Adapted Laplacian Operator For Hybrid Quad/Triangle Meshes

Alexander Pinzón Fernández

Universidad Nacional de Colombia

Facultad de Ingenieria, Departamento de Sistemas e Industrial

Grupo de Investigación CIM@LAB

Bogotá, Colombia

2014

Adapted Laplacian Operator For Hybrid Quad/Triangle Meshes

Alexander Pinzón Fernández

A thesis submitted in partial fulfillment of the requirements for the degree of:

Master in Systems Engineering and Computer Science

Advisor:
Eduardo Romero Castro , Ph.D.

Research Area:
Computer Graphics

Universidad Nacional de Colombia
Facultad de Ingenieria, Departamento de Sistemas e Industrial
Grupo de Investigación CIM@LAB
Bogotá, Colombia
2014

Dedicación

A Beatriz y Campo Elias mis padres que siempre me dieron la libertad de escoger, con su comprensión y apoyo me permitieron dedicar mi tiempo a la ciencia.

A mis padres

Beatriz Fernandez Vargas
Campo Elias Pinzon Rojas

Acknowledgement

I wold like to thank my advisor professor Eduardo Romero and CIM&LAB research group for his support in this thesis.

I would like to thank all the workers in Colombia who fund the public education with their work.

This work was supported in part by the Blender Foundation, Google Summer of code program at 2012 - 2013.

Abstract

In the last two decades three-dimensional modeling methods used by artist have been evolving and developing rapidly thanks to the use of vector operators of differential geometry how the Laplacian operator this operator allows to model in a simple way the behavior of complex applications such as noise reduction, enhancement, remeshing, UV mapping, posed, skeletonization, among other. The Laplacian operator is theoretically defined in a continuous and smooth domain named manifold in practice manifolds are often approximated by discrete polygon meshes composed by triangles and quadrangles which represent the real world three-dimensional objects that artists work. In these meshes spectral structure is calculated using a discrete Laplacian operator, the discrete version of the Laplacian operator given by Pinkall in 1993 works only with triangle meshes, and Xiong in 2011 working exclusively with quads meshes. This thesis proposes an original extension of the Laplacian operator that allows working with hybrid meshes composed by triangles and quadrangles.

Along with the operator, this work presents new sculpting and modeling applications with base on the enhancement, applications on subdivision surfaces using smoothing, mesh posing using differential coordinates and skeletonization using iterative contraction. This series of applications demonstrates the quality, predictability and flexibility of the proposed operator.

The proposed operator was successfully used in real production environments within software Blender how new tools. Currently these tools are available as open source software.

Resumen

En las dos últimas décadas los métodos de modelado tridimensional han ido evolucionando y desarrollándose rápidamente, en parte gracias al uso de operadores vectoriales de geometría diferencial como el operador Laplaciano ha sido uno de los más ampliamente estudiado y usado gracias a las propiedades exhibidas por sus vectores propios en novedosas aplicaciones como: reducción de ruido, realce, remallado, mapeo UV, posado, esqueletonización, entre otras. Este operador diferencial es definido en un dominio continuo y suave llamado variedad, las variedades son a menudo aproximadas por mallas discretas de polígonos compuestas por triángulos y cuadrángulos que a su vez representan objetos tridimensionales del mundo real. En estas mallas se calcula la estructura espectral con el uso de algún operador Laplaciano discreto, la versión discreta del operador Laplaciano propuesta por Pinkall en el 1993 trabaja únicamente con mallas compuestas por triángulos, y la de Xiong en el 2011 trabaja exclusivamente con cuadrángulos. Esta tesis propone una extensión original del Operador Laplaciano que permite trabajar con mallas híbridas compuestas por triángulos y cuadrángulos.

Junto con el operador, este trabajo presenta nuevas aplicaciones en esculpido y modelamiento con base en el realce, aplicaciones en subdivisión de superficies con el uso de suavizado, posado de mallas con el uso de coordenadas diferenciales y esqueletonización usando contracción iterativa. Esta serie de aplicaciones demuestra la calidad, predictibilidad y flexibilidad del operador propuesto.

El operador propuesto fue usado de forma exitosa en ambientes reales de producción dentro del software Blender como nuevas herramientas. Actualmente estas herramientas están disponibles como programas de código abierto.

Keywords: laplacian operator; smooth; enhance; sculpting; subdivision surface

Contents

Acknowledgement iv

Abstract v

1 Introduction 2

2 Mathematical Foundation and Background 4
2.1 Related work . . . 4
2.2 Manifolds . . . 6
2.3 Laplace Operator . . . 7
2.3.1 Discrete Laplace Operator Setting . . . 7
2.3.2 Gradient of Voronoi Area . . . 8
2.4 Laplace Beltrami Operator . . . 8

3 Shape Inflation With an Adapted Laplacian Operator For Hybrid Quad/Triangle Meshes 9
3.1 Introduction . . . 10
3.2 Related work . . . 11
3.3 Laplacian Smooth . . . 12
3.3.1 Gradient of Voronoi Area . . . 12
3.3.2 Laplace Beltrami Operator . . . 13
3.4 Proposed Method . . . 13
3.4.1 Laplace Beltrami Operator for Hybrid Quad/Triangle Meshes TQLBO 13
3.4.2 The Shape Inflation . . . 16
3.5 Sculpting . . . 17
3.6 Subdivision surfaces . . . 18
3.7 Results . . . 19
3.8 Implementation . . . 24
3.9 Conclusion and future work . . . 25

4 Mesh smoothing based on curvature flow operator in a diffusion equation 26
4.1 Synopsis . . . 26
4.2 Benefits to Blender . . . 26

4.3 Deliverables . . . 27
4.4 Project Details . . . 27
4.5 Project Schedule . . . 27
4.6 Mesh Smoothing . . . 28
4.7 Results and Conclusions . . . 29

5 Mesh Editing with Laplacian Deform **32**
5.1 Synopsis . . . 32
5.2 Benefits to Blender . . . 32
5.3 Deliverables . . . 33
5.4 Project Details . . . 33
5.5 Project Schedule . . . 33
5.6 Laplacian Deform . . . 34
5.7 Performance Testing of Solvers . . . 35
5.7.1 Hardware Specification . . . 35
5.7.2 Software Specification . . . 35
5.7.3 Numeric Solvers Used . . . 36
5.8 Results . . . 37

6 Skeleton Extraction **40**
6.1 Results . . . 44

7 Conclusion and future work **46**

Bibliography **46**

List of Figures

2-1 Area of the Voronoi region around v_i in dark blue.v_j belong to the first neighborhood around v_i. α_j and β_j opposite angles to edge $\overrightarrow{v_j - v_i}$. 8

3-1 A set of 48 successive shapes enhanced, from $\lambda = 0.0$ in blue to $\lambda = -240.0$ in red, with steps of -5.0. 9

3-2 Area of the Voronoi region around v_i in dark blue.v_j belong to the first neighborhood around v_i. α_j and β_j opposite angles to edge $\overrightarrow{v_j - v_i}$. 12

3-3 $t^*_{j1} \equiv\triangle\ v_i v_j v'_j$, $t^*_{j2} \equiv\triangle\ v_i v'_j v_{j+1}$, $t^*_{j3} \equiv\triangle\ v_i v_j v_{j+1}$ Triangulations of the quad with common vertex v_i proposed by [Xiong 2011] to define Mean LBO. . . . 14

3-4 The 5 basic triangle-quad cases with common vertex V_i and the relationship with V_j and V'_j. (a) Two triangles [Desbrun 1999]. (b) (c) Two quads and one quad [Xiong 2011]. (d) (e) Triangles and quads (TQLBO) our contribution. . 15

3-5 Family of cups generated with our method, from a coarse model (a), (c): the shape, obtained from the Catmull-Clark Subdivision (b), (d), is inflated. Soft constraints, over the coarse model, is drawn in red and blue (c). 18

3-6 (a) Original Model, (b) Model with Catmull-Clark Subdivision. Models with Laplacian smoothing: (c) and (d). Models with a first Laplacian filtering $\lambda = 60.0$, $\lambda_e = 12.0$ and before applying shape inflation: (e) and (f). 19

