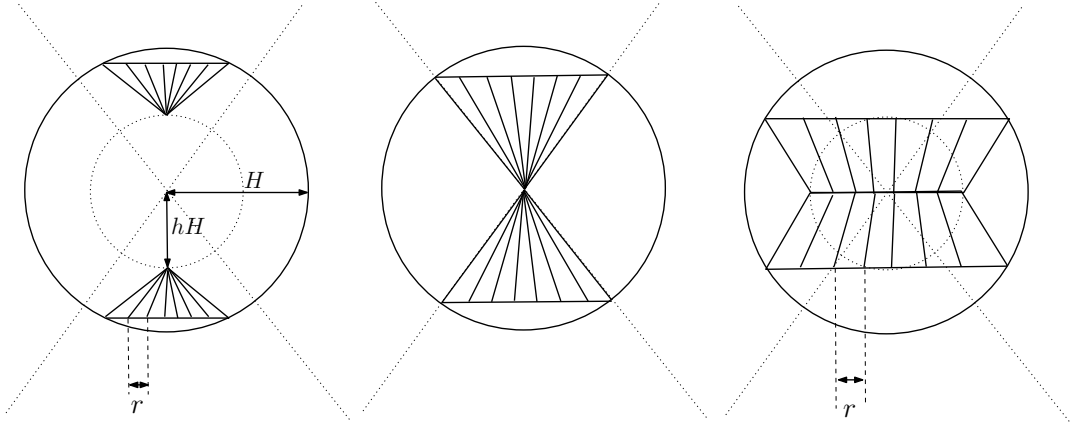


Figure 4.7: Topology change from left to right: a special sampling strategy from a two-sheeted hyperboloid to a one-sheeted hyperboloid. Similarly, the reverse is true for topology change from right to left.

The case of one sheet of the hyperboloid at time  $t$  as shown in the left figure of Figure 4.7 is considered, whereby a special point is added to the intersection of

---

the hyperboloid and the dotted sphere. The intersection between the hot sphere surface and one sheet of the hyperboloid is a circle. Let  $a_0, a_1, \dots, a_{m-1}$  be the vertices of a regular polygon with  $m$  number of sides along the circle. These points are mirrored across the symmetry plane of the hyperboloid in order to obtain the same points on the other sheet.

During the topology change from a two-sheeted hyperboloid to a one-sheeted hyperboloid, the edge length of the regular polygon with  $m$  number of sides,  $r$ , increases (Figure 4.7). It decreases when the change is reversed. In this project, the  $r$  value for the topology change from a two-sheeted hyperboloid to a one-sheeted hyperboloid is chosen as half of the minimum edge length which satisfies the [L] condition. In the reversed case of topology change from a one-sheeted hyperboloid to a two-sheeted hyperboloid, the value of  $r$  is double the edge length of the smallest edge which satisfies the [L] condition.

### 4.3 Surface Point Scheduling for Changing Status

There are in total five states for each surface point: creation, trajectory I, trajectory II, trajectory III and deletion. During the deformation process, all surface points have a *scheduling time* which indicates the time to change the status. When the program reaches the scheduling time, the whole deformation process is paused while waiting for the particular surface point  $p$  to change its status. Subsequently, the program is resumed and a new schedule time is assigned to  $p$ .

The following are involved in the surface point scheduling calculation:

- *Escaping time scheduling.* Escaping time scheduling provides the time when the surface point  $p$  touches its mixed cell boundary. When the escaping

---

time is reached in the program, the surface point  $p$  is transferred to another mixed cell and its trajectory is changed.

- *Metamorphing time scheduling.* Metamorphosis occurs when the weight of mixed cells Type (3,2,2), (3,1,1) and (2,2,1) reaches zero. When the hyperboloid patches in mixed cells of Type (3,2,2), (3,1,1) and (2,2,1) enter the hot sphere, the trajectory of the surface point in these mixed cells is changed to Trajectory II in order for the point to cross the asymptote.
- *Sphere Creation or Deletion scheduling.* A mesh sphere is created or deleted when the topology is changed for mixed cells of Type (3,3,3), (2,1,0) and (3,0,0).

### 4.3.1 Escaping Time Scheduling

In general, the escaping time of  $p(t)$  in mixed cell  $\mu_X$  is calculated as:

$$\langle p(t), n_{XY} \rangle = \langle m_{XY}(t), n_{XY} \rangle, \quad (4.2)$$

where  $n_{XY} = z_Y(t) - z_X(t)$  is a constant vector perpendicular to the boundary of  $\mu_X$  and  $\mu_Y$ . The Equation 4.2 is computed for all possible neighborhood mixed cell  $\mu_Y$  of  $\mu_X$ . The minimum value of  $t$  is the escaping time for  $p(t)$ .

At the escaping time  $t$ , the surface point  $p$  may proceed to one of the following:

- transfer to another mixed cell (trajectory I  $\rightarrow$  trajectory I, trajectory II  $\rightarrow$  trajectory II, trajectory I  $\rightarrow$  trajectory II or trajectory II  $\rightarrow$  trajectory I),
- move along the mixed cell boundary (trajectory I  $\rightarrow$  trajectory III, trajectory II  $\rightarrow$  trajectory III or trajectory III  $\rightarrow$  trajectory III).

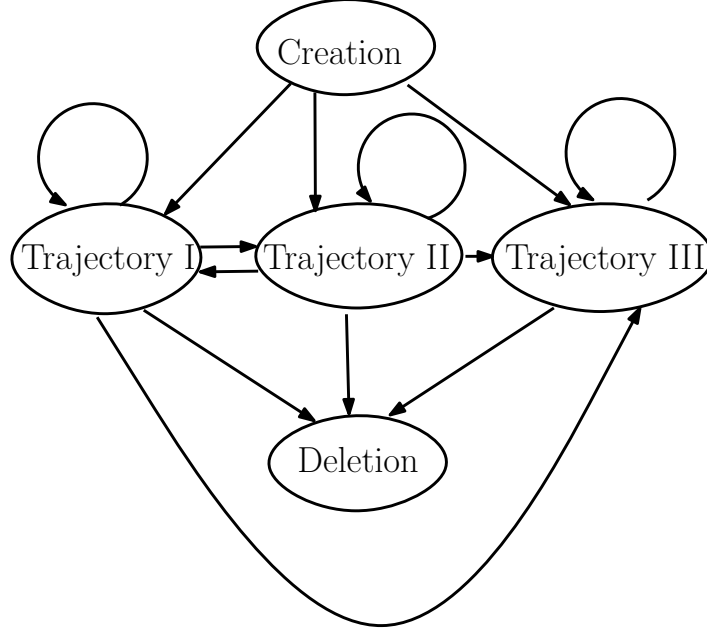


Figure 4.8: Transfer of states for a surface point. Arrows indicate the direction of transfer between the states.

#### 4.3.1.1 Trajectory I Escaping Time Calculation

The computation for escaping time is straight-forward in trajectory I. Equation 4.1 for  $p(t)$  is substituted into Equation 4.2 and the quadratic equation of  $t$  is solved. There are at most two escaping times for every surface point that belongs to each mixed cell. However, it is not always the case that both roots from the quadratic equation are valid. Some roots is possibly imaginary from  $\langle p(t), n_{XY} \rangle = -\langle m_{XY}(t), n_{XY} \rangle$ , since the Equation 4.2 is squared during the calculation.

---

#### 4.3.1.2 Trajectory II Escaping Time Calculation

The escaping time of  $p(t)$  for trajectory II is calculated as  $p(t)' = p(t_0)' + l \cdot \hat{n}$ . By combining with Equation 4.2,

$$\langle p(t_0)' + l \cdot \hat{n}, n_{XY} \rangle = \langle m_{XY}(t), n_{XY} \rangle. \quad (4.3)$$

By combining with the hyperboloid or sphere equation in  $\mu_X$ , both the unknowns  $l$  and escaping time  $t$  in trajectory II are solved.

#### 4.3.1.3 Trajectory III Escaping Time Calculation

The escaping time in trajectory III is similar to that of trajectory II. The only difference is during sticking point scheduling,  $p(t)$  moves along with the boundary center  $m_{XY}(t)$ , implying that  $p(t)' = m_{XY}(t)' + l \cdot \hat{n}$ .

By combining with the hyperboloid or sphere equation in  $\mu_X$ ,  $l$  is eliminated and the escaping time  $t$  in trajectory III is solved.

#### 4.3.1.4 Special Situation: Escaping Degenerate Mixed Cells

A surface point  $p(t)$  may escape to a degenerate mixed cell with zero volume (see Section 3.3.1). Because no point lies inside a degenerate mixed cell, the surface point  $p(t)$  escapes immediately after it enters the mixed cell. The Equation 4.2 does not work in this case, since the floating point numbers are not capable to identify the entering time and escaping time.

The task becomes identifying the escaping neighborhood mixed cell from the degenerate mixed cell without computing the escaping time. Under GGP As-

---

sumption, there are three types of mixed cells possibly degenerate, namely, Type (3,2,2), (3,1,1) and (2,2,1). For a degenerate mixed cell of Type (3,2,2), it has two neighborhood mixed cells of Type (3,3,3). The surface point  $p(t)$ , which enters the degenerate mixed cell, must come from one mixed cell and escape to the other mixed cell of Type (3,3,3). Similarly, for a mixed cell of Type (3,1,1) or (2,2,1), there are two neighborhood mixed cells sharing the same Voronoi center. The surface point  $p(t)$  from one such neighborhood mixed cell will escape to the other neighborhood mixed cell immediately.

### 4.3.2 Metamorphosis Scheduling

In mixed cells of Type (3,2,2), (3,1,1) and (2,2,1), the surface points on hyperboloid patches have to switch from trajectory I to trajectory II, when the hyperboloid patches enter the hot sphere:

$$-2H^2 < w_X(t) < 2H^2. \quad (4.4)$$

Equation 4.4 provides another scheduling time for surface points to change trajectories.

### 4.3.3 Make or Delete Sphere Scheduling

In contrast to mixed cells of Type (3,2,2), (3,1,1) and (2,2,1), Equation 4.4 provides a scheduling time to create or delete a mesh sphere in mixed cells of Type (3,3,3), (2,1,0) and (3,1,0). All surface points on the mesh sphere are created or deleted accordingly.

---

In conclusion, the minimum value of the results of all the above scheduling calculations provides the escaping time for a surface point  $p$  to change its status as shown in Figure 4.8.

## 4.4 Mesh Refinement Maintaining Triangle Quality

During deformation, homeomorphism between the mesh and the skin surface is guaranteed by its triangle quality, which satisfies Conditions [U] and [L] [16].

The quality of triangles and edges may become bad as surface points change their positions. Triangles and edges are categorized into three zones based on Conditions [U] and [L]: *acceptable zone*, *buffer zone*, and *unacceptable zone*. The main purpose of the maintenance is to check and refine an element, i.e. an edge or a triangle, before it enters the unacceptable zone. An element is in the buffer zone if it is not acceptable but satisfies the Conditions [U] and [L] with a lower quality  $Q_1$ . In this zone, the element can be refined into acceptable elements by edge contraction or point insertion. The refinement methods include edge contractions (Figure 4.10) and triangle circum-center insertion (Figure 4.9).

All triangles and edges are required to be refined before they enter the unacceptable zone. Since all surface point trajectories and the time to change the trajectory can be computed, the triangle refinement scheduling is calculated before the triangles and edges enter the unacceptable zone, and the schedules are inserted into the program. When the triangle scheduling time is reached in the program, the deformation process is paused, the triangle is refined and then the

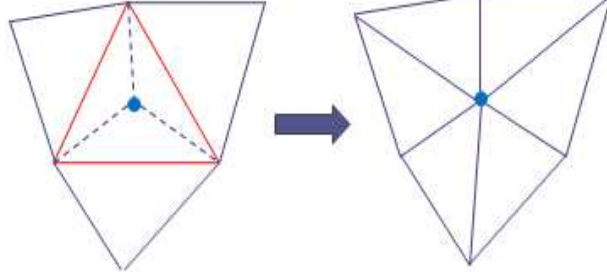


Figure 4.9: Triangle refinement when condition [U] is violated. A point near the circumcenter of the triangle (in red) is inserted. Edges are flipped to maintain the Delaunay property.

