5-6-7 MESHES

by

Nima Aghdaii

B.Sc., University of Tehran, 2008

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Computing Science
Faculty of Applied Sciences

APPROVAL

Name:	Nima Aghdaii
Degree:	Master of Science
Title of Thesis:	5-6-7 Meshes

Examining Committee: Dr. Veronica Dahl
Chair

____

Dr. Hao Zhang, Senior Supervisor

____

Dr. Binay Bhattacharya, Supervisor

____

Dr. Ligang Liu, Examiner,
Professor of Computing Science,
Zhejiang University, China

Date Approved: June 28, 2012

Abstract

In this thesis, we introduce a new type of meshes called 5-6-7 meshes. For many mesh processing tasks, low- or high-valence vertices are undesirable. At the same time, it is not always possible to achieve complete vertex valence regularity, i.e., to only have valence-6 vertices. A 5-6-7 mesh is a closed triangle mesh where each vertex has valence 5, 6, or 7. An intriguing question is whether it is always possible to convert an arbitrary mesh into a 5-6-7 mesh. In this thesis, we answer this question in the positive. We present a 5-6-7 remeshing algorithm which converts any closed triangle mesh with arbitrary genus into a 5-6-7 mesh, which a) closely approximates the original mesh geometrically, e.g., in terms of feature preservation, and b) has a comparable vertex count as the original mesh. We demonstrate the results of our remeshing algorithm on meshes with sharp features and different topology and complexity.

Keywords: geometry processing; remeshing; connectivity

Acknowledgments

I would like to thank all the people whose support and encouragement have significantly helped me throughout these years. First, I would like to thank Dr. Hao Zhang, my senior supervisor, Dr. Binay Bhattacharya, my co-supervisor, and Dr. Ligang Liu, my examiner, for their support, encouragement, criticism, and guidance throughout these years. Secondly, I would also like to thank Hamid Younesy, the co-author on our paper on 5-6-7 Meshes. His knowledge and experience besides all the effort he put into this work cannot be expressed in words and without his assistance, this work would have never been at this stage. Moreover, I would also like to take this opportunity to thank all my friends and colleagues at the GRaphics, Usability, and VIsualization (GrUVi) lab, for the very fruitful discussions, reading groups, and for their constant help throughout these years. I would like to specifically appreciate Ramsay Dyer's bright hints and comments on this work. Finally, I would like to thank my parents for their patience, support and love. And my great sister, for her moral support during the hard times.

Contents

List of Tables

List of Figures

Preface

This work initially started as a project for the course "Geometric Modeling in Computer Graphics". The main idea was proposed to me by my supervisor, Hao Zhang, as a yes/no question of "Is it possible to have a 5-6-7 remeshing for any arbitrary triangle mesh?". My first intuition about the problem was not positive since I started to think about the simplest geometric shapes, such as a tetrahedron, and initially it was not clear how they can be transformed to a 5-6-7 mesh. However, noticing simple local remeshing approaches to remove vertices of degree 3 inspired the possibility of the idea. By the end of the course, we had a 5-6-7 remeshing with a more complicated structure for removing high degree vertices and not a visually appealing set of results. Following that trend, a better approach for removing high degree vertices based on vertex splits reduced the size of the 5-6-7 mesh and thanks to the simplification and geometry enhancement steps we could produce 5-6-7 meshes with a satisfactory geometry quality and a comparable vertex count.

As the first challenge of this work was the core idea and the execution phase, the second wall to climb was answering questions regarding the motivation of this work. Questions such as why 5-6-7 remeshing is better than other remeshing algorithms? or what are we going to benefit from 5-6-7 meshes? In order to motivate this work and answer these questions rationally, we first studied the advantages and disadvantages of regular and irregular vertices and presented comparisons to support the plus points of 5-6-7 meshes. Secondly, I hope that besides the theoretical value of this work, it awakens possibilities in the area of remeshing, specially constrained regularity remeshing. At the end, we sum it up by a set of future works and possible directions to follow.

Chapter 1

Introduction

In computer graphics applications and numerical analysis, one of the most common ways to represent a shape is using surface meshes. Surface meshes are generally constructed by a set of polygons that approximate the surface of the shape. Among various choices of polygons, triangles and quadrangles, are widely used to build triangle meshes and quad meshes, respectively. In this thesis our focus is on triangle meshes, since triangles are the simplest form of a 2D structure, they always create a planar structure (despite quadrangles), and generally they are ubiquitous in computer graphics and geometry processing.

Triangle meshes are usually represented as a set of points in the space –known as the geometry– and the relation between these points –known as the graph connectivity. Although generally these two quantities are considered to be independent, [24] showed that there is a significant relationship between them. For example, achieving regularity in the geometry, requires regularity in the connectivity as well. In this thesis, we heed our attention to vertex degrees in the connectivity graph and strive to provide a guaranteed bound for the regularity of vertex degrees in the connectivity graph.

The degrees of vertices in a triangle mesh often have an impact on how certain mesh processing algorithms perform. When triangle quality is of concern, neither low- nor high-valence vertices are desirable since they often imply large or small face angles in a triangle mesh. It is commonly known that the regular vertex valence in a triangle mesh is 6 but complete regularity can be achieved only on a tessellation of genus-one surfaces. An intriguing question is whether it is always possible to completely eliminate low- and high-valence vertices, only keeping valences close to 6, e.g., 5, 6, and 7, for meshes tessellating surfaces of any arbitrary genus.

In this thesis, we answer the above question in the positive. Specifically, we show that given an arbitrary closed triangle mesh with any genus, we can always remesh it to a *5-6-7 mesh*, i.e., a

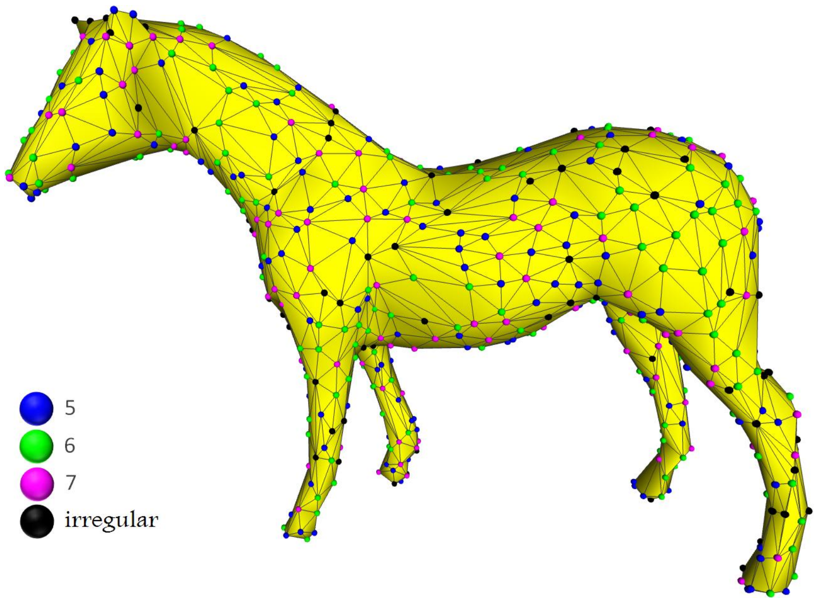


Figure 1.1: An example of an input mesh. Vertices with valences 5, 6, and 7 are colored in blue, green, and pink, respectively, and other vertices are in black. In this work we intend to remove all the black vertices.

triangle mesh whose vertex valences only take on values 5, 6, or 7. We also show how to keep the face count comparable to the original mesh, while respecting features on the original mesh. Figure 1.1 shows an example of an input mesh, with vertices of valences 5, 6, and 7 colored in blue, green, and pink and other vertices colored in black.

1.1 Background

We often use the terms *vertex degrees* and *vertex valencies* alternatively, which are synonyms and the choice of either is totally subjective, without influencing the meaning at all. Also, in order to refer to a vertex of degree *x* we use the term *vx*. For instance, a *v*6 vertex or a *v*7 vertex.

In computer graphics and geometry processing a regular vertex usually refers to a v6 vertex and any vertex with valence other than 6 is considered as irregular. Vertices with a valence lower than 6 are considered as low valence vertices and higher than 6 are called high valence vertices. However, in this work since our goal is to achieve vertices of degrees 5, 6, or 7, we refer to vertices of valence less than 5 as low valencies and higher than 7 as high valencies.

1.2 Motivation and applications

Our interest in the specific valences 5, 6, and 7 is motivated by Euler Characteristic formula, from which it can be shown that the average valence in a closed manifold triangle mesh is $6\,(1-\frac{(2-2g)}{n})$, where n is the number of vertices and g is the genus of the mesh. As such, by increasing the number of vertices, we will maintain an average valence of 6. However, since it is not always possible to have a mesh consisting of vertices of valence 6 only, vertices with valences higher than 6 and/or lower than 6 are generally inevitable. Thus the “next best scenario” in bounding the vertex valences close to valence 6 would be to produce 5-6-7 meshes, meshes with vertices of degrees 5, 6, and 7.

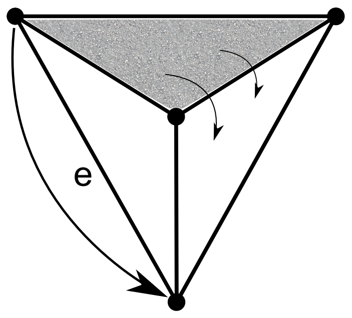


Figure 1.2: Face fold over in the vicinity of a v3 vertex. Collapsing edge ’e’ will produce a fold over face, which is shaded in this figure.

The first question here would be ”Why removing irregularities?”. In order to answer this question, we take a closer look at irregularities in a triangle mesh. For example, valence-three vertices will cause an edge collapse operator to generate non-manifold vertices [32], as shown in Figure 1.2. Also, both degree 3 and 4 vertices in planar regions will produce large obtuse angles, which is not desired in many applications such as parametrization algorithms. On the other hand, high valence vertices produce slim triangles, which are prone to numerical/precision errors. Regular vertices are generally much easier to handle in subdivision algorithms [39]. Moreover, high valence vertices lead to visual artifacts, known as *polar artifact*, in rendering applications using mesh subdivision (shown in Figure 1.3).

Besides the disadvantages of irregular vertices and the advantages of regular vertices, another important aspect of having a 5-6-7 mesh is explained in the work by Li et al. [27]. In that paper the authors provide an interactive tool to edit vertices of degrees 5 and 7, and for their application they require the input mesh to be a 5-6-7 mesh. They accomplished this task by developing a tool for the

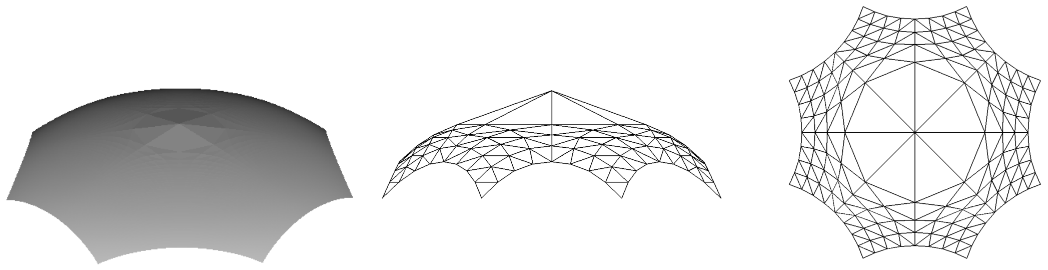


Figure 1.3: Polar artifact is a known rendering artifact around high valence vertices, when using subdivision algorithms. Figure is taken from [5]

user to manually change/edit the connectivity of the mesh by moving irregularities on the surface of the mesh.

1.3 Overview of our approach

We provide a *5-6-7 remeshing* algorithm to transform an arbitrary triangle mesh to a 5-6-7 mesh. Our 5-6-7 remeshing algorithm works in two phases. First is the initial conversion which is guaranteed to convert an arbitrary closed mesh into a 5-6-7 mesh. This step keeps the changes to the mesh geometry to minimal but will increase the face count and may produce uneven vertex distribution. In the second phase, the refinement phase, we perform mesh decimation and enhancement, while maintaining the 5-6-7 property. Figure 1.4 depicts a summary of our algorithm in 4 steps, as mentioned above.

The first phase of our method changes the geometry only slightly at high valence vertices. However, in order to reduce the size (face count) of the 5-6-7 mesh back to the size of the initial mesh, we apply decimation and geometric enhancement which may change the shape geometry to some extent. For this, user only needs to specify a feature preservation threshold and the number of iterations to relax. Our implementation provides an interactive tool to assist the user in choosing a reasonable value.

1.3.1 5-6-7 remeshing

We start by removing vertices with valence lower than 5, without introducing any *geometric error*. After that we apply a planar subdivision scheme, which does not change the geometry either, to push high valence vertices away from each other in the connectivity graph and surround each high

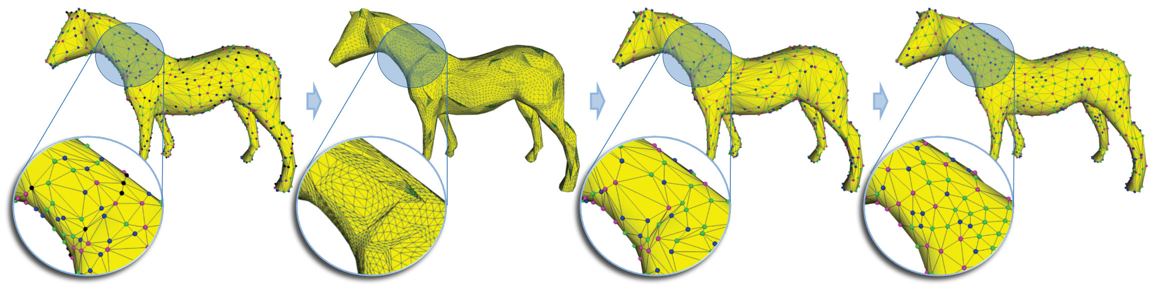


Figure 1.4: A triangle mesh (left) with low- and high-valence vertices (marked by black dots) remeshed into a 5-6-7 mesh with the same vertex count (right). Intermediate steps produce an initial 5-6-7 mesh (middle left) and a decimated version (middle right) which preserves the 5-6-7 property. The final mesh is produced with redistribution of vertices to improve sampling regularity, while respecting the features. Vertices of degrees 5, 6, and 7 are coloured by blue, green, and red, respectively.

valence vertex with an unshared set of regular vertices. Then we remove vertices with a valence greater than 7 using only *local* remeshing. This operation may introduce some geometric error.

1.3.2 5-6-7 mesh refinement

We perform a 5-6-7 preserving decimation to turn the mesh towards having its original vertex count. The decimation is governed by quadric error [14] and we only allow edge collapses that preserve the 5-6-7 property. In other words, the decimation step does not produce any low or high valence vertices. Then a relaxation step is applied both to improve the triangle qualities and to reduce the geometric error produced by the decimation step.

1.4 Contributions

In this work, we answer the theoretical question of ”Is it possible to have a 5-6-7 remeshing for any given mesh with an arbitrarily small error?” in the positive, by providing a constructive proof. We provide the 5-6-7 remeshing algorithm that works fully automatically and produces the 5-6-7 mesh. However, in order to improve the quality of the 5-6-7 mesh the algorithm requires some parameter tuning.

Although there has been an abundance of works in the area of remeshing, and most of them produce very satisfactory results, there has not been any work on remeshing with a guaranteed bound on vertex valences. Many of the substantial state-of-the-art works in this area either produce meshes with a majority of v6 vertices (without any guarantees on other vertices), or they use subdivision or refinement approaches to produce a guaranteed bound for the vertex valences, which is based on a new *base mesh*, which itself is tricky to have a guaranteed bound for its valences. Despite the refinement approaches, our method works pretty much locally around irregularities (except for one subdivision step), it can be considered as an interpolating approach and it is possible to keep the error very small.

