Stony Brook University

Variational Delaunay Triangulation

A Thesis Presented

by

Phanindra Bhagavatula

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Computer Science

Stony Brook University

May 2012

Stony Brook University

The Graduate School

Phanindra Bhagavatula

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis.

Xianfeng David Gu - Thesis Advisor
Associate Professor, Department of Computer Science

Jie Gao - Chairman of Thesis Committee
Associate Professor, Department of Computer Science

Xiangmin Jiao
Associate Professor, Department of Applied Mathematics and Statistics

This thesis is accepted by the Graduate School.

Charles Taber
Interim Dean of the Graduate
School

Abstract of the Thesis

Variational Delaunay Triangulation

by

Phanindra Bhagavatula

Master of Science

in

Computer Science

Stony Brook University

2012

In this thesis I present an algorithm and its implementation for 2D and 3D simplicial mesh optimization. An energy function for each simplex of a mesh in $\mathbb{R}^n$ $2 \leq n \leq 3$ is defined as the volume of the hyperbolic simplex in $\mathbb{H}^{n+1}$ constructed from the said simplex. It has been proven otherwise and mentioned here as well that a regular simplex has maximum energy. Thus maximizing this energy by reshaping each individual simplex of the mesh will improve the overall quality of the mesh. The algorithm maximizes this energy to achieve an optimum mesh by displacing vertices and updating connectivity of the mesh conforming to the Delaunay property by following a gradient descent method. The details of the energy function, proof of correctness and implementation details are presented herewith.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A Mesh is a discretization of a space into simple geometric constructs to model a real world object and represent the same digitally. The study of meshes is a large subfield of computer graphics and geometric modeling. Different representations of such meshes are used for various applications and goals.

Mesh generation algorithms are essential tools in many areas. Typically, meshes are used in numerical solvers for solving partial differential equations (finite element methods) in computational science, geometric modeling, physics-based simulations, realistic simulation of deformable objects in computer graphics etc. Tetrahedral meshes are a popular choice for discretization of three dimensional domains and can be generated using *advancing front*, *Delaunay* or *octree* methods. In all applications, quality of a mesh plays a critical role and influences the accuracy and speed of numerical computations. Irregular meshes having slivers (skinny tetrahedra), can introduce numerical errors and increase time needed to find a solution. Isotropic meshing is desirable in common case where nearly-regular tetrahedra are preferred. Such a combinatorial optimization leads to good geometric properties.

Creating High quality meshes is a difficult task for various reasons. The size of a tetrahedral mesh is generally huge and handling the mesh requires robust data structures and algorithms. Degenerate tetrahedra are not uncommon even with evenly spaced vertices. Thus developing algorithms for 3D meshing and suitable error analysis for the same is very challenging.

In this thesis I present an implementation of a novel mesh quality optimization technique. The details of the implementation and an analysis of quality of the mesh generated are provided.

Chapter 2

Background

2.1 Delaunay Triangulation

The most popular triangular and tetrahedral meshing techniques are those utilizing Delaunay [6, 7] criterion. The Delaunay Property states that for a mesh in n dimensions the circumsphere of a nD Simplex should not contain any other vertex of the mesh. Figure 2.1 depicts the criterion for a two dimensional mesh. For every triangle (2D simplex) the circus circle of it doesn't contain any other vertices of the mesh. Following the same analogy, in a 3D mesh, every tetrahedron (3D simplex) has a circumsphere which is void of any other vertices in the mesh.

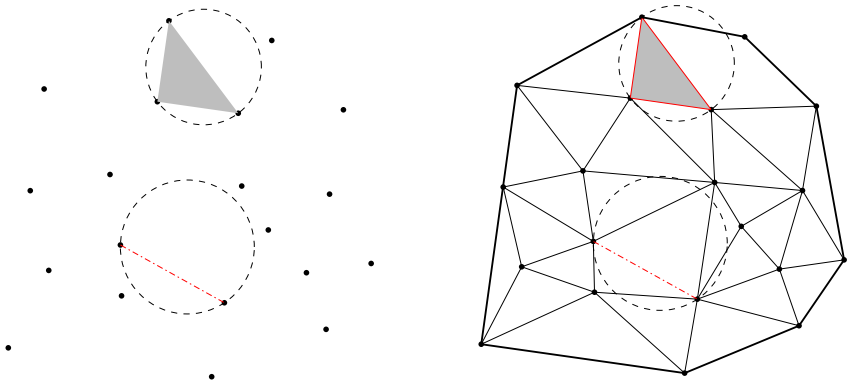


Figure 2.1: 2D-Delaunay Criterion[4]

Formally for a given set of points $P = \{p_1, p_2, \ldots, p_n\}$ in a d dimensional space $\mathbb{R}^d$, a k **Simplex** s $(0 \leq k \leq d)$ is defined as a k dimensional polytope which is the smallest convex set of its $k+1$ vertices where a convex set of a

set of points is a set of all possible convex combinations of the set of points. For example, a triangle is a 2-simplex, a tetrahedron is a 3-simplex. A line segment can be considered as a 1-simplex and a point as a 0-simplex.

A **Convex Hull** of a set of points $P = \{p_1, p_2, \ldots, p_n\}$ in $\mathbb{R}^d$ is the smallest convex set containing all the points. More precisely, it is the intersection of all possible convex sets containing the points.

A **Delaunay triangulation** of a set of points $P = \{p_1, p_2, \ldots, p_n\}$ in a d dimensional space $\mathbb{R}^d$ is defined as a subdivision of the convex hull of the set of points into simplices such that for any two simplices s_1 and s_2, the intersection $s_1 \cap s_2$ is ϕ or a k simplex where $0 \leq k < d$

Delaunay triangulation of a set of points in $\mathbb{R}^d$ is unique if the points are in general position i.e there exists no $k - flat$ containing $k + 2$ points or a $k - sphere$ containing $k + 3$ points for $1 \leq k \leq d - 1$. There are many algorithms to compute delaunay triangulation for a given set of points in $\mathbb{R}^d$. Below is a brief about each of the techniques.

2.1.1 Algorithms

Delaunay triangulation from Convex Hull

A Delaunay triangulation is closely related to convex hulls in that, a convex hull in $\mathbb{R}^d$ corresponds to a a projection of convex hull in $\mathbb{R}^d + 1$ onto $\mathbb{R}^d$. For a point $p_i = \langle x_1, x_2, ..., x_d \rangle \in \mathbb{R}^d$, we define a $liftpoint$ $p^+ = \langle x_1, x_2, ..., x_d, x_{d+1} \rangle \in \mathbb{R}^d + 1$, where $p_{d+1} = \sum_{i=1}^{d} {x_i}^2$. For a point set P in $\mathbb{R}^d$ we define a point set $P^+ = \{p_i^+ | p_i \in P\}$. P^+ is a set of points "lifted" from points in $\mathbb{R}^d$ to a paraboloid in $\mathbb{R}^d + 1$. Then the convex hull $conv(P^+)$ is a $(d + 1)$-dimensional polytope. The Delaunay triangulation of P can be produced by projecting $conv(P^+)$ into d-dimensions. Thus convex hull algorithms can be used to compute Delaunay triangulations. Figure 2.2 depicts the relationship between a delaunay triangulation of a set of points in $\mathbb{R}^2$ and the corresponding convex hull of the lifted points in $\mathbb{R}^3$.

Quick Hull Algorithm[8]

The Quick Hull algorithm computes convex hull of a given set of points in an arbitrary dimension (d). The current implementation assumes points to be present in general position (i.e., no set of $d + 1$ points defines a $(d - 1)$ flat) so that the convex hull is a simplicial complex. The convex hull is represented by its vertices and $(d - 1)$-dimensional faces called *facets*. Each facet consists of a set of vertices, a set of $d - 2$ simplices called *ridges* and a hyperplane equation. Each ridge is an intersection of the vertices of two neighboring facets.

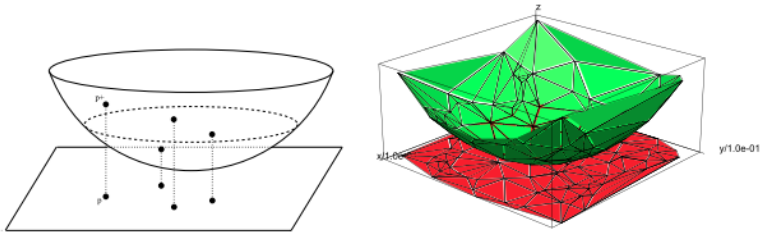


Figure 2.2: Delaunay Triangulation in $\mathbb{R}^2$ and Convex hull of *lifted* points in $\mathbb{R}^3$[4]

Quickhull uses two geometric operations, *oriented hyperplane through d points* and *signed distance to hyperplane*. A hyperplane is represented by its outward-pointing unit normal and its signed offset from the origin. The signed distance of a point to a hyperplane is the inner product of the point and the normal plus the offset. If distance of a point is positive, it is considered above the hyperplane.