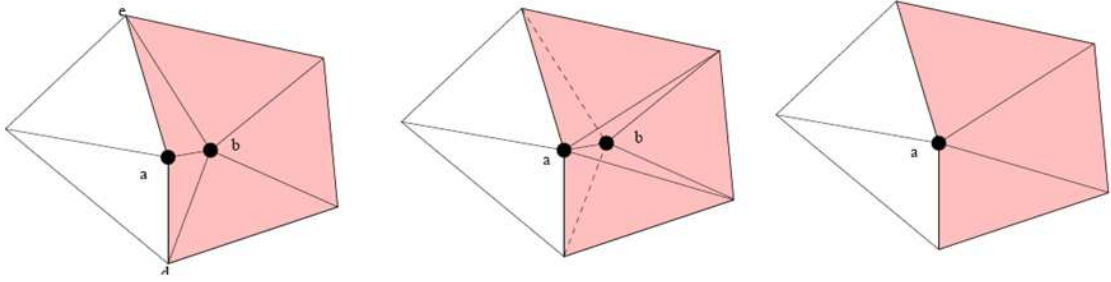


Figure 4.10: Edge contraction when condition [L] is violated. The edges are flipped until vertex  $b$  is in degree 3. This prevents surface points from crossing edges after  $ab$  is removed.

deformation process is resumed.

When given an element that satisfies the two Conditions at time  $t_0$ , it is proven that there is a time increment  $\Delta t$  whereby the element will not become unacceptable at time  $t_0 + \Delta t$ , which is the triangle refinement scheduling time. An element is not checked if all its vertices belong to the same mixed cell and it is moving in trajectory I. Every element is scaling towards or away from the mixed cell center in trajectory I. Scheduling is unnecessary for these elements until one of their vertex exits the mixed cell or transfers to another trajectory.

---

#### 4.4.1 Scheduling Edge Update

Let  $p(t_0)q(t_0)$  be an edge at time  $t_0$  that satisfies the Condition [L]  $\|p(t_0) - q(t_0)\| \geq 2\frac{C}{Q_0}\varrho_{p(t_0)q(t_0)}$ . A time  $t_1 = t_0 + \Delta t$  is computed such that  $\|p(t_1) - q(t_1)\| \geq 2\frac{C}{Q_1}\varrho_{p(t_1)q(t_1)}$  when  $p(t_0)$  and  $q(t_0)$  move to  $p(t_1)$  and  $q(t_1)$  respectively. Let  $\|v_{pq}(t)\|$  be the maximum speed of  $p(t)$  and  $q(t)$ ;  $\|p(t_1) - q(t_1)\|$  has a lower bound of:

$$\|p(t_1) - q(t_1)\| \geq \|p(t_0) - q(t_0)\| - 2 \int_{t_0}^{t_1} \|v_{pq}(t)\| dt. \quad (4.5)$$

As  $C$ ,  $Q_0$  and  $Q_1$  are all constants, the local length scale  $\varrho_{p(t_0)q(t_0)}$  has the largest increment while  $v_{pq}(t)$  follows trajectory I and moves away from the mixed cell center:

$$\varrho_{p(t_1)q(t_1)} \leq \varrho_{p(t_0)q(t_0)} + \int_{t_0}^{t_1} \|v_{pq}(t)\| dt. \quad (4.6)$$

By combining Equations 4.5 and 4.6,  $t_1$  is solved in

$$\|p(t_0) - q(t_0)\| - 2 \int_{t_0}^{t_1} \|v_{pq}(t)\| dt \geq 2\frac{C}{Q_1}(\varrho_{p(t_0)q(t_0)} + \int_{t_0}^{t_1} \|v_{pq}(t)\| dt), \quad (4.7)$$

as a loose bound for  $\|p(t_1) - q(t_1)\| \geq 2\frac{C}{Q_1}\varrho_{p(t_1)q(t_1)}$ . It is guaranteed that the edge  $p(t)q(t)$  with in the time interval of  $(t_0, t_1)$  is longer than  $2\frac{C}{Q_1}\varrho_{p(t)q(t)}$ .

#### 4.4.2 Scheduling Triangle Update

Let  $R(t_0)$  be the circumradius of the triangle  $p(t_0)q(t_0)u(t_0)$  that satisfies Condition [U] at time  $t_0$ , namely,  $\varrho_{p(t_0)q(t_0)u(t_0)} \cdot CQ_0 > R(t_0)$ . The value  $t_1 = t_0 + \Delta t$  is computed such that the relaxed condition  $\varrho_{p(t_1)q(t_1)u(t_1)} \cdot CQ_1 > R_1$  is satis-

---

fied. Let  $u(t_0)'$  be the orthogonal projection of  $u(t_0)$  onto  $p(t_0)q(t_0)$  and  $H(t_0) = \|u(t_0) - u(t_0)'\|$ . The circumradius is  $R(t_0) = \frac{\|p(t_0) - u(t_0)\| \cdot \|q(t_0) - u(t_0)\|}{2H(t_0)}$  (Figure 4.11).

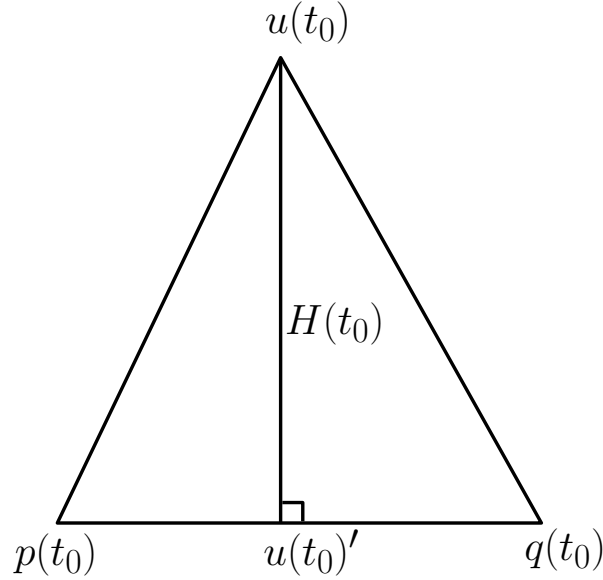


Figure 4.11: Labeling of triangle elements at time  $t_0$  for triangle scheduling.

The maximal speed of  $p(t)$ ,  $q(t)$  and  $u(t)$  is denoted by  $\|v_{pqu}(t)\|$ . The largest increment of the circumradius  $R(t_0)$  occurs when  $\|p(t_0) - u(t_0)\|$  and  $\|q(t_0) - u(t_0)\|$  have the largest increment and  $H(t_0)$  has the largest reduction:

$$R(t_1) \leq \frac{(\|p(t_0) - u(t_0)\| + 2 \int_{t_0}^{t_1} \|v_{pqu}(t)\|) \cdot (\|q(t_0) - u(t_0)\| + 2 \int_{t_0}^{t_1} \|v_{pqu}(t)\|)}{2(H(t_0) - 2 \int_{t_0}^{t_1} \|v_{pqu}(t)\|)}.$$

In a similar manner as edge scheduling, the local length scale  $\varrho_{p(t_0)q(t_0)u(t_0)}$  has the largest increment while  $v_{pqu}(t)$  follows trajectory I and moves towards the mixed cell center, which is given by:

$$\varrho_{p(t_1)q(t_1)u(t_1)} \geq \varrho_{p(t_0)q(t_0)u(t_0)} - \int_{t_0}^{t_1} \|v_{pqu}(t)\|$$

---

The loose bound for  $\varrho_{p(t_1)q(t_1)u(t_1)} \cdot CQ_1 > R_1$  is computed by

$$\frac{(\|p(t_0) - u(t_0)\| + 2 \int_{t_0}^{t_1} \|v_{pqu}(t)\|) \cdot (\|q(t_0) - u(t_0)\| + 2 \int_{t_0}^{t_1} \|v_{pqu}(t)\|)}{2(H(t_0) - 2 \int_{t_0}^{t_1} \|v_{pqu}(t)\|)} \leq \varrho_{p(t_0)q(t_0)u(t_0)} - \int_{t_0}^{t_1} \|v_{pqu}(t)\|. \quad (4.8)$$

The solution  $t_1$  of Equation 4.8 guarantees that the triangle quality,  $\varrho_{p(t_1)q(t_1)u(t_1)} \cdot CQ_1 > R_1$ , is always satisfied in the time period  $(t_0, t_1)$ .

## 4.5 Combine Point Scheduling with Triangle Scheduling

Finally, all scheduled times for all skin surface mesh points, edges and triangles are combined and inserted into a priority queue. The time  $t$  is given the first priority. A function for minimum time extraction is called each time to provide the next schedule time. In surface point scheduling, the program is paused, the point is transferred to its new status, the new scheduling time is calculated and then the new schedule is inserted into the priority queue. In edge or triangle scheduling, the triangle is refined, the new scheduling time is calculated and then the new schedule is inserted into the priority queue as well.

---

The edge and triangle scheduling times are based on the current status of the contributing vertices. The minimum scheduling time among all contributing vertices of the edge or triangle determines the maximum time period for the next edge or triangle scheduling. It is possible that several edge or triangle schedules have the same time  $t$  as some surface point schedules. In these situations, priority is given to the surface point schedules instead of the edge or triangle schedules. The reason for this second order of priority is because the new scheduling time after triangle refinement for edge or triangle schedules depends on the new scheduling time of the contributing vertices.

## 4.6 Summary

In this chapter, the ways to generate the skin surface mesh at time  $t_1$  when given the mesh at time  $t_0$  is described. First, after each surface point is located in its mixed cell, the initial moving trajectory for that point is determined. The skin surface mesh is free to deform according to the sphere and hyperboloid equations until one of its mesh point hits its escaping time. The status of that particular surface point is transferred and the deformation in its new mixed cell is resumed. Second, surface point movements cause short edges and large triangles. The time for each edge to become too short and thus violate the [L] condition, is scheduled by edge scheduling. Similarly, the time for each triangle to become too large and thus violate the [U] condition, is scheduled by triangle scheduling. When either the edge scheduling time or triangle scheduling time is reached, edge contraction or circum-center insertion is performed immediately in order to maintain the triangulation quality. Finally, the times for metamorphosis and

---

creation/deletion spheres are scheduled. All four types of topology changes are automatically handled when the time for topology change is reached.

## Chapter 5

# Simplified General Skin

## Deformation

New difficulties arise when the number of input weighted points is extremely large and the intermediate complex structure becomes overwhelmingly dense. Given two input weighted point sets with  $m$  and  $n$  points, the intermediate complex is constructed by  $mn$  weighted points. Due to the complexity of Delaunay triangulation, the overall complexity is  $O(m^2n^2)$ . For example, a mannequin model skin ( $B_0$ ) with 12,680 weighted points is deformed to a fist model skin ( $B_1$ ) with 5,461 weighted points as shown in Figure 5.1. The intermediate complex consists of 69 million weighted points. The complexity of this intermediate complex is the square of 69 million, which is impossible for a 32-bit machine to handle. Moreover, the deformation of skin surface over such a dense intermediate complex results in computationally intensive *scheduling* [16; 21; 30], which is directly associated with the overall computation time of GSD.