Chapter 2

Related Work

The quality of a surface mesh is crucial for many purposes, including 3D visualization and numerical simulation. Therefore, there has been an abundance of remeshing algorithms proposed in the past two decades to improve the quality of a given mesh. While some remeshing algorithms are based on improving the geometry of the mesh by redistributing points on the underlying surface, other works look more into the mesh connectivity, and strive to reduce the degree variance of the connectivity graph by removing irregularities, or at least moving them to more appropriate locations.

Here we investigate some of the previous works in two separate sections. First, we look at the objectives of different remeshing algorithms and second we classify them by their methods. In other words, the first and the second sections are organized based on "What" the objectives are, and "How" the objectives are achieved technically.

2.1 Remeshing Objectives

The main distinction between remeshing algorithms lies in their objective. Some of these objectives include triangle quality, point distribution, angle constraints, and valence regularity. These objectives are crucial for a myriad number of applications such as mesh compression, feature preservation/retrieval, efficient rendering, interactive free-form shape modelling, etc. Among all of these objectives, regularity as a common goal is an appropriate anchor point to categorize them. Regularity can be achieved in different ways, and there are also different types of regularity. In the following subsections we study the different types of regularity by providing examples of the previous works that lie in that category.

2.1.1 Semi Regularity

Semi-regular meshes are obtained by either recursive subdivision or refinement of a base mesh. Starting with a base mesh that roughly approximates the desired shape, we iteratively subdivide/refine the mesh to achieve a smooth approximation of the original shape. Semi regular meshes are heavily utilized in generating smooth surfaces, multiresolution analysis/modeling, and level-of-detail applications because of their hierarchical nature. In this type of remeshing, all of the added vertices are regular and the only irregular vertices are those from the initial base mesh. Figure 2.1 shows an example of semi-regular remeshing.

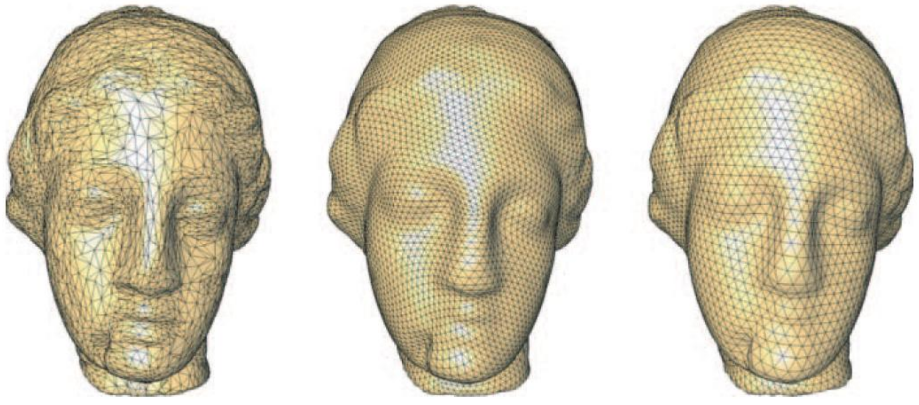


Figure 2.1: An exmaple of semi-regularity remeshing. The figure is taken from [3]

2.1.2 Complete Regularity

In completely regular meshes (a regular grid), the connectivity is implicit. In other words, the surface of the mesh is parametrized to a completely regular structure, such as *geometry image* proposed by Gu et al. [17]. As shown in Figure 2.2, the mesh is first cut such that 1) it has the topology of a disk 2) parametrization distortion is minimized, and finally parametrized as a simple $n \times n$ matrix of [x, y, z] values. Other attributes of the mesh such as vertex normals and colors also can be stored in separate matrices. Since all of these attributes share the same parametrization with the geometry, parametrization is basically implicit. Geometry images are heavily utilized for simplifying the hardware rendering pipeline, and texturing. However, since global parametrization is inevitably prone to severe distortion, Sander et al. [38] proposed a specific atlas construction to map the mesh on charts of arbitrary shape to alleviate that issue .

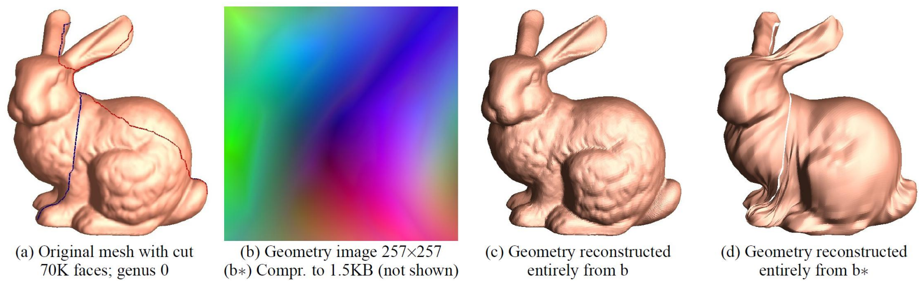

(a) Original mesh with cut
70K faces; genus 0
(b) Geometry image 257×257
(b∗) Compr. to 1.5KB (not shown)
(c) Geometry reconstructed
entirely from b
(d) Geometry reconstructed
entirely from b∗

Figure 2.2: Geometry images, used to acquire complete regularity implicitly. The figure is taken from [17]

2.1.3 High Regularity

Highly regular meshes are generally produced by converting the original mesh into a mesh with the majority of v6 vertices. Most of the remeshing methods targeting high regularity, work based on relaxation algorithms [29][36], local/global parametrization [43], or re-sampling of the points in a certain domain [44]. After applying these type of remeshing algorithms, most of the vertices will have a regular valence, however there is no guaranteed upper/lower bound on the valence of the vertices. Figure 2.3 illustrates an example of the high regularity remeshing, by Surazhsky, et al. [43]. Most of the high regularity remeshings, produce meshes a very visually appealing connectivity and some of them also attempt to work on the distribution of the points in favor of features, curvature, or other surface properties.

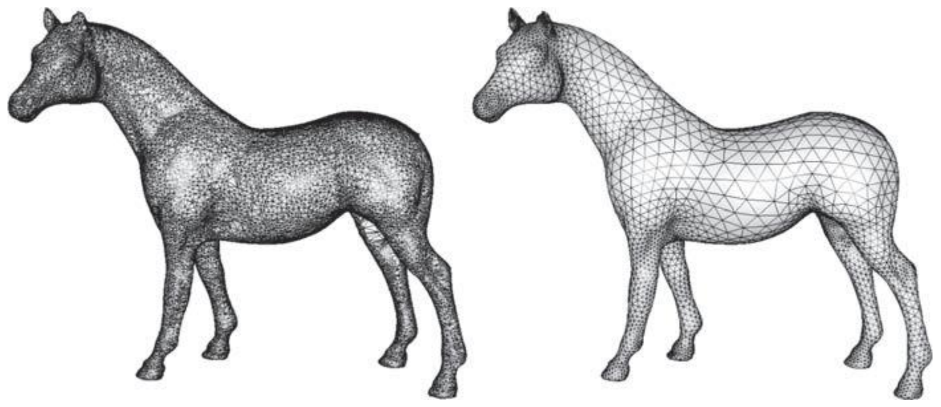


Figure 2.3: High regularity remeshing. The figure is taken from [3]

2.1.4 Constrained Regularity

In this category, some constraint on the mesh needs to be satisfied. This constraint might be on the triangle quality, vertex valences, or any other useful property. For example, in non-obtuse remeshing [28] (shown in Figure 2.4) every angle in the mesh should be less than or equal to $\frac{\pi}{2}$. Many of the constrained regularity remeshings are mainly beneficial to theoretical or application oriented uses, and they are less considered as a remeshing that portray aesthetic improvements.

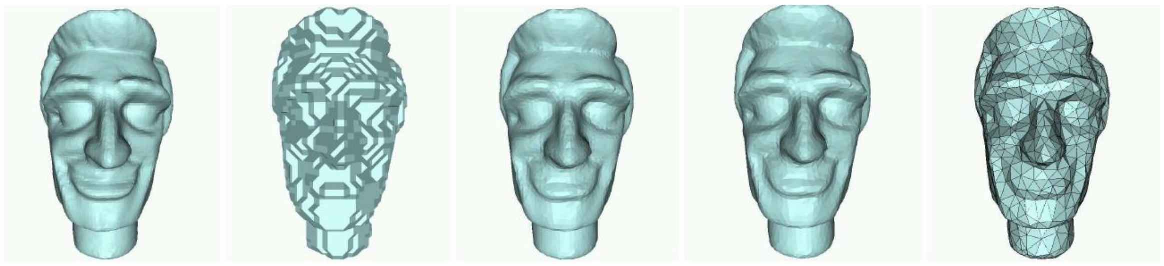


Figure 2.4: Steps of the non-obtuse remeshing algorithm by Li et al. [26]. The figure is taken from [26]

However, the work on constrained regularity remeshing on the connectivity graph of the mesh is very limited, and the only existing algorithms in this area are based on structured meshes for specific shapes. 4-8 subdivision [48] for modeling terrain data is a well known example of this category, as they produce a remeshing that only contains vertices of valences 4 and 8. The 4-8 mesh structure they introduce is shown in Figure 2.5. However, in such approaches, the target mesh is very specific and a new structured mesh replaces the original mesh, which might not be desired in some cases, specially when preserving small features is of great importance, or only local modifications are allowed. Our 5-6-7 remeshing algorithms can be considered as one of the pioneers of this category of remeshing.

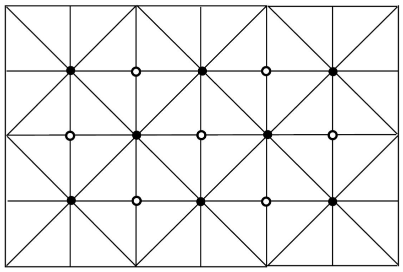


Figure 2.5: 4-8 mesh based on recursive subdivision. The figure is taken from [48]

2.2 Remeshing Methods

It is also interesting to classify remeshing algorithms based on their approach. Some of them use a relaxation approach to redistribute points, while others work in a parametrized domain. Several other works just rely on structured remeshing or directly editing the connectivity. This classification is important to us because we can analyze a variety of the previous works just based on their category. For instance, all the methods based on global parametrization suffer from distortion issues, and timely nature of parametrization.

2.2.1 Parametrization Based Methods

A noticeable portion of remeshing algorithms work based on re-sampling of the points on the mesh to achieve a well distributed point set. Some of these methods keep the connectivity intact while others update the connectivity after updating the point set.

These approaches usually rely on a parametrization of the original mesh. The point set is re-distributed in the parametrized domain and mapped back onto the surface of the mesh. For instance, [23] introduced an interactive remeshing of triangulated surfaces based on global parametrization. As it is shown in Figure 2.6, given the original mesh, the feature edges are extracted using a dihedral angle threshold (other feature extraction methods can be used as well). In the next step, the mesh is parameterized, while keeping track of the feature edges in the parameterized domain. Then we remove the mesh and we sample the parametrization area based on a specified density distribution. The density distribution can be determined by a user or automatically generated to respect the feature backbone. Subsequently, a 2D constrained triangulation is used to re-triangulate that domain and finally in order to improve the quality of the triangles, a fast optimization step is performed to perfect the final mesh by improving the quality of the triangles. This method provides a desirable control over the density distribution of the point set, which enables us to achieve a balance between uniform sampling and curvature sensitivity, by adding more points around the curvature backbone.

Unfortunately, dealing with large meshes, performing the global parametrization of the whole mesh is computationally costly. Additionally, on meshes with severe isotropic distortion, the global parametrization will be inaccurate due to numerical precision issues and the known inevitable distortion of global parametrization.

An alternative workaround to the global parametrization is to work directly on the surface of the mesh, and perform a series of local parametrizations on the mesh. One of the most famous state of the art algorithms based on local parametrization is the work by [43]. The mesh is divided into

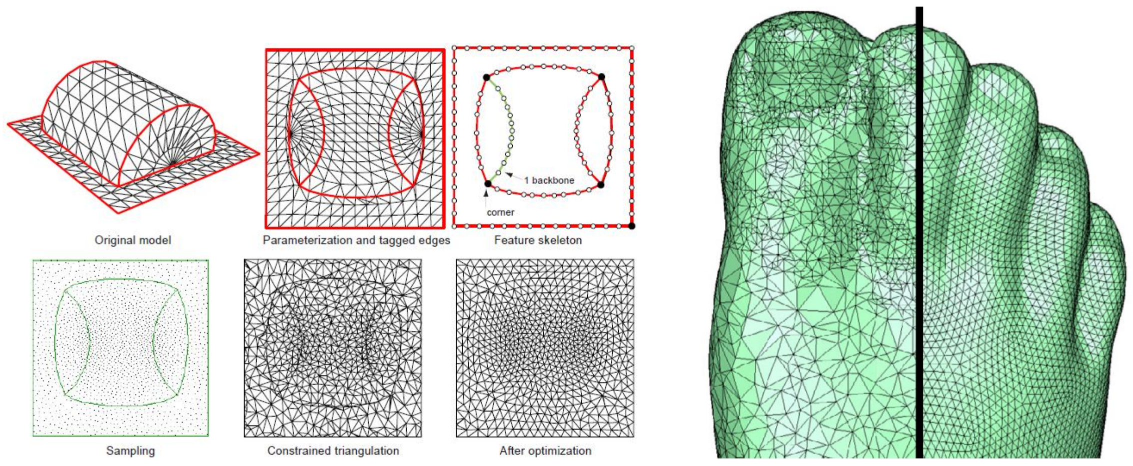


Figure 2.6: Left: the upper left figure shows the original mesh and the lower right figure depicts the final mesh in parametrized domain. Feature edges are colored in red. Right: the comparison between the original mesh and the resulting mesh. The Figure is taken form [3]

small overlapping patches and the vertex sampling and triangle quality of each patch is improved. In order to improve the sampling distribution of the points, they use an area-based remeshing approach that is done by an optimization with the goal of equalizing the area of the triangles. It is known that equalizing triangle areas does not necessarily mean achieving a desirable triangle quality but Surazhsky and Gotsman [42] discovered that a 2D triangulation with nearly equal triangle areas represents an almost uniform spatial vertex sampling. As depicted in Figure 2.7, starting with a patch, they use an optimization step to equalize the triangle areas as much as possible. Discarding the connectivity of the mesh, the more uniform distribution of the point set is noticeable. This new point set is re-triangulated using Delaunay triangulation (or you can just apply Delaunay edge flips). After a few iterations of area equalizing and re-triangulating, we achieve an almost uniformly distributed point set with triangles close to equilateral.

In a nutshell, parametrization provides us the 2D version of the problem which can be a much easier case, and different variety of algorithms can be utilized to improve the point set distribution. Another alternative to the area equalization algorithm would be the *Umbrella Operator*, which moves every vertex to the barycenter of its neighbors and updates mesh connectivity [49].