For processing a point, quick hull uses a simplification of Grünbaum's Beneath-Beyond Theorem. [9]

Theorem. SIMPLIFIED BENEATH-BEYOND *Let H be a convex hull in R^d, and let p be a point in $R^d - H$. Then F is a facet of conv($p \cup H$) if and only if*

1. *F is a facet of H, and p is below F; or*
2. *F is not a facet of H, and its vertices are p and the vertices of a ridge of H with one incident facet below p and other incident facet above p.*

Quickhull algorithm has a simple way to determine visible facets. After the initialization, every unprocessed vertex p is assigned to a facet's outside set. By definition, the facet is visible to the vertex. Every time a new set of facets are created by the algorithm, it builds new outside set for the new facets using vertices from the previously visible facets. If a vertex is above multiple facets, it is assigned to the outside set of one of the facets. Additionally the farthest vertex among the outside set is maintained for each facet.

Algorithm 1 QuickHull Algorithm

```
procedure QuickHull(Set of points)
    Create a simplex of d+1 points
    for each Facets F
        for each unassigned vertex p
            if p is above F then assign p to F's outside bucket
            end if
        end for
    end for
    for each Facet F with a non empty outside bucket
        Select the furthest point p in F's outside bucket
        Initialize the Visible set V to F
        for each Unvisited neighbors N of facets in V do
            if p is above N then
                Add N to V
            end if
        end for
        The boundary of V is the set of Horizon ridges H
        for each Ridge R in H
            Create a new facet R and p
            link the new facet to its neighbors
        end for
        for each new facet F'
            for each Unassigned point q in outside bucket of a facet in V
                if q is above F' then
                    Assign q to outside bucket of F'
                end if
            end for
        end for
        Delete the facets in V
    end for
end procedure
```

Incremental Delaunay Construction

The incremental algorithm bases itself on the concept of *flips*. A *flip* is an operation to transform non-locally Delaunay simplices into a set of simplices which are locally delaunay.

Definition and Classification of flips. For a set P of $d + 2$ points in

$\mathbb{R}^d$, according to Lawson [10] there are exactly two ways to triangulate P. If the points are lifted to $\mathbb{R}^d+1$ as described above, the two ways of triangulation are the ones corresponding to the two sides of $d+1$ simplex in $\mathbb{R}^d+1$. There can be no other triangulation because a $d+1$ simplex exhausts all d simplices as its facets owing to the Radon's theorem [11]. A *flip* is the operation that replaces one triangulation with the other.

In $\mathbb{R}^2$, the two cases correspond to the fact that a tetrahedron of the lifted points in $\mathbb{R}^3$ corresponds to a triangle or a quadrilateral in $\mathbb{R}^2$. Figure 2.3 depicts the flips in $\mathbb{R}^2$.

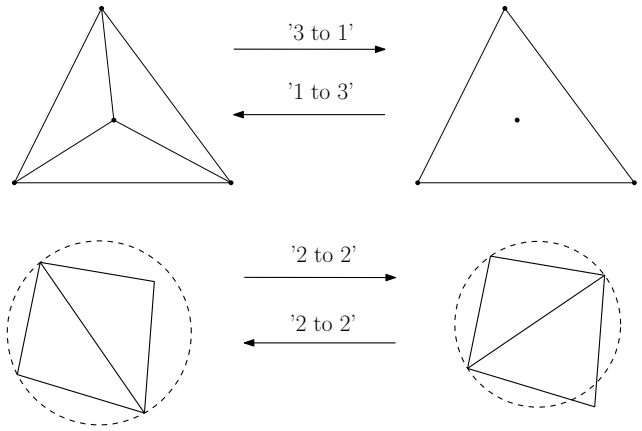


Figure 2.3: Lawson Edge Flips $\mathbb{R}^2$ [5]

In case of $\mathbb{R}^3$, the 4-simplex in $\mathbb{R}^4$ projects to a single or a double tetrahedron in $\mathbb{R}^3$. Flips in $\mathbb{R}^3$ are depicted in figure 2.4

An incremental algorithm for a set of points in $\mathbb{R}^d$ starts by constructing a huge d-simplex containing all the points. Incrementally, the points are inserted in random order. The Simplex in which the new point lies is determined and the simplex is split by adding new edges joining the new point to each vertex of the simplex ('1 to 3' flip in 2D and '1 to 4' flip in 3D). The each edge then is locally flipped if it is not locally delaunay. Eventually producing a delaunay triangulation of the given set of points.

2.2 Mesh Quality [1]

The utility of tetrahedral meshes increases many folds if the shapes of the tetrahedrons is so optimized that Finite Element Methods produce error free and fast results. The quality of a tetrahedral mesh denotes how suitable it is

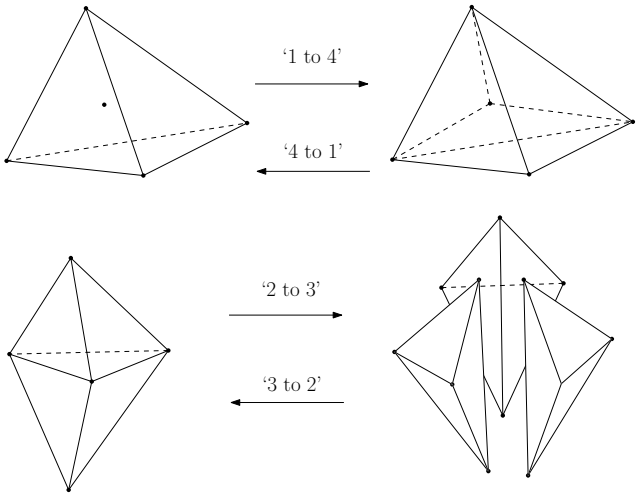


Figure 2.4: Lawson Edge Flips $\mathbb{R}^3$ [5]

for such numerically intensive operations. Many factors contribute to quality of a tetrahedron. Dihedral angles between pairs of faces of a tetrahedron are of particular importance. Figure 2.5 illustrates a dihedral angle between two triangular faces.

In finite element methods, large dihedral angles cause large interpola-

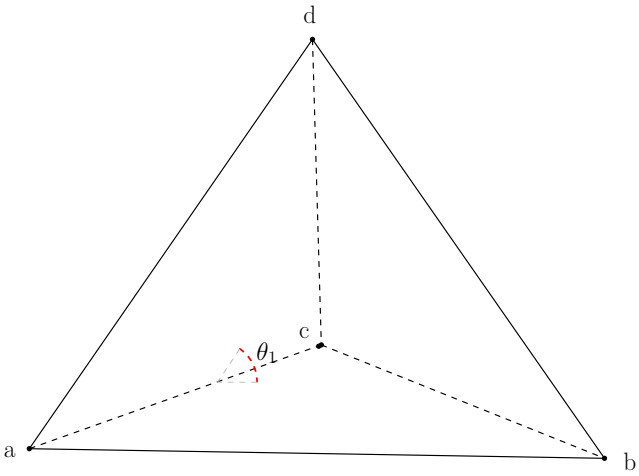


Figure 2.5: Dihedral Angle

tion errors, which hurt the accuracy of a simulation. Shewchuk's *What is a good Linear Finite Element* [12] explains in detail the costs of extreme dihedral angles in finite element meshes. Examples of some tetrahedra with extreme

dihedral angles are shown in figure 2.6.