3-7 (a) Original Model. (b) Simple subdivision. (c), (d) (e) Laplacian smoothing with $\lambda = 7$ and 2 iterations: (c) for triangles, (d) for quads, (e) for triangles and quads random chosen. 20

3-8 Top row: Original camel model in left. Shape inflation with $\lambda = -30.0$, $\lambda = -100.0$, $\lambda = -400.0$. Bottom row: Shape inflation with weight vertex group, $\lambda = -50.0$ and 2 iterations for the legs, $\lambda = -200.0$ and 1 iteration for the head and neck. 21

3-9 The method is pose insensitive. The inflation for the different poses are similar in terms of shape. Top row: Original walk cycle camel model. Bottom row: Shape inflation with weight vertex group, $\lambda = -400$ and 2 iterations. 22

3-10 Top row: (a) Leg Camel, (b) Inflate brush for leg into blue circle, (c) Inflate shape brush for leg into red circle. Bottom row: (a) Hand, (b) Inflate brush for fingers into blue circle, (c) Shape inflation brush for fingers in red circle. . 22

3-11 Performance of our dynamic shape inflation brush in terms of the sculpted vertices per second. Three models with 12K, 40K, 164K vertices used for sculpting in real time. 23
3-12 (a) Bottom row: Original Model. Top row: Original model scaled by 4. (b) Top and bottom row: inflating with Normalized-TQLBO $\lambda = -50$ (c) Top and bottom row: inflating with TQLBO $\lambda = -50$. 24

4-1 Panel inside blender user interface of the Laplacian Smooth modifier tool. . . 29
4-2 Noise attenuation in face model with Laplacian smoothing tool using only one iteration and changing λ. (a) Original Model. (b) Smoothing $\lambda = 0.5$. (c) Smoothing $\lambda = 2.5$ (d) Smoothing with $\lambda = 5.0$. 30
4-3 Smoothing boundary changing λ_{Border} factor. (a) Original Model. (b) Smoothing $\lambda_{Border} = 1.0$. (c) Smoothing $\lambda_{Border} = 2.5$ (d) Smoothing with $\lambda_{Border} = 10.0$. 30
4-4 Use of weights per vertex to constrain the effect of mesh smoothing. (a) Original Model. (b) Smoothing with $\lambda = 1.5$ (c) red vertices $weight = 1.0$, blue vertices $weight = 0.0$. (d) Smoothing with $\lambda = 2.5$. Red vertices were only smoothing. 31

5-1 Difference between v_i and the center of mass of its neighboors $v_1, ..., v$. 34
5-2 Plot of Vertices Vs Seconds, Initial factorization performance. 37
5-3 Panel inside blender user interface of the Laplacian Deform modifier tool. . . 37
5-4 Anchor vertices in blue color. (a) Original Model, (b,c,d) new poses change only anchor vertices, system find positions for vertices in yellow color. 38
5-5 (a) Original cactus model. (b) Rotate 70^{o} to right the blue segments with basic interpolation (c) Rotate 70^{o} to right the blue segments with Laplacian deform tool. 39
5-6 (a) Original Horse model. (b) Translate and rotate the blue segments with basic interpolation (c) Translate and rotate the blue segments with Laplacian deform tool. 39

6-3 From left to right iterative mesh contraction. 40
6-1 Poster *Software para la Extracción del Esqueleto por Contracción y Suavizado* presented at 7th International Seminar on Medical Image Processing and Analysis SIPAIM 2011. 41
6-2 Poster *Análisis Experimental de la Extracción del Esqueleto por Contracción con Suavizado Laplaciano* presented at 6th International Seminar on Medical Image Processing and Analysis SIPAIM 2010. 42
6-4 Left: The vertex x_i move along line constraint. Right: the distance of vertex x_i to plane 1 and plane 2 when change the position in every iteration. 43

6-5 Model with different poses and skeleton obtained with our skeleton extraction software. 45

List of Tables

5-1 Vertices Vs Seconds, Laplacian Deform initial factorization performance. . . 36

1 Introduction

The discrete versions of Laplace Beltrami Operator have been used in the last years for the develop of new geometric modeling tools. In the work of Pinkall [31] was introduced the cotangent version of Laplace operator, that allow to find the minimal surface when compute a discete harmonic map with the Laplacian operator, this version has been widely studied and applied in various problems of computer geometric modeling. The manifolds that are continuous domains homeomorphic to $\mathbb{R}^n$ are represented in computers by polygon meshes. These polygons are generally composed of triangles and quadrangles, while work with laplacian operator in this hybrid composition are not a mathematical challenge, most research uses only meshes composed by triangles [31, 17, 27, 38, 4, 5, 21], in other recent studies [25, 44] the Laplacian operator is working exclusively with quadrangles. But in the artistic scope topology and the way the edges are distributed, the triangle and quadrangles, directly affects the processes of animation, interpolation, textured, etc. As discussed by [28] who use a manual connection of a couple of vertices to perform animation processes and interpolation. It is then of paramount importance to develop operators that easily interact with such meshes, eliminating the need of preprocessing the mesh to convert it to triangles and change the original topology.

Modeling techniques able to generate a variety of realistic shapes, have been developed [8]. Editing techniques have evolved from affine transformations to advanced tools like sculpting [13, 19, 42], editing, creation from sketches [22, 20], and complex interpolation techniques [38, 46]. Catmull-Clark based methods however require to interact with a minimun number of control points for any operation to be efficient, or in other words, a unicity condition is introduced by demanding a smooth surface after any of these shape operations. Hence, traditional modeling methods for subdividing surfaces from coarse geometry have become widely popular [10, 41]. These works have generalized a uniform B-cubic spline knot insertion to meshes, some of them adding some type of control, for instance with the use of creases to produce sharp edges [15], or the modification of some vertex weights to locally control the zone of influence [6]. Neverteless these methods are difficult to deal with since they require a large number of parameters and a very tedious customization. Instead, the presented applications requires a single parameter that controls the global curvature, which is used to maintain realistic shapes, creating a family of different versions of the same object and therefore preserving the detail of the original model and a realistic appearance.

The shape inflation and shape exaggeration can thus be used as such brush in the sculpting

process, when inflating a shape since current brushes end up by losing detail when moving vertices [42]. In contrast, the presented enhance method inflates a mesh by moving the vertices towards the reverse curvature direction, conserving the shape and sharp features of the model.

Contributions This work presents an extension of the Laplace Beltrami Operator for hybrid quad/triangle meshes, representing a larger mesh spectrum from what has been presented so far. The method eliminates the need of preprocessing and allows preservation of the original topology. Likewise, along with this operator, it is proposed a method to generate a family of parameterized shapes, in a robust and predictable way. This method enables customization of the smoothness and curvature, obtained during the subdivision surfaces process. Finally, it is proposed a new brush for inflating the silhouette mesh features in modeling and sculpting.

This work is organized as follows: Chapter 2 presents works related to the Laplacian operator, applications in digital sculpting, deformation, and offsetting methods for polygonal meshes also it is described the theoretical framework of the Laplacian operator for polygon meshes; In chapter 3.4, it is presented the extension of the Laplacian Operator for hybrid meshes and applications of shape inflation ,subdivision of surfaces , deformation, skelenotization and sculpting; finally some Laplacian Operator results, to hybrid quad/triangle meshes are graphically shown as well as results of the shape inflation applications in sculpting, subdivision, deformation, skelenotization and modeling.

2 Mathematical Foundation and Background

This chapter studies basic mathematical foundation on differential geometry to understand the differential operators and the Laplace beltrami operator.

The differential geometry studies curvatures and geodesics [21]. The differential operators show a deep relationships betwen the geoemetry (curvatures, geodesics) and topology of the manifold. This differential operators are very used in applications on computer geometric modelling over the last years [36, 2].

2.1 Related work

Many tools have been developed for modeling, based on the Laplacian mesh processing. These different tools preserve the surface geometric details when using laplacian operators for different processes such as smoothing, enhancing, free-form deformation, fusion, morphing and other applications [35].