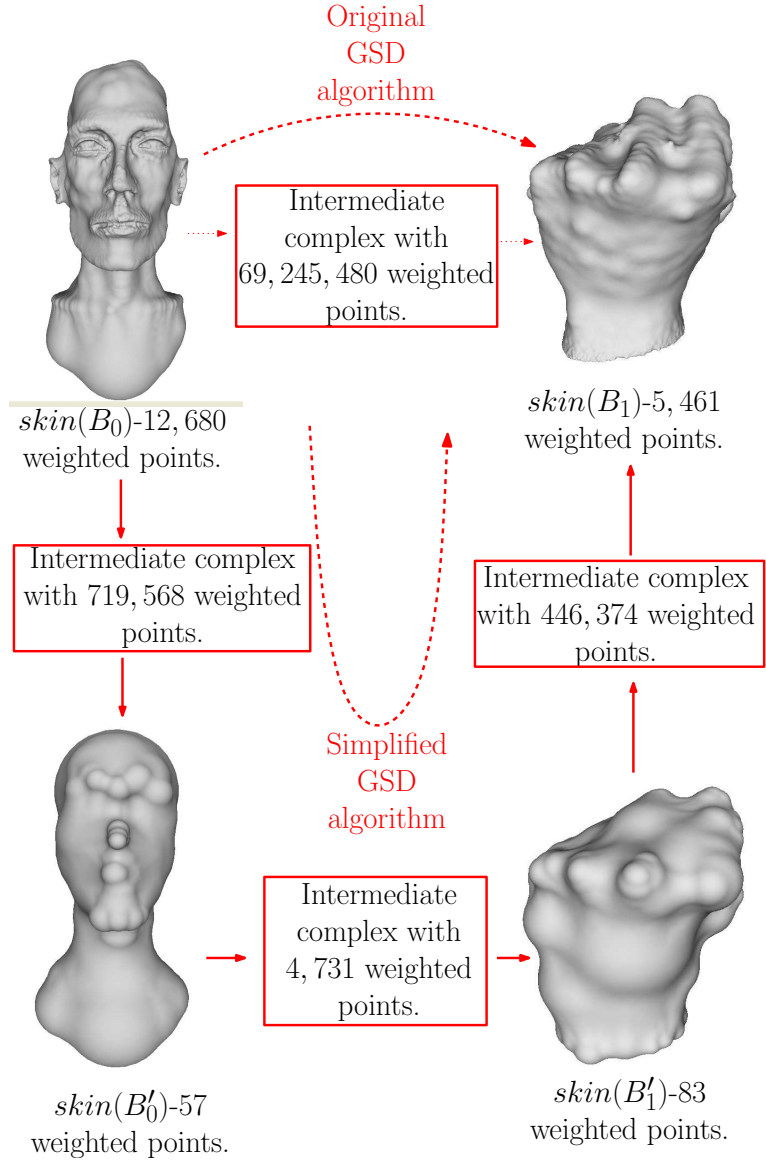


Figure 5.1: The new SGSD algorithm partitions the original GSD into three distinct deformations: two simplification deformations and one GSD. The intermediate complex in the SGSD algorithm is much simpler as compared to that of the original GSD algorithm. (The original mannequin and fist union of ball models are generated by power crust by Amenta et al. [52].)

---

Simplification of skin models reduces the complexity of intermediate complexes and hence is of importance when the number of weighted points is large [72]. A new simplified GSD (SGSD) algorithm is introduced to reduce the complexity of the intermediate complex during deformation. As shown in Figure 5.1, the weighted point set  $B_0$  is simplified by reducing the weighted points and adjusting the weights of the remaining weighted points in order to create a new weighted point set  $B'_0$  with  $m'$  weighted points, where  $m'$  is defined by the user. The weighted point set  $B'_1$  is obtained by simplifying  $B_1$  in the same manner. Deformation is then performed from skin( $B_0$ ) to skin( $B'_0$ ) (Figure 5.2) and from skin( $B'_1$ ) to skin( $B_1$ ). The new intermediate complex between skin( $B'_0$ ) and skin( $B'_1$ ) is reduced from 69 million (by GSD) to 43 thousand weighted points. Thus, deformation from the mannequin model to the fist model becomes feasible. The overall complexity of SGSD (see the arrows in Figure 5.1), when  $m'$  and  $n'$  are user defined constants, is reduced to  $O(m'^2 + n'^2)$ . Although three deformations are performed instead of one and three intermediate complexes are generated instead of one, the SGSD algorithm is much more efficient than the original GSD algorithm.

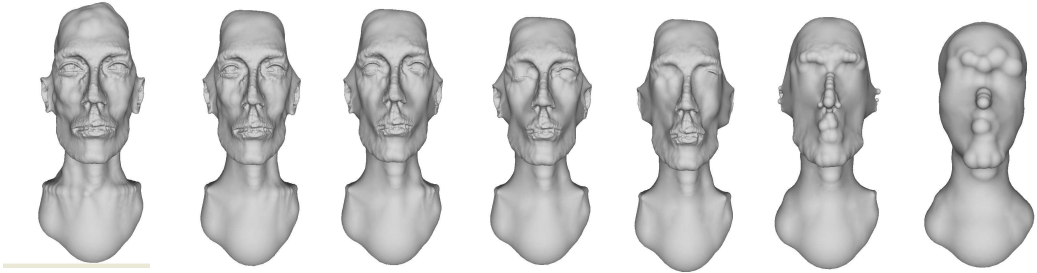


Figure 5.2: The simplification process in the SGSD algorithm.

---

## 5.1 Simplified General Skin Deformation Algorithm

A SGSD algorithm for improving the efficiency and reducing the complexity of intermediate complex in GSD is proposed. When given two skin surfaces,  $\text{skin}(B_0)$  with  $m$  weighted points, and  $\text{skin}(B_1)$  with  $n$  weighted points, the overall complexity of GSD is improved from  $O(m^2n^2)$  to  $O(m^2 + n^2)$  in three steps:

1. ***Simplify weighted point set.*** The input weighted point sets are considered as unions of balls. The unions of balls are simplified with the number of simplified weighted points as defined by the user. Skin surfaces for both the original weighted point sets and the simplified weighted point sets are generated as inputs for the SGSD algorithm.
2. ***Deform skin surfaces.*** The whole SGSD process is divided into three parts. The three parts are named as Deformation I, II and III, for deformation from  $\text{skin}(B_0)$  to  $\text{skin}(B'_0)$ , from  $\text{skin}(B'_0)$  to  $\text{skin}(B'_1)$  and from  $\text{skin}(B'_1)$  to  $\text{skin}(B_1)$  respectively.
3. ***Connect the three deformation processes.*** The whole deformation process is constructed by connecting Deformation I, II and III. The result from the previous deformation is used for later deformations like Deformation II and III. The mesh points therefore move continuously between different deformations with this connection.

## 5.2 Simplification of Weighted Point Set

Weighted point sets, which are considered as unions of balls, are simplified by two steps: removal and recovering volume. The simplification mentioned in this

---

thesis is different from the geometry simplification in two aspects,

1. Typical geometry simplification usually refers to *level of detail* [2; 3; 40].

The purpose of the simplification is to use less geometry elements and keep most visible features of the original model. However, the main goal of the simplification in this thesis is to mimic an intermediate shape in the original deformation process. Thus, sharp features (small balls on the surface) may be removed since they are usually covered by large balls during the original GSD process.

2. A way of recovering volume is allowed after the removal step.

Many research works have been done to simplify unions of balls. One method by Attali and Montanvert examines the skeleton of the model and removes branch  $T$  with two criteria [4],

1. The homotopy class of the shape is preserved when  $T$  is removed.
2.  $T$  is not relevant with a fixed threshold, which is calculated considering both surfacial skeleton and wireframe skeleton.

This method preserves the characteristics of the original shape. However, only one threshold is not enough for both surfacial skeleton and wireframe skeleton. Another method developed by Tam and Heidrich [61] builds the medial axis of the model and prunes the medial axis iteratively. This is a similar work to the work by Attali and Montanvert with more sophisticated parameters used in each iteration instead of a single threshold. In 2009, Bordignon proposed a new method to simplify a union of balls called Scale-Space [9]. The algorithm smoothes the shape and preserves the volume by analyzing the scale-space properties of bi-dimensional

---

curvature motion. It has nice properties such as avoiding disconnection and self-intersection. However, the algorithm tends to simplify all shapes into a single sphere.

The simplification method, which is used in SGSD, summarizes all previous works stated above. Two variables are introduced, the feature variable ( $K$ ) and the density variable ( $J$ ). The original union of balls is simplified by deleting balls which are,

- Not affecting the homotopy group of the object,
- Not relevant according to certain criterion which takes in consideration of  $K$  and  $J$ , and
- Contributing small volume to the union, and the volume can be recovered by enlarging the rest of balls.

After the deletion, we enlarge the weight of each ball from  $w_i$  to  $w_i + \alpha$  with a value of  $\alpha$  that gives the least difference of the volumes between the union of the original and simplified set of balls. This increment by  $\alpha$  does not change the underlying Voronoi complex. Koehl’s code [33] is used in the volume calculations.

The ultimate goal is to simplify a given large set of balls into a set with  $m'$  (defined by the user) number of balls with a similar shape. We also provide suggestions for reasonable small number  $m'$  by suggesting the numbers for the two variables,  $K$  and  $J$  (See Section 5.2.2 for details). The word ‘reasonable’ is defined as not setting a tight bound on the difference between the volumes of the two unions of balls. This is because the simplified version is one of the intermediate shapes in the deformation sequence.

---

The ideal result of the simplified union of balls provides a similar intermediate weighted point set in the original GSD process. Experimental result shows that, with identical homology classes and tolerable volume difference, the deformation process with SGSD is similar to the one with original GSD (Section 7.2).

The calculation of the volume and volume difference of the union of balls is introduced in section 5.2.1. The simplification method consists of 2 steps. First, the relevant spheres are removed, as described in section 5.2.2. Second, the balls are enlarged in the resultant model in an effort to recover some lost volume, as described in section 5.2.3.

### **5.2.1 Volume and Volume Difference of Union of Balls**

The volume of the union of balls can be calculated in a few ways. One method is implemented in the program called Volume [13]. The purpose of the program is to break the balls into pyramids with flat bottoms and pyramids with rounded bottoms, which are called restrictions. The volume of the union of balls is the sum of all these restrictions.

Another method is the inclusion-exclusion formula. A program that employs this method is written by Patrice Koehl named AlphaBalls, which is written in Fortran and C [33]. While most of the code is written in Fortran for historical reasons, this software has proven greatly useful and fast. For an input of about 20,000 balls, the software takes about 5 seconds for computation. The software also computes the volume and surface area that each sphere contributes to the model. This information is useful for deciding which spheres can be deleted or replaced.

---

After we simplify a union of balls  $A$  to a smaller set  $A'$ , we do not simply take the volume difference between  $A$  and  $A'$  to evaluate the shape difference. Instead, we take the symmetric volume difference as in Equation 5.1.

$$V = \frac{(A \cup A') - (A \cap A')}{A \cup A'} \cdot 100\%. \quad (5.1)$$

The percentage shown in  $V$  evaluates the shape difference between  $A$  and  $A'$ .

### 5.2.2 Removal

Removal is further broken down into 2 steps. First, spheres that contribute much surface to the model are removed. This is because, when the alpha value changes, the shape of the model is alter significantly as the surface spheres change their volume. The followings describe the criteria for the removal of a sphere  $b_i$ :

- The sphere  $b_i$  contributes some surface to the model.
- The volume that  $b_i$  contributes is less than the feature variable ( $K$ ) multiply the average contribution of all the spheres.
- Topology is preserved.

The second step involves preventing large spheres in the model that contribute some surface to the model from being removed. Interestingly, this step tends to remove smaller features, after which, a smaller set of spheres remains. However, since geometric models are dealt with in the current work, this remaining set will likely be very densely packed. The sphere  $b_i$  is removed, beginning with the smallest spheres, according to the following criteria:

- 
- The sphere  $b_i$  contributes less than the density variable ( $J$ ) of its volume to the model.
  - Topology is preserved.

This step is necessary because if the effects are approximated with the enlargement of all the spheres, these spheres will probably be swallowed up by their neighbors.

### 5.2.3 Recovering Volume

The volume is recovered by enlarging the balls in the resultant model according to Equation 5.2.

$$r_i(\alpha) = \sqrt{r_i^2 + \alpha^2} \quad (5.2)$$

All the balls in the resultant model increase their radius  $r_i$  with the same  $\alpha$  value. The benefits for this are:

1. to avoid skinny intermediate mixed cells [16], and
2. to reduce the complexity of the intermediate complex.

For example, in Figure 5.3, a skin surface skin ( $B_0$ ) with four weighted points is deformed to its simplified model skin skin ( $B'_0$ ) by deleting one weighted point in  $B_0$ . The intermediate complex is built based on the super-imposition of the two Voronoi complexes of  $B_0$  and  $B'_0$ , namely  $V(B(t))$ . The complexity of the intermediate complex is equivalent to the number of simplices in  $V(B(t))$ . In Figure 5.3, there are four intermediate Voronoi vertices in the intermediate complex. However, if different weight changes in  $B'_0$  are allowed, there are more than

---

four intermediate Voronoi vertices after the super-imposition (Figure 5.4). In addition, very slim intermediate Delaunay cells are created in the intermediate Delaunay triangulation.