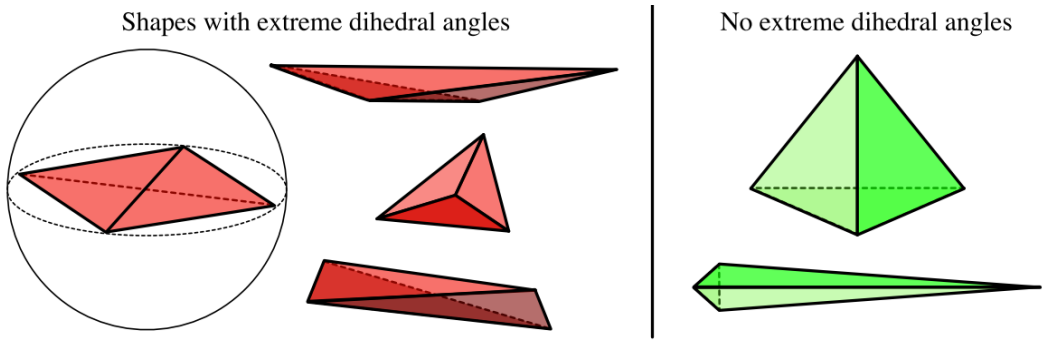


Figure 2.6: Left: Tetrahedra with extreme dihedral angles. Right: Tetrahedra with good dihedral angles [1]

2.2.1 Quality Measures [1]

Tetrahedral meshes are employed in numerous applications. Thus a single metric determining mesh quality can't be relevant to every application. Mesh improvement algorithms are still expected to generate meshes that are suitable for many different applications. Quality of a tetrahedron is thus determined by various user functions that the user could choose. Many quality measures are available as described in [12, 13]. Here I present a summary of these measures. The table 2.1 lists the formulae for each of the measures and its gradient. Figure 2.7 illustrates the quantities used to compute the measures in table 2.1

Minimum Sine Measure

The *minimum sine measure* of a tetrahedron is the minimum of the sines of its six dihedral angles. Since sines of 0° and 180° are both zero, this metric ensures bad quality measure for a tetrahedron for such angles. For a regular tetrahedron, where all dihedral angles are equal, the minimum sine measure has a maximum value for a tetrahedron which is $2\sqrt{2}/3$.

Volume-length measure

The *Volume-length measure* suggested by Parthasarathy, Graichen, and Hathaway [14] and denoted V/l^3_{rms} is the signed volume of a tetrahedron di-

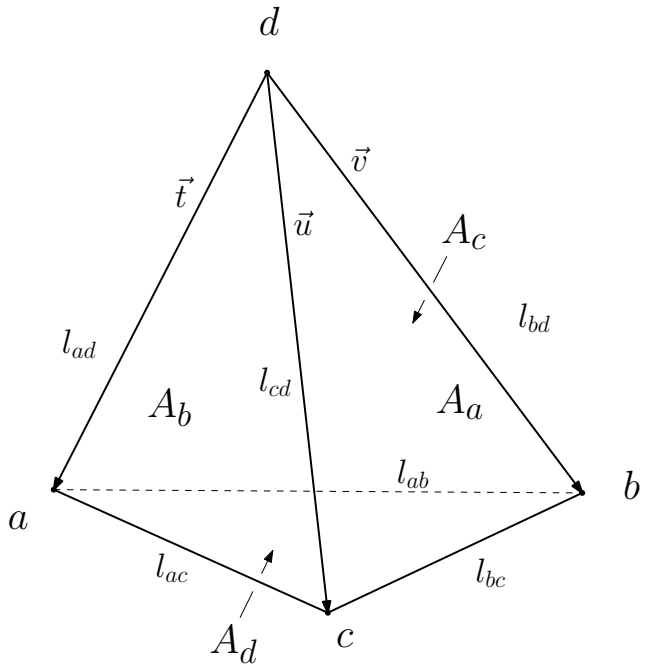


Figure 2.7: Tetrahedron labeling. l is edge length, A is face area and **t, u, v** are edge vectors

vided by the cube root of its root-mean-squared edge length. Intuitively this metric credits 'fat' tetrahedra. A tetrahedron has a maximum quality according to this metric when it is regular and the value is $\sqrt{2}/12$.

Radius Ratio(and its square root)

The *radius ratio* was suggested by Cavendish, Field, and Frey [15] is the ratio of in-radius (radius of inscribed sphere) to circus-radius (radius of circumscribed sphere). This metric too credits 'fat' tetrahedra. The regular tetrahedron has a maximum value of this ratio which is 1/3.

2.3 Variational Approaches to Meshing

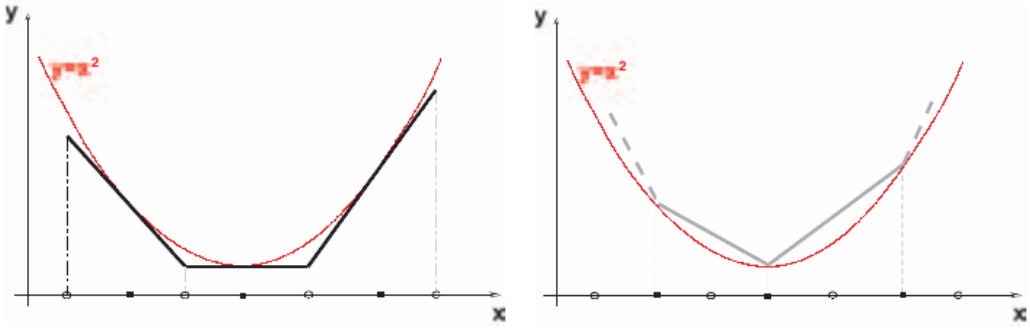
Mesh improvement techniques which employ minimization of energy defined on the mesh are categorized under variational approach to meshing. Triangular and tetrahedral meshes used in computer graphics, industrial modeling and structural simulations can be improved using this powerful and robust tool. Various methods have been discussed in [3, 16–18]. These methods

Table 2.1: Formulae for tetrahedron quality measures and their gradients. Quantities are labelled in figure 2.7 and formulae for their computation are in table 2.2. Numbers used as vertex subscripts 1, 2, 3, 4 correspond to a, b, c, d respectively[1]

Measure	Formula
Minimum Sine	$\frac{3V}{2} \min_{1 \le k < l \le 4} \frac{l_{kl}}{A_k A_l}$
Volume-Length	$\frac{6}{\sqrt{2}} \frac{V}{l_{rms}^3}$
Square Root of Radius Ration	$6\sqrt{3} \frac{V}{\sqrt{Z(A_a+A_b+A_c+A_d)}}$

Table 2.2: Formulae for quantities with respect to d needed to compute quality measures.

$\boldsymbol{f(d)}$	**Formula**	$\boldsymbol{f(d)}$	**Formula**
l_{ad}	$\lvert\boldsymbol{t}\rvert$	l_{rms}	$\sqrt{\frac{1}{6}(\sum_{\{i,j \in \{a,b,c,d\} \mid i \neq j\}} l_{ij}^2)}$
l_{bd}	$\lvert\boldsymbol{u}\rvert$	A_a	$\frac{\lvert\boldsymbol{u} \times \boldsymbol{v}\rvert}{2}$
l_{cd}	$\lvert\boldsymbol{v}\rvert$	A_b	$\frac{\lvert\boldsymbol{v} \times \boldsymbol{t}\rvert}{2}$
l_{ab}	$\lvert a - b\rvert$	A_c	$\frac{\lvert\boldsymbol{t} \times \boldsymbol{u}\rvert}{2}$
l_{bc}	$\lvert b - c\rvert$	A_d	$\frac{\lvert\boldsymbol{u}-\boldsymbol{v}\rvert \times \lvert\boldsymbol{t}-\boldsymbol{v}\rvert}{2}$
l_{ac}	$\lvert c - a\rvert$	V	$\frac{det[\boldsymbol{tuv}]}{6}$
		Z	$\lvert\lvert\boldsymbol{t}\rvert^2 \boldsymbol{u} \times \boldsymbol{v} + \lvert\boldsymbol{u}\rvert^2 \boldsymbol{v} \times \boldsymbol{t} + \lvert\boldsymbol{v}\rvert^2 \boldsymbol{t} \times \boldsymbol{u} + \rvert$



(a) Piecewise Linear Approximation(CVT) (b) Optimal Delaunay Triangulations(ODT)

Figure 2.8: [3]

define (often highly) non-convex energies that they minimize through vertex displacement or connectivity changes in the mesh. Below is a short summary of a few methods.