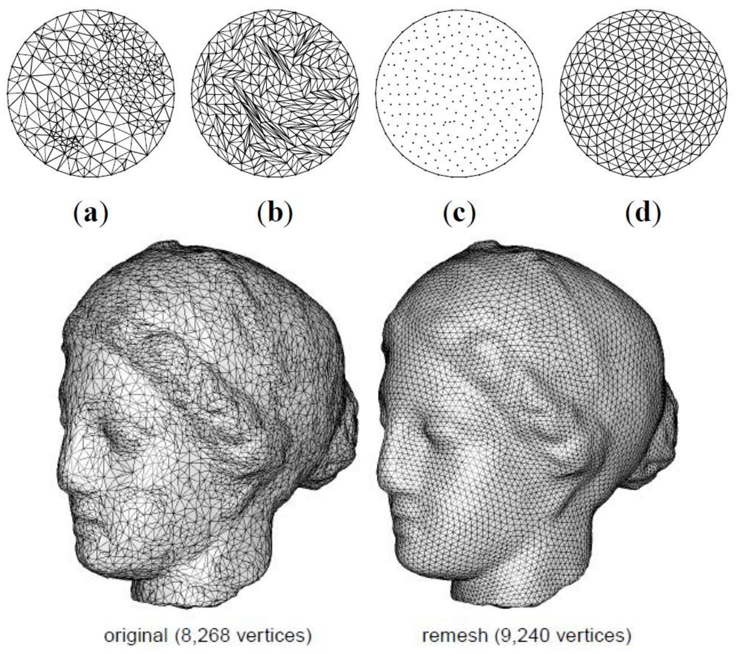


Figure 2.7: Remeshing based on area equalization: (a) The input mesh. (b) The area equalized mesh. (c) The point set after discarding the original edges. (d) The final mesh. The figure is taken from [43]

2.2.2 Relaxation Algorithms

An intuitive solution to achieve a good point distribution is relaxation of the points. One of the well-known methods is Lloyd relaxation [29], which produces an Isotropic placement of the points. Lloyd relaxation is an iterative approach that alternatively calculates the Voronoi cells and then moves the generator of each cell to its centroid. Lloyd relaxation produces Central Voronoi Diagrams [9], in which the centroid of each Voronoi cell coincides with its corresponding generator. The equal distribution of energy in each cell after applying Lloyd relaxation is shown by Gersho [15]. The iterations of Lloyd relaxation are shown in Figure 2.8.

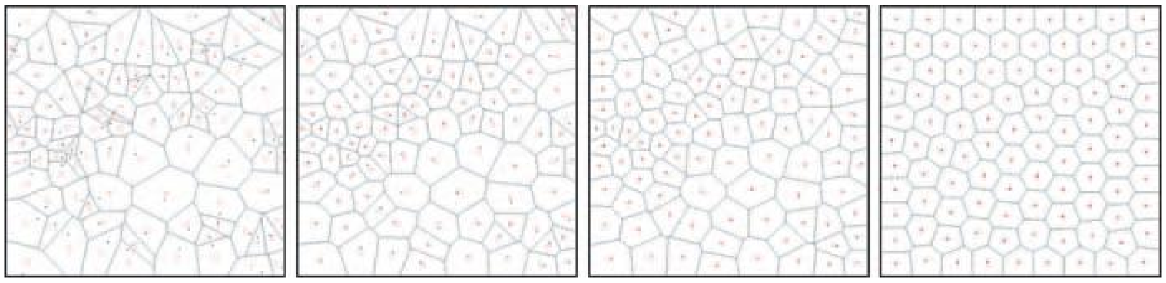


Figure 2.8: Lloyd relaxation. The Voronoi diagram of the given point set is shown on the left and the result, a central voronoi diagram, on the right. You can see the steps of relaxation in between. The figure is reproduced by [3] based on the original paper [9]

Two major remeshing works are based on Lloyd relaxation are by Alliez et al. [4] and Surazhsky et al. [41]. In order to use relaxation methods on 3D meshes, we have to either parametrize the mesh and perform the relaxation steps in 2D or adapt the relaxation algorithm to 3D by using the geodesic distance on triangle meshes to generate geodesic based Voronoi diagrams [36]. The result of the two mentioned works are shown in Figures 2.9 and 2.10, respectively.

2.2.3 Refinement Approaches

Despite many other approaches that strive to improve the mesh quality in several steps starting from the original mesh, Refinement algorithms use the original mesh only as a ground truth and try to approximate it with a very well-structured regular mesh, such as a grid or a very specific pattern. In a refinement approach we usually start with a base mesh, which is a very coarse representation of the original mesh, and recursively subdivide it to better approximate the original geometry.

Semi-regular meshes obtained from mesh refinement are ideal for multiresolution modeling/analysis, morphing, editing, visualization, and level of detail applications. Due to their very specific structure,

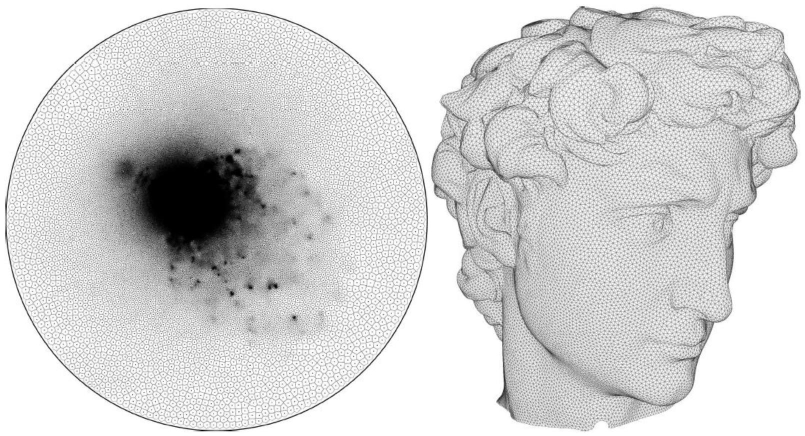


Figure 2.9: Isotropic surface remeshing, by Alliez et al. The figure is taken from [4]

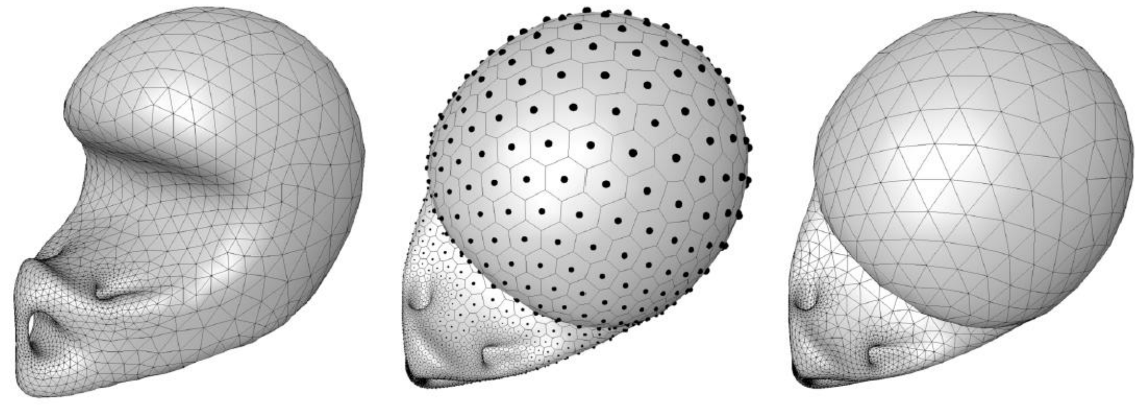


Figure 2.10: Isotropic remeshing of surfaces: a local parametrization approach by Surazhsky et al. The image is taken from [41]

it is possible to compress and store them very efficiently. One famous example is *4-8 Meshes* [48] that is used to model 3D shapes, terrain data, etc.

In most of the refinement algorithms, all of the recursively added vertices are regular vertices and the only irregular vertices are the ones from the base mesh. So, usually it is important to start with a good base mesh. Figure 2.11 represents an example of a refinement subdivision on quad meshes [3].

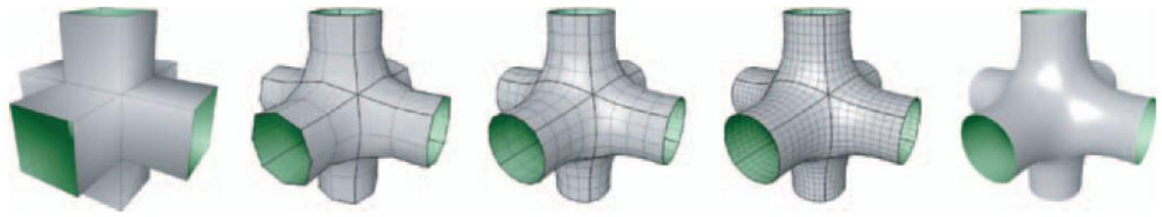


Figure 2.11: Refinement based remeshing. Starting from a base mesh on the left, the mesh is iteratively refined to the final smooth mesh on the left. The figure is taken from [3].

There are also other techniques such as [22], in which the mesh is taken into parametrized domain and then the whole domain is replaced by a very specific and regular 2D mesh such as a regular grid, and then projected back to the 3D space. This regular grid can be refined in different localities to adapt for multi-resolution purposes, or feature preservation applications. This method is shown in Figure 2.12.

Since in many cases the distortion from global parametrization is not tolerable, methods that don't utilize parametrization are very welcome, such as the work by Kobbelt et al. [25] that is based on ray shooting to find the correspondence and then shrink/warps the new mesh onto the initial mesh. Figure 2.13 shows how this approach works.

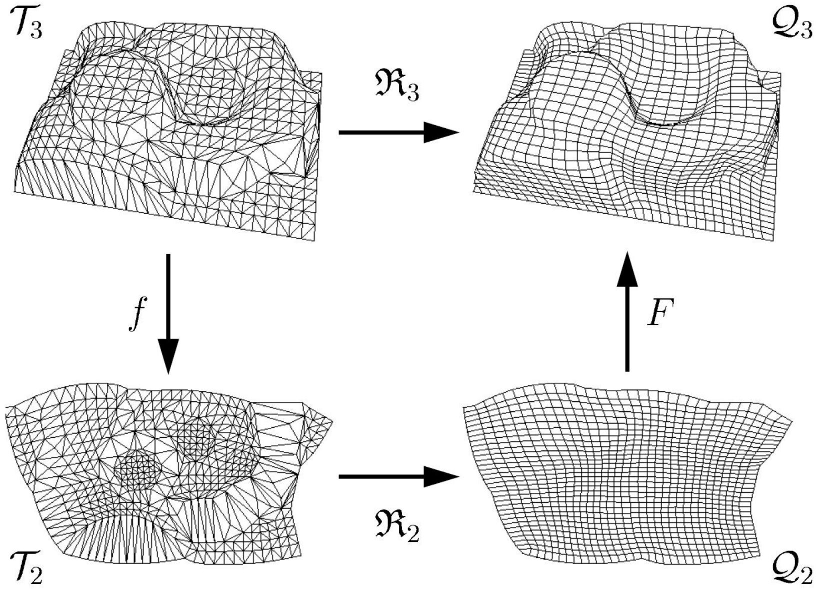


Figure 2.12: Refinement based remeshing based on regular grids. The figure is taken from [22]

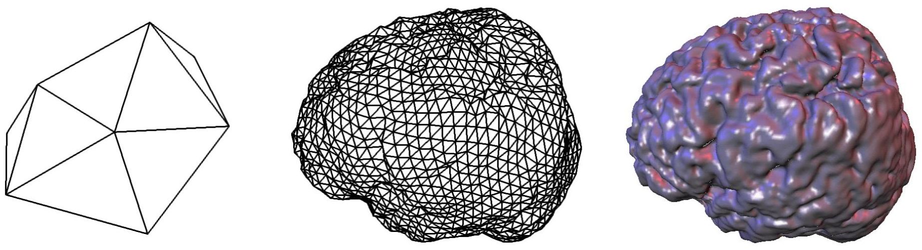


Figure 2.13: A refinement approach by Kobbelt et al. based on warping-shrinking the mesh to the original surface. The figure is taken from [25]

2.2.4 Connectivity Editting

Isenberg et al. [24] showed that there is an intrinsic connection between the geometry and connectivity of a manifold and they are not totally independent. They argued that connectivity encodes some information about the geometry of the mesh.Therefore, a low quality connectivity usually imposes a deficient geometry as well. Another evidence to support that theory is that low valence vertices occur at points with positive curvature, while high valencies are usually found at points with negative curvature i.e. saddle points. This claim can be verified by *Discrete Gaussian Curvature*, which is defined on a triangle mesh per every vertex as below, for a point p, with incident angles λ_i, and the geometry quantity E,

$$K = \frac{2\pi - \sum_i \lambda_i}{E}$$

Replacing λ_i by the average angle of $\frac{\pi}{3}$, for a vertex of degree d the formula becomes,

$$K = \frac{2\pi - \frac{d\pi}{3}}{E}$$

Clearly, discrete Gaussian curvature is positive for $d < 6$ and it is negative for $d > 6$, and vertices of degree 6 usually correspond to planar regions.

Having all of these said, it is now crucial to pay attention to the location of irregularities on a mesh. For example it might be desirable to move low valencies to positively curved locations on the mesh and high valencies to points with negative curvature. It might be also possible to cancel some low and high valencies with each other.

Two recent works by Li et al. [27] in 2010, shown in Figure 2.14 and Peng et al. [19] in 2011, shown in 2.15, followed this idea. They first provided some theoretical study of irregularities on triangle and quad meshes, and then provided a tool for the user to manipulate the connectivity of the mesh by interactively move irregularities around the mesh. However, in [27] having a 5-6-7 mesh to start with is a requirement. Their method to remove low/high valencies is manual and not very explicit how they can guarantee to end up with a 5-6-7 mesh.

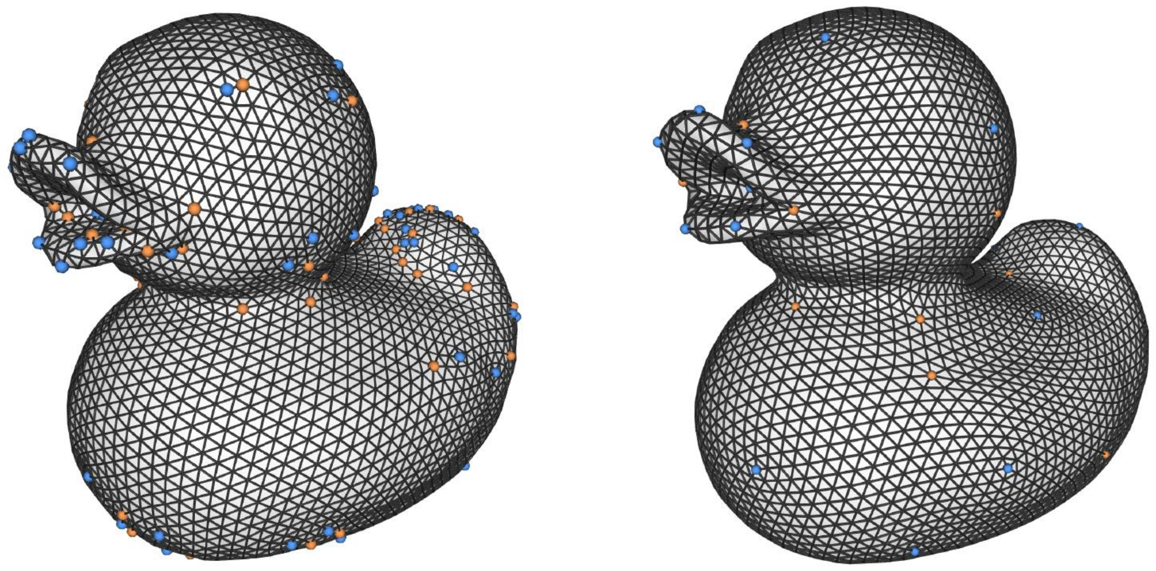


Figure 2.14: Connectivity Editing on triangle meshes. The figure on the left is the input mesh, and the figure on the right is the output of their algorithm. Some of the irregular vertices are eliminated or moved around on the mesh. The Figure is taken from 2.14

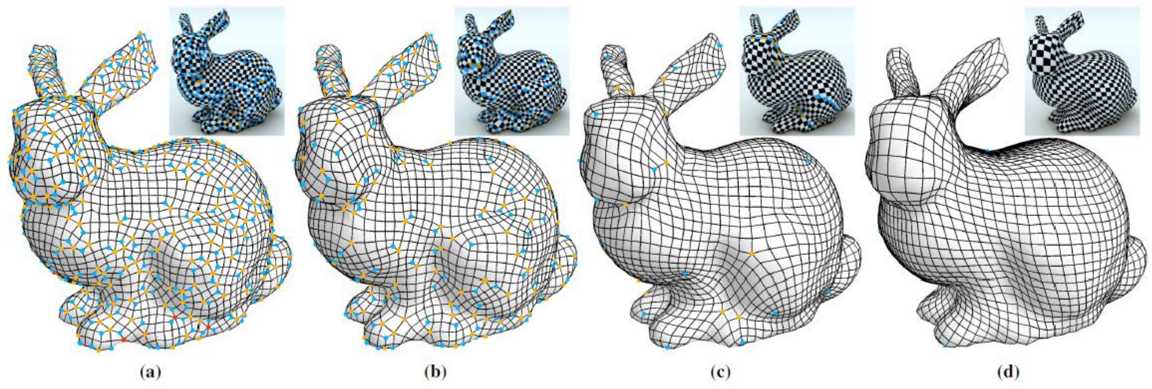


Figure 2.15: Connectivity Editing on quad meshes. The figure on the left is the input mesh and irregular vertices are cancelled out or moved around towards the right figure. The Figure is taken from 2.15

Chapter 3

Remeshing to 5-6-7 mesh

In this chapter, we present a set of local remeshing schemes, and a subdivision step to convert an arbitrary closed manifold triangle mesh to a 5-6-7 mesh. We start by eliminating vertices of degree 3 or 4 by specific patterns that only affect a very local region around those vertices, without introducing any error to the shape of the mesh.