The most used discretization of Laplace Beltrami operator Δ_Ω over a triangulated mesh Ω was proposed by Pinkall [31].

$$\Delta_\Omega(u) = \frac{1}{2}\sum_{j\in N_1(i)}(\cot\alpha_j + \cot\beta_j)(x_i - x_j)$$

Where $N_1$is the 1-ring neighboorhood, α and β are the opposite angles to edge between vertex i and vertex j. In this work the discretize laplacian operator was used to find the minimal surface based on minimization energy strategy using the Dirichlet's energy of the function u over a manifold represented by triangulated mesh Ω.

$$E_D(u) = \frac{1}{2}\int_\Omega \left|\nabla u\right|^2$$

Then Taubin works [43] for first time treat the problem of noise reduction in digital poligonal meshes from signal proccessing point of view. Taubin extends Fourier analysis to signals defined on polygonal meshes, Taubin observe that Fourier transform is a decomposition of the signal into an eigenvectors of the Laplacian operator, and reconstruct the signal with

a linear combination of these eigenvectors. Desbrun works [17] consider the same approach of Taubin, but Desbrun use a curvature normal ($\bar{\kappa}\mathbf{n}$) based on cotangen Laplace operator version for noise reduction over a difussion process. This it's the most important and popular Laplace Beltrami opertator discretization [25], many works for mesh smoothing and fairing have been developed based on this LBO discretization [16, 27, 37, 29].

$$\frac{\partial x_i}{\partial t} = -\bar{\kappa_i}\mathbf{n_i}$$

$$-\bar{\kappa_i}\mathbf{n_i} = \frac{1}{A}\sum_{j\in N_1(i)}\left(\cot\alpha_j + \cot\beta_j\right)$$

Where A is the area surrounding vertex i. This laplacian operator L was used for reduce the noise in a mesh X over a difussion process.

The convergence of the Laplace Beltrami operator has been very important in fields how numerical analysis, given that implications in the simulation process and geometric partial differentials equations. In the works of [45] is established the convergence of several discrete Laplace Beltrami operators over triangulated meshes with numerical results that support the theoretical analysis. Over quadrilateral meshes Liu [25] presents a discrete Laplace Beltrami Operator based on a bilinear interpolation and its convergence over this type of meshes composed only by quads.

In the work of Sorkine [39] the laplacian operator was used to repose a meshes while preserving geometry details of the surface. The details was stored in differential coordinates δ_i for every vertex v_i.

$$\delta_i = \sum_{j\in N_1(i)}\frac{1}{2}\left(\cot\alpha_j + \cot\beta_j\right)\left(v_i - v_j\right)$$

The differential coordinates represents the difference between the absolut coordinate of $v_i$and the center of mass of its immediate neighboors.

Offset methods for polygon meshing, based on the curvature defined by the Laplace Beltrami operator, have been developed. These methods adjust the shape offset by a constant distance, with enough precision. Nevertheless, these methods fail to conserve sufficient detail because of the smoothing, a crucial issue which depends on the offset size [47]. In volumetric approaches, in case of point-based representations, the offset boundary computation is based on the distance field and therefore when calculating such offset, the topology of the model may be different to the original [12].

[18] propose automatic feature detection and shape edition with feature inter-relationship preservation. They define salient surface features like ridges and valleys, characerized by their first and second order curvature derivatives, see [30], and angle-based threshold. Likewise, curves have been also classified as planar or non-planar, approximated by lines, circles,

ellipses and other complex shapes. In such case, the user defines an initial change over several features which is propagated towards other features, based on the classified shapes and the inter-relationships between them. This method works well with objects that have sharp edges, composed of basic geometric shapes such as lines, circles or ellipses. However, the method is very limited when models are smooth since it cannot find the proper features to edit.

Digital sculpting have been traditionally approached either under a polygonal representation or a voxel grid-based method. Brushes for inflation operations only depend on the vertex normal [42]. In grid-based sculpting, some other operations have allowed to add or remove voxels since production of polygonal meshes require a processing of isosurfaces from a volume [19]. The drawback comes from the difficulty of maintaining the surface details during larger scale deformations.

In the literature, several studies have described the skeleton extraction systems and different metrics that identify appropriate methods given an application specific [3]. One of the best methods reported in the literature for the extraction of the skeleton is the Laplacian smoothing method for its advantages of homotopy representation and hierarchical connections between parts. The Skeleton extraction methods permits simplify the dimension of the object preserving the topological structure [14], Au et. al. [4] present a skeleton extraction method based on iterative smooothing-contraction, in this method the several constraints are used to waranty that the process converge to skeleton formed by branchs and joins, the constraints based on laplacian operator, the low frequencies of the mesh are preserving with the use of attractor to original mesh, while the iterative smoothin process remove high frecuencies.

2.2 Manifolds

A manifold is a topological space M with the following properties:

If $x \in M$, then there is some neighborhood $N(x)$ and some integer $n \geq 0$ such that $N(x)$ is homeomorphic to $\mathbb{R}^n$ [40].

This work is related to manifolds that represent a surface in an three-dimensional Euclideean Space. This manifolds are homeomorphic to $\mathbb{R}^2$.

The manifols are represented by polygonal meshes with points connected by triangles and quads.

2.3 Laplace Operator

In computer graphics a manifold is often approximated by a discrete mesh [35], then is necessary to define a discrete laplacian operator that act on functions defined on such meshes.

Consider a smooth compact manifold M of dimension m isometrically embedded in a Euclidean space $\mathbb{R}^d$.

Given a twice continuously differentiable function $f \in C^2(M)$, let $\nabla_{\mathrm{M}} f$ denote the gradient vector field of f on M.

The Laplace-Beltrami operator Δ_{M} of f is defined as the divergence of the gradient; that is [40],

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0$$

$$\nabla^2_{\mathrm{M}} f = \Delta_{\mathrm{M}} f = 0$$

$$\Delta_{\mathrm{M}} f = div\left(\nabla_{\mathrm{M}} f\right) \tag{2-1}$$

2.3.1 Discrete Laplace Operator Setting

Discrete Laplacian operators are linear operators that act on functions defines on meshes. These functions are specified by their values at the vertices.

Thus, if a mesh M has n vertices, then functions on M will be represented by vectors with n components and a mesh Laplacian will be described by an $n \times n$ matrix [35].

The Laplacian operator locally takes the difference between the value of a function at a vertex and a weighted average of its values at the first-order or one-ring neighbor vertices, then a Laplacian matrix L has a local form given by

$$L(f)_i = b_i^{-1} \sum_{j \in N(i)} w_{ij} \left(f_i - f_j\right)$$

Where $w_{ij}$are the weights between the vertex i and the vertex j. b_i^{-1} are the factors depending of the boundary region over vertex i. $N(i)$ are the neighbors that share a edge with vertex i.

2.3.2 Gradient of Voronoi Area

Consider a surface S composed of a set of triangles around vertex v_i. Let us define the *Voronoi Region* of v_i as show in figure **2-1**, The area change produced by the movement of v_i is called the gradient of *Voronoi region* [31, 17, 27].

$$\nabla A = \frac{1}{2}\sum_{j} \left(\cot \alpha_j + \cot \beta_j\right)\left(v_i - v_j\right) \tag{2-2}$$

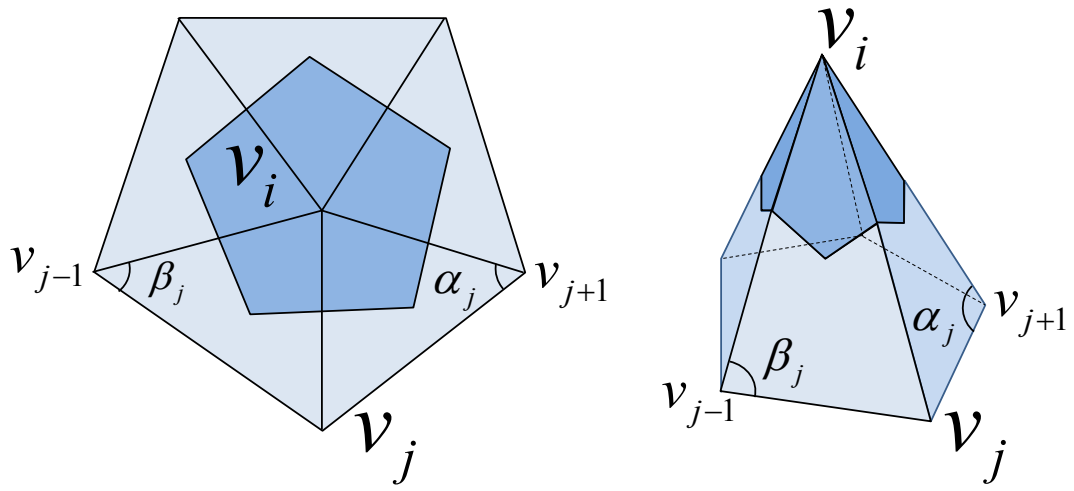


Figure **2-1**: Area of the Voronoi region around v_i in dark blue.v_j belong to the first neighborhood around v_i. α_j and β_j opposite angles to edge $\overrightarrow{v_j - v_i}$.