2.3.1 Centroidal Voronoi Tessellation [2]

The CVT tries to minimize a quadratic energy defined on a mesh. It generates a mesh which is dual to optimal Voronoi diagrams. The quadratic energy is defined by

$$E_{cvt} = \sum_{i=1..N} \int_{V_i} \|\boldsymbol{x} - \boldsymbol{x_i}\|^2 dx$$

where the $\boldsymbol{x_i}$ are vertex positions and $\boldsymbol{V_i}$ a local cell associated with each $\boldsymbol{x_i}$. The union of these cells forms a partition of the domain Ω. The algorithm computes the Voronoi diagram for a given set of vertices restricted to the domain Ω since it is energetically optimal position for the current vertex position. In the second phase the partition is held fixed and the vertex position of $\boldsymbol{x_i}$ is optimized. Each of these steps decreases the same energy. The analysis of E_{cvt} [3] the minimization of energy corresponds to the minimization of the volume between a paraboloid $f(x) = \|x\|^2$ and an underlaid circumscribed piece wise linear approximate f_{PWL}^{dual} which is formed by planar patches tangent to the paraboloid (Figure 2.8a depicts this).

$$E_{CVT} = \|f - f_{PWL}^{dual}\|_{\mathcal{L}^1}$$

Tests have shown [3] that using CVT gives rise to numerous degenerate sliver tets. It has been attributed to the fact that E_{CVT} tends to optimize the compactness of the Voronoi cells but not Delaunay triangulation. Thus

presence of a sliver is not penalized by this energy function.

2.3.2 Optimal Delaunay Triangulations

An energy optimization algorithm was proposed by Chen [19] which works on the input mesh itself instead of its dual. The energy function used is

$$E_{ODT} = \|f - f_{PWL}^{primal}\|\mathcal{L}^1$$

$$E_{ODT} = \frac{1}{n+1} \sum_{i=1..N} \int_{\Omega_i} \|\boldsymbol{x} - \boldsymbol{x_i}\|^2 dx$$

which is the volume between the paraboloid and an *overlaid, circumscribing piecewise linear approximate* $f_{PWL}primal$ formed by a linear interpolation of points on the paraboloid (Figure 2.8b). For every vertex position $\boldsymbol{x_i}$, the energy is calculated over the *1-ring* region Ω_i also called the star of vertex $\boldsymbol{x_i}$ which is the set of simplices incident at the vertex $\boldsymbol{x_i}$. This method doesn't formally guarantee an optimum mesh. The smoothing technique presented updates the mesh connectivity through local edge flips only. This method doesn't carry over to 3D since there is no theorem to prove that an arbitrary mesh is only finite number of flips away from the optimal connectivity.

2.3.3 Variational Tetrahedral Meshing [3]

Variational Tetrahedral Meshing by Alliez minimizes the E_{ODT} by a minimization procedure by modifying both vertex positions and connectivity of the mesh. Connectivity Optimization is easily achieved by following the Delaunay property. For a vertex $\boldsymbol{x_i}$, E_{ODT} is minimized by the Delaunay connectivity. For a given mesh $\mathcal{M}$ with vertices $\boldsymbol{x_i}$, the E_{ODT} can be written as:

$$E_{ODT} = \frac{1}{4} \sum_i \boldsymbol{x_i} |\Omega_i| - \int_{\mathcal{M}} \boldsymbol{x}^2 d\boldsymbol{x}$$

where $|\Omega_i|$ is the measure of the 1-ring neighborhood of vertex $\boldsymbol{x_i}$. [3] states that the last term is constant given a fixed boundary $\partial\mathcal{M}$. A simple derivation of the quadratic energy in $\boldsymbol{x_i}$ leads to the following optimal position $\boldsymbol{x_i^*}$

$$\boldsymbol{x_i^*} = -\frac{1}{2|\Omega_i|} \sum_{T_j \in \Omega_i} \left(\nabla_{\boldsymbol{x_i}} |T_j| \left[\sum_{x_k \in T_j, x_k \neq x_i} \|\boldsymbol{x_k}\|^2 \right] \right)$$

The term $\nabla_{x_i}|T_j|$ is the gradient of the volume of the tet T_j with respect to $\boldsymbol{x_i}$. Replacing function $f(\boldsymbol{x}) = \|\boldsymbol{x}\|^2$ by the translated function $f(\boldsymbol{x}) = \|\boldsymbol{x} - \boldsymbol{x_i}\|^2$, we get the vertex position as

$$\boldsymbol{x_i^*} = \boldsymbol{x_i} - \frac{1}{2|\Omega_i|} \sum_{T_j \in \Omega_i} \left(\nabla_{\boldsymbol{x_i}} |T_j| \left[\sum_{x_k \in T_j, x_k \neq x_i} \|\boldsymbol{x_i} - \boldsymbol{x_k}\|^2 \right] \right)$$

Chapter 3

System Overview

We propose a variational mesh optimization algorithm to improve mesh quality. In this chapter I define the energy function used by our algorithm and the motivation behind the same. For a given mesh in $\mathbb{R}^n$ ($n \leq 3$) we define an energy function $E_{mesh} = \sum_m E^i_{simplex}$ where the mesh is a union of m simplices and each simplex i has energy $E^i_{simplex}$ defined as the volume of the ideal hyperbolic simplex in the hyperbolic space $\mathbb{H}^{n+1}$ constructed from the euclidean simplex in $\mathbb{R}^n$. Further sections state some identities associated to a hyperbolic simplex in $\mathbb{H}^{n+1}$ and the construction of the same from a euclidean simplex in $\mathbb{R}^n$.

3.1 Triangular Mesh Optimization

3.1.1 Volume of an ideal hyperbolic 3-simplex

Theorem. *Consider an ideal Hyperbolic 3-simplex, that is a simplex Δ with all four vertices on the sphere of points at infinity. If α, β, γ are the dihedral angles along three edges meeting at a common vertex, then $\alpha + \beta + \gamma = \pi$, and*

$$volume(\Delta) = \Lambda(\alpha) + \Lambda(\beta) + \Lambda(\gamma)$$

It does not matter which particular vertex is chosen since the opposite dihedral angles in an ideal hyperbolic tetrahedron are equal.

Proof. Milnor [20] proved this theorem as follows. We consider a Beltrami upper half-space model of the hyperbolic space with the metric $ds^2 = (dx^2 + dy^2 + dz^2)/z^2$ and the associated volume element $dxdydz/z^3$ where $z > 0$. Let us position the ideal simplex Δ such that one of its faces lies on the hemisphere $z = (1 - x^2 - y^2)^{1/2}$ and its opposite vertex lies at infinity. Projecting Δ

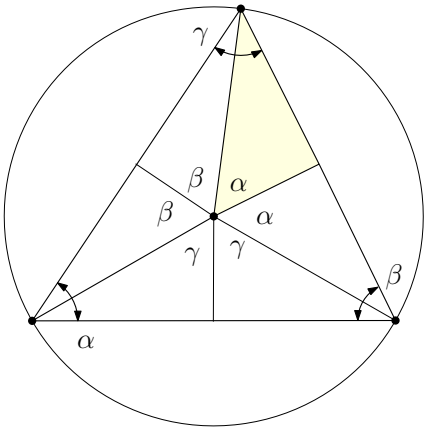


Figure 3.1: Triangle inscribed in a unit circle

orthogonally to the unit disc in the x,y-plane, we obtain a triangle inscribed in the disc with angles α, β, γ. These are angles of euclidean triangle, so $\alpha+\beta+\gamma=\pi$. It follows from figure 3.1 that these angles determine Δ up to congruence, and are subject to no other restriction. Barycentrically dividing the triangle from the origin of the circumcircle, we obtain six right triangles as illustrated in 3.1. Considering the region in the upper half space lying over one of the triangles like the one shaded in the figure, we must integrate the Beltrami volume element $dxdydz/z^3$ over the region defined by inequalities

$$z \geq \sqrt{1-x^2-y^2}, \qquad 0 \leq y \leq x\tan\alpha, \qquad 0 \leq x \leq \cos\alpha$$

Integrating with respect to z, we obtain $\frac{1}{2}dxdy/(1-x^2-y^2)$. Integrating with respect to y, we obtain

$$\frac{dx}{4A}[log(A+y)-log(A-y)]_o^{x\tan x} = \frac{dx}{4A}log\frac{A\cos\alpha+x\sin\alpha}{A\cos\alpha-x\sin\alpha}$$

where $A=(1-x^2)^{1/2}$. Substituting $A=\sin\theta$, $x=\cos\theta$, $dx=-Ad\theta$, and integrating we get

$$\begin{aligned} volume &= \frac{1}{4}\int_\alpha^{\pi/2} log\frac{2\sin(\theta+\alpha)}{2\sin(\theta-\alpha)}d\theta \\ &= \Big(-\Lambda\Big(\frac{\pi}{2}+\alpha\Big)+\Lambda(2\alpha)+\Lambda\Big(\frac{\pi}{2}-\alpha\Big)-\Lambda(0)\Big)/4 \end{aligned}$$

Now if we substitute the identity

$$\Lambda(2\alpha) = 2\Lambda(\alpha)+2\Lambda(\alpha+\pi/2),$$

this reduces easily to $volume = \Lambda(\alpha)/2$. Adding the volumes of the other five regions we obtain $volume = \Lambda(\alpha) + \Lambda(\beta) + \Lambda(\gamma)$ □

Corollary. *The maximum possible volume of a hyperbolic 3-simplex is* $3\Lambda(\pi/3)$.