After the low valency removal step, we target high valence vertices and replace each of them by a set of 5-6-7 vertices. This step needs to be preceded by a subdivision step, to guarantee a certain level of separation between irregularities. While the subdivision step does not change the shape at all, vertex splits may produce an error, which is arbitrarily low.

3.1 Removing low-valence vertices

We start by removing all the low-valence vertices. During this step, all the v3 and v4 vertices are eliminated while the region outside the 2-ring neighbourhood of each target vertex is kept intact. The k-ring neighborhood of a vertex is the subgraph induced by the set of vertices with distance at most k from the target vertex. It is important to note that in this step no error is added to the shape of the initial model.

Perhaps the most straightforward approach to remove a valence-3 vertex is to split an edge of one of its incident faces and connect its midpoint to both this vertex and also the other vertex on the opposite side of the edge, as shown in Figure 3.1. Although this solution works but it converts the v3 vertex into a v4 vertex and also introduces a new v4 vertex. Since we aim to eliminate v4 vertices in the next step, this solution is not very efficient. There are also other approaches to remove a $v3$ vertex but they either generate $v4$ vertices or they add too many new vertices to the mesh.

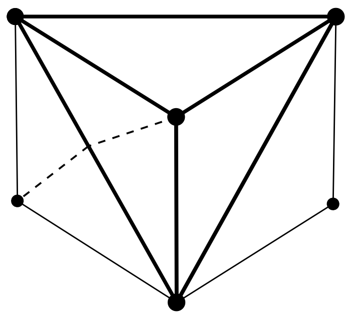


Figure 3.1: A simple scheme to remove a vertex of valence 3

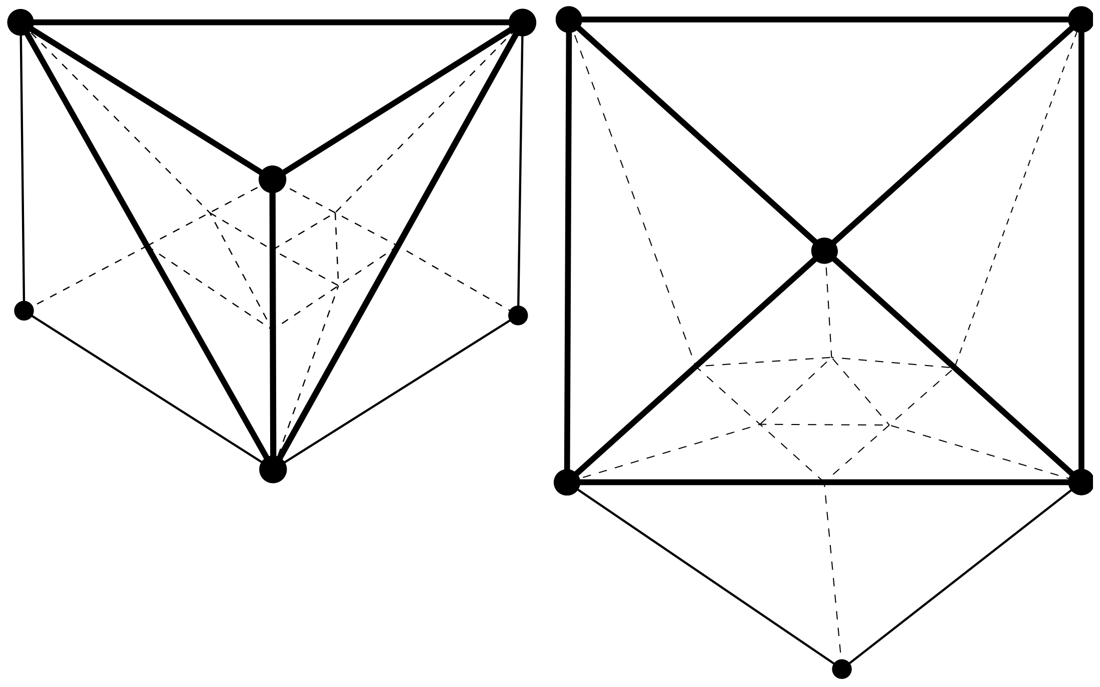


Figure 3.2: Elimination of a vertex with valence 3 (left) or 4 (right); the low-valence vertex is marked by a dark dot. Solid lines represent edges in the original mesh and dashed lines are the new edges added by the remeshing. Note that the original mesh edges are exactly preserved geometrically hence no change to mesh geometry occurs with the remeshing.

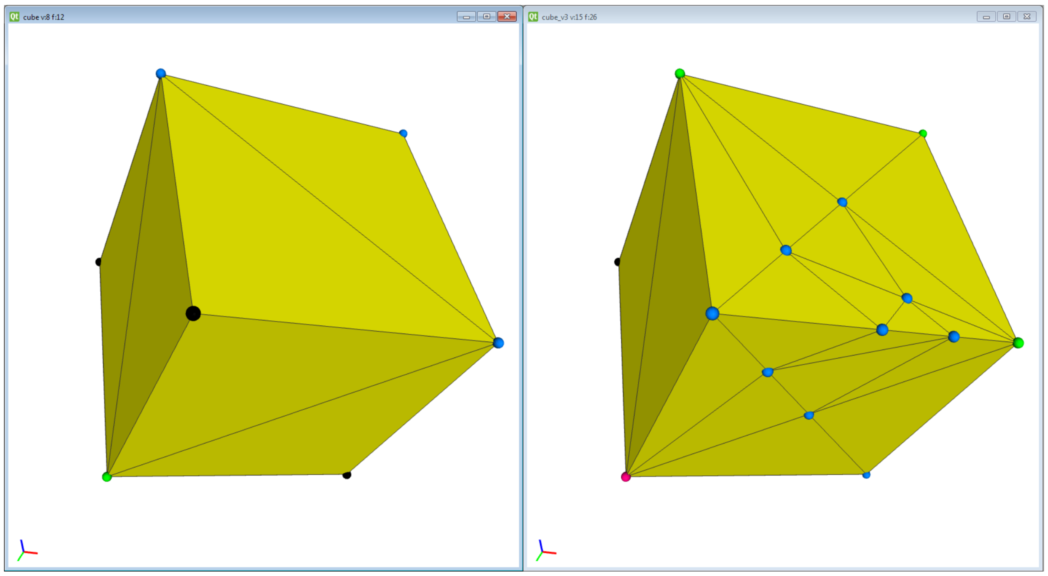


Figure 3.3: V3 removal on a 3D surface. On the left is the input mesh with a degree 3 vertex and on the right is the mesh after replacing the v3 vertex.

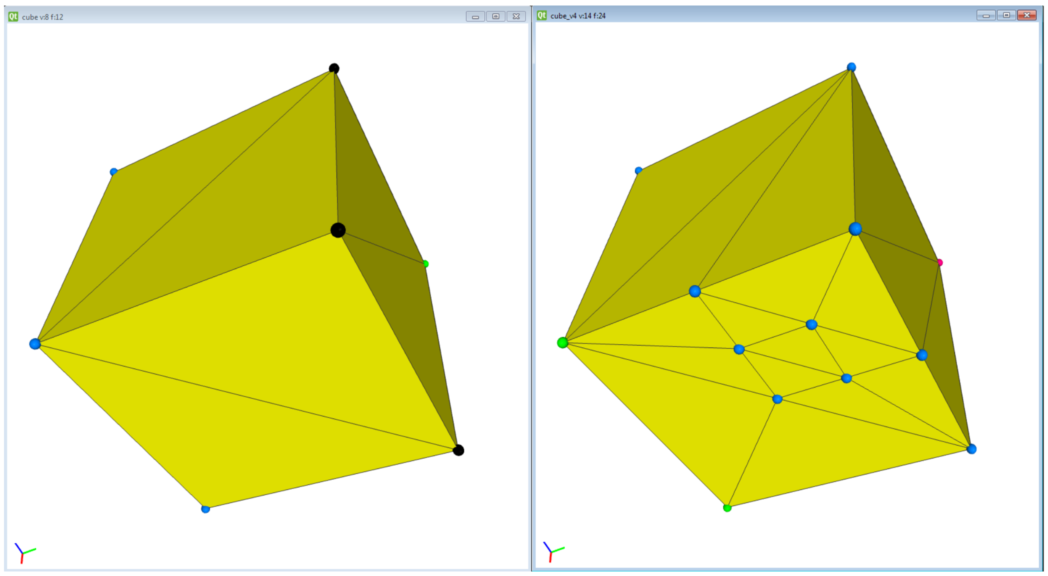


Figure 3.4: V3 removal on a 3D surface. On the left is the input mesh with a degree 4 vertex and on the right is the mesh after replacing the v4 vertex.

We have chosen to replace all the vertices of valence 3 or 4 by a 5-6-7 structure, as shown in Figure 3.2. It can be observed that these two structures can be adapted to any arbitrary geometry corresponding to a $v3$ or $v4$ vertex without introducing any geometric error and without reducing the valence of any other vertex in the mesh. As a result, no new $v3$ or $v4$ vertices would be generated. Figures 3.3 and 3.4 exhibit the result of removing low valencies in our algorithm on the mesh of a cube. As you can observe, no error is added to the shape of the mesh and all the new vertices and faces are on the original surface of the mesh. In addition, our effort to minimize the modification to the locality of the target vertex ends up having a non-symmetric structure for V3 removal. However, it does not affect the final results very much since the decimation step will modify the connectivity of the vertices and the geometry enhancement modify the geometry of the points.

3.2 Subdivision

In order to remove high-valence vertices, our method requires those vertices to be far apart from each other connectivity-wise. More specifically, any vertex with a valence higher than 7 should have a one-ring neighbourhood consisting of only regular vertices ($v6$). Often this is not the case, and in order to guarantee this condition, we apply a planar subdivision to subdivide all faces of the mesh into 9 smaller faces, as shown in figure 3.5.

This condition is strictly required because in the next step, high valency removal, we replace every vertex with degree higher than 7 with a 5-6-7 structure, and we make sure that none of the vertices on the one-ring of that vertex will receive an additional valency of more than one. Although this step of the algorithm does not change the shape of the mesh, it increases the size of the mesh, in terms of vertex/triangle count. More precisely, it multiplies the number of faces by 9.

3.3 Removing high-valence vertices

In order to remove a high-valence vertex, we iteratively split it to v7 vertices until the remaining vertex has a degree less than or equal to 7. More specifically, we denote the vertex with degree > 7 as our *pivot*. Next, we replace the pivot with a vertex of degree 7 and another vertex of degree $deg(pivot) - 3$. The new vertex of degree $deg(pivot) - 3$ becomes our new pivot and we repeat the process until the degree of the pivot becomes less than or equal to 7.

Moreover, it is important to notice that after each vertex split, the valency of the two other vertices adjacent to them is increased. Therefore, several vertex splits might result in one or more

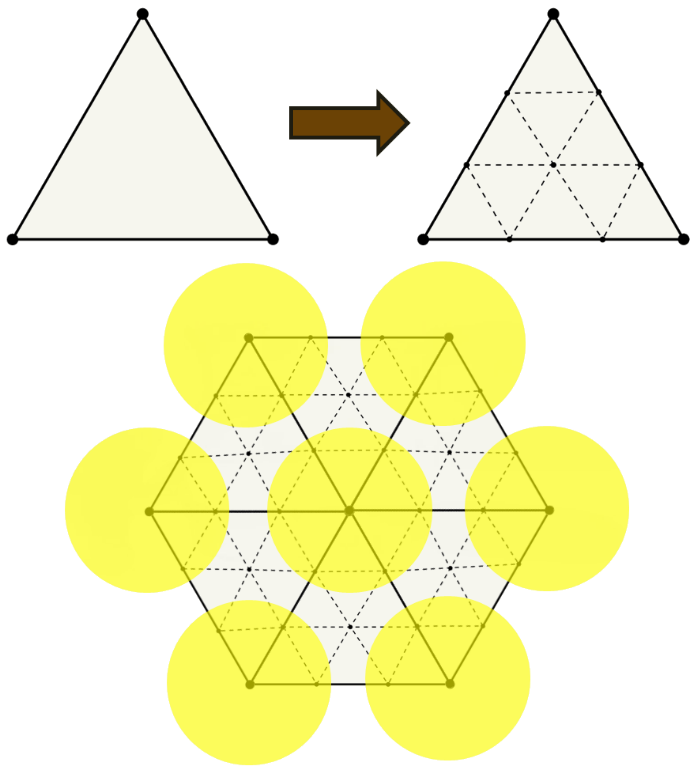


Figure 3.5: Topological planar subdivision scheme to keep all the high-valence vertices sufficiently far apart from each other, i.e., they are separated by at least two *v*6 vertices, and every initial vertex is surrounded by a unique set of *v*6 vertices.

high valency in that locality. In order to avoid this issue, we flip one of the newly generated edges after splitting the vertex. Also, the series of flips should happen on the same side at every split. Figure 3.7 illustrates the iterations involved for removing a vertex of valence 14. The newly added edges are in colored in red and it is crucial to notice that the degree of none of the vertices in the one ring of the original high valency vertex is increased by more than one. This is an important point because it guarantees that on the one-ring of that vertex we will not have any vertex of degree higher than 7.

It can be shown that a vertex of valence h, where $h > 7$, can be replaced by $\lfloor \frac{h-2}{3} \rfloor - 1$ vertices with valence 7, and one v567 vertex, while increasing the valence of the vertices in the one-ring neighbourhood by at most one (refer to the appendix for the proofs). Since the subdivision process has provided each high-valence vertex with a unique one-ring of regular vertices, the high-valence removal step replaces high-valence vertices with a 5-6-7 structure without introducing new high or low valence vertices.