If the gradient in equation 2-2 is normalized by the total area of the 1-ring neighborhood around v_i, the *discrete mean curvature normal* of a surface S is obtained, as shown in equation 2-3.

$$2\kappa\mathbf{n} = \frac{\nabla A}{A} \tag{2-3}$$

2.4 Laplace Beltrami Operator

The *Laplace Beltrami operator* LBO noted as Δ is used for measuring the mean curvature normal to the Surface S [31].

$$\Delta_S = 2\kappa\mathbf{n} \tag{2-4}$$

3 Shape Inflation With an Adapted Laplacian Operator For Hybrid Quad/Triangle Meshes

Alexander Pinzon, Eduardo Romero
Cimalab Research Group
Universidad Nacional de Colombia
Bogota-Colombia
Email: apinzonf@unal.edu.co, edromero@unal.edu.co

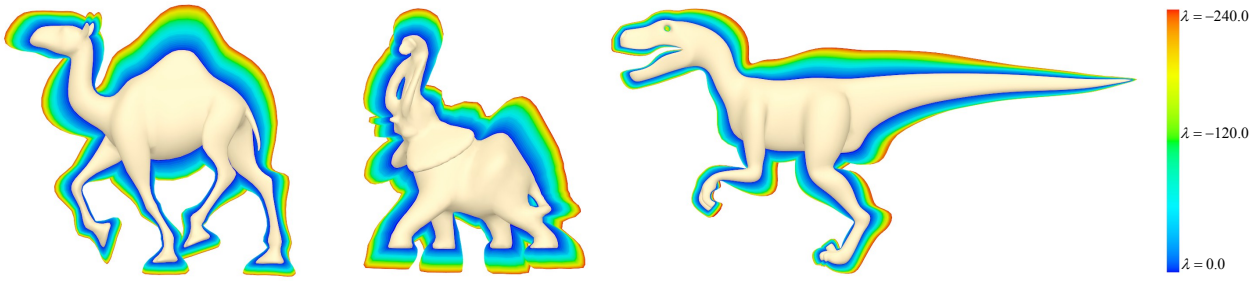


Figure **3-1**: A set of 48 successive shapes enhanced, from $\lambda = 0.0$ in blue to $\lambda = -240.0$ in red, with steps of -5.0.

This paper was accepted and presented in the Brazilian Symposium on Computer Graphics and Image Processing SIBGRAPI 2013. The conference was held on August 5-8 of 2013, in the city of Arequipa, Perú [34].

Abstract

This paper proposes a novel modeling method for a hybrid quad/triangle mesh that allows to set a family of possible shapes by controlling a single parameter, the global curvature. The method uses an original extension of the Laplace Beltrami operator that efficiently

estimates a curvature parameter which is used to define an inflated shape after a particular operation performed in certain mesh points. Along with the method, this work presents new applications in sculpting and modeling, with subdivision of surfaces and weight vertex groups. A series of graphics examples demonstrates the quality, predictability and flexibility of the method in a real production environment with software Blender.

keywords- laplacian smooth; curvature; sculpting; subdivision surface

3.1 Introduction

Over the last years several, modeling techniques able to generate a variety of realistic shapes, have been developed [8]. Editing techniques have evolved from affine transformations to advanced tools like sculpting [13, 19, 42], editing, creation from sketches [22, 20], and complex interpolation techniques [38, 46]. Catmull-Clark based methods however require to interact with a small number of control points for any operation to be efficient, or in other words, a unicity condition is introduced by demanding a smooth surface after any of these shape operations. Hence, traditional modeling methods for subdividing surfaces from coarse geometry have become widely popular [10, 41]. These works have generalized a uniform B-cubic spline knot insertion to meshes, some of them adding some type of control, for instance with the use of creases to produce sharp edges [15], or the modification of some vertex weights to locally control the zone of influence [6]. Nevertheless these methods are difficult to deal with since they require a large number of parameters and a very tedious customization. Instead, the presented method requires a single parameter that controls the global curvature, which is used to maintain realistic shapes, creating a family of different versions of the same object and therefore preserving the detail of the original model and a realistic appearance.

Interest in meshes composed of triangles and quads has lately increased because of the flexibility of modeling tools such as Blender 3D [7]. Nowadays, many artists use a manual connection of a couple of vertices to perform animation processes and interpolation [28]. It is then of paramount importance to develop operators that easily interact with such meshes, eliminating the need of preprocessing the mesh to convert it to triangles. The shape inflation and shape exaggeration can thus be used as such brush in the sculpting process, when inflating a shape since current brushes end up by losing detail when moving vertices [42]. In contrast, the presented method inflates a mesh by moving the vertices towards the reverse curvature direction, conserving the shape and sharp features of the model.

Contributions

This work presents an extension of the Laplace Beltrami operator for hybrid quad/triangle meshes, representing a larger mesh spectrum from what has been presented so far. The

method eliminates the need of preprocessing and allows preservation of the original topology. Likewise, along with this operator, it is proposed a method to generate a family of parametrized shapes, in a robust and predictable way. This method enables customization of the smoothness and curvature, obtained during the subdivision surfaces process. Finally, it is proposed a new brush for inflating the silhouette mesh features in modeling and sculpting.

This work is organized as follows: Section 3.2 presents works related to the Laplacian mesh processing, digital sculpting, and offsetting methods for polygonal meshes; In section 3.3, it is described the theoretical framework of the Laplacian operator for polygon meshes; In section 3.4, it is presented the method for shape inflation and applications of subdivision of surfaces and sculpting; finally some Laplacian operator results, to hybrid quad/triangle meshes are graphically shown as well as results of the shape inflation applications in sculpting, subdivision and modeling.

3.2 Related work

Many tools have been developed for modeling, based on the Laplacian mesh processing. Thanks to the advantages of the Laplacian operator, these different tools preserve the surface geometric details when using them for different processes such as free-form deformation, fusion, morphing and other applications [38].

Offset methods for polygon meshing, based on the curvature defined by the Laplace Beltrami operator, have been developed. These methods adjust the shape offset by a constant distance, with enough precision. Nevertheless, these methods fail to conserve sufficient detail because of the smoothing, a crucial issue which depends on the offset size [47]. In volumetric approaches, in case of point-based representations, the offset boundary computation is based on the distance field and therefore when calculating such offset, the topology of the model may be different to the original [12].

[18] propose automatic feature detection and shape edition with feature inter-relationship preservation. They define salient surface features like ridges and valleys, characerized by their first and second order curvature derivatives, see [30], and angle-based threshold. Likewise, curves have been also classified as planar or non-planar, approximated by lines, circles, ellipses and other complex shapes. In such case, the user defines an initial change over several features which is propagated towards other features, based on the classified shapes and the inter-relationships between them. This method works well with objects that have sharp edges, composed of basic geometric shapes such as lines, circles or ellipses. However, the method is very limited when models are smooth since it cannot find the proper features to edit.

Digital sculpting have been traditionally approached either under a polygonal representation or a voxel grid-based method. Brushes for inflation operations only depend on the vertex

normal [42]. In grid-based sculpting, some other operations have allowed to add or remove voxels since production of polygonal meshes require a processing of isosurfaces from a volume [19]. The drawback comes from the difficulty of maintaining the surface details during larger scale deformations.