Proof. To find the max volume the continuous function $\Lambda(\alpha) + \Lambda(\beta) + \Lambda(\gamma)$ has to be maximized subject to the constraints $\alpha, \beta, \gamma \geq 0$ and $\alpha + \beta + \gamma = \pi$. Since the degenerate cases, where one dihedral angle is zero, all have zero volume, the maximum must occur at an interior point where the derivative $\Lambda'(\theta) = -\log(2\sin\theta)$ is defined and satisfies

$$\Lambda'(\alpha) = \Lambda'(\beta) = \Lambda'(\gamma)$$

hence $\sin\alpha = \sin\beta = \sin\gamma$. Thus $\alpha = \beta = \gamma = \pi/3$ is the only interior solution. □

3.1.2 Construction of Hyperbolic simplex in $\mathbb{H}^3$ for a given simplex in $\mathbb{R}^2$

The above corollary can be used to optimize triangular meshes. Given an arbitrary triangle, we can construct a hyperbolic tetrahedron in Beltrami upper half plane model in $\mathbb{H}^3$ such that the triangle is a projection of a face on the hemisphere centered at the circumcenter of the triangle and the opposite vertex is at infinity as depicted in figure 3.2. Changing the vertex positions of the triangle such that the volume of the hyperbolic tetrahedron thus constructed is maximum leads to transformation of an irregular triangle to an equilateral triangle. This volume is defined as the energy of a triangle in $\mathbb{R}^2$. The energy of a triangular mesh is the summation of volumes of the hyperbolic tetrahedra corresponding to each triangle in the mesh.
Since the energy is a function of the angles of the tetrahedron and thus a function of the vertex positions, we define a gradient of energy of a triangle w.r.t each vertex. The new position of a vertex is determined by gradient descent method. In case of a mesh, the gradient of energy at a vertex is a summation of gradient of each incident triangle w.r.t that vertex. We reshape the triangular mesh such that the energy of the whole mesh is maximized.

Let energy of a triangle E_{tri} be defined as

$$E_{tri} = \Lambda(\alpha) + \Lambda(\beta) + \Lambda(\gamma) \qquad \alpha, \beta, \gamma \text{ are the interior angles of the triangle}$$

$$E_{mesh} = \sum_i E^i_{tri}$$

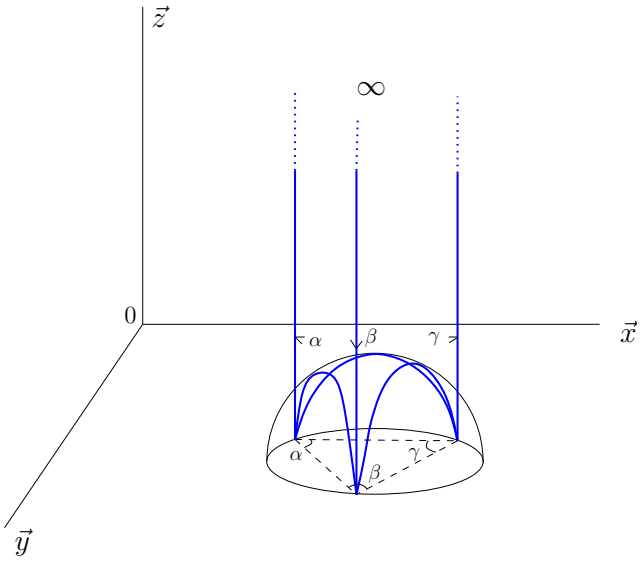


Figure 3.2: 3D Hyperbolic Simplex constructed from a triangle on 2D plane

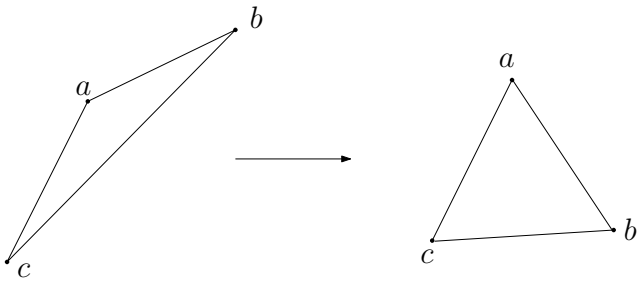


Figure 3.3: Transformation of a triangle from irregular to regular

The gradient of the energy with respect to a vertex v_i of the triangle is

$$\begin{aligned} E'_{tri} &= \frac{d}{dv_i}(\Lambda(\alpha) + \Lambda(\beta) + \Lambda(\gamma)) \\ &= -\log(2\sin\alpha)\frac{d\alpha}{dv_i} + -\log(2\sin\beta)\frac{d\beta}{dv_i} + -\log(2\sin\gamma)\frac{d\gamma}{dv_i} \end{aligned}$$

The algorithm stated below summarizes the procedure.

Algorithm 2 Variational 2D Delaunay Triangulation

```
procedure VARIATIONAL DELAUNAY(Set of points)
    Generate a Delaunay triangulation for the set of points.
    for each Interior Vertex v
        GradientVector G_vector ← 0
        for each 2D Simplex incident on v
            Calculate the Energy Gradient Vector for the 2D Simplex
            G_vector ← G_vector + Energy Gradient Vector
        end for
        Normalize(G_vector)
        pos(v) ← pos(v) + ϵ × G_vector
        Check Delaunay Legality of Each edge incident at v
        for each Illegal Edge
            Flip Edges to legalize the Mesh
        end for
    end for
end procedure
```

3.1.3 Proof Of Uniqueness

The energy function defined guarantees a unique and optimum mesh. We first state the following lemma based on which the uniqueness theorem is proven.

Definition. *For an Internal edge of a mesh as shown in figure 3.4 the* ***Edge cross ratio*** *is defined as*

$$\rho(e) = \frac{a * b'}{a' * b}$$

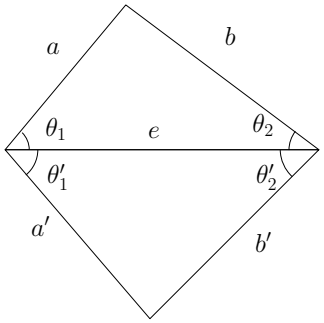


Figure 3.4: An interior edge e and the edges of incident triangles

Lemma 1. *When Hyperbolic tetrahedron volume for neighboring triangles is optimized, all the edge cross ratios equal to one.*

$$\rho(e) = \frac{a * d}{b * c} = 1$$

Proof. Consider the following deformations

$$\theta_1 = \theta_1 + \epsilon\theta_2 = \theta_2 - \epsilon\theta_1' = \theta_1' - \epsilon\theta_2' = \theta_2' + \epsilon$$

Then at optimal point the following holds true

$$\frac{d[\Lambda(\theta_1 + \epsilon) + \Lambda(theta_2 - \epsilon) + \Lambda(theta_1' - \epsilon) + \Lambda(theta_2' + \epsilon)]}{d\epsilon} = 0$$

Lobachevsky function

$$\Lambda(\theta) = -\int^{\theta} \log|2\sin t|dt$$

$$\therefore \frac{d\Lambda(\theta + \epsilon)}{d\epsilon} = -\log|2\sin(\theta + \epsilon)|,$$
$$\frac{d\Lambda(\theta - \epsilon)}{d\epsilon} = \log|2\sin(\theta - \epsilon)|$$

Therefore we get

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{\sin\theta_1'}{\sin\theta_2'}$$

$$\therefore \text{By law of sines } a'b = b'a$$

□

Theorem 1. *If a triangulation is given and its boundary curvature is fixed, then if optimal solution exists, it must be unique*

Proof. If the triangulation is fixed, the edge cross ratio is fixed, then only the edge lengths are deformed. This deformation is equivalent to the discrete Yamabe flow. Let $u : V \to \mathbb{R}$ is a function for edge $[v_i, v_j]$, the length of which is given b

$$l_{ij} = e^{\frac{u_i+u_j}{2}}$$

The planar triangulation with given boundary curvature is equivalent to set all the target curvatures. For interior vertices the target curvatures are 0.