In terms of positioning of the newly created vertices, although we can move all of them to the same location of the original high valence vertex and introduce no geometric error, the resulting mesh will have a degenerate geometry, which is not desirable. To solve this problem, we initially place all the newly created vertices at the position of the initial vertex, which is degenerate. Then we iteratively move each vertex towards the centroid of its adjacent vertices, while giving the vertex itself a higher weight to keep it close to its initial position. The new positions are iteratively calculated by the following equation:

$$u_{i_{new}} = \frac{\alpha u_i + \sum\limits_{p \in N(u_i)} p}{\alpha + |N(u_i)|},$$

where u_i and $u_{i_{new}}$ are the positions of the i-th vertex before and after each iteration, $N(u_i)$ is the set of adjacent vertices for vertex u_i and α is a constant weight, which is chosen to be 50 in our implementation. Higher values of α keeps vertices u_i close to their original position and a value of zero moves them to the centroid of their neighbours. Since at each iteration two of the degenerate vertices are pulled away, for k newly added vertices, we require a minimum of $\lfloor \frac{k}{2} \rfloor$ iterations.

This positioning of the vertices might distort the mesh around u_i, which can be controlled by the value of α. However, the decimation and geometry enhancement process in the next step will move these vertices around in a geometry-aware manner. So, this equation is just used to create a non-degenerate mesh without fold-overs to start with.

It is worth noting that there are cases where it is inevitable to tolerate some error for the high valence removal step, unless if we move all the generated vertices to the location of the original vertex, resulting in a degenerate geometry that has no geometric error. For example, imagine the shape of the tip of a juice reamer, or any fan-shaped one-ring with non-coplanar incident faces (Figure 3.6). In these cases our method smoothes out the tip of the reamer a little. First we replace the target vertex by the 5-6-7 structure and position them very close to the initial vertex. Then, the decimation and geometry enhancement steps will decide about the proper positioning of the new points.

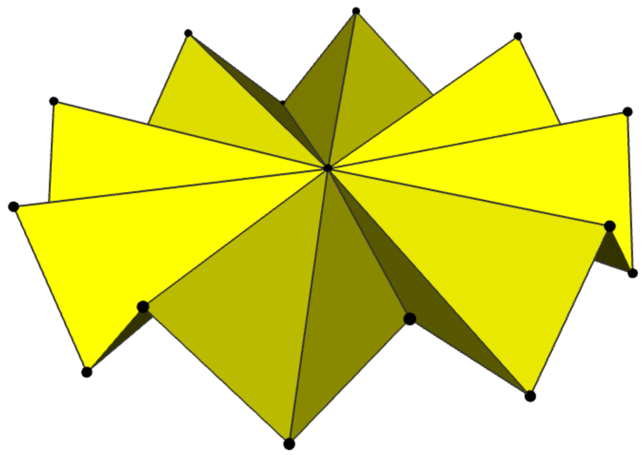


Figure 3.6: The fan-shaped one-ring of a high valence vertex. It is observable that no non-degenerate alternative structure with lower valency can replace this structure without modifying the shape in that region.

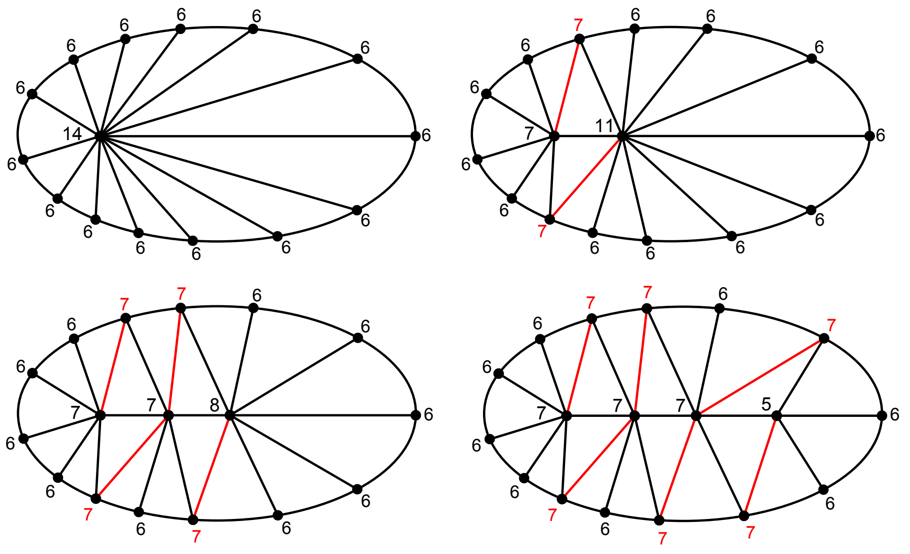


Figure 3.7: Elimination of a high-valence vertex ($v17$)is done by a series of vertex splits. It results in the addition of several $v7$ vertices and one $v567$ vertex, while increasing the valence of some of the vertices on the boundary by one (thus from $v6$ to $v7$).

Chapter 4

Decimation and Enhancement

So far we have created a 5-6-7 mesh by slightly changing the geometry. However, during this process, we have increased the size of the mesh by a factor of approximately 10. Therefore, a *5-6-7 preserving* simplification algorithm is applied with an attempt to reduce the size of the final mesh back to the size of the initial mesh, while preserving the 5-6-7 property, and respecting the shape of the initial model.

4.1 Mesh decimation

We simplify the mesh using the edge collapse simplification, and we only allow those edge collapses, that still preserve the 567 property of the mesh. As shown in Figure 4.1(left), an edge collapse between vertices v_1 and v_2, with valences d_1 and d_2, creates a merged vertex of valence $d_1 + d_2 - 4$ (Lemma 1) and reduces the valence of vertices u_1 and u_2 by 1. Therefore, if either u_1 or u_2 has a valence of 5, we cannot collapse the edge. Moreover, we should maintain the condition $5 \leq d_1 + d_2 - 4 \leq 7$ or in other words, v_1 and v_2 can be either $v5$-$v5$ or $v5$-$v6$. We apply edge collapse mesh decimation governed by Quadric error measurement, while considering the 5-6-7 preserving constraints.

The simplification might stop at some point without any more possible edges to collapse. At this point, we iterate over all the edges of the mesh and mark those that create more collapsible edges, as shown in Figure 4.1 (right) and then we flip all of them. However, allowing an arbitrary edge to be collapsed may be dangerous and can result in a substantial error. Therefore we only allow an edge to be flipped if the dot product of the normal of its incident faces is beyond a user defined threshold, that varies between 0 and 1. We also perform the same check for the adjacent faces of the flipped edge. In our implementation a threshold of 0.9 is used. Note that increasing this threshold allows less edges to be flipped for the sake of a lower geometric error. One observation here is that we usually have many flips without any geometric error because the subdivision step generated many adjacent coplanar faces, for which flipping an edge between them will not introduce error.

The decimation process will alternatively decimate the mesh and then flip edges, until either the target face count is achieved or no more edges can be flipped.

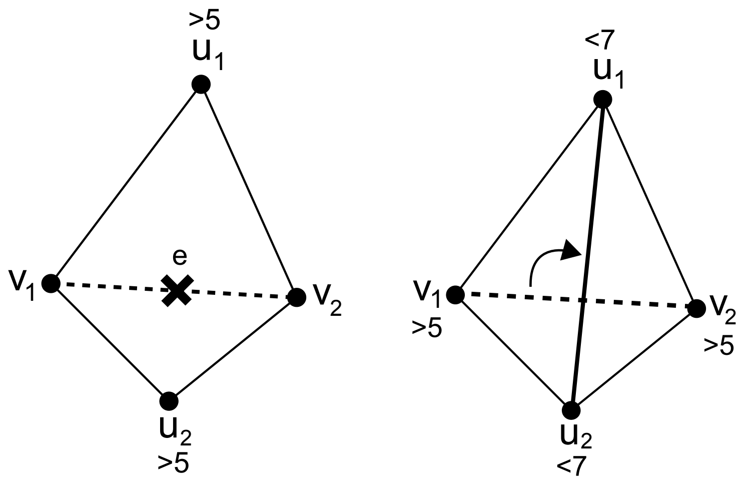


Figure 4.1: Edge collapse (left) and edge flip (right) that preserve the 5-6-7 property.

4.2 Geometry enhancement

Although the decimated mesh maintains the 5-6-7 connectivity, the quality of the resulting mesh is not always desirable. Due to the edge flips and also because of the constraints on the quadric simplification, the decimation process is not able to collapse some of the low error edges and instead it has to consider the next candidate edges. Besides the geometric error, we observed that the decimation process decreased the quality of the triangles as well, by creating very small or long triangles. In order to address these issues, we apply an enhancement heuristic to relax the points on the surface of the original mesh, while respecting the geometry features.

More specifically, we apply a Laplacian smoothing followed by a back projection of the points to the surface of the original mesh. In order to project the points back to the original surface, we first find a mapping from the points of the decimated 5-6-7 mesh to the points of the original mesh. The mapping is constructed by considering the closest vertex from the original mesh to the vertex on the decimated mesh. Then, every vertex is projected back to the closest point on the one-ring neighbourhood of its corresponding vertex on the original mesh, which is the closest point to one of the triangles that is incident to its corresponding vertex.

After projecting the points back to the original mesh, we may need to update the correspondence that we calculated earlier, since the vertices are moving around. Let $c[u]$ be the corresponding vertex for u in the original mesh, we update $c[u]$ to the closest point from $N(c[u]) \cup \{c[u]\}$ to u, where $N(c[u])$ is the set of vertices in the one-ring of $c[u]$. The one-ring neighborhood of a vertex is the set of vertices, faces and edges incident to the target vertex. Here, $N(c[u])$ only refers to the incident vertices.

A drawback of this heuristic is that it smoothes out feature points after each iteration. To work around this, we detect feature points and fix their positions. We first examine the normals of every vertex's incident faces, and mark that as a feature vertex if there are two faces f and f' with $dot(\vec{n_f}, \vec{n_{f'}}) < \sigma_{threshold}$. The value of $\sigma_{threshold}$ is determined by the user. For instance, in Figure 4.2, a small positive value will work to fix the vertices on the edge of the cylinder. We also provide an interface to visualize the marked feature vertices as the user changes the value. A small value for $\sigma_{threshold}$ lets some of the feature points disappear but allows a better distribution of the points on the surface of the mesh and a larger value will fix more points, making the mesh less flexible towards changes.

In cases where the increased size of the mesh is not an issue, for example when the initial mesh has a low face count, we can bypass the decimation step and run the geometry enhancement right after we create a 5-6-7 mesh. In such a case, the size of the mesh is increased but the process will become faster as the decimation step is the slowest step of our algorithm, and we also will not have the extra error added by the decimation step. The exact time complexity, empirical time measurements and corresponding statistics about the decimation step is provided in the results section. Figure 4.2 illustrates a cylinder before and after converting it to a 5-6-7 mesh (without decimating) followed by the geometric enhancement step. So, the final mesh has significantly more faces than the original one but the high valence vertex is replaced by a set of 5-6-7 vertices, which are relaxed on the surface after the geometry enhancement.

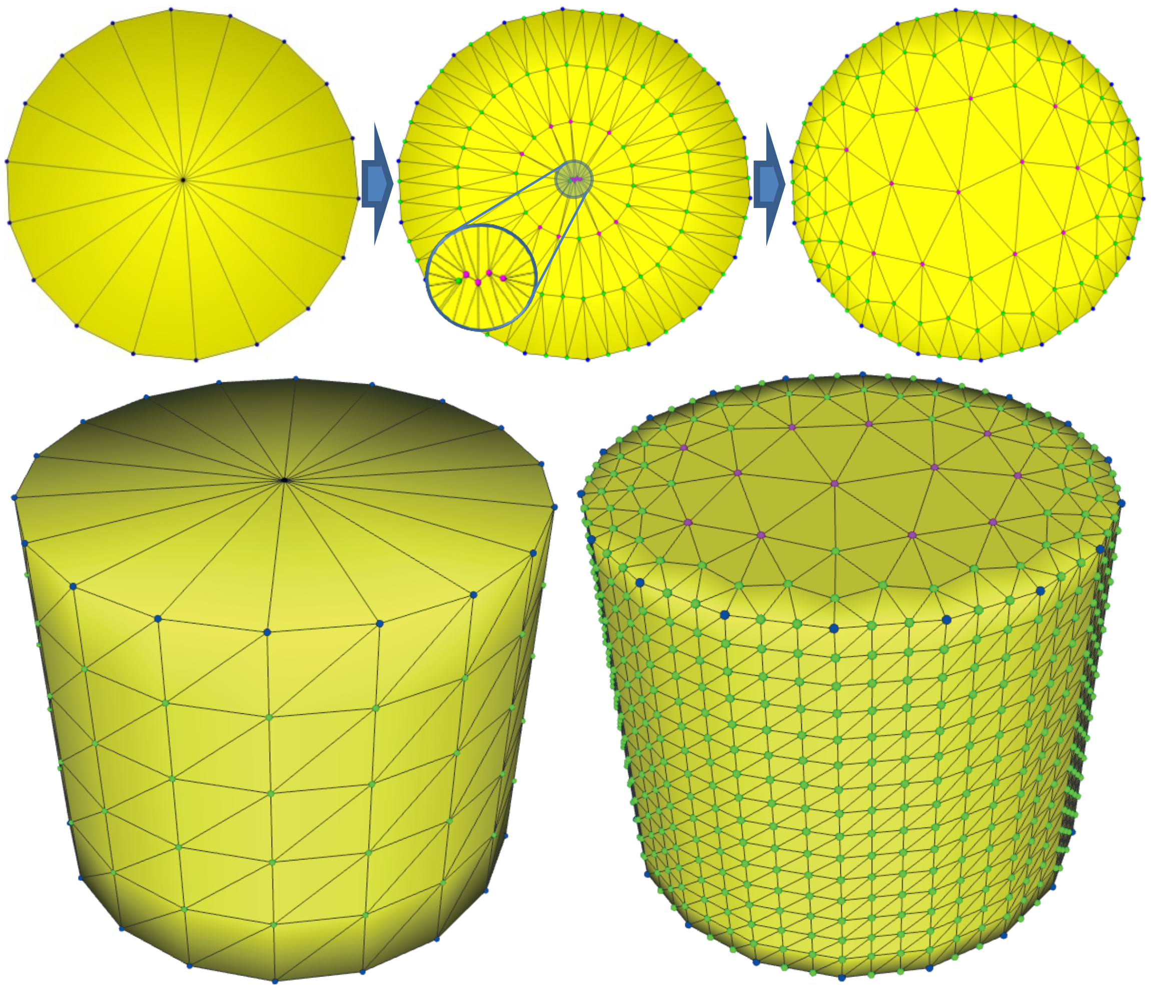


Figure 4.2: Comparison of the original cylinder mesh(left) with the 5-6-7 enhanced mesh(right). The top row from left to right: the original, 5-6-7, and 5-6-7 enhanced mesh (without decimation).

Chapter 5

Results

Our implementation of 5-6-7 remeshing takes a closed triangle mesh and turns it into a 5-6-7 mesh, which is simplified afterwards and then geometrically enhanced. Although the 5-6-7 remeshing has a linear time complexity, the decimation process can be fairly slow and take minutes to run on a mesh with 25K vertices. Although quadric mesh decimation has a well defined time complexity, the constrained quadric decimation that is used in our work does not have a known complexity. Its running time highly depends on the initial triangulation and how lucky we are in the progression of the quadric decimation. Sometimes the mesh can be decimated to the initial size, and sometimes it takes several switches between edge collapses and edge flips to achieve the desired size.