3.3 Laplacian Smooth

Computer objects, reconstructed from the real world, are usually noisy. Laplacian Smooth techniques allow a proper noise reduction on the mesh surface with minimal shape changes, while still preserving a desirable geometry as well as the original shape.

Many smoothing Laplacian functionals regularize the surface energy by controlling the total surface curvature S.

$$E(S) = \int_S \kappa_1^2 + \kappa_2^2 dS$$

Where κ_1 and κ_2 are the two principal curvatures of the surface S.

3.3.1 Gradient of Voronoi Area

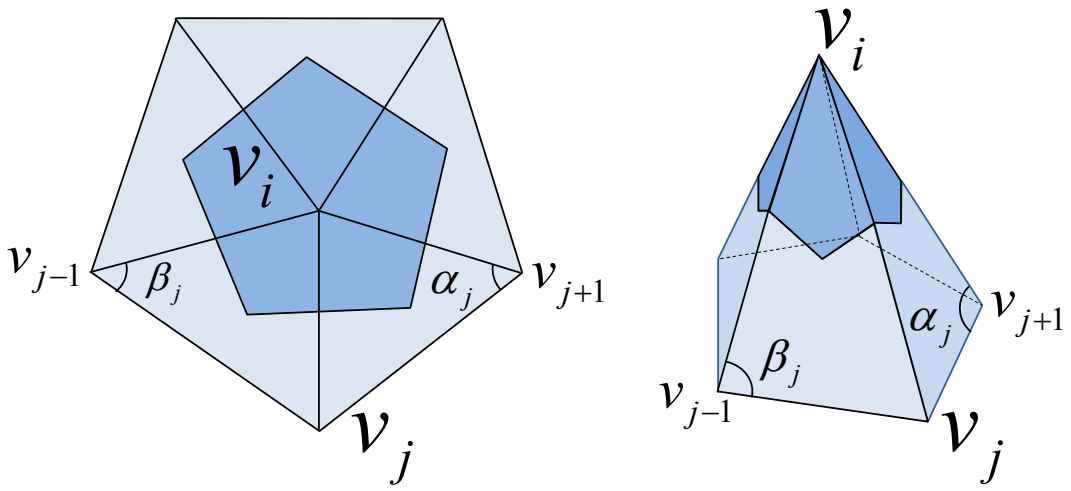


Figure **3-2**: Area of the Voronoi region around v_i in dark blue.v_j belong to the first neighborhood around v_i. α_j and β_j opposite angles to edge $\overrightarrow{v_j - v_i}$.

Consider a surface S composed of a set of triangles around vertex v_i. Let us define the *Voronoi region* of v_i as show in figure **3-2**, The area change produced by the movement of v_i is called the gradient of *Voronoi region* [31, 17, 27].

$$\nabla A = \frac{1}{2}\sum_{j} (\cot \alpha_j + \cot \beta_j) \tag{3-1}$$

If the gradient in equation (3-1) is normalized by the total area of the 1-ring neighborhood around v_i, the *discrete mean curvature normal* of a surface S is obtained, as shown in equation (3-2).

$$2\kappa \mathbf{n} = \frac{\nabla A}{A} \tag{3-2}$$

3.3.2 Laplace Beltrami Operator

The *Laplace Beltrami operator* LBO noted as $\triangle$ is used for measuring the mean curvature normal to the Surface S [31].

$$\triangle S = 2\kappa \mathbf{n} \tag{3-3}$$

The LBO has desirable properties: the LBO points out to the direction of the minimal surface area.

3.4 Proposed Method

This method exaggerates a shape using a Laplacian smoothing operator in the reverse direction, i.e., the new shape is a modified version in which those areas with larger curvature are magnified. The operator amounts to a generator of a set of models which conserves the basic silhouette of the original shape. In addition, the presented approach can be easily mixed with traditional or uniform subdivision of surfaces. This method is based on an original extension of the Laplace Beltrami operator for hybrid quad/triangle meshes, mixing arbitrary types of meshes, exploiting the basic geometrical relationships and ensuring good results with few algorithm iterations.

3.4.1 Laplace Beltrami Operator for Hybrid Quad/Triangle Meshes TQLBO

Given a mesh $M = (V, Q, T)$, with vertices V, quads Q, triangles T. The area of 1-ring neighborhood $A(v_i)$ corresponds to a sum of the quad faces $A(Q_{v_i})$ and the areas of the triangular faces $A(T_{v_i})$ adjacent to vertex v_i.

$$A(v_i) = A(Q_{v_i}) + A(T_{v_i})$$

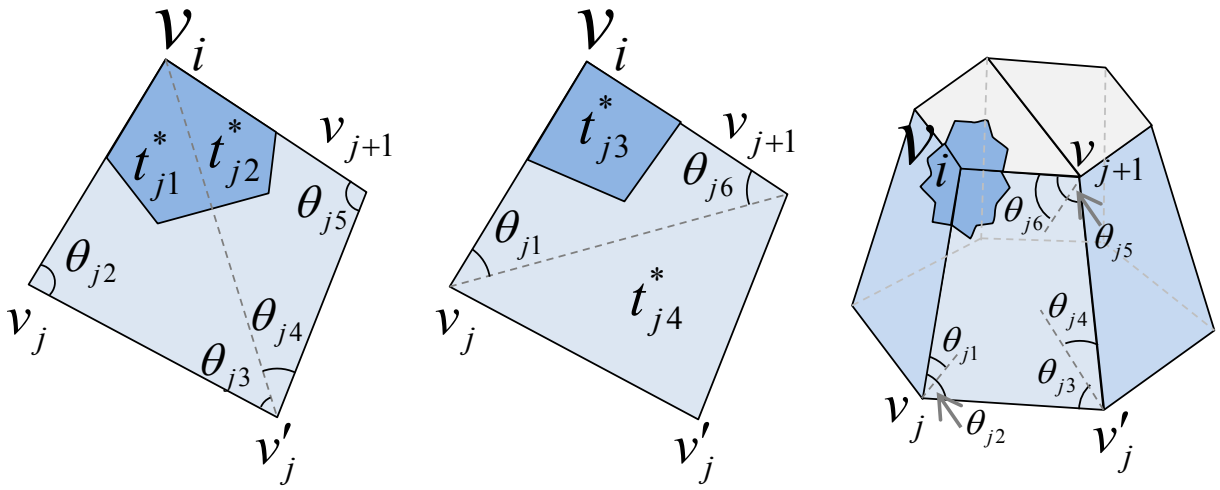


Figure **3-3**: $t^*_{j1} \equiv\triangle\ v_i v_j v'_j$, $t^*_{j2} \equiv\triangle\ v_i v'_j v_{j+1}$, $t^*_{j3} \equiv\triangle\ v_i v_j v_{j+1}$ Triangulations of the quad with common vertex v_i proposed by [Xiong 2011] to define Mean LBO.

Applying the mean average area, according to [44], from all possible triangulations, as show in figure **3-3**, the area for quads $A(Q_{v_i})$ and triangles $A(T_{v_i})$ is.

$$A(v_i) = \tfrac{1}{2^m}\sum_{j=1}^{m} 2^{m-1} A(q_j) + \sum_{k=1}^{r} A(t_k)$$

Where $q_1, q_2, ..., q_j, ..., q_m \in Q_{v_i}$ and $t_1, t_2, ..., t_k, ..., t_r \in T_{v_i}$

$$A(v_i) = \frac{1}{2}\sum_{j=1}^{m}\left[A\left(t^*_{j1}\right) + A\left(t^*_{j2}\right) + A\left(t^*_{j3}\right)\right] + \sum_{k=1}^{r} A(t_k) \tag{3-4}$$

Applying the gradient operator to (3-4)

$$\nabla A(v_i) = \tfrac{1}{2}\sum_{j=1}^{m}\left[\nabla A\left(t^*_{j1}\right) + \nabla A\left(t^*_{j2}\right) + \nabla A\left(t^*_{j3}\right)\right] + \sum_{k=1}^{r} \nabla A(t_k)$$