$$\overline{K} = 0, \forall v_i \notin \partial M$$

We define a 1-form

$$\omega := \sum_{i=1}^{n} (\overline{K}_i - K_i) du_i$$

The 1-form energy is closed. The energy

$$E(u_1, u_2, \ldots, u_n) = \int_{(0,0,\ldots,0)} (u_1, u_2, \ldots, u_n)\omega$$

is well defined. Furthermore, the energy is convex. So the optimal point is unique. The desired configuration is the unique optimal point. □

Theorem 2. *Delaunay triangulation optimizes triangle energy. Thus the optimum Triangulation is Delaunay.*

Proof. Delaunay Triangulation maximizes the minimal angle. Thus given an edge, the total energy of the incident triangles, is optimized if the edge is flipped to restore Delaunayhood. Thus the optimum triangulation will always be Delaunay. □

3.2 Volume of an ideal hyperbolic 4-simplex

A hyperbolic 4-simplex corresponding to a tetrahedron in $\mathbb{R}^3$ can be constructed in a similar fashion as a hyperbolic tetrahedron from a euclidean triangle. The volume of the hyperbolic 4-simplex Δ is given by the following formula as mentioned in section 11.3 of [21]

$$Volume(\Delta) = \frac{4}{3}\pi^2 - \frac{\pi}{3}\sum_{i<j} \theta_{ij}$$

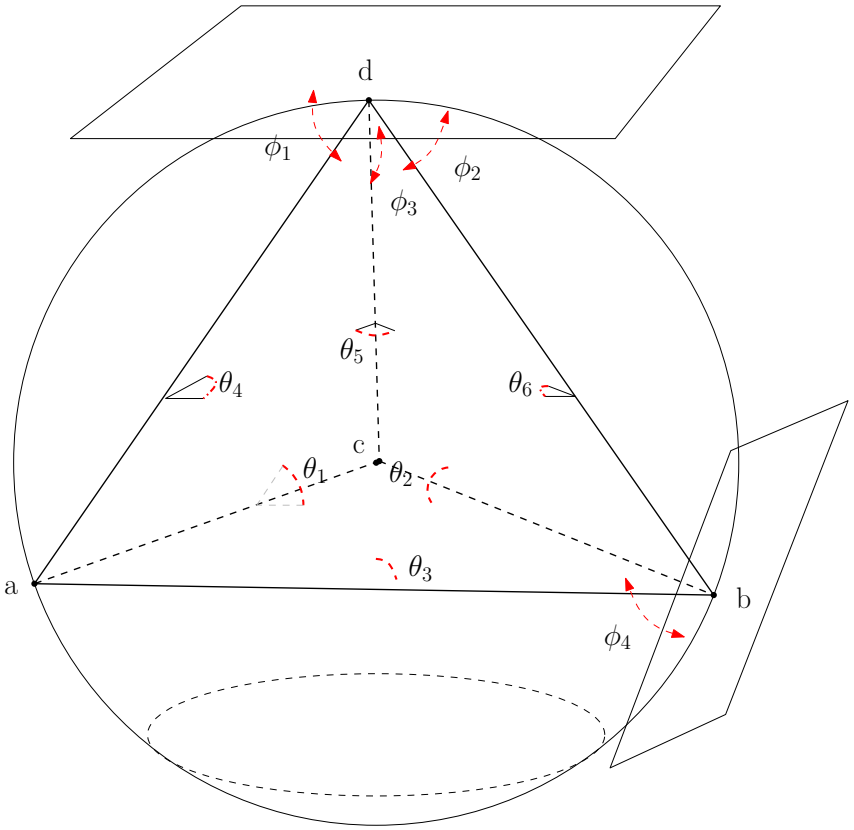


Figure 3.5: Angles of Tetrahedron Corresponding to Dihedral angles of Hyperbolic 4-simplex

where θ_{ij} is the dihedral angle between the faces of the hyperbolic 4-simplex. As seen earlier in case of a triangle, the dihedral angles of hyperbolic 4-simplex constructed from a euclidean tetrahedron is the angles depicted in the figure 3.5

3.3 Implementation Details for Triangular Mesh Optimization

Triangular meshes are represented using a variety of data structures. One of the most efficient ones is the *Half Edge Data Structure*. The half edge data structure represents each triangle as an interconnection of its elements. We define class for each element viz. the **Face** class, the **edge** class, the **vertex** class and the **halfEdge** class. The halfEdge class gives orientation of the winding for vertices of each face. It helps in determining the face normals

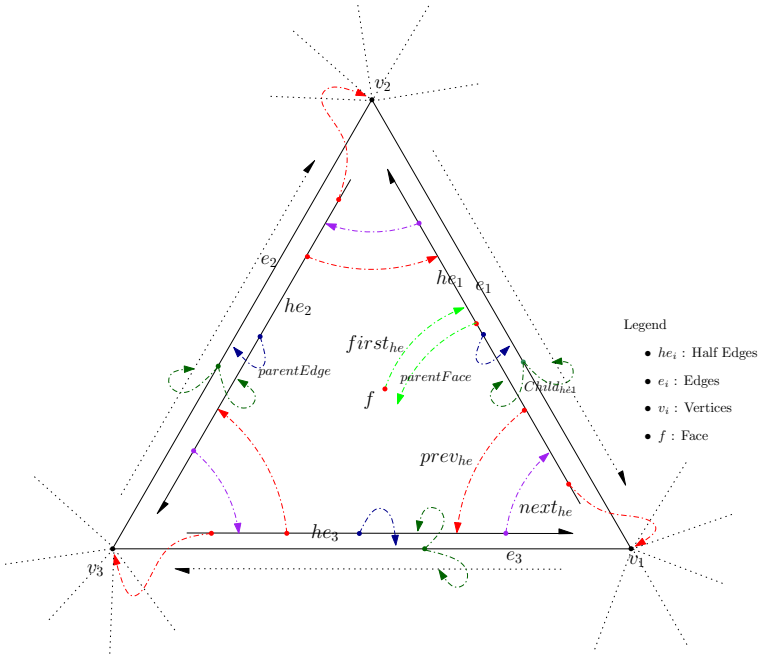


Figure 3.6: Half Edge Data Structure

without much orientation checking calculations. The interconnection between each of these data structures is depicted in the figure 3.6. The implementation generates a set of random points in a circle and triangulates the vertices using Delaunay triangulation algorithm. For the case mesh in $\mathbb{R}^2$ we use a simple incremental Delaunay triangulation algorithm. The algorithm starts with an initial large triangle and each point is inserted and 2D-*flips*[5] as described earlier are used to ensure Delaunay property is maintained as each point is inserted.

3.4 Implementation Details for Tetrahedral Mesh Optimization

The code to optimize a 3D mesh is presented. We use a standard representation of the mesh using C++ Classes emulating interconnection of entities of a mesh.

3.4.1 Data Structures

The triangulation is represented by a set of classes for each element of the mesh. We use an aggregation style of design. The complete object is represented by the **Solid** class. Each tetrahedron is represented by the **Cell** class, each face by the **Face** class and each vertex by the **Vertex** class. The solid is an aggregation of each of the smaller elements (simplices of lower dimensions). Thus we maintain a list of all cells, faces and vertices in the solid class. Each cell, face or vertex has a unique id which helps identify the desired object from the list. We use a map data structure in the Solid class to maintain the list of each entity. Each of the simplices is an aggregation of simplices of the lower dimensions. For example, a cell is formed by a set of vertices and faces, where each of the faces is a combination of vertices which form the face. We maintain a list of the lower dimension simplices which a given simplex is made up of. For robustness of the application and for simplicity, we only maintain the list of id's representing the desired simplex.

The implementation optimizes a mesh generated from a number of random points in a unit cube. The code has two main parts. The Delaunay Triangulator and the Optimizer.

3.4.2 Delaunay Triangulator

This part of the code takes as input a set of random points and generates a Delaunay tetrahedralization of the those in 3D. The code implements the QuickHull algorithm explained earlier to generate a convex hull of the projected points in 4D. The quick hull requires representation of simplices in 4d and hence a different connectivity relationship between simplices of each dimension. We thus use a different set of classes derived from a set of Base classes representing each simplex. The Delaunay Triangulator gives as output the 4D mesh which is a convex hull. This Delaunay triangulation in 3D can be obtained by projecting this 4D convex hull into 3D. The optimizer code does that once the mesh is returned by the quickHull algorithm.