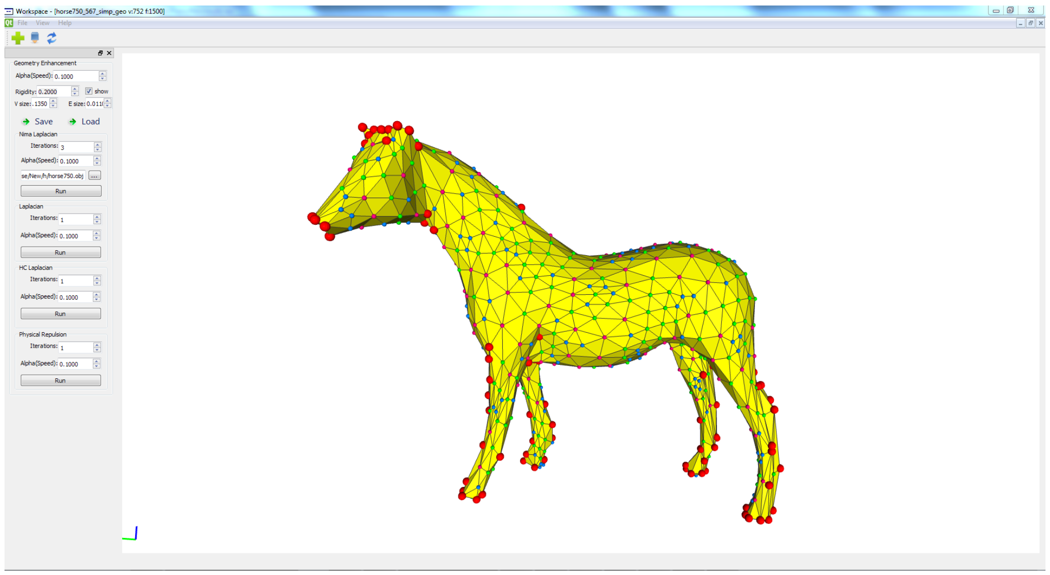
5.1 Parameters

Our 5-6-7 remeshing does not require any parameter tuning or user interaction to produce the 5-6-7 mesh, except for α. This parameter is used to control the distance of the split vertices from the original location of the high pivot vertex. Value of zero corresponds to geometric degeneracy and zero error, and higher values will add distortion in that locality to the mesh.

During the decimation step, we switch between edge collapses and edge flips, and edge flips can produce geometric error. Therefore, only edges with almost flat incident faces are allowed to be flipped. To detect the flatness, the dot product of the normal of the faces has to be below some threshold β, for that edge to be legit for a flip.

Also, the geometry enhancement step needs two parameters: s and r. Parameter s governs the speed of Laplacian smoothing. A value of 1.0 would be the basic Laplacian smoothing. And, parameter r is the rigidity factor, that is used in the feature detection step. A higher value for the rigidity factor implies a more rigidly constrained feature detection, which results in detection of more feature points. However, a lower value will let more vertices of the mesh to move around, which increases the flexibility for the cost of smoothing out some feature areas.

In our implementation, we used the values of $\alpha = 50$, $\beta = 0.1$, and $s = 0.1$ which are fixed for all of the experiments. Only the value of r needs to be tuned for each model. We provide an interactive tool for the user to pick an appropriate value for r as shown in Figure 5.1. Most of the times a value between 0.1 and 0.2 is selected.



Workspace - [horse750_567_simp_geo v:752 f:1500]

Geometry Enhancement
Alpha(Speed): 0.1000
Rigidity: 0.2000 ✓ show
V size: .1350 E size: 0.011

Nima Laplacian
Iterations: 3
Alpha(Speed): 0.1000
se/New/h/horse750.obj

Laplacian
Iterations: 1
Alpha(Speed): 0.1000

HC Laplacian
Iterations: 1
Alpha(Speed): 0.1000

Physical Repulsion
Iterations: 1
Alpha(Speed): 0.1000

Figure 5.1: Our interactive interface for the user to pick an appropriate value for the rigidity factor. Vertices detected as features are highlighted in red color with a larger size.

5.2 Test data

We tested our algorithm on meshes with different topologies and different types of features. Figure 5.4 demonstrates the result of applying our method on a genus two manifold and Figure 5.2 exhibits our feature preservation mechanism. For the geometry enhancement, the user needs to set the rigidity factor ($\sigma_{threshold}$) and the number of iterations. Since picking the rigidity factor is not always intuitive, an interactive tool assists the user to pick a reasonable value.

We also incorporate edge lengths in quadric errors such that, shorter edges become better candidates to be collapsed. This attempts to eliminate small dense groups of vertices and tends to equalize the edge lengths of the mesh, which gives a more uniformly distributed vertex set.

Model	Metro error	#Faces	Valence					Time
			V_{low}	V_5	V_6	V_7	V_{high}	
fish	0.010	1K	8.9%	24.5%	36.1%	22.5%	8%	3.3s
horse	0.018	1.5K	7.7%	27.7%	33.6%	22.1%	8.9%	4.5s
venus	0.020	1.5K	9%	25.7%	35.6%	19%	10.7%	4.2s
eight	0.010	1.5K	5.1%	23.2%	42.6%	23.9%	5.2%	5.1s
cow	0.020	6K	3.2%	17.9%	61.7%	12.5%	4.7%	18.7s
armhand	0.007	25K	3.6%	26.2%	42.7%	22.2%	5.3%	13m

Table 5.1: Various statistics related to 5-6-7 remeshing on several meshes. Metro error is computed between the original and the final mesh. Time represents the total running time of the 5-6-7 remeshing algorithm and decimation. The number of vertices (in percentages) of different valences before remeshing is also included.

In order to quantify the quality of the resulting mesh, we compute an approximation error measured by the well-known Metro tool [8], which calculates the Hausdorff distance between the original mesh and the final mesh after geometry enhancement. The error is mostly a result of the simplification process, and 5-6-7 remeshing itself does not add significant error to the mesh. The error as well as various other statistics related to our remeshing algorithm, such as the execution time, are shown in table 5.1.

5.3 Evaluation

In this section we would like to perform two more evaluations to show the advantages of 5-6-7 meshes. Although we compared some example meshes with their 5-6-7 version, a wise mind would still argue that this comparison is a bit biased because the initial mesh might have a very bad vertex positioning, and we are comparing it with a mesh that is geometrically enhanced at the end of the algorithm. So, the fair evaluation would be comparing the 5-6-7 mesh with the geometrically enhanced version of the initial mesh. In other words, starting with a sample triangle mesh, we apply our 5-6-7 remeshing algorithm and all the post processing, and we also apply our geometry enhancement algorithm on the initial mesh, to see the fair comparison. Figure 5.6 and 5.7, show the aforementioned comparison for the horse and fish meshes, respectively.

As you can clearly observe, the existence of irregular vertices, especially in the flat regions of the mesh, enforces a non-uniform distribution of the points even after applying the Laplacian relaxation on the surface of the mesh (which is our geometry enhancement). Although the geometry enhancement step tries to improve the point distribution on the surface of the mesh, it is evident that

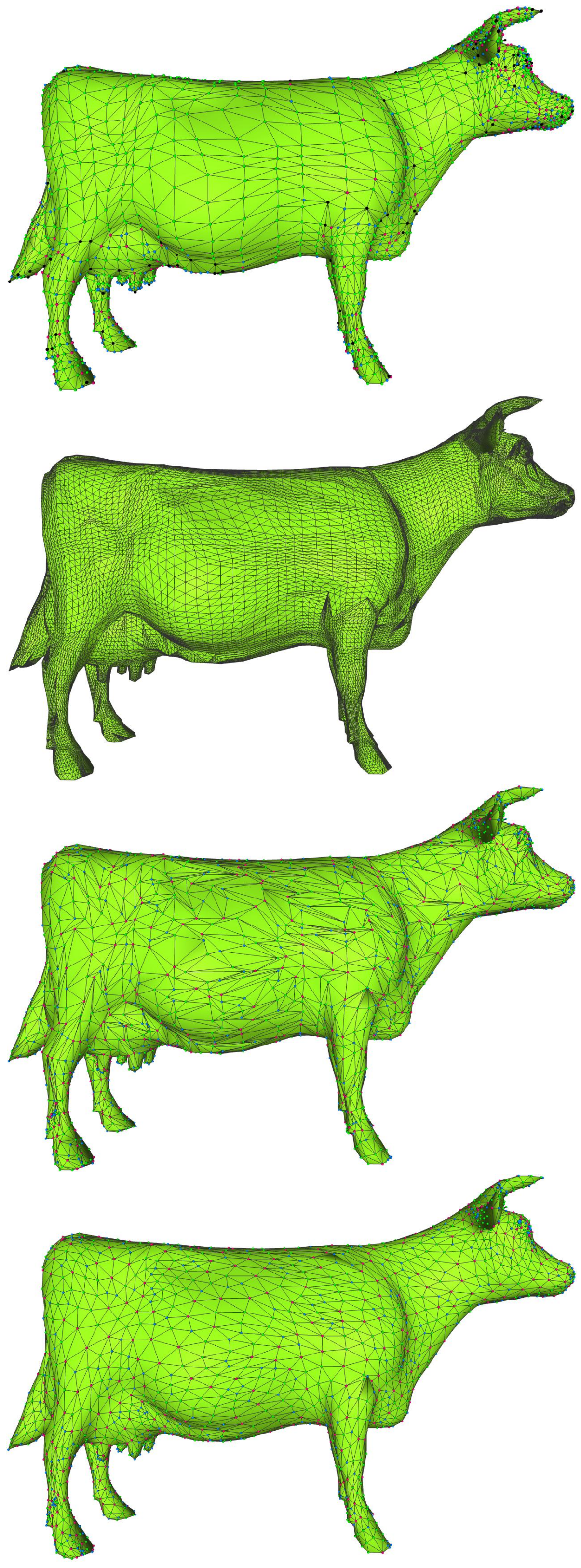


Figure 5.2: An example of respecting sharp features on a complex mesh. From top to bottom: the original mesh, the 5-6-7 mesh, the decimated 5-6-7 mesh and finally the enhanced 5-6-7 mesh.

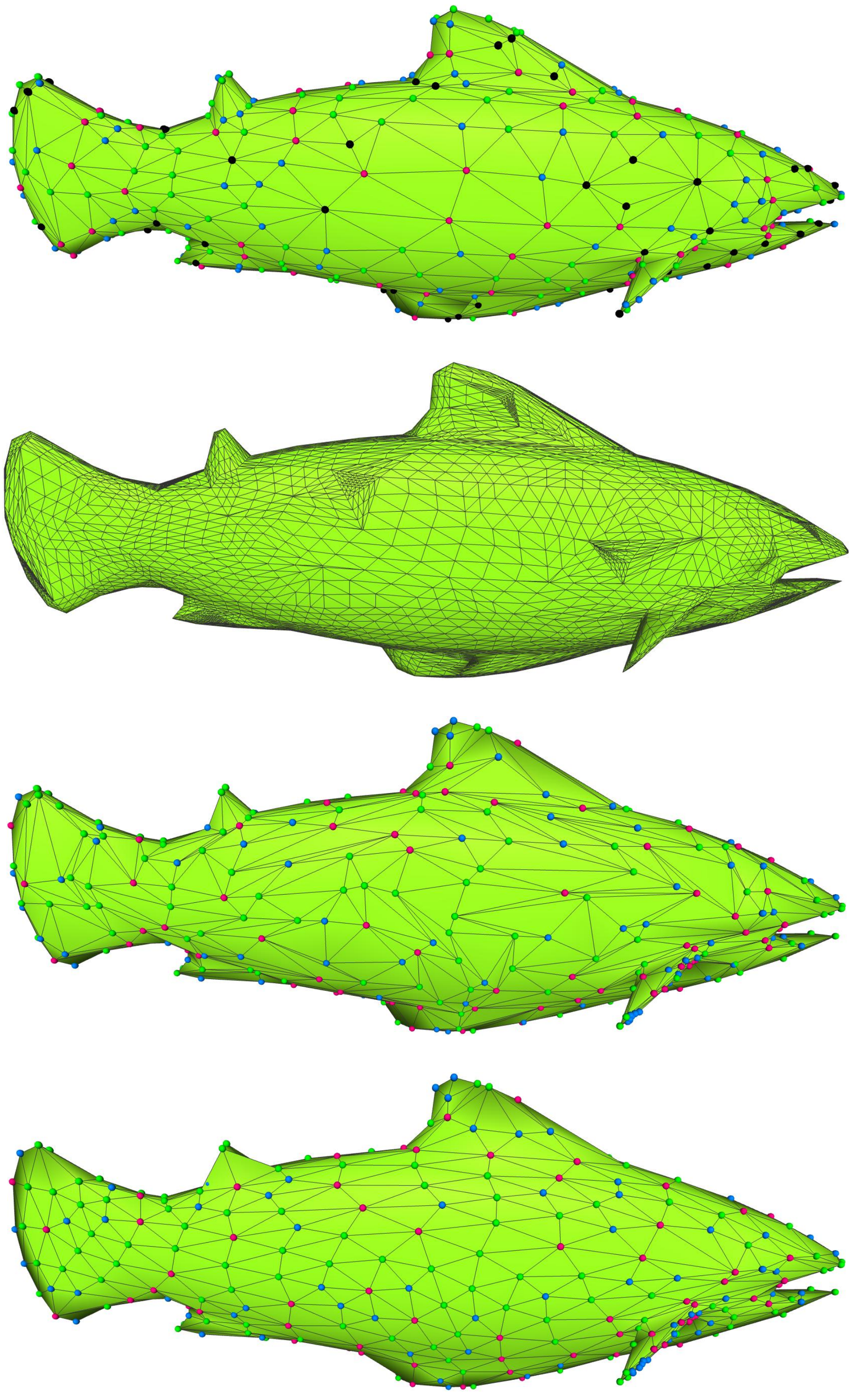


Figure 5.3: 5-6-7 remeshing of Fish mesh with fairly flat areas as well as some sharp feature regions. From top to bottom: the original mesh, the 5-6-7 mesh, the decimated 5-6-7 mesh and finally the enhanced 5-6-7 mesh.