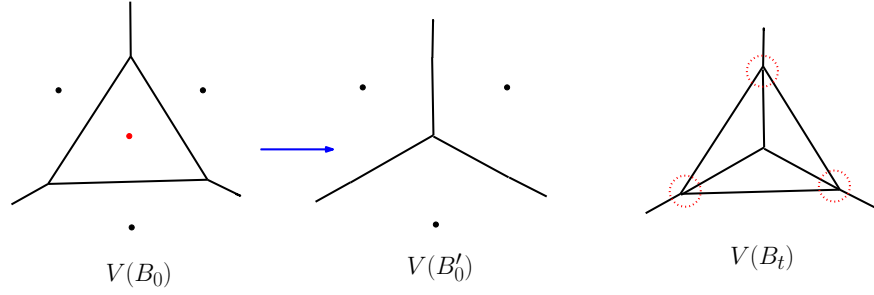


Figure 5.3: A four weighted point set  $B_0$  deforms to a subset  $B'_0$  by deleting one of its weighted point. The Voronoi diagram of  $B_0$ ,  $B'_0$  and the intermediate weighted point set  $B(t)$  are shown. The vertices circled in  $V(B(t))$  are intermediate Voronoi vertices with trapezoid intermediate Delaunay cells.

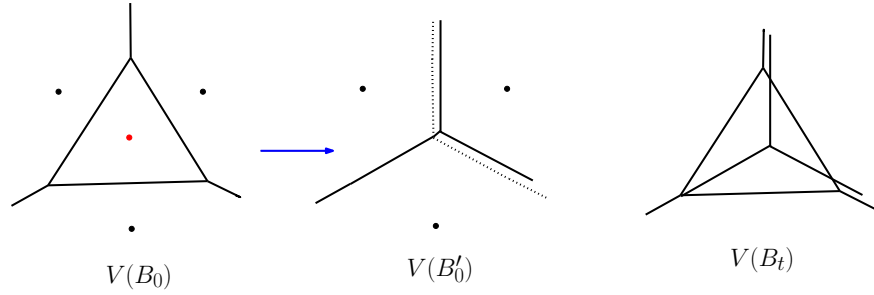


Figure 5.4: An  $\alpha$  value is given to the weighted point on the upper-right corner. There are 6 intermediate Voronoi vertices after super-imposition.

The minimum  $\alpha$  value is calculated by the bisection method. The symmetric volume difference is expected to have one global minimum, where only positive  $\alpha$  values are considered (Figure 5.5).

Figure 5.6 shows the simplified unions of balls of the mannequin and fist models used in this thesis. For different values of  $K$  and  $J$ , it shows the number of balls reduced and the symmetric volume difference. In both cases, as the values

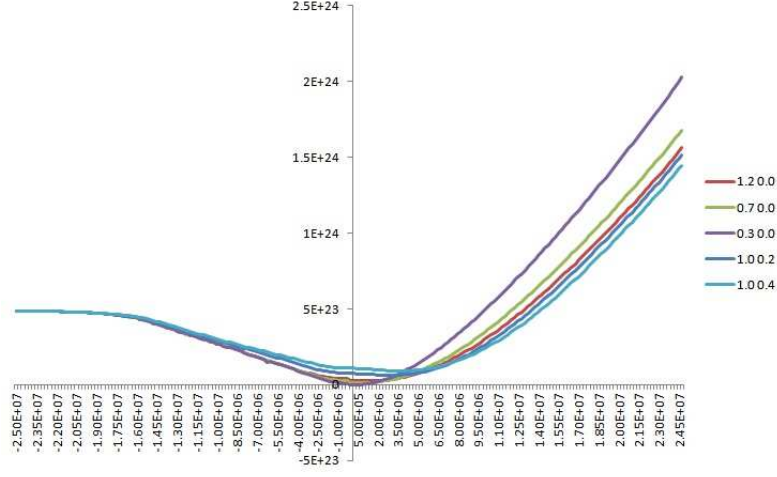


Figure 5.5: Symmetric volume difference as alpha is varied for the mannequin model

of the variables decrease, the proportion of balls removed becomes smaller. The symmetric volume difference as a consequence becomes smaller as well. Koehl's code is used for all the above computations [33].

One bottleneck is the check for introduction of holes. This check takes long as it involves a search through the Delaunay triangulation. The impact of this check is apparent in both the feature decreasing step and the density decreasing step. Moreover, because of this check, fewer balls are removed in the feature decreasing step, leaving more balls to be processed by the density reducing step. The worst combination for performance is a low feature variable, a non-zero density variable and a large model with the check. In those cases, the removal process runs for a few days. Therefore, we recommend a range of values to the user. For the feature variable ( $K$ ), a value greater than 0.5 and less than 1.5 is recommended. For the density valuable ( $J$ ), a value between 0 and 0.5 is suggested. In Chapter 7, we show experiments comparing the original GSD versus the SGSD. Similar

---

deformation processes can be obtained with proper choice of variables  $K$  and  $J$ .

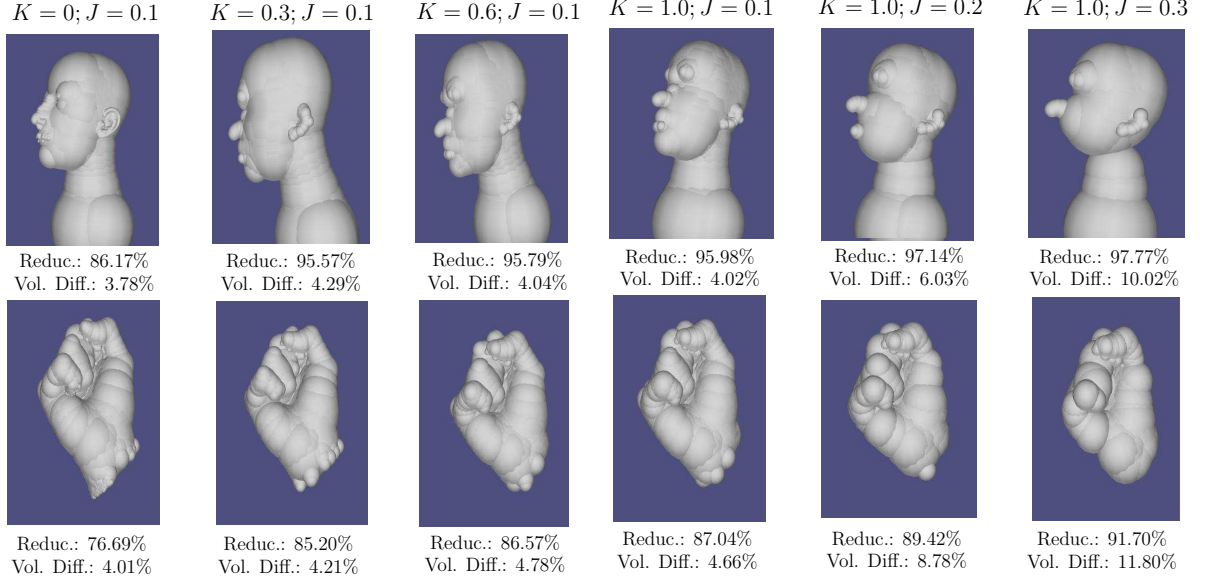


Figure 5.6: The simplified unions of balls with different  $K$  and  $J$  values. The reduction percentage and the symmetric volume difference are shown.

### 5.3 Degeneracies in Intermediate Delaunay and Voronoi Complexes

Degeneracies arise in the intermediate complex of Deformation I and III. The GGP Assumption is again unavoidably violated when  $B_0$  and  $B'_0$  share the same point locations. First, the degenerate case in  $\mathbb{R}^2$  is illustrated. In  $\mathbb{R}^2$ , after the Voronoi complexes in Deformation I and III are super-imposed, there are Voronoi edges from different Voronoi complexes that intersect at one of their end-points (see Figure 5.3 as an example). This degenerate case is represented by Type (1,0,0). The corresponding intermediate Delaunay cell is a trapezoid which is obtained by deforming a Delaunay edge to form a Delaunay triangle (Figure 5.7).

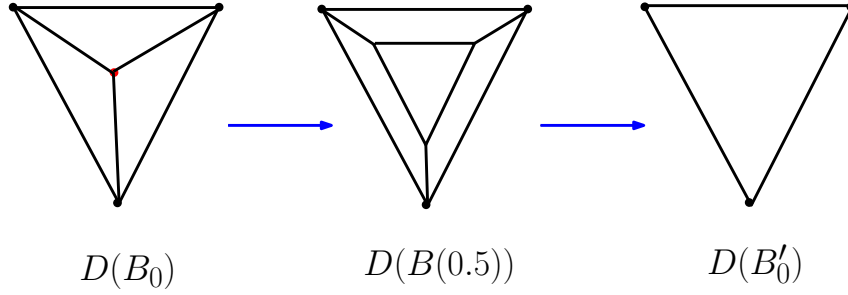


Figure 5.7: The intermediate Delaunay triangulation at  $t = 0.5$  (middle) for the original simplification (shown in Figure 5.3). The intermediate Delaunay triangulation involves three trapezoids and one triangle

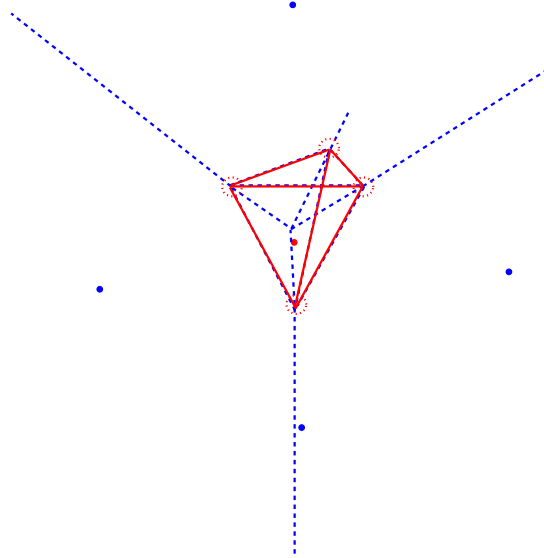


Figure 5.8: An intermediate Voronoi complex after deleting one point in a five weighted point set. The deleted point is shown in red. There are six degenerate Voronoi edges (in red), and four degenerate Voronoi vertices (circled in red)

In  $\mathbb{R}^3$ , there are two more degenerate cases in the intermediate complex. First, it is indicated by Type (2,1,1) that a Voronoi face and a Voronoi edge from different Voronoi complexes intersect at one boundary edge of the Voronoi face. For example, as shown in Figure 5.8, one weighted point from a five weighted point set is deleted. There are six such degenerate intermediate Voronoi edges,

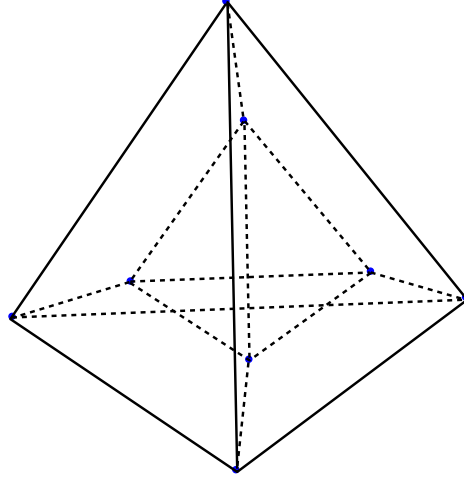


Figure 5.9: The intermediate Delaunay complex that corresponds to the intermediate Voronoi complex in Figure 5.8 at time  $t = 0.5$ . The intermediate Delaunay triangulation consists of four frustums and one tetrahedron.

Table 5.1: All possible combinations of intermediate Voronoi cells and Delaunay Cells. The new degenerate cases are shown in bold.