3.4.3 Optimizer

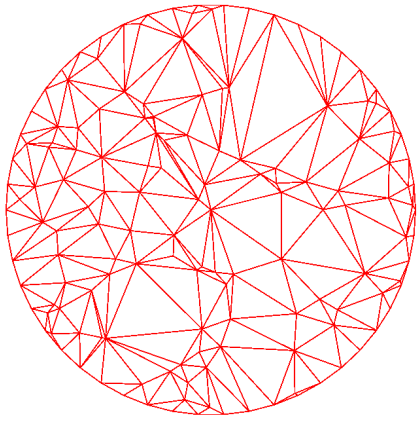
The optimizer code does the major processing. This code initially generates a set of random points and calls for generation of 3D Delaunay tetrahedralization. Once the mesh representing 4D convex hull is obtained, the same is projected into 3D and the Delaunay mesh for the set of points is obtained. The optimizer while creating the mesh, also identifies the *faces* and *vertices*

which belong to the convex hull of the mesh in 3D. This information is used by the variational algorithm to ensure the boundary vertices are not moved. The optimizer then parses through each tetrahedron and calculates the energy of the tetrahedron and the energy gradient with respect to each vertex of the tetrahedron. This information is maintained as a map in the **Cell** class. The energy is calculated using the formula derived in the previous section. For simplicity of calculation, the energy gradient is calculated using the numerical derivative formulae. Once this data has been obtained, the code now determines the new position for each internal vertex such that the energy of the complete mesh is reduced. The connectivity of each vertex needs to be checked to ensure the new position of the vertex doesn't destroy the Delaunay property of the mesh. Thus we check each face of the tetrahedra incident on the said vertex for Delaunay legality and employ a series of flips as described in [5] to restore delaunayhood. This series of flips is not guaranteed to stop or produce a Delaunay triangulation. But we employ this method for overall speed of the algorithm. The algorithm continues to run until the energy of the mesh is not constant at which point, every vertex is at its optimum position.

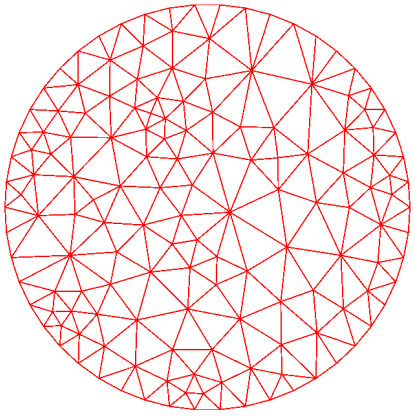
Chapter 4

Results

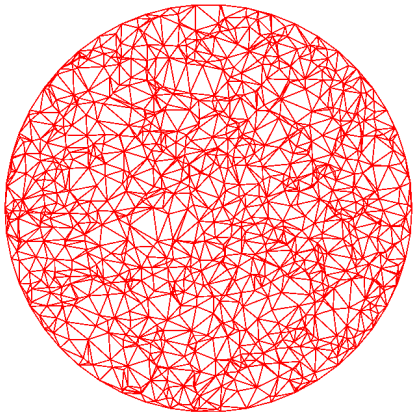
Figure 4.1 shows Mesh Optimization results for a 2D mesh in $\mathbb{R}^2$ for 10, and 100 vertices A histogram of the interior angles of triangles of the mesh with 1000 vertices is shown in figure 4.2. The optimization reduces the obtuse angles, increases all acute angles and most of the triangles are equilateral. Figure 4.3 shows the Energy E_{tri} calculated using the formula described above after each iteration of optimization. The energy of the mesh increases to the optimum after which it is almost constant within a small factor. Figure 4.4 shows the Cross Ratio of edges corresponding to triangles incident on each edge as explained earlier in the previous chapter. The histogram clearly shows how the proposed algorithm makes the mesh optimum and changes the edge lengths such that all edge ratios are in the range of 0.9 to 1.1. Figure **??** shows a histogram of the dihedral angles of tetrahedrons in the 3D mesh before and after optimization. The algorithm improves the mesh quality slightly but not much since there were not many iterations that could be run due to slow nature of the algorithm.



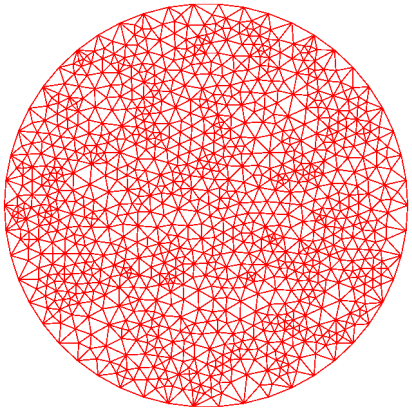
(a) Delaunay triangulation of 100 vertices



(b) Variational Delaunay Triangulation of 100 vertices



(c) Delaunay triangulation of 1000 vertices



(d) Variational Delaunay Triangulation of 1000 vertices

Figure 4.1: Variational Delaunay triangulation

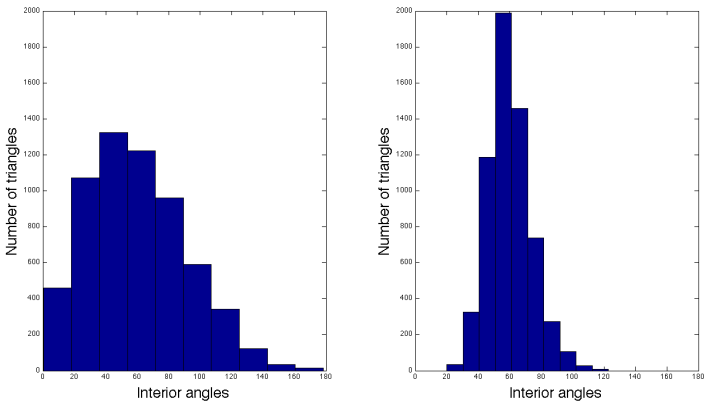


Figure 4.2: Histogram Of Interior angles of 2D Mesh before and after optimization