According to (3-1), we have

$$\nabla A\left(t^*_{j1}\right) = \frac{\cot\theta_{j3}(v_j - v_i) + \cot\theta_{j2}\left(v'_j - v_i\right)}{2}$$

$$\nabla A\left(t^*_{j2}\right) = \frac{\cot\theta_{j5}\left(v'_j - v_i\right) + \cot\theta_{j4}(v_{j+1} - v_i)}{2}$$

$$\nabla A\left(t_{j3}^{*}\right)=\frac{\cot\theta_{j6}(v_j-v_i)+\cot\theta_{j1}(v_{j+1}-v_i)}{2}$$

$$\nabla A\left(t_k\right)=\frac{\cot\alpha_k(v_k-v_i)+\cot\beta_{k+1}(v_{k+1}-v_i)}{2}$$

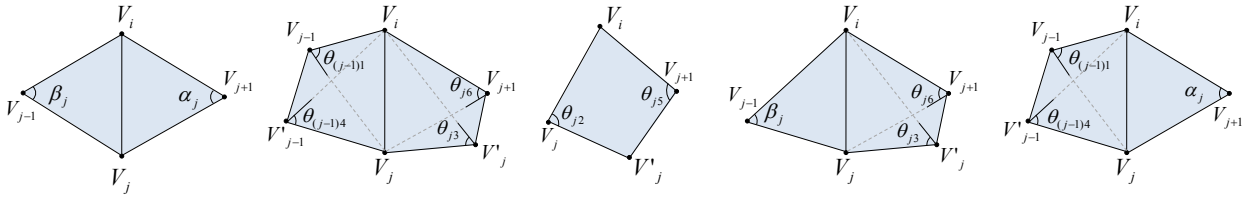


Figure **3-4**: The 5 basic triangle-quad cases with common vertex V_i and the relationship with V_j and V_j'. (a) Two triangles [Desbrun 1999]. (b) (c) Two quads and one quad [Xiong 2011]. (d) (e) Triangles and quads (TQLBO) our contribution.

Triangle and quad configurations of the 1-ring neighborhood faces, adjacent to v_i, can be simplified to five cases, as shown in figure **3-4**.

According to equation (3-2), (3-3), and five simple cases defined in figure **3-4** the TQLBO (Triangle-Quad LBO) of v_i is

$$\triangle_S\left(v_i\right)=2\kappa\mathbf{n}=\frac{\nabla A}{A}=\frac{1}{2A}\sum_{v_j\in N_1(v_i)}w_{ij}\left(v_j-v_i\right)$$

$$w_{ij}=\begin{cases}\left(\cot\alpha_j+\cot\beta_j\right) & \text{case } a.\\ \frac{1}{2}\left(\cot\theta_{(j-1)1}+\cot\theta_{(j-1)4}+\cot\theta_{j3}+\cot\theta_{j6}\right) & \text{case } b.\\ \left(\cot\theta_{j2}+\cot\theta_{j5}\right) & \text{case } c.\\ \frac{1}{2}\left(\cot\theta_{j3}+\cot\theta_{j6}\right)+\cot\beta_j & \text{case } d.\\ \frac{1}{2}\left(\cot\theta_{(j-1)1}+\cot\theta_{(j-1)4}\right)+\cot\alpha_j & \text{case } e.\end{cases} \tag{3-5}$$

We define a TQLBO as a matrix equation

$$L\left(i,j\right)=\begin{cases}-\frac{1}{2A_i}w_{ij} & \text{if } j\in N\left(v_i\right)\\ \frac{1}{2A_i}\sum\limits_{j\in N(v_i)}w_{ij} & \text{if } i=j\\ 0 & \text{otherwise}\end{cases} \tag{3-6}$$

Where L is a $n\times n$ matrix, n is the number of vertices of a given mesh M, w_{ij} is the TQLBO defined in equation (3-5), $N\left(v_i\right)$ is the 1-ring neighborhood with shared face to v_i, A_i is the ring area around v_i.

Normalized equation of the TQLBO

$$L(i,j) = \begin{cases} -\frac{w_{ij}}{\sum\limits_{j \in N(v_i)} w_{ij}} & \text{if } j \in N(v_i) \\ \delta_{ij} & \text{otherwise} \end{cases} \tag{3-7}$$

Where δ_{ij} being the Kronecker delta function.

3.4.2 The Shape Inflation

The shape is inflated by using the reverse direction of the curvature flow, moving the vertices towards those mesh portions with larger curvature. A standard diffusion process is applied:

$$\frac{\partial V}{\partial t} = \lambda L(V)$$

To solve this equation, implicit integration is used as well as a normalized version of TQLBO matrix

$$(I - |\lambda dt| W_p L) V' = V^t \tag{3-8}$$

$$V^{t+1} = V^t + \text{sign}(\lambda)(V' - V^t)$$

The vertices V^{t+1} are inflated, along their reverse curvature direction, by solving the linear system: $Ax = b$, where $A = I - |\lambda dt| W_p L$, L is the Normalized TQLBO defined in the equation (3-7), $x = V'$ are the smoothing vertices, $b = V^t$ are the actual vertices positions, W_p is a diagonal matrix with vertex weights, and λdt is the inflate factor that supports negative and positive values: negative for inflation and positive for smoothing.

The method was devised to use with weighted vertex groups, which specify the final shape inflation of the solution, meaning \$0\$ as no changes and 1 when a maximal change is applied. The weights modify the influence zones, where the Laplacian is applied, as shown in equation 3-8 . Interestingly, the generated family of shapes may change substantially with the weights of specific control points.

The curvature cannot be calculated at the boundary of the meshes that are not closed, for that reason we use the scale-dependent operator proposed by Desbrun et al. [17], the inflation factor for boundary is represented by λ_e.

The model volume increases as the lambda is larger and negative, this can be counteracted with a simple volume preservation. However, the mesh may suffer large displacements when $\lambda < -1.0$ or after multiple iterations. A simple volume conservation algorithm is: If v_i^{t+1} is a mesh vertex of V^{t+1} in the $t+1$ iteration, we define $\overline{v}$ as:

$$\overline{v} = \frac{1}{n} \sum_{v_i \in V} v_i$$

$\overline{v}$is the mesh center, vol_{ini} is an initial volume, and vol_{t+1} is the volume at the iteration $t+1$, n is the number of vertices, then the scale factor

$$\beta = \left(\frac{vol_{ini}}{vol_{t+1}} \right)^{\frac{1}{3}}$$

allows to scale the vertices to:

$$v_{i\,new}^{t+1} = \beta \left(v_i^{t+1} - \overline{v} \right) + \overline{v}$$

The shape inflation use a negative curvature flow that is an unstable process when performing many iterations, however, our method uses less than 3 iterations to get good results, and with 3 iterations or less the method behaves stable.

3.5 Sculpting

A new sculpting brush is herein proposed and aims to inflate the shape, magnifying the shape curvatures of a polygon mesh in real time. This brush works properly with the stroke method *Drag Dot*, allowing to pre-visualize the model changes before the mouse is released. Also, it allows to move the mouse along the model to match the shape zone which is supposed to be inflated.

Brushes that perform a similar inflation can introduce mesh distortions or produce mesh self-intersections, provided these brushes only move the vertices along the normal without any global information. In contrast, the present method searches for a proper inflation while preserving the global curvature, retaining the original shape and main model features. In addition, this method simplifies the work required for the inflation since it needs not different brushes for inflating, softening or styling. The inflated brush can make all these operation in a single step. Real-time brushes require the Laplacian matrix is constructed with the vertices that are within the sphere radius defined by the user, reducing the matrix to be processed, the center of this sphere depending on the place where the user clicks on the canvas and the three-dimensional mesh placed where the click is projected. Special handling is required for the boundary vertices with neighbors that are not within the brush radius: these vertices are marked as boundary and the curvature is not there calculated, but they must be included in the matrix so that every vertex has their corresponding neighbors within the selection. The sculpting Laplacian matrix reads as.

$$L(i,j) = \begin{cases} -\frac{w_{ij}}{\sum\limits_{j \in N(v_i)} w_{ij}} & \text{if } \|v_i - u\| < r \wedge \|v_j - u\| < r \\ 0 & \text{if } \|v_i - u\| < r \wedge \|v_j - u\| \geq r \\ \delta_{ij} & \text{otherwise} \end{cases}$$

Where $v_j \in N(v_i)$, u is the sphere center of radius r. The matrices should remove rows and columns of vertices that are not within the radius.