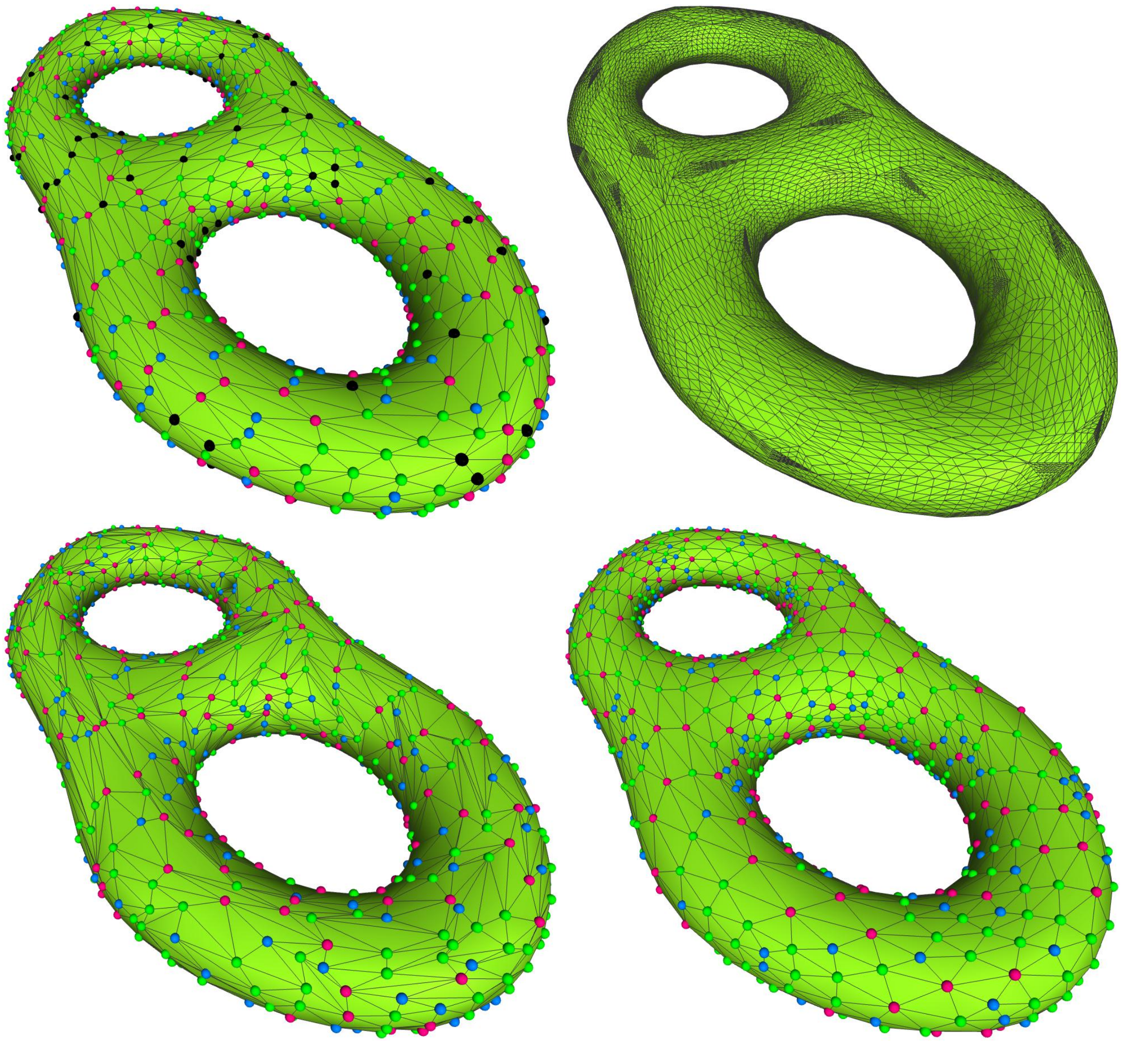


Figure 5.4: 5-6-7 remeshing on a genus 2 mesh. From top left to bottom right: the original mesh, the 5-6-7 mesh, the decimated 5-6-7 mesh and finally the enhanced 5-6-7 mesh.

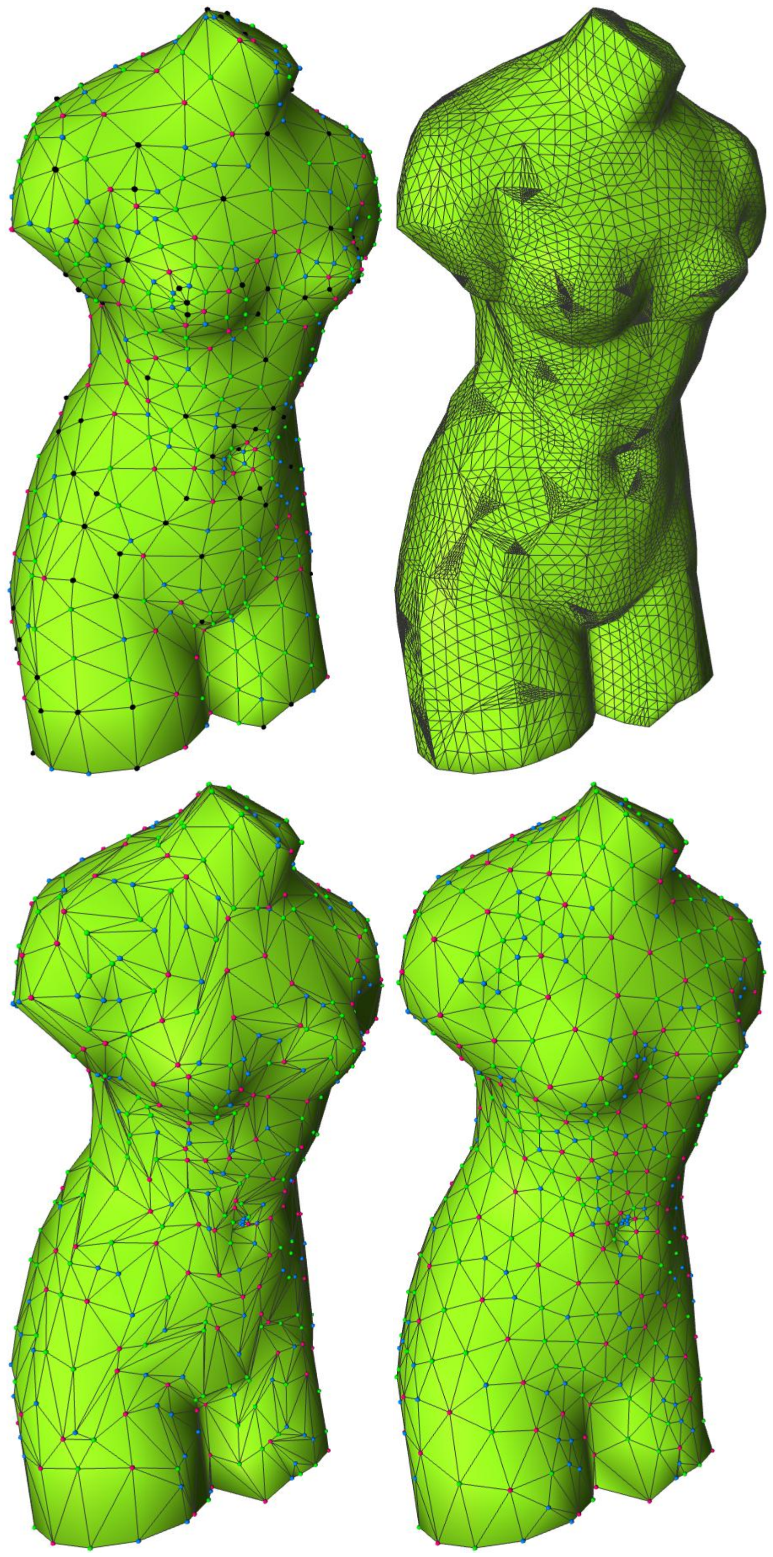


Figure 5.5: 5-6-7 remeshing of Venus mesh with subtle feature areas. From top left to the bottom right: the original mesh, the 5-6-7 mesh, the decimated 5-6-7 mesh and finally the enhanced 5-6-7 mesh.

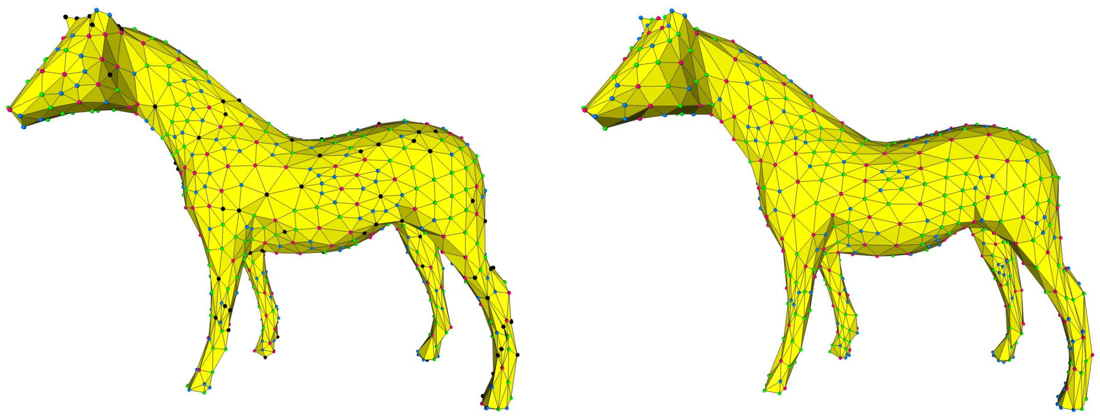


Figure 5.6: Comparison between the geometrically enhanced version of the initial mesh and the 5-6-7 mesh. On the left you see the initial mesh after running the geometry enhancement algorithm on it, versus the 5-6-7 mesh on the right.

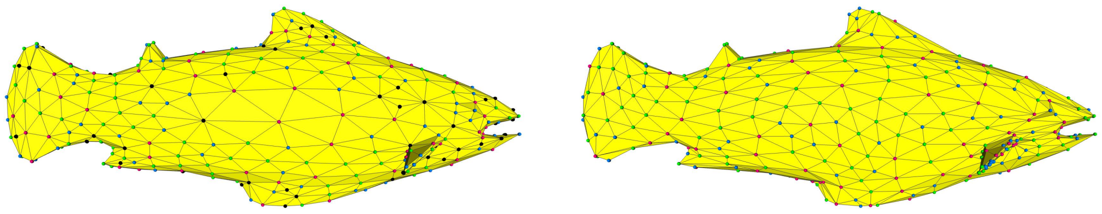


Figure 5.7: Comparison between the geometrically enhanced version of the initial mesh and the 5-6-7 mesh. On the left you see the initial mesh after running the geometry enhancement algorithm on it, versus the 5-6-7 mesh on the right.

the bad point distribution is inherent in the low quality connectivity graph. For example, take a look at the v3 vertices near the eye of the fish, or the high valence vertex on the center of the fish's body.

Our second evaluation is the comparison of the initial mesh and the 5-6-7 mesh after applying subdivision algorithms. We used Loop subdivision [30] to subdivide the faces of both meshes for 2 steps. Figure 5.8 shows the results of Loop subdivision on both the original mesh and the 5-6-7 mesh. Also, Figure 5.9 presents the same results on the sphere, with material, lighting and under different shading algorithms. You can observe the wrinkles generated in the smooth shaded version of the original mesh after applying subdivision, which is inevitable around high valence vertices after applying subdivision algorithms. Although in our 5-6-7 mesh we have more non-valence-6 vertices(rather than two extremely high valence vertices) but since their valences are bounded well enough, we have less visual artifacts appearing on a 5-6-7 mesh.

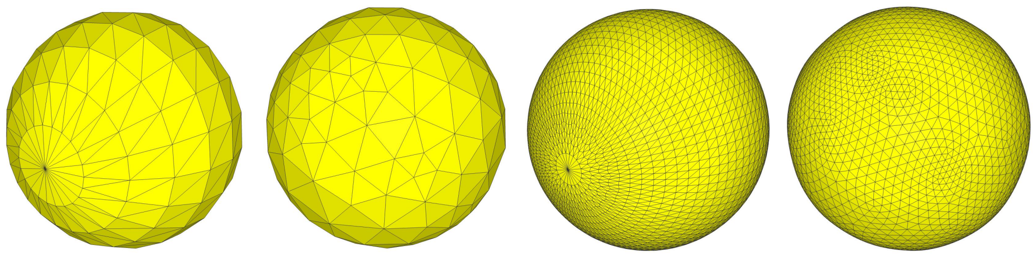


Figure 5.8: Results of subdivision on a sphere and the 5-6-7 sphere. Figures from left to right: 1) initial sphere 2) 5-6-7 sphere with the same vertex count 3) initial sphere after applying 2 steps of Loop subdivision 4) 5-6-7 sphere after applying 2 steps of Loop subdivision.

5.4 Limitations

Time Complexity: One of the drawbacks of our remeshing algorithm is the time complexity of the decimation step. Although other parts of the algorithm have a linear time complexity and run efficiently, the decimation step requires alternating between the edge collapse and edge flips, till it reaches the original mesh size. In the worst case, there might be $O(E)$ number of switches between these two steps, and edge collapses and edge flips have a time complexity of $O(ElogE)$ and $O(E)$, respectively, where E is the number of edges in the mesh. In total a time complexity of $O(E^2logE)$ in total, which can be easily reduced to $O(E^2)$ by selecting the best collapsible edge at each step in $O(E)$, rather than constructing the priority queue of edges. However, in most of the experiments, we observed that at each edge collapse iteration, the size of the mesh is reduced to half. Which reduces

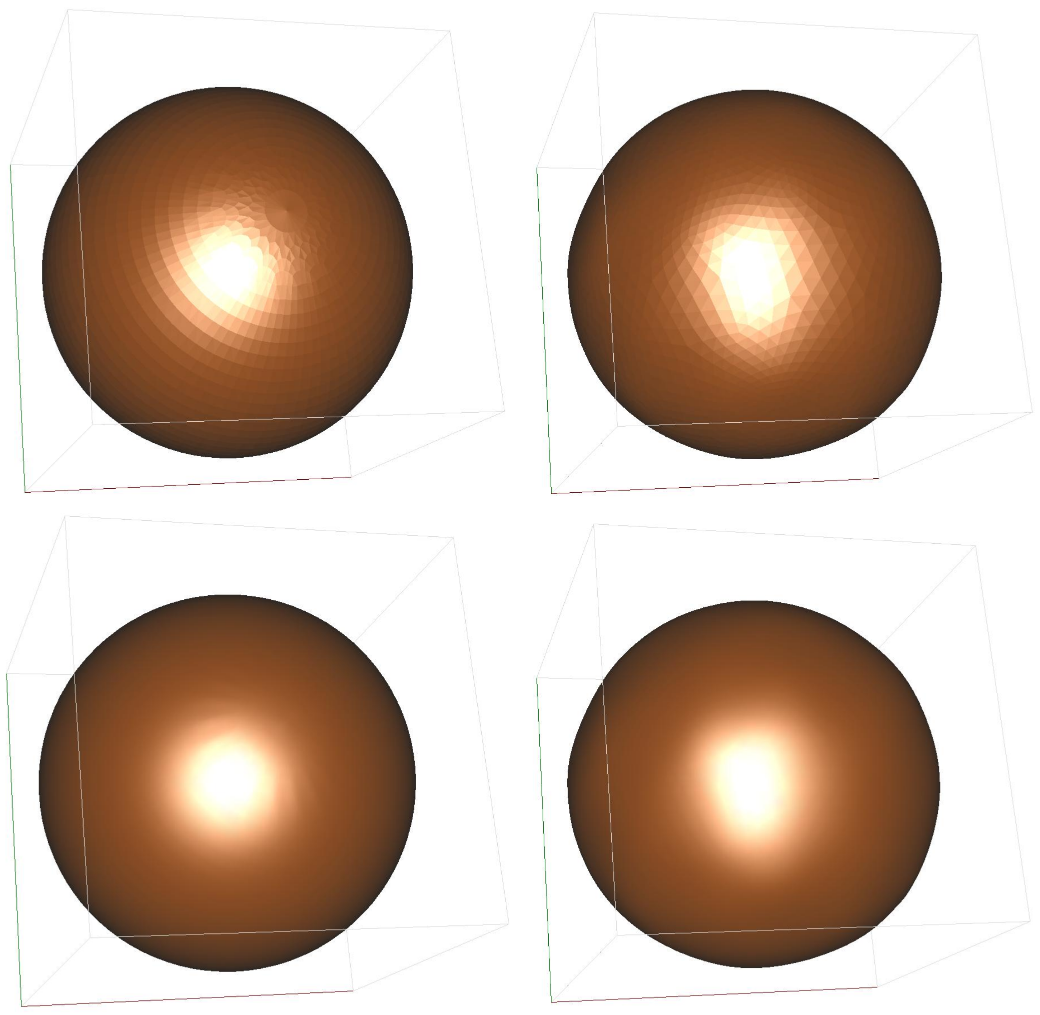


Figure 5.9: Shaded results of subdivision on a sphere and the 5-6-7 sphere. The two figures on the top are the flat shaded sphere and 5-6-7 sphere, and the two figures on the bottom are the same spheres shaded smoothly.

the time complexity back to $O(ElogE)$.