Type	Voronoi	Delaunay	Patch
(3,3,3)	Polyhedron	Vertex	Sphere
(3,2,2)	Polygon	Edge	Hyperboloid
(3,1,1)	Edge	Triangle	Hyperboloid
(3,0,0)	Vertex	Tetrahedron	Sphere
(2,2,1)	Edge	Parallelogram	Hyperboloid
(2,1,0)	Vertex	Triangle Prism	Sphere
<b>(2,1,1)</b>	<b>Edge</b>	<b>Trapezoid</b>	<b>Hyperboloid</b>
<b>(1,0,0)</b>	<b>Vertex</b>	<b>Tri. Frustum</b>	<b>Sphere</b>

each of which is shared by four Voronoi regions. The corresponding intermediate Delaunay cells are trapezoids. Second, Type (1,0,0) represents an intersection between a Voronoi edge and a Voronoi point from different Voronoi complexes. The corresponding intermediate Delaunay cell is a frustum which deforms from a triangle to a tetrahedron (Figure 5.9). Table 3.1 is expanded to include the two degeneracies in SGSD (Table 5.1).

## 5.4 Escaping Time in New Degenerate Mixed Cells

New degenerate mixed cells are found in mixed cell Type  $(2,1,1)$  and  $(1,0,0)$ . In Figure 5.10, a triangle frustum is interpolated from a triangle to a tetrahedron. At time  $t = 0$ , the single mixed cell of Type  $(1,0,0)$  and four mixed cells of Type  $(2,1,1)$  have volume of zero. Therefore, all these five mixed cells are degenerate at  $t = 0$ .

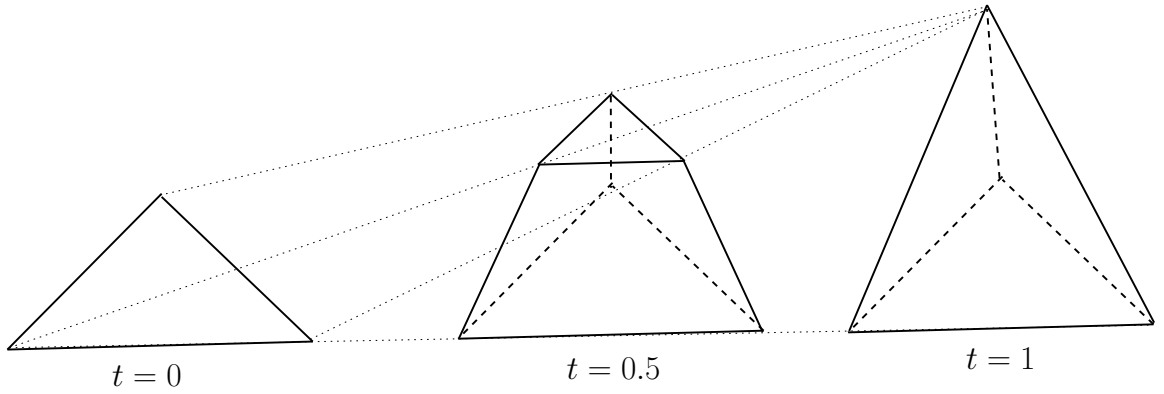


Figure 5.10: A triangle frustum is interpolated from a triangle to a tetrahedron.

The Equation 4.2 fails to handle the escaping time calculation for these new degenerate mixed cells because of the numerical errors in floating point numbers. The numerical errors exist while the time  $t$  is greater but close to 0. The front view of the mixed cells of Type  $(2,1,1)$  and  $(1,0,0)$  for the triangle frustum in Figure 5.10 with time  $t$  approaching to 0, is drawn in Figure 5.11.

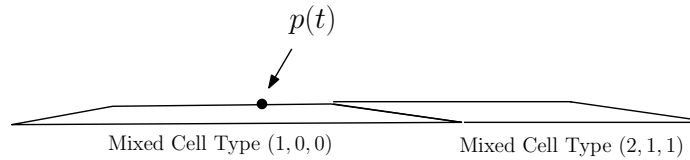


Figure 5.11: The front view of mixed cells Type  $(2,1,1)$  and  $(1,0,0)$  for the triangle frustum in Figure 5.10 with time  $t$  approaching to 0 .

---

Since the escaping time is not possible to be computed in degenerate mixed cell, the surface point  $p(t)$  escapes immediately after it enters the degenerate mixed cell. However, different from the case in Section 4.3.1.4, all mixed cells, except the one that  $p(t)$  comes from, are possible destinations, since the time  $t$  is not exactly 0. There are several heuristics available to choose the most reasonable escaping mixed cell. The best solution is to choose the boundary nearest to  $p(t)$  along its trajectory with opposite normal, comparing to the normal of the boundary which the surface point  $p(t)$  comes from.

## 5.5 Complexity Analysis

The complexity for the new SGSD reaches  $O(m^2 + n^2)$ , assuming that the user inputs  $m'$  and  $n'$  are small as compared with  $m$  and  $n$ . The number of weighted points in Deformation I is  $mm'$  and the complexity can be treated as  $O(m)$  when  $m'$  is a small constant. The overall complexity is  $O(m^2) + O((m'n')^2) + O(n^2) = O(m^2 + n^2)$ , by treating Deformation III in the same manner.

The overall complexity can be verified by studying the intermediate Voronoi complexes. In Deformation I and III, the complexity is equal to that of one of the intermediate Voronoi complexes. Deformation I from  $\text{skin}(B_0)$  to  $\text{skin}(B'_0)$  is used to illustrate this point. When the two Voronoi complexes  $V_0$  and  $V'_0$  of  $B_0$  and  $B'_0$  are super-imposed, the extra Voronoi regions in the intermediate Voronoi complex are the cells in  $V'_0$  within the Voronoi cells of the deleted weighted points in  $V_0$ . In the worst case, each cell of the deleted points come into contact with all the cells in  $V'_0$  and each of these  $m - m'$  deleted points generate  $m'$  new cells. However, the cells of  $V'_0$  are far larger in size than those of the deleted points in

---

$V_0$ . The number of cells in  $V'_0$  in the cell of each of the deleted points in  $V_0$  are not proportional to  $m'$ , and it will not increase the complexity of the intermediate Voronoi complex to  $O(m^2m'^2)$ .

## 5.6 Conclusion

In this chapter, a new simplified general skin deformation algorithm is introduced to improve the efficiency of the original GSD algorithm. A solution to deform skin surfaces with huge weighted point sets, which is originally impossible for the GSD algorithm to handle, is provided. It is demonstrated that the complexity of the intermediate complex can be improved from the original  $O(m^2n^2)$  to  $O(m^2 + n^2)$ . This improvement reduces the number of intermediate mixed cells in the deformation and therefore reduces the number of schedules in the GSD algorithm and improves the overall running time.

In addition, a new aspect of GSD is investigated, which is to study the degeneracies where the GPA is unavoidably violated. In this chapter, a small portion of the degeneracies found during the simplification deformation process is discussed. More degeneracies occur in other situations, for example, in partial movements such as the bending of an elbow with only the lower part of the arm changing position. All possible shapes of intermediate Delaunay and Voronoi complexes in  $\mathbb{R}^2$  and  $\mathbb{R}^3$  are listed in Chapter 6.

## Chapter 6

# Special Deformation: Partial Movements

More degenerate cases are discovered when points from part of the weighted point set are allowed to change their positions. For example, more intermediate Delaunay types are required for a bending elbow with only the lower part of the arm changing position. In molecular dynamics, molecules change part of their shape, while the rest of their atoms remain unchanged. When this happens, the GGP Assumption is unavoidably violated and more degenerate cases will be created in the intermediate Voronoi complex because the following is allowed,

$$\dim(\nu_X \cap \nu_Y) > \dim(\nu_X) + \dim(\nu_Y) - d.$$

---

## 6.1 More Degenerate Types of Intermediate Complexes

In Figure 6.3, a ‘molecule movement’ with only one atom moving away from its original position in  $\mathbb{R}^2$  is demonstrated. In the figure, both  $\nu_X$  and  $\nu_Y$  are in dimension 1. However,  $\nu_{XY} = \nu_X \cap \nu_Y$  is also in dimension 1 and violates the GGP Assumption.

In  $\mathbb{R}^3$ , there are seven additional degenerate cases that violate the GGP Assumption, namely (2,2,2), (2,1,1), (2,0,0), (1,1,1), (1,1,0), (1,0,0) and (0,0,0) (Table 7.2).

Table 6.1: Additional types of intermediate Voronoi, Delaunay and mixed cells.

Type	Voronoi cells	Delaunay cells	Surface Patch
(2,2,2)	Polygon	Edge	Hyperboloid
(2,1,1)	Edge	Quadrangle or Pentagon	Hyperboloid
(1,1,1)	Edge	Polygon with 3 to 6 vertices	Hyperboloid
(1,1,0)	Vertex	Polyhedron with 7 to 9 vertices	Sphere
(2,0,0)	Vertex	Polyhedron with 6 to 7 vertices	Sphere
(1,0,0)	Vertex	Polyhedron with 6 or 11 vertices	Sphere
(0,0,0)	Vertex	Polyhedron with 4 to 12 vertices	Sphere

The shapes of Voronoi cells and Delaunay cells are shown using Equation 3.1. Two coplanar Voronoi polygons overlap with each other in Type (2,2,2), and the intersection is a polygon (Figure 6.1(a)). There are two Voronoi regions sharing this polygon. The two weighted points of these two regions form a convex hull that is a Delaunay edge. A Voronoi edge intersects a Voronoi face and forms another Voronoi edge in Type (2,1,1) (Figure 6.1(b)). As a result, the intermediate Voronoi edge is surrounded by four or five Voronoi regions, which

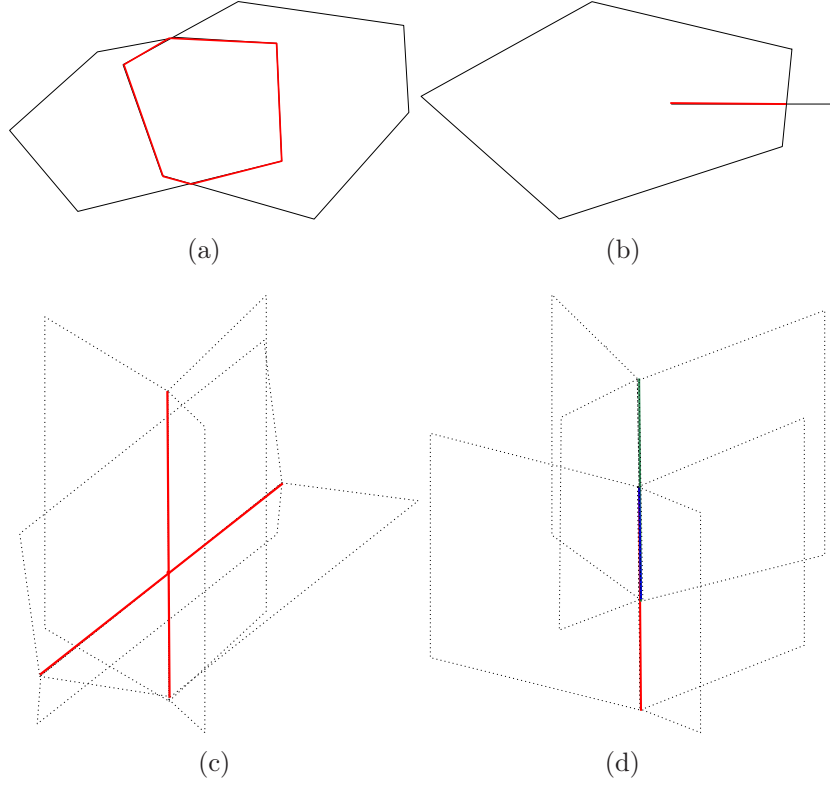


Figure 6.1: (a) Two Voronoi polygons overlap to form a new intermediate Voronoi polygon. (b) A Voronoi polygon intersects a Voronoi edge to form another Voronoi edge. (c) Two Voronoi edges intersect at an intermediate Voronoi vertex. (d) Two Voronoi edges overlap to form an intermediate Voronoi edge.

are contributed by four or five intermediate weighted points. The convex hull of these weighted point centers is a quadrangle or a pentagon. Two Voronoi edges intersect at a Voronoi vertex in Type (1,1,0), (Figure 6.1(c)). The Voronoi vertex is surrounded by seven, eight or nine Voronoi regions. The intermediate Delaunay cell is a polyhedron with seven to nine vertices. Two Voronoi edges overlap and form an intermediate Voronoi edge in Type (1,1,1) (Figure 6.1(d)). There are a minimum of three and a maximum of six Voronoi regions surrounding this intermediate Voronoi edge. The intermediate Delaunay cell is a polygon with three to six vertices. The intermediate Voronoi cell is always a vertex for Type

---

$(2,0,0)$ ,  $(1,0,0)$  and  $(0,0,0)$ . The intermediate Delaunay cells are obtained by counting the number of surrounding Voronoi regions.