3.6 Subdivision surfaces

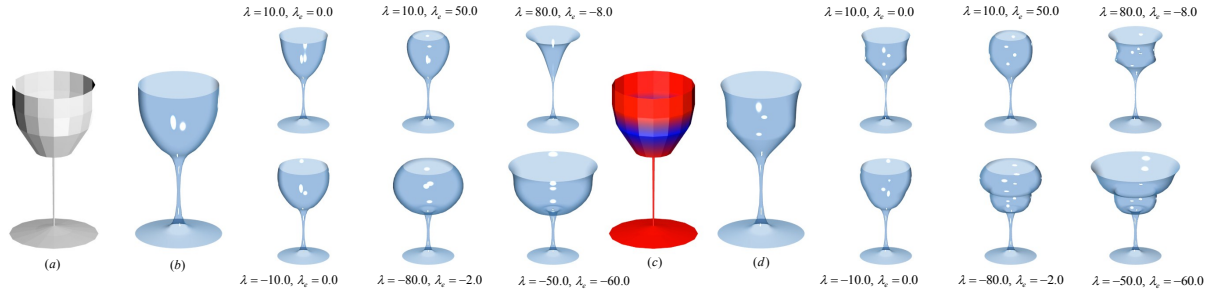


Figure **3-5**: Family of cups generated with our method, from a coarse model (a), (c): the shape, obtained from the Catmull-Clark Subdivision (b), (d), is inflated. Soft constraints, over the coarse model, is drawn in red and blue (c).

The Catmull-Clark subdivision transformation is used to smooth a surface, as the limit of a sequence of subdivision steps [41]. This process is governed by a B-spline curve [26], performing a recursive subdivision transformation that refines the model into a linear interpolation that approximates a smooth surface. The model smoothness is automatically guaranteed [15].

Catmull-Clark subdivision surface methods generate smooth and continuous models from a coarse model and produce quick results because of the simplicity of implementation. Nevertheless, changes to the global curvature are hardly implantable. The Catmull-Clark subdivision surfaces together with shape inflation can easily generate families of shapes by changing a single parameter, allowing to handle a model with very few vertices. In practice, this would allow an artist to choose a model from a similar set of options that would meet his/her needs without having to change each of the control vertices. Likewise, the presented method allows the use of vertex weight paint over the control points. The weights can be applied to a coarse model, followed by a Catmull-Clark subdivision where weights are interpolated, producing weights with smooth changes in the influence zones, as shown in figure **3-5**.c.

In equation 3-8, W_p is a diagonal matrix with weights corresponding to each vertex. Weights at each vertex produce a different solution so that the matrix must be placed in the diffusion equation since families that are generated may change substantially with weighted of specific control points.

3.7 Results

The results of the shape inflation method with the extension of the Laplace Beltrami operator for hybrid quad/triangle meshes with several example models shown in figures **3-1**, **3-5**, **3-6**, **3-7**, **3-8**, **3-10**, **3-11**, **3-12**, **3-9**. The shape inflation was assessed with TQLBO method on a PC with AMD Quad-Core Processor @ 2.40 GHz and 8 GB RAM.

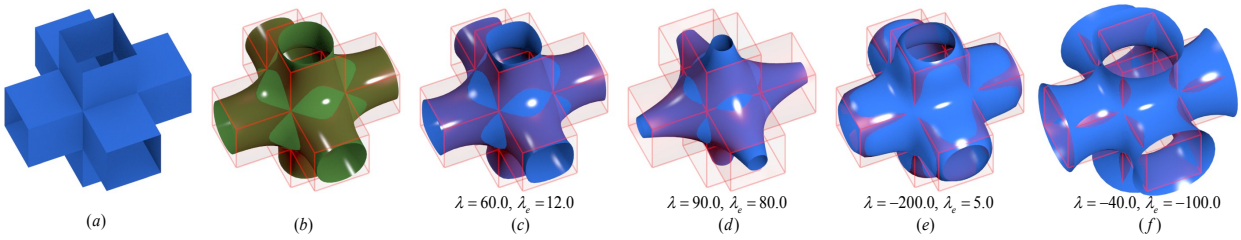


Figure **3-6**: (a) Original Model, (b) Model with Catmull-Clark Subdivision. Models with Laplacian smoothing: (c) and (d). Models with a first Laplacian filtering $\lambda = 60.0$, $\lambda_e = 12.0$ and before applying shape inflation: (e) and (f).

Figure **3-7** shows the results when applying the Laplace Beltrami Operator TQLBO of equation (3-6) in a model with a simple subdivision. In column (c) the Laplacian smoothing was applied to a model consisting of only quads. In column (d) the model was converted to triangles and then the Laplacian smoothing was applied. In column (e) the model was randomly converted from some quads into triangles and then the Laplacian smoothing is applied, showing similar results to those meshes composed only of triangles or quads.

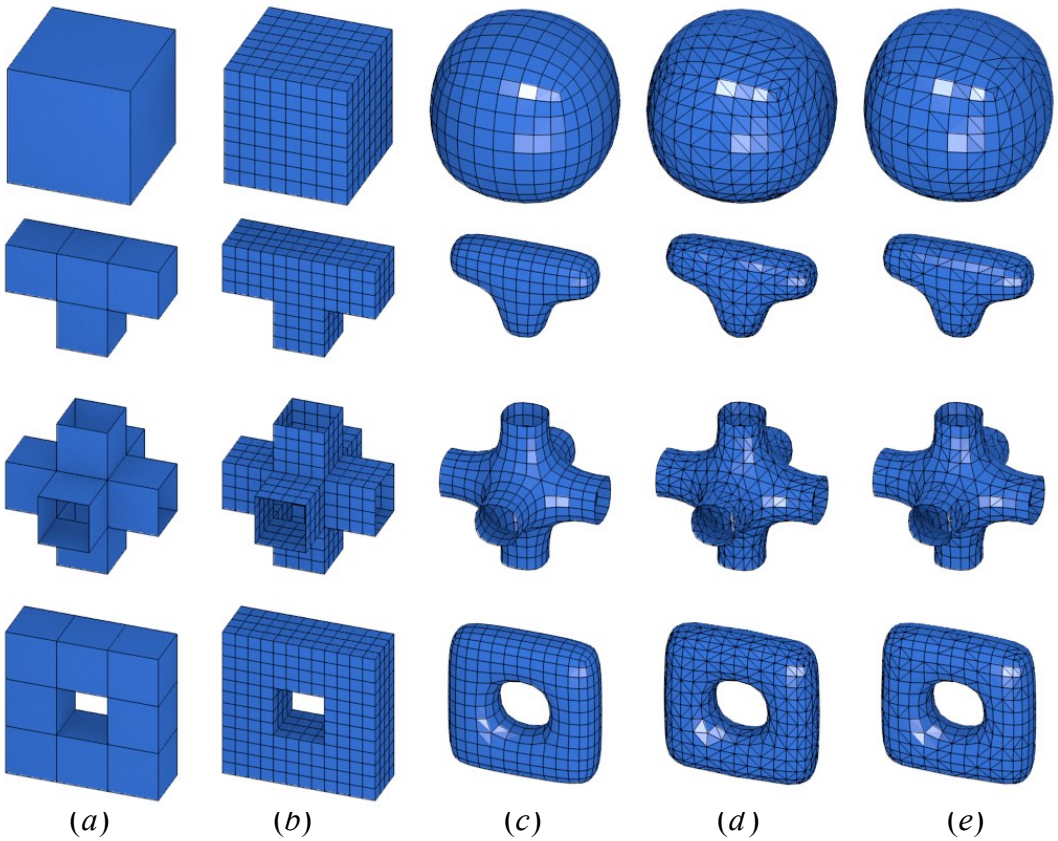


Figure **3-7**: (a) Original Model. (b) Simple subdivision. (c), (d) (e) Laplacian smoothing with $\lambda = 7$ and 2 iterations: (c) for triangles, (d) for quads, (e) for triangles and quads random chosen.

Methods using the Catmull-Clark subdivision surface and the inflation allows to modify the curvature that is obtained with the process of subdivision, as shown in figure **3-5**. This test used a coarse cup model, in which the subdivision was performed, followed by a Laplacian smoothing and inflation. In figure **3-5**.c, **3-5**.d shows also the use of weight vertex groups over coarse models, with subdivision surfaces that allowed to generate the weights for the new interpolated vertices. These new weights were used for the inflation obtained on the 6 cups that are at the right of the figure **3-5**.d.

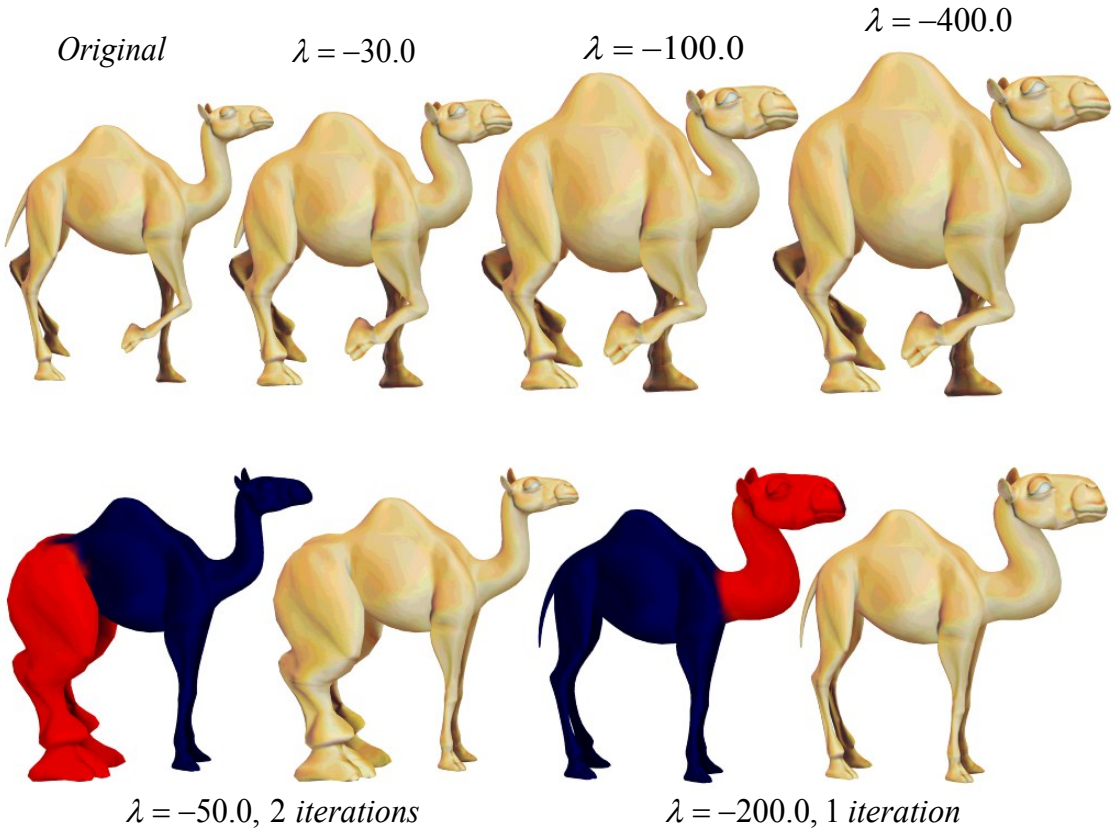


Figure **3-8**: Top row: Original camel model in left. Shape inflation with $\lambda = -30.0$, $\lambda = -100.0$, $\lambda = -400.0$. Bottom row: Shape inflation with weight vertex group, $\lambda = -50.0$ and 2 iterations for the legs, $\lambda = -200.0$ and 1 iteration for the head and neck.

Laplacian smoothing applied with simple subdivision (see figure **3-6**.c.) may produce similar results to those obtained with Catmull-Clark (see figure **3-6**.b.), whose models are in average equal triangles. The one obtained with the Laplacian smoothing is shown in panel (c), (d) and those curvature modified versions are in (e) and (f). As can be observed, different versions of the original sketch can be obtained by parameterizing a single model value, a great advantage of the presented method. Figure **3-8** shows the generation of different versions of a camel according to the λ parameter. In the top row, it is shown the shape inflation results, as λ becomes larger and negative, the resultant shape is observed as if the model would inflate the more convex parts, as shown in figure **3-1**. The larger the λ parameter the larger the model feature inflations. The bottom row of figure **3-8** shows the use of weighted vertex groups, specifying which areas will be inflated. On the left, the inflation of the camel legs produces an organic aspect, notice that the border is not distorted and smooth.

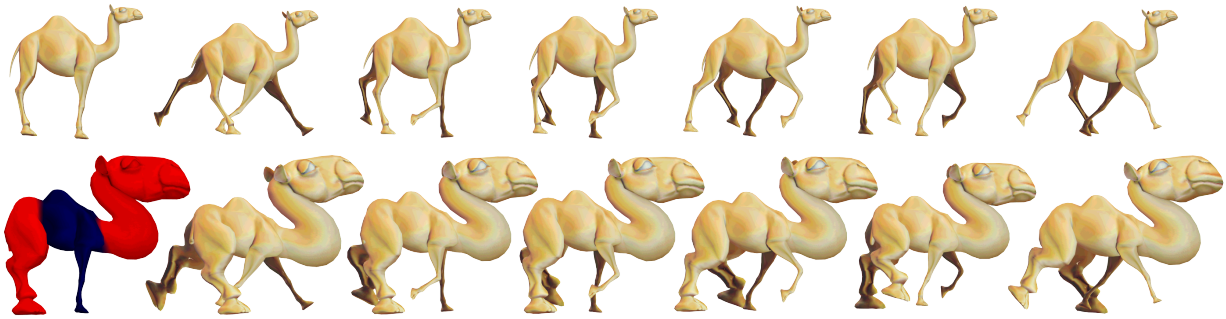


Figure **3-9**: The method is pose insensitive. The inflation for the different poses are similar in terms of shape. Top row: Original walk cycle camel model. Bottom row: Shape inflation with weight vertex group, $\lambda = -400$ and 2 iterations.

The inflation of the silhouette features is predictable and invariant under isometric transformations, as those classically used in some animations (see Figure **3-9**). In this figure, the animation shows some camel poses during a walk, the inflation is performed at the neck and legs, as shown in the bottom left camel in figure **3-9**. Local modifications produced by the pose interpolation or animation rigging practically do not affect the result. In spite of at any pose of the camel legs there is a clear difference, the inflation method allows a flesh-like shape in the original pattern produced by the artist. This is due to the mesh restricted diffusion process so that small local changes are treated without affecting the global solution. The method therefore is rotation invariant since it depends exclusively on the normal mesh field, which is rotation invariant.

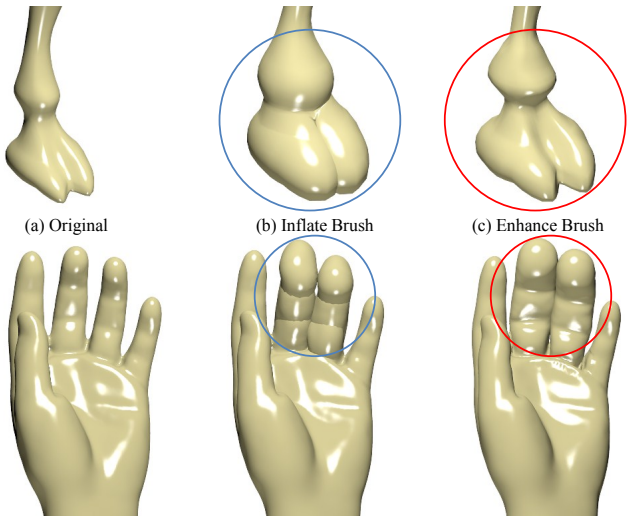


Figure **3-10**: Top row: (a) Leg Camel, (b) Inflate brush for leg into blue circle, (c) Inflate shape brush for leg into red circle. Bottom row: (a) Hand, (b) Inflate brush for fingers into blue circle, (c) Shape inflation brush for fingers in red circle.