Subdivision Step: Our algorithm requires one subdivision step to guarantee a required level of separation between the irregularities. This subdivision step multiplies the size of the mesh by 9, which is not desirable for most of the large meshes. Although it does not change the complexity of our algorithm (since 9 is a constant multiplier) but of course it slows down the following steps and also this is one of the main reasons we require a decimation step afterwards.

Geometry Enhancement Step: The final step of our algorithm, geometry enhancement, is required to both improve the triangle qualities and to reduce the possible distortion produced by the decimation step. We basically apply this step to show that in a 5-6-7 mesh we can achieve a better vertex distribution and besides the theoretical results of our work, the visual aspect is improved as well. However, this step needs some human interaction to pick the proper rigidity threshold. Parameter tuning on one hand, and the possible face fold overs on the other hand are the main downsides of this step. Face fold overs can happen for two reasons: a) face fold over is a known downside of Laplacian relaxation, in general. b) our feature preservation fixes several vertices, which reduces the flexibility of the mesh and increases the possibility of producing face fold overs. One possible improvement would be allowing feature vertices to move along the feature creases. Since this step is considered mainly as a post processing on our 5-6-7 remeshing, we considered the implementation of feature line adaptation as an option.

Chapter 6

Conclusion and future work

6.1 Conclusion

In this thesis, we show that a closed triangle mesh with an arbitrary genus can always be converted into a 5-6-7 mesh, a mesh with only valence-5, 6, and 7 vertices. The initial conversion scheme removes low- and high-valence vertices one by one and during the process, it creates a fairly large number of new vertices. We address this issue with a mesh decimation and geometry enhancement algorithm, while preserving the 5-6-7 property. In the end, we obtain a 5-6-7 mesh that closely approximates the original mesh, i.e. features are respected and has a comparable vertex count. However, It might not always be possible to decimate the mesh to its original face count, and, in some cases it is even theoretically impossible to decimate the mesh to its original face count (e.g. a tetrahedron or a great stellated dodecahedron, which are some of the worst cases of our algorithm, as shown in Figure 6.1).

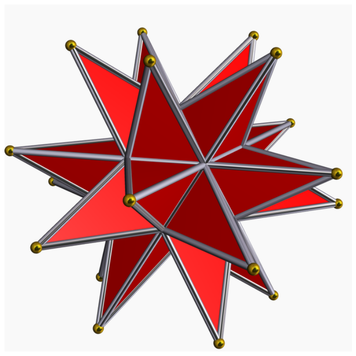


Figure 6.1: The great stellated dodecahedron. (taken from Wikipedia)

A summary of our approach can be divided into the following four steps:

1. **Low Valence Removal:** v3 and v4 Vertices are removed efficiently without introducing error.

2. **Subdivision and High Valence Removal:** Vertices with valence greater than 7 are removed and an arbitrarily low error is introduced. An error of zero is achievable in a degenerate geometry. By the end of this step, we have a 5-6-7 mesh that has a low geometric error and the error only appears at high valence vertices. However, the size of the mesh is multiplied by approximately 10.

3. **Decimation:** In this step we decimate the mesh towards the initial face count as much as possible, while preserving the 5-6-7 property. This step is computationally expensive and might introduce a considerable error to the geometry.

4. **Geometry Enhancement:** We use a heuristic to improve the quality of the triangles, and projecting the mesh back to the surface of its original mesh. To preserve the features of the mesh, we detect them and fix their position. While this heuristic works in many cases, there are situations that it might generate a considerable error.

6.2 Future Work

In the following we present some possible future directions and some interesting problems we encounter during this work, which can be taken as a future work.

- **Low Valency Removal** It is interesting to either provide a more efficient low valency removal scheme or to prove that the low valency removal algorithm we used is the most efficient. Intuitively, in order to make the scheme optimum, the new vertices should have as low valence as possible (which is valency 5 in a 5-6-7 mesh). The schemes we used only introduces v5 vertices but the main challenge is taking into account the increased valencies in the neighborhood of the original vertex.

- **Local vs. Global Remeshing** In this work, we ignore the geometry and correct the connectivity mostly in local areas, then we position the new elements such that minimum error is added to the geometry. However, there is a strong innate relationship between the geometry and connectivity [24], and based on this connection, a more promising approach to 5-6-7 remeshing might be taking into account the shape of the model in the first place to produce a rather global remeshing.

- **Separating Irregularities** One drawback of our approach is the subdivision step that is necessary to separate irregular vertices from each other. It still remains an intriguing question to find a less global approach to surround irregular vertices with unique regular vertices.

- **Geometry Enhancement** We utilized a rather straightforward approach for our Geometry Enhancement, which is based on the combination of Laplacian smoothing and a heuristic, which requires parameter tuning. A more solid work on geometry enhancement would be a profound contribution to the area of mesh processing.

- **5-6 and 6-7 Remeshing** The formula for the average vertex valency over a mesh suggests full regularity on a tessellation of genus 1, less than 6 for genus < 1, and greater than 6 for genus > 1. This clue suggests an even more constrained remeshing, which is 5-6 remeshing for genus < 1, and 6-7 remeshing for genus > 1. And, of course a fully regular remeshing for genus = 1.

Appendix

Lemma 1. *The result of merging two vertices of degrees d_1 and d_2 is a vertex of degree $d_1 + d_2 - 4$.*

Lemma 2. *On a closed manifold M with the connectivity graph G_M, let $V(G_M)$ be the vertex set of G_M and let $V' \subseteq V(G_M)$. Also, let ω be the result of merging V' into one vertex. The degree of ω is given by the following formula, if the subgraph G' induced by V' is a tree.*

$$deg(\omega) = \sum_{v'_i \in V'} deg(v'_i) - 4\,(|V'| - 1) \tag{1}$$

Proof. The proof can be done by induction over the size of the tree and using lemma 1. □

Lemma 3. *The remaining vertex from the splitting algorithm is a 5-6-7 vertex.*

Theorem 1. *Every vertex of degree h, $h > 7$, will be replaced by $\lfloor \frac{h-2}{3} \rfloor - 1$ number of $v7$ vertices and one 5-6-7 vertex.*

Proof. Using lemma 3, we know that the remaining vertex will be a 5-6-7 vertex. Now suppose that after the split procedure we have $|V'|$ vertices consisting of $|V'| - 1$ number of $v7$ vertices and one 5-6-7 vertex. By merging these vertices together we will recover the original high valence vertex, which had a degree of h. Let $h = deg(\omega)$ and let υ_{567} be the remaining vertex from the splitting algorithm. Now, using lemma 2 we have,

$$\begin{aligned}
h &= \sum_{v'_i \in V'} deg(v'_i) - 4\,(|V'| - 1) \\
&= (7\,(|V'| - 1) + deg(\upsilon_{567})\,) - 4\,(|V'| - 1) \\
&= 3\,(|V'| - 1) + deg(\upsilon_{567}) \\
|V'| &= \frac{h - deg(\upsilon_{567}) + 3}{3} \qquad (2) \\
&= \frac{(h-2) - t}{3} \qquad (3)
\end{aligned}$$

Where $t \in \{0, 1, 2\}$ and it makes $h - 2$ divisible by 3. So,

$$|V'| = \left\lfloor \frac{h-2}{3} \right\rfloor \tag{4}$$

Which is $|V'| - 1$ vertices of degree 7 and one 5-6-7 vertex. □

Bibliography

[1] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Lévy, and Mathieu Desbrun. Anisotropic polygonal remeshing. *ACM Trans. on Graphics*, pages 485–493, 2003.

[2] Pierre Alliez, Mark Meyer, and Mathieu Desbrun. Interactive geometry remeshing. *ACM Trans. Graph.*, pages 347–354, 2002.

[3] Pierre Alliez, Giuliana Ucelli, Craig Gotsman, and Marco Attene. Recent advances in remeshing of surfaces. In *Shape Analysis and Structuring*, pages 53–82. Springer Berlin Heidelberg, 2008.

[4] Pierre Alliez, Éric Colin de Verdière, Olivier Devillers, and Martin Isenburg. Isotropic surface remeshing. *Proc. IEEE Conf. on Shape Modeling and Applications*, pages 49–58, 2003.

[5] Ursula H. Augsdorfer, Neil A. Dodgson, and Maclolm A. Sabin. Removing polar rendering artifacts in subdivision surfaces. *journal of graphics, gpu, and game tools*, pages 61–76, 2009.

[6] Suddha Basu and Jack Snoeyink. Terrain representation using right-triangulated irregular networks. In *CCCG*, pages 133–136, 2007.

[7] C. Gotsman C. Touma. Triangle mesh compression. In *Proceedings of Graphics Interface*, page 2634, 1998.

[8] P Cignoni, C Rocchini, and R Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, pages 167–174, 1998.

[9] Q. Du, V. Faber, and M. Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM review*, pages 637–676, 1999.

[10] Nira Dyn, Kai Hormann, Sun-Jeong Kim, and David Levin. *Optimizing 3D triangulations using discrete curvature analysis*, pages 135–146. Vanderbilt University, 2001.

[11] Michael S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, pages 231–250, 1997.

[12] Pascal J. Frey. About surface remeshing. *Proceedings of 9th International Meshing Roundtable*, 2000.

[13] Pascal J. Frey and Houman Borouchaki. Geometric surface mesh optimization. *Computing and Visualization in Science*, pages 113–121, 1998.

[14] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, 1997.

[15] A. Gersho. Asymptotically optimal block quantization. *Information Theory, IEEE Transactions on*, pages 373–380, 1979.

[16] Ozgur Gonen. Modeling planar 3-valence meshes. Master's thesis, Texas A&M University, 2007.

[17] X. Gu, S.J. Gortler, and H. Hoppe. Geometry images. *ACM Transactions on Graphics*, pages 355–361, 2002.

[18] Igor Guskov, Kiril Vidimče, Wim Sweldens, and Peter Schröder. Normal meshes. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 95–102, 2000.

[19] Chi han Peng, Eugene Zhang, Yoshihiro Kobayashi, and Peter Wonka. Connectivity editing for quadrilateral meshes. *ACM Trans. on Graphics (Proc. SIGGRAPH Asia)*, 2011.

[20] Hugues Hoppe. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, 1996.

[21] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 19–26, 1993.

[22] K. Hormann and G. Greiner. Quadrilateral remeshing. In *Proceedings of Vision, Modeling, and Visualization*, pages 153–162, 2000.

[23] K Hormann, U Labsik, and G Greiner. Remeshing triangulated surfaces with optimal parameterizations. *Computer-Aided Design*, pages 779–788, 2001.

[24] Martin Isenburg, Stefan Gumhold, Craig Gotsman, Stefan Gumhold, and Craig Gotsman. Connectivity shapes. In *In IEEE Visualization 2001 Conference Proceedings (2001*, pages 135–142, 2001.

[25] L.P. Kobbelt, J. Vorsatz, and U. Labsik. A shrink wrapping approach to remeshing polygonal surfaces. In *Computer Graphics Forum*, pages 119–130, 1999.

[26] J. Y. S. Li and H. Zhang. Nonobtuse remeshing and mesh decimation. In *Proc. Symp. on Geom. Processing (SGP)*, pages 235–238, 2006.

[27] Yuanyuan Li, Eugene Zhang, Yoshihiro Kobayashi, and Peter Wonka. Editing operations for irregular vertices in triangle meshes. In *ACM Trans. on Graphics (Proc. SIGGRAPH Asia)*, pages 153–165, 2010.

[28] Rong Liu and Hao Zhang. Mesh segmentation via spectral embedding and contour analysis. *Computer Graphics Forum (Special Issue of Eurographics)*, pages 385–394, 2007.

[29] S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, pages 129–137, 1982.

[30] C. Loop. Smooth Subdivision Surfaces Based on Triangles. Department of mathematics, University of Utah, 1987.

[31] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM Trans. on Graphics (Proc. SIGGRAPH)*, pages 163–169, 1987.

[32] David Luebke, Benjamin Watson, Jonathan D. Cohen, Martin Reddy, and Amitabh Varshney. *Level of Detail for 3D Graphics*. Elsevier Science Inc., 2002.

[33] Oleg R. Musin. Properties of the delaunay triangulation. In *Proceedings of the 13th annual symposium on Computational Geometry*, pages 424–426, 1997.

[34] Renato Pajarola. Overview of quadtree-based terrain triangulation and visualization overview of quadtree-based terrain triangulation and visualization. *Evaluation*, pages 1–16, 2002.

[35] J. Peters and X. Wu. The distance of a subdivision surface to its control polyhedron. *J. Approx. Theory*, pages 491–507, 2009.

[36] G. Peyré and L. Cohen. Surface segmentation using geodesic centroidal tesselation. In *Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium*, pages 995–1002, 2004.

[37] A. Rassineux, P. Villon, J. M. Savignat, and O. Stab. Surface remeshing by local hermite diffuse interpolation. In *International Journal for Numerical Methods in Engineering*, 2000.

[38] P.V. Sander, Z.J. Wood, S.J. Gortler, and H. Hoppe. Multi-chart geometry images. In *Symposium on Geometry Processing Proceedings: Aachen, Germany*, 2003.

[39] P. Schröder, D. Zorin, T. DeRose, DR Forsey, L. Kobbelt, M. Lounsbery, and J. Peters. Subdivision for modeling and animation. *SIGGRAPH 2000 Course Notes*, 2000.

[40] T. Surazhsky, E. Magid, O. Soldea, G. Elber, and E. Rivlin. A comparison of gaussian and mean curvatures estimation methods on triangular meshes. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, pages 1021–1026, 2003.

[41] V. Surazhsky, P. Alliez, and C. Gotsman. Isotropic remeshing of surfaces: A local parameterization approach. In *In Proceedings of 12th International Meshing Roundtable*, 2003.

[42] V. Surazhsky and C. Gotsman. High quality compatible triangulations. *Engineering with Computers*, pages 147–156, 2004.

[43] Vitaly Surazhsky and Craig Gotsman. Explicit surface remeshing. In *Proc. Symp. on Geom. Processing (SGP)*, pages 20–30, 2003.

[44] Andrzej Szymczak, Jarek Rossignac, and Davis King. Piecewise regular meshes: construction and compression. *Graph. Models*, pages 183–198, 2002.

[45] C. Touma and C. Gotsman. Triangle mesh compression. *Proceedings of Graphics Interface*, 1998.

[46] Greg Turk. Re-tiling polygonal surfaces. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 55–64, 1992.

[47] Evan VanderZee, Anil Hirani, Damrong Guoy, and Edgar Ramos. Well-centered planar triangulation an iterative approach. In *Proceedings of the 16th International Meshing Roundtable*, pages 121–138. Springer Berlin Heidelberg, 2008.

[48] Luiz Velho. Using semi-regular 4-8 meshes for subdivision surfaces. *J. Graph. Tools*, pages 35–47, 2000.

[49] J. Vorsatz, Ch. Rössl, and H.-P. Seidel. Dynamic remeshing and applications. In *Proceedings of the 8th ACM symposium on Solid modeling and applications*, pages 167–175, 2003.

[50] J. Vorsatz, C. Rssl, L. P. Kobbelt, and H.-P. Seidel. Feature sensitive remeshing. *Computer Graphics Forum (Proc. EUROGRAPHICS)*, pages 393–401, 2001.

[51] Weining Yue, Qingwei Guo, Jie Zhang, and Guoping Wang. 3d triangular mesh optimization in geometry processing for cad. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 23–33, 2007.