Although the number of intermediate Voronoi cell types is increased, the types of skin patches remain the same as that of hyperboloids and spheres. The construction of intermediate skin surface mesh is similar to that of GSD and SGSD.

## 6.2 Conclusion

Degenerate intermediate Delaunay and Voronoi cells violating the GGP Assumption are dealt with, such that the GSD Algorithm can be applied to partial movements.

By relaxing the GGP Assumption, new types of degenerate Voronoi cells are formed and dealt with in order to enable the GSD Algorithm to perform partial motion modeling. Such modeling in a real time manner facilitate the investigation and execution of experiment simulations (see Figure 6.2). In this chapter, all the cases in  $\mathbb{R}^2$  and  $\mathbb{R}^3$  are enumerated.

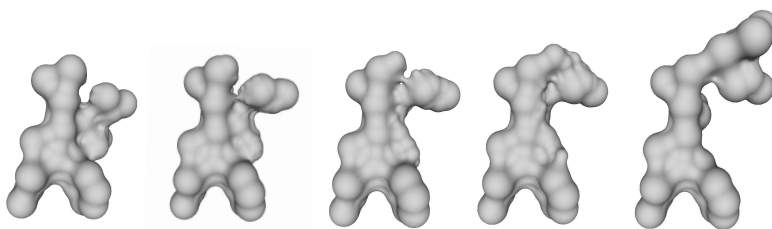


Figure 6.2: An example of partial molecular movement.

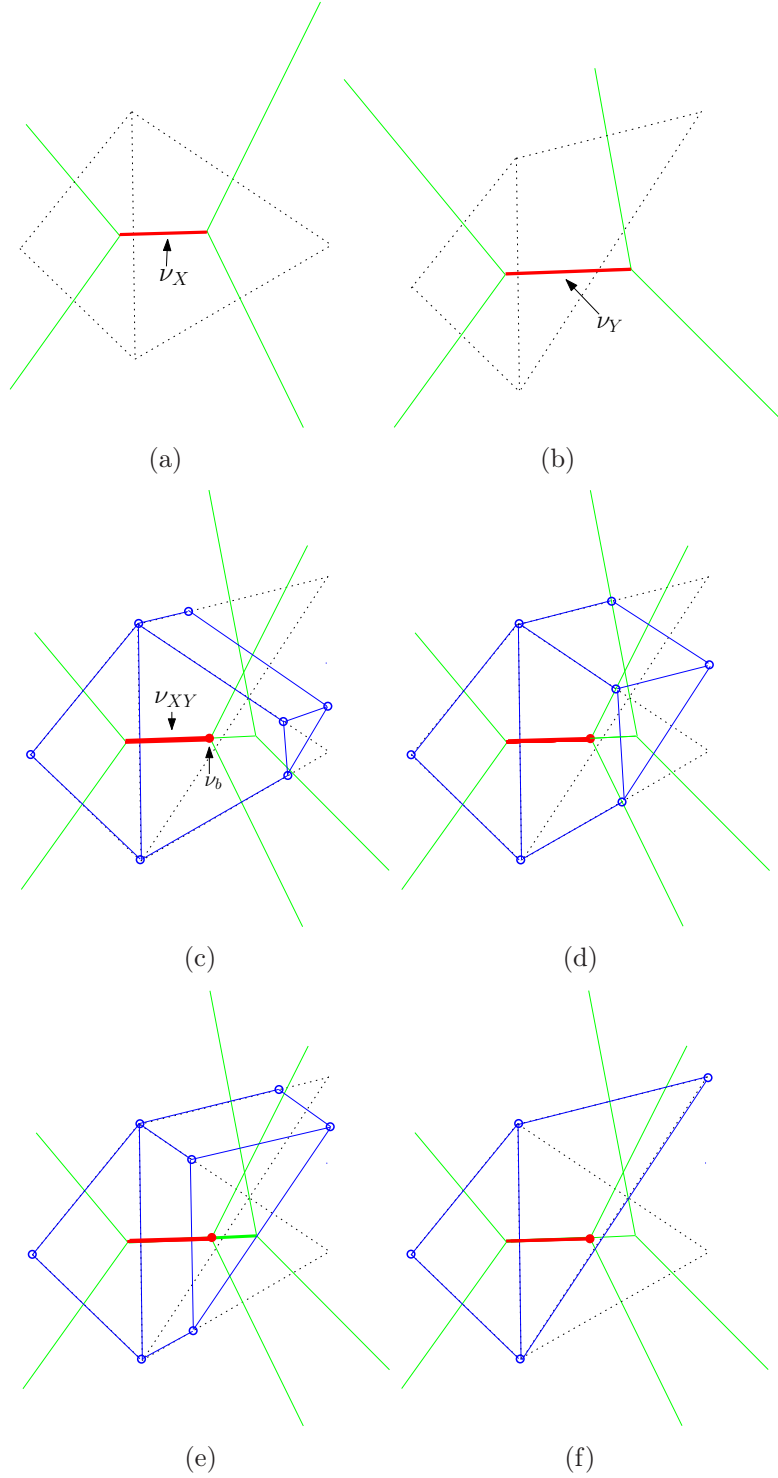


Figure 6.3: Movement of a simple ‘molecule’ with four atoms from status in (a) to a new status in (b). The two Voronoi complexes are super-imposed in (a) and (b), and a degenerate intermediate Voronoi vertex  $\nu_b$ , which has a trapezoid intermediate Delaunay cell, is formed. (c) - (f) demonstrate the deformation process with different  $t$  values at (c)  $t = 0.25$ , (d)  $t = 0.5$ , (e)  $t = 0.75$  and (f)  $t = 1$ .

# Chapter 7

## Software Development and Experiment Results

A general skin deformation software has been developed in Visual C++ with QT4 and OpenGL under windows platform. The software is available at our project development page: <http://www.comp.nus.edu.sg/~yanke2/skin/skin.htm>.

In this chapter, it is first demonstrated that our GSD algorithm is faster than existing static skin mesh algorithms. Second, subsequent improvement in efficiency from GSD to SGSD is shown.

### 7.1 Software for General Skin Deformation

Experimental results show that the GSD algorithm is much faster than existing static skin mesh algorithms in terms of generating each frame of intermediate skin mesh model with triangle quality guaranteed. The time taken in three different sets of skin model deformation is shown in Table 7.1: a mannequin skin model to

---

a question mark skin model (Figure 1.1); a bunny model to a torus model (Figure 1.2); a woman skin model to a fish skin model. All experiments are performed with Intel Duo Core 2.33GHz and 4GB RAM. In order to show a smooth deformation processes, one thousand frames (intermediate skin model meshes) are generated for each example. Therefore, if 24 frames per second is required for a short video clip, a deformation process of approximately one minute is produced. The GSD algorithm is compared with Kruithof’s algorithm [47] developed in CGAL [14] and the static skin section of the dynamic skin deformation algorithm (DST) developed by Cheng [16; 30] in Table 7.1.

In the first experiment (Figure 1.1), a mannequin skin mesh model (built by 12,684 spheres) is deformed to form a question mark (7 spheres). The details of the triangulation quality is shown in Figure 7.1. It is shown in Table 7.1 that the GSD algorithm is hundreds of times faster than both static skin mesh algorithms (CGAL-4.0.1 and DST). These static skin algorithms are slow because they generate each intermediate skin mesh model from scratch. They have the potential to be faster if the skin models are built by fewer spheres. In the example of Woman.skn (8 spheres) deforming to Fish.skn(8 spheres), our algorithm is only about two times faster than CGAL-4.0.1. Meshes with better triangle quality is produced by our program as compared to CGAL-4.0.1.

The faster performance of the GSD algorithm is also due to the new way of linear vertex movement. Figure 7.2 shows the minority of triangles that are being tested and refined during the time interval of two frames. The total number of triangles being refined is *constant* for each deformation regardless of how many frames are generated in the whole deformation process.

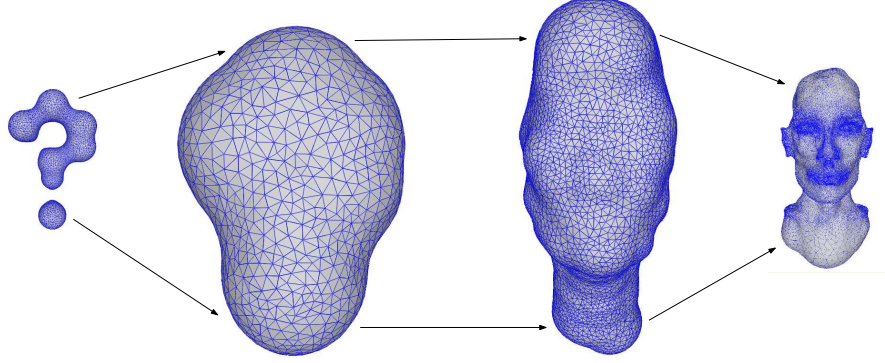


Figure 7.1: A zoomed-in snapshot of the general deformation between a mannequin skin model and a question mark skin model. Triangles are shown in wireframe.

Experiments	GSD	CGAL-4.0.1	DST
Mannequin.skn ↔ Question_Mark.skn	1.748 sec	323.8 sec	695.0 sec
Bunny.skn ↔ Torus.skn	0.908 sec	79.98 sec	93.76 sec
Woman.skn ↔ Fish.skn	0.615 sec	1.025 sec	1.875 sec

Table 7.1: Average time taken by different algorithms for different examples. Each deformation process is run with 1,000 frames.

## 7.2 Software for Simplified General Skin Deformation

The SGSD algorithm, which significantly reduces the number of intermediate mixed cells, the number of schedules and the overall running time as compared to the original GSD algorithm, is employed in the second version of GSD. A comparison of the results of deformation for three pairs of skin models is made: a bunny skin model deforming to form a cow skin model (Figure 7.3), a dragon skin model deforming to form a bunny skin model and a mannequin head skin model deforming to form a fist skin model (Figure 7.4). Two simplification lev-

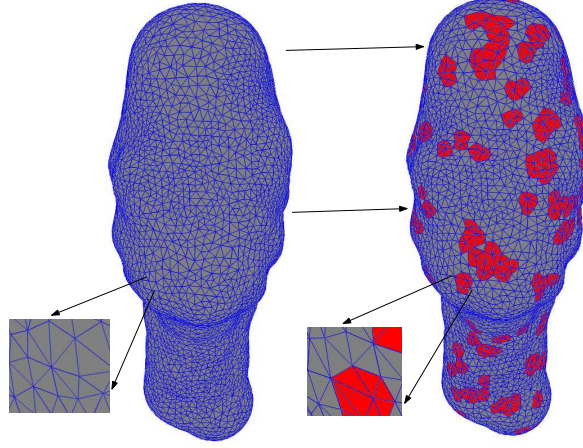


Figure 7.2: Triangle and vertex correspondences for frame 344 and 345 during the deformation. The red regions in frame 345 are triangles that are being refined from frame 344.

els are introduced to compare the performance by choosing different values of feature variable  $K$  and density valuable  $J$ . As the current available source (32-bit machine) is unable to store the original intermediate complex (149,381,672 intermediate mixed cells) of the last pair of skin models (mannequin head and fist), the original GSD of this pair of skin models is not tested. All starting weighted point sets are obtained from the power crust project developed by Nina Amenta et al. [52] at <http://www.cs.ucdavis.edu/amenta/powercrust.html> and the sphere-tree construction toolkit developed by Bradshaw et al. [12] at <http://isg.cs.tcd.ie/spheretree/>. The input skin meshes are generated by the quality skin mesh software developed by Cheng and Shi [18; 19] (Table 7.2).

Both GSD and SGSD algorithms are tested in a 32-bit windows machine with Intel Duo Core 2.33GHz and 4GB RAM. For the SGSD, we select two levels of simplification. In the first level, we choose  $K = 0.6$  and  $J = 0.1$  and we call it SGSD-1. Second, we choose  $K = 1.0$  and  $J = 0.3$  in the second level, or SGSD-2.

---

Table 7.2: Input skin meshes.

Input	# W. Points	# Vertices	# Triangles
Mannequin	12680	274206	482568
Fist	5461	217736	348616
Bunny	658	79030	110422
Cow	732	93813	149282
Dragon	869	100420	162231

In Figure 7.4 and 7.3, we show the comparison of actual deformation processes between the original GSD, SGSD-1 and SGSD-2. Similar deformation processes are achieved although we use much simpler intermediate weighted point sets.

First, a comparison of the total number of intermediate mixed cells is made, as shown in Table 7.3. The total number of intermediate mixed cells in all three sub-deformations in SGSD are also shown in Table 7.3. The result shows that the intermediate complex is significantly simplified in SGSD. Second, the whole deformation process is divided into 1,000 frames and the average number of schedules (Table 7.4) and average running time (Table 7.5) are compared in each frame for both algorithms. Based on the statistics collected, both the number of schedules and running time are reduced due to the simpler intermediate complexes in SGSD.

Table 7.3: Number of intermediate mixed cells for different deformation models.

Model pair	GSD	SGSD-1	SGSD-2
Mannequin $\leftrightarrow$ Fist	149,381,672	913,053	317,192
Bunny $\leftrightarrow$ Cow	577,614	182,254	94,911
Dragon $\leftrightarrow$ Bunny	891,161	221,658	103,746

The number of intermediate weighted points is maintained at less than 3 million in GSD. When the number of intermediate weighted points exceeds 3 million,

---

Table 7.4: Average number of schedules.

Model pair	GSD	SGSD-1	SGSD-2
Mannequin $\leftrightarrow$ Fist	-	16,491	5,719
Bunny $\leftrightarrow$ Cow	10,855	3,712	1,731
Dragon $\leftrightarrow$ Bunny	13,774	4,073	2,442

Table 7.5: Average time taken in each frame.

Model pair	GSD	SGSD-1	SGSD-2
Mannequin $\leftrightarrow$ Fist	-	3.18 sec	1.38 sec
Bunny $\leftrightarrow$ Cow	5.40 sec	1.87 sec	0.65 sec
Dragon $\leftrightarrow$ Bunny	6.14 sec	1.92 sec	0.75 sec

for example, the direct GSD between the mannequin and fist skin models in Figure 5.1, it is impossible for a 32-bit machine to handle such a large intermediate complex (indicated by the empty cell in Table 7.3 and 7.4). However, It is possible to deform the mannequin model to the fist model using the SGSD algorithm by two simplification deformations and one GSD (Figure 7.4).

The result of SGSD can be very much similar to the original GSD algorithm as the simplification processes guarantees the volume difference to be small. Although the simplified objects may lose sharp features of the original objects, it is visually tolerable since it is only one intermediate frame in the whole deformation process.

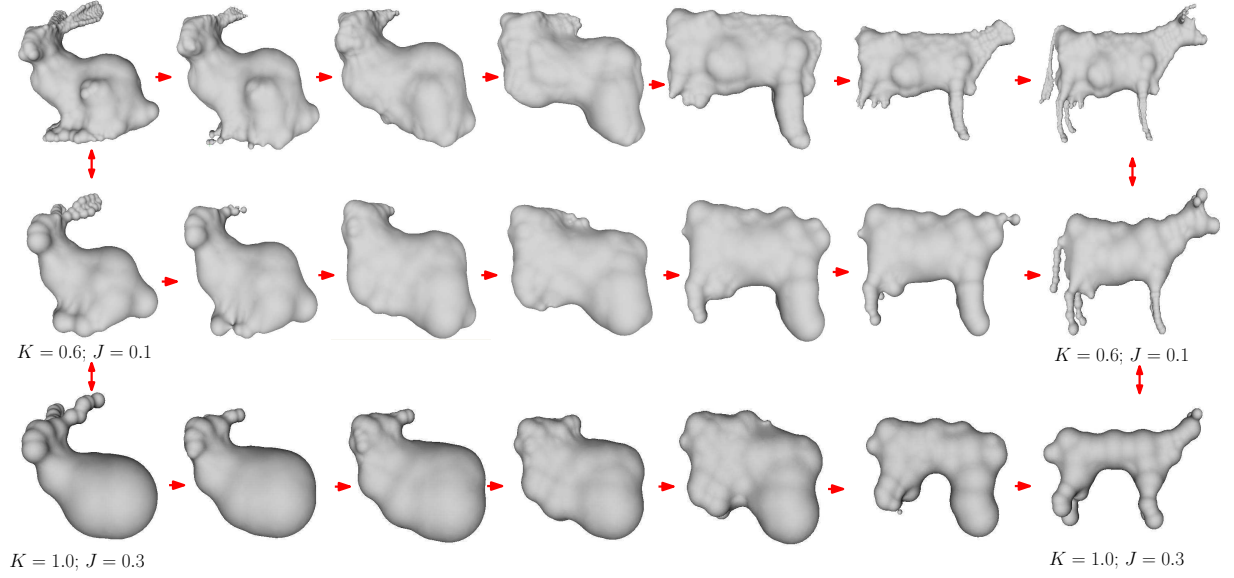


Figure 7.3: Different simplification level break down for deformation between bunny and cow.

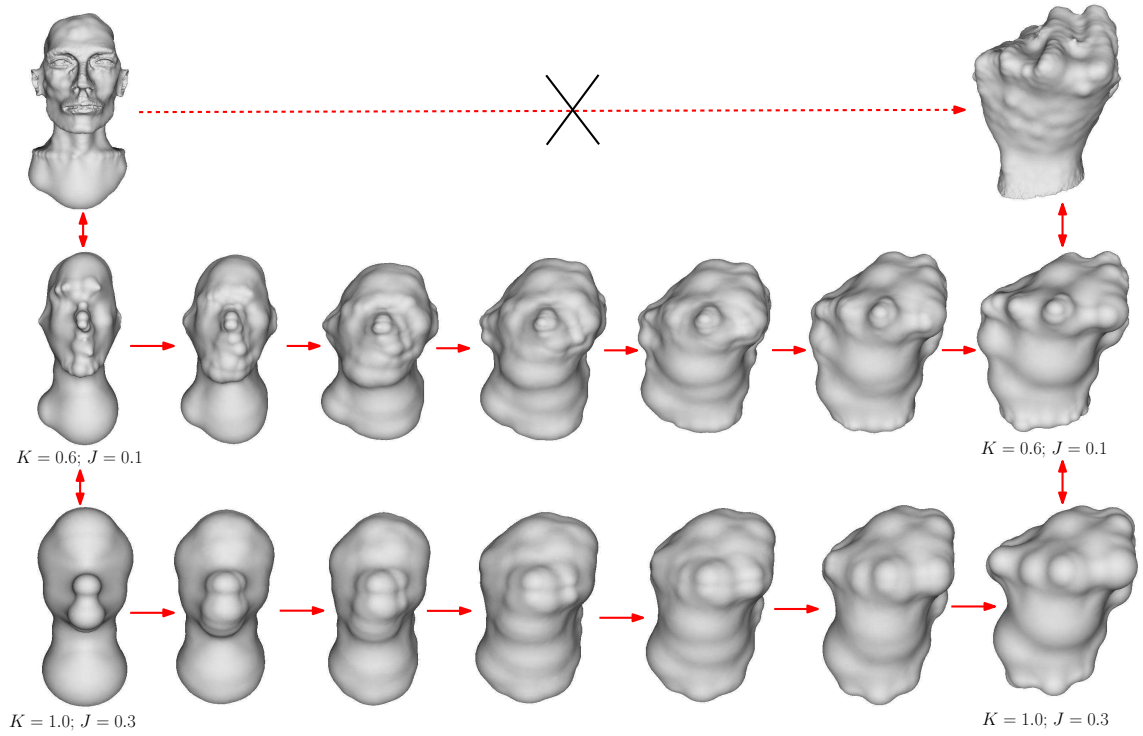


Figure 7.4: Different simplification level break down for deformation between mannequin and fist.

# Chapter 8

## Conclusions

An algorithm called general skin deformation (GSD) that allows automatic free form deformation between any two objects is presented in this thesis. This algorithm provides the speed boost required for deformation with additional advantages, such as automatic topology change handling, quality triangulation and surface point correspondence mapping. These enhancements do not only enable shape synthesis to become possible in computer animation, engineering and biogeometry applications, but also facilitate shape manipulations such as shape space searching, simplification or compression. The surface correspondence, guaranteed triangle quality and homeomorphism enable robust computation for engineering simulations.

The efficiency of GSD is improved with a simplified version of GSD, which is called SGSD. In SGSD, most features are inherited from GSD and the overall complexity is improved from  $O(m^2n^2)$  to  $O(m^2 + n^2)$ . This improvement greatly reduces the program running time as it simplifies the intermediate complex structure and abandons unnecessary topology changes. This improvement also enables some impossible deformations with large input sets to become possible. New degeneracy problems arise in the simplified deformation process and they are solved

---

by introducing new types of intermediate complexes.

Both GSD and SGSD algorithms are suitable for deformation between objects of greatly dissimilar shapes, since all input weighted points are assumed in general positions. In fact, for two shapes that are too similar, both algorithms suffer from degeneracy problems. Several new intermediate Delaunay types are introduced because of identical point positions in source and target weighted point sets (examples are shown in Chapter 5 and in Chapter 6).

Fully automated deformation algorithms are still far from being practical in real world applications, such as movie and cartoon animations. In GSD and SGSD, a solution to escape from the restriction of similarity is provided, both source and target shapes are converted into weighted point sets and the weighted points are interpolated from the two sets. The limitation of this approach is that sharp features may be lost in the intermediate shapes. It is recommended that the future work addresses this limitation by introducing additional reference shapes during the deformation[15].

Future research includes controlling the deformation locally, for example, reducing the topology changes during deformation so that the changes can be controlled. Another area of interest is to investigate new types of possible vertex movement schemes, such as movements along directions that are orthogonal to the hyperboloids surface to enhance performance and quality. Another exciting challenge is to extend our algorithm to construct a deforming *volume tetrahedral mesh* for physical simulation purposes. The tetrahedral refinement in a deforming body is more complex than that of the surface mesh. A study has been done to show that a good surface mesh of the boundary of an object aids the construction of the volumetric mesh [18].

# References

- [1] M. Alexa and O. Sorkine. As-rigid-as-possible shape interpolation. *In Proc. SGP*, 2007. 2, 10
- [2] C. Andujar, D. Ayala, and P. Brunet. Volume-based polyhedra simplification using tg-maps. *Technical Report LSI-97-21-R*, 1997. 72
- [3] C. Andujar, D. Ayala, P. Brunet, R. Joan, and J. Sole. Automatic generation of multiresolution boundary representation. *Computer Graphics Forum*, 1996. 72
- [4] Dominique Attali and Annick Montanvert. Computing and simplifying 2d and 3d continuous skeletons. *Comput. Vis. Image Underst.*, 1997. 72
- [5] F. Aurenhammer. Voronoi diagrams - a study of a fundamental geometric data structure. *ACM Comput. Surveys* 23, pages 345–405, 1991. 14
- [6] I. Baran and J. Popovic. Automatic rigging and animation of 3d characters. *ACM TOG, vol.26, no.3*, 2007. 2
- [7] M. Bertalmio, G. Sapiro, and G. Randall. Region tracking on level-sets methods. *IEEE Transactions on Medical Imaging*, 18:448–451, 1999. 8

## REFERENCES

---

- [8] j. Bloomenthal, C. Bajaj, J. Blinn, Marie-Paule, Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill. *Introduction to implicit surfaces*. Morgan-Kaufmann, 1997. 3
- [9] Alex Bordignon, Betina Vath, Thales Vieira, Marcos Craizer, Thomas Lewiner, and Cynthia O.L. Ferreria. Scale-space for union of 3d balls. *In Proceeding of the 2009 XXII Brazilian Symposium on Computer Graphics and Image Processing.*, 2009. 72
- [10] Mario Botsch and Leif Kobbelt. Real-time shape editing using radial basis functions. *Comput. Graph. Forum 24(3): 611-621*, 2005. 2
- [11] Mario Botsch, Mark Pauly, Markus H. Gross, and Leif Kobbelt. coupled prisms for intuitive surface modeling. *Symposium on Geometry Processing*, 2006. 10
- [12] G. Bradshaw and C. O’Sullivan. Sphere-tree constuction using medial axis approximation. *SIGGRAPH Symposium on Computer Animation SCA*, 2002. 10, 93
- [13] Frederic Cazals, Harshad Kanhere, and Sébastien Lorient. Computing the volume of a union of balls: A certified algorithm. *ACM Trans. Math. Softw.*, 38. 74
- [14] CGAL. <http://www.cgal.org>. 7, 8, 9, 91
- [15] C. Chen and H.-L. Cheng. Superimposing Voronoi complexes for shape deformation. *Int. J. Comput. Geometry Appl*, 2006. 9, 10, 12, 14, 25, 30, 98

## REFERENCES

---

- [16] H.-L. Cheng, Tamal K. Dey, H. Edelsbrunner, and J. Sullivan. Dynamic skin triangulation. *Discrete Comput. Geom*, 2001. 7, 9, 10, 11, 13, 21, 23, 25, 54, 55, 61, 68, 76, 91
- [17] H.-L. Cheng, H. Edelsbrunner, and P. Fu. Shape space from deformation. *Comput. Geom. Theory Appl.*, pages 191–204, 2001. 8, 14, 25
- [18] H.-L. Cheng and X. Shi. Quality tetrahedral mesh generation for macro-molecules. *ISAAC*, pages 203–212, 2006. 9, 10, 13, 14, 93, 98
- [19] H.-L. Cheng and X.W. Shi. Quality mesh generation for molecular skin surfaces using restricted union of balls. *IEEE Visualization*, 2005. 7, 8, 9, 10, 13, 14, 93
- [20] H.-L. Cheng and T. Tan. Approximating polyhedral objects with deformable smooth surfaces. *Computational Geometry, Thoery and Applications*, 18:104–117, 2008. 4, 6, 10, 22
- [21] H.-L. Cheng and K. Yan. Mesh deformable smooth manifolds with surface correspondences. *Mathematical Foundations of Computer Science.*, 2010. 4, 9, 14, 68
- [22] Holun Cheng. Algorithms for smooth and deformable surface in 3d. *Ph.D Thesis*, 2002. 14
- [23] S.-W. Cheng, H. Edelsbrunner, P. Fu, and P. Lam. Design and analysis of planar shape deformation. *Comput. Geom. Theory Appl.*, pages 205–218, 2001. 8, 25

## REFERENCES

---

- [24] M. K. Chung, K. J. Worsley, S. Robbins, T. Paus, J. Taylor, J. N. Giedd, J. L. Rapoport, and A. C. Evans. Deformation-based surface morphometry applied to gray matter deformation. *NeuroImage*, 18:198–213, 2003. 2
- [25] M. L. CONNOLLY. Analytic molecular surface calculation. *J. Appl. Crystallogr.*, 1983. 16
- [26] T.E. Creighton. Proteins structures and molecular principles. *Freeman, New York*, 1984. 9
- [27] K. G. Der, R. W. Sumner, and J. Popovic. Inverse kinematics for reduced deformable models. *ACM SIGGRAPH, Computer Graphics Proceedings, Annual Coference Series*, 2006. 2
- [28] Olivier Devillers. On deletion in delaunay triangulations. *SCG'99 Miami Beach Florida*, 1999. 36
- [29] H. Edelsbrunner. Deformable smooth surface design. *Discrete Comput. Geom*, pages 87–115, 1999. 4, 8, 9, 13, 14
- [30] H. Edelsbrunner and A. Ungor. Relaxed scheduling in dynamic skin triangulationă. *Japanese Conf. Comput. Geom*, 2002. 11, 23, 68, 91
- [31] H. Edelsbrunner and C. J. A. Delfinado. An incremental algorithm for betti numbers of simplicial complexes on the 3-sphere. *Comput. Aided Geom. Design* 12, 1995. 9
- [32] H. Edelsbrunner, M. Facello, P. Fu, and J. Liang. Measuring proteins and voids in proteins. *Proc. 28th Hawaii Intern. Conf. Syst. Sci. 1995*, 1995. 9

- [33] H. Edelsbrunner and P. Koehl. The geometry of biomolecular solvation. *Combinatorial and Computational Geometry (MSRI Publications)*., pages 243–275, 2005. 73, 74, 78
- [34] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topology persistence and simplification. *In Proc. 41st Ann. IEEE Sympos. Found. Comput. Sci. (2000)*., 2000. 9, 14
- [35] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graphics*, pages 43–72, 1994. 9
- [36] H. Edelsbrunner and R. Seidel. Voronoi diagrams and arrangement. *Discrete Compt. Geom.* 1, 1986. 14
- [37] James E. Gain and Neil A. Dodgson. Preventing self-intersection under free-form deformation. *IEEE Trans. Vis. Comput. Graph.* 7(4): 289-298, 2001. 6
- [38] Don L. Gibbons, Marie-Christine Vaney, Alain Roussel, Armelle Vigouroux, Brigid Reilly, Jean Lepault, Margaret Kielian, and Félix A. Rey. Conformational change and protein-protein interactions of the fusion protein of semliki forest virus. *Nature* 427, 320-325 (22 January 2004), 2004. 1
- [39] P. Giblin. Graphs, surfaces, and homology. *Chapman and Hall, London.*, 1981. 9
- [40] P. S. Heckbert and M. Garland. Multiresolution modelling for fast rendering. *Proc. Graphics Interface*, pages 43–50, 1994. 72

- [41] Jin Huang, Xiaohan Shi, Xinguo Liu, Kun Zhou, Liyi Wei, Shanghua Teng, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Subspace gradient domain mesh deformation. *ACM Transactions on Graphics (SIGGRAPH 2006)*, 2006. 10
- [42] Tao J., Qian-Yi Z., and Shi-Min H. Editing the topology of 3d models by sketching. *In ACM SIGGRAPH '07*, 2007. 2, 6
- [43] Doug L. James and Christopher D. Twigg. Skinning mesh animations. *ACM SIGGRAPH 2005.*, 2005. 8
- [44] S. Karan. Skinning characters using surface-oriented free-form deformations. *In In Graphics Interface 2000*, pages 35–42, 2000. 8
- [45] S. Kircher and M. Garland. Free-form motion processing. *ACM TOG, vol27, no.2*, 2008. 2
- [46] Leif Kobbelt and Mario Botsch. Freeform shape representations for efficient geometry processing . *Invited paper at Shape Modeling International*, 2003. 6
- [47] N.G.H. Kruithof and G. Vegter. Meshing skin surfaces with certified topology. *Computational Geometry*, 36:166–182, 2007. 8, 9, 13, 91
- [48] B. LEE and F. M. RICHARDS. The interpretation of protein structures: estimation of static accessibility. *J. Mol. Biol.*, 1971. 16
- [49] J.P. Lewis, M. Cordener, and N. Fong. Pose space deformation: Aunified approach to shape interpolation and skeleton-driven deformation. *ACM SIG-*

- GRAPH, Computer Graphics Proceedings, Annual Coference Series*, 2000.
- 10
- [50] David Mason and Geoff Wyvill. Blendeforming: Ray traceable localized foldover-free space deformation. *Computer Graphics International 183-192*, 2001. 6
- [51] T.E. Morthland, P.E. Byrne, D.A. Tortorelliand, and J.A.D. Optimal riser design for metal castings. *Metallurgical and Material Transcations*, 1995. 1
- [52] Sunghee Choi Nina Amenta and Ravi Kolluri. The power crust. *Proceedings of 6th ACM Symposium on Solid Modeling, 2001.*, pages 249–260, 2001. 10, 69, 93
- [53] N.Provatas, N.Goldendeld, and J.A.Dantzig. Adaptive grid methods in solidification microstructure modeling. *Phys.Rev.Lett*, 1998. 1
- [54] Klaus Schulten Paul Grayson, Emad Tajkhorshid. Mechanisms of selectivity in channels and enzymes studied with interactive molecular dynamics. *Biophysical Journal.*, pages 36–48, 2003. 1
- [55] W. Sumner Robert and P. Jovan. Deformation transfer for triangle meshes. *ACM SIGGRAPH, Computer Graphics Proceedings, Annual Coference Series*, 2004. 2
- [56] Jonathan Richard Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications 22(1-3):21-74*, May 2002. 36

## REFERENCES

---

- [57] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Roessl, and Hans-Peter Seidel. Laplacian surface editing. *Symposium on Geometry Processing*, 2004. 10
- [58] J. Starck and A. Hilton. Correspondence labelling for wide-timeframe free-form surface matching. pages 1–8, 2007. 8
- [59] Jorge Stolfi. Primitives for computational geometry. *Technical Report 36, DEC SRC*, 1989. 41
- [60] Vitaly Surazhsky and Craig Gotsman. Intrinsic morphing of compatible triangulations. *International Journal of Shape Modeling*, 2003. 2
- [61] Roger Tam and Heidrich Wolfgang. Shape simplification based on the medial axis transform. *In Proceedings of the 14th IEEE Visualization 2003 (VIS'03), Washington, DC, USA.*, 2003. 72
- [62] R.D. Taylor, P.J. Jewsbury, and J.W. Essex. A review of protein-small molecule docking methods. *J Comput Aided Mol Des.* 9
- [63] Seth Teller and Michael Hohmeyer. Determining the line through four lines. *Journal of Graphic Tools*, 1999. 41
- [64] David M. Mount Thomas Kao. Dynamic maintenance of delaunay triangulations. *Proceeding of Auto-Carto 10*, pages 219–233, 1991. 30, 35
- [65] G. Turk and J. O'Brien. Shape transformation using variational implicit functions. *Proc. SIGGRAPH*, 1999. 3
- [66] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. *Proc. SIGGRAPH '94*, 1994. 6

- [67] W. von Funck, H. Theisel, and H.-P. Seidel. Vector field based shape deformations. *In ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2006)*, 2006. 10
- [68] Rui Wang, Kun Zhou, John Snyder, Xinguo Liu, Hujun Bao, Qunsheng Peng, and Bining Guo. Variational sphere set approximation for solid objects. *Visual Comput*, 2006. 10
- [69] Y. Wang, B. Peterson, and L. Staib. Shape-based 3d surface correspondence using geodesics and local geometry. In *CVPR*, pages 2644–2651, 2000. 8
- [70] A. Watt and M. Watt. *Advance Animation and Rending Techniques*. Addison-Wesley, 1992. 2
- [71] O. Weber, O. Sorkine, Y. Lipman, and C. Gotsman. Context-aware skeletal shape deformation. *Computer Graphics Forum, vol 26, no.3*, 2007. 2, 10
- [72] K. Yan and H.-L. Cheng. Local modification of skin surface mesh: Towards free-form skin surface deformation. *In Proceedings of The China-Japan Joint Conference on Computational Geometry, Graphs and Applications (CGGA).*, 2010. 6, 70
- [73] K. Yan and H.-L. Cheng. On simplifying deformation of smooth manifolds defined by large weighted point sets. *In Proceedings of Thailand-Japan Joint Conference on Computational Geometry and Graphs.*, 2012. 13
- [74] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph.* 23(3): 644-651, 2004. 10

## REFERENCES

---

- [75] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliaschi. Deformation-driven shape correspondence. *In Proc. SGP*, 2008. 2
- [76] Kun Zhou, Jin Huang, John Snyder, Xinguo Liu, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Large mesh deformation using the volumetric graph laplacian. *ACM Transactions on Graphics*, July 2005. 10