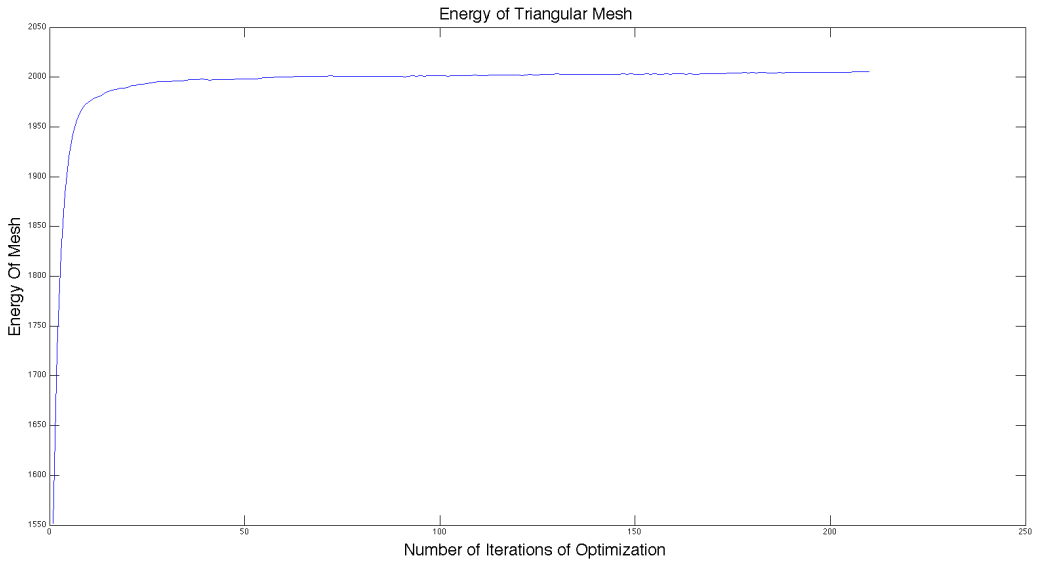


Figure 4.3: Energy E_{tri} of mesh

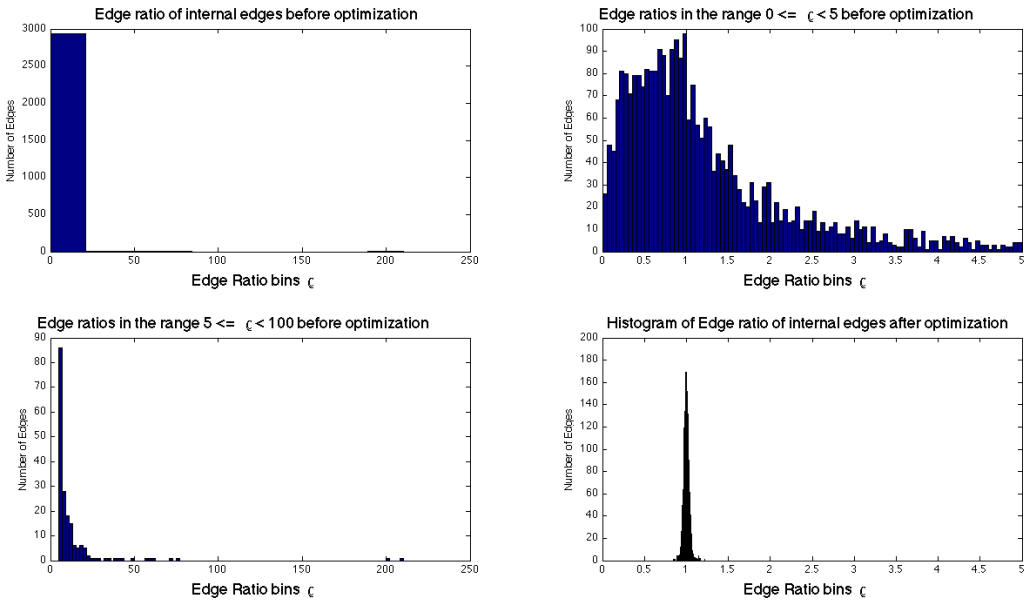


Figure 4.4: Edge Cross Ratio Histogram

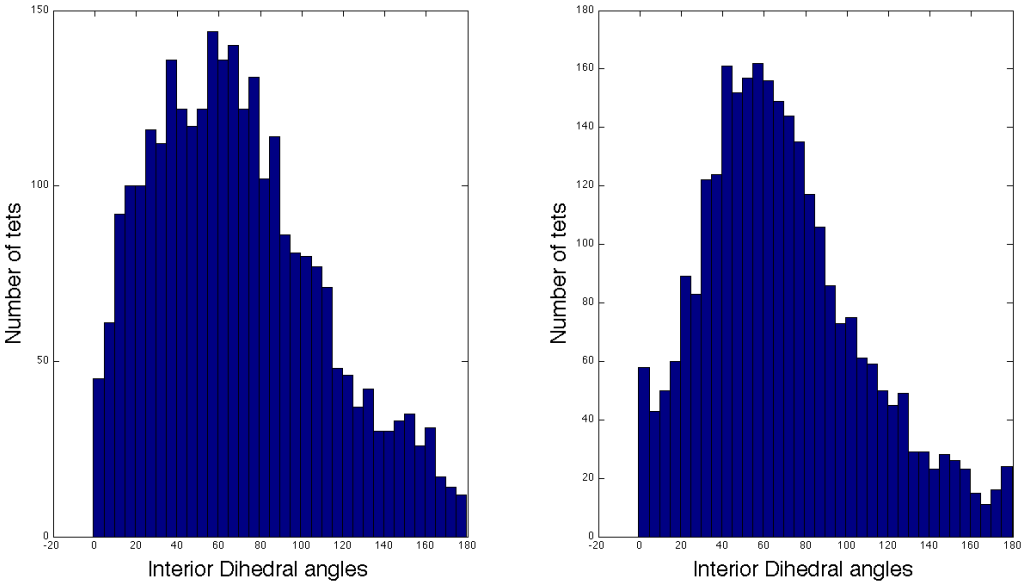


Figure 4.5: Dihedral Angles of Tetrahedrons of 3D mesh before and after optimization

Chapter 5

Conclusion and Future work

This thesis presents an algorithm which can be a practical method to produce high quality meshes. The Energy function described earlier is a good measure of quality of a mesh and thus an appropriate parameter to improve mesh quality. There are many improvements that the current algorithm and the implementation requires.

Speed

The major shortcoming of the current software is speed of execution. Since the algorithm is an iterative algorithm where each internal vertex is iterated over to determine the energy gradient for the incident cells w.r.t the vertex and determine the new position of the vertex, the complexity of the algorithm is very high. The complexity further increases due to the fact that every movement of a vertex requires restoration of the Delaunay property of the mesh. The restoration of Delaunay hood can be sped up by using an efficient local Delaunay fixing methods. The software currently uses Lawson's flipping method [22] which doesn't guarantee a solution in 3D for some special cases. Star Splaying method by Shewchuk [23] is a fast local delaunay fixing method which can be used to speed up the algorithm.

Termination of Optimization

There is no good way to determine convergence in variational algorithms. The current method relies on optimum position of vertices. Determination of optimum position depends solely on the mesh density and the least count of vertex displacement. A higher least count of vertex displacement could cause wobbling of a vertex in a small region due to change in direction of gradient

at each new position. If the least count is too small, the convergence would take a lot of time to be achieved.

Extension to practical use

The algorithm can be extended in future to practical applications where remeshing objects pose constrained boundaries which need special treatment to maintain the delaunay property of the mesh. Constrained Delaunay algorithms can be used in conjunction with the optimization algorithm to achieve higher quality meshes.

Parallelization

The algorithm has high scope of parallelization. Determination of new vertex position can be parallelized by a GPU implementation. It would improve the speed tremendously thus achieving optimum meshes quickly.

Bibliography

[1] Bryan Klingner and Jonathan Shewchuk. Tetrahedral mesh improvement. (March), 2008.

[2] Qiang Du and Desheng Wang. Tetrahedral mesh generation and optimization based on centroidal voronoi tessellations. *International Journal for Numerical Methods in Engineering*, 56(9):1355–1373, 2003.

[3] Pierre Alliez, David Cohen-Steiner, Mariette Yvinec, and Mathieu Desbrun. Variational tetrahedral meshing. *ACM Transactions on Graphics*, 24(3):617, 2005.

[4] Hang Si. A quality tetrahedral mesh generator and. *Control*, 9:62, 2006.

[5] H Edelsbrunner and N R Shah. Incremental topological flipping works for regular triangulations. *Algorithmica*, 15(3):223–241, 1996.

[6] S J Owen. *A Survey of Unstructured Mesh Generation Technology*, volume 3, pages 239–267. Citeseer, 1998.

[7] B Delaunay. Sur la sphere vide. *Izv Akad Nauk SSSR Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(7):793–800, 1934.

[8] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[9] B. Grünbaum. Measure of symmetry for convex sets. *Proceedings of the 7th Symposium in Pure Mathematics of the American Mathematical Society, Symposium on Convexity*, 1961.

[10] Charles L Lawson. Properties of n-dimensional triangulations. *Computer Aided Geometric Design*, 3(4):231–246, 1986.

[11] J Radon. Mengen konvexer körper , die einen gemeinsamen punkt enthalten. *Mathematische Annalen*, 83:113 – 115, 1921.

[12] Jonathan R Shewchuk. What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures. *Proceedings of the 11th International Meshing Roundtable*, 94720:115 – 126, 2002.

[13] David A Field. Qualitative measures for initial meshes. *International Journal for Numerical Methods in Engineering*, 47(4):887–906, 2000.

[14] V N Parthasarathy Et Al. A comparison of tetrahedron quality measures. *Finite Elements in Analysis and Design*, 15(3):255–261, 1993.

[15] J C Cavendish, D A Field, and W H Frey. An approach to automatic three-dimensional finite element mesh generation. *Internat J Numer Methods Eng*, 21:329–347, 1985.

[16] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. *Proceedings of the 20th annual conference on Computer graphics and interactive techniques SIGGRAPH 93*, d(Annual Conference Series):19–26, 1993.

[17] N Molino, R Bridson, J Teran, and R Fedkiw. *A crystalline, red green strategy for meshing highly deformable objects with tetrahedra*, pages 103–114. 2003.

[18] Lori A Freitag and Carl Ollivier-gooch. A comparison of tetrahedral mesh improvement techniques. *5th International Meshing Roundtable*, pages 87–100, 1996.

[19] Long Chen and Jinchao Xu. Optimal delaunay triangulations. *Journal of Computational Mathematics*, 22(2):299–308, 2004.

[20] John W Milnor. Hyperbolic geometry: The first 150 years. *Bulletin of the American Mathematical Society*, 6(1):9–25, 1982.

[21] J G Ratcliffe. *Foundations of Hyperbolic Manifolds*, volume 149. Springer-Verlag, 1994.

[22] C Lawson. Transforming triangulations. *Discrete Mathematics*, 3:365–372, 1971.

[23] Jonathan Richard Shewchuk. Star splaying : An algorithm for repairing delaunay triangulations and convex hulls. *Star*, page 237, 2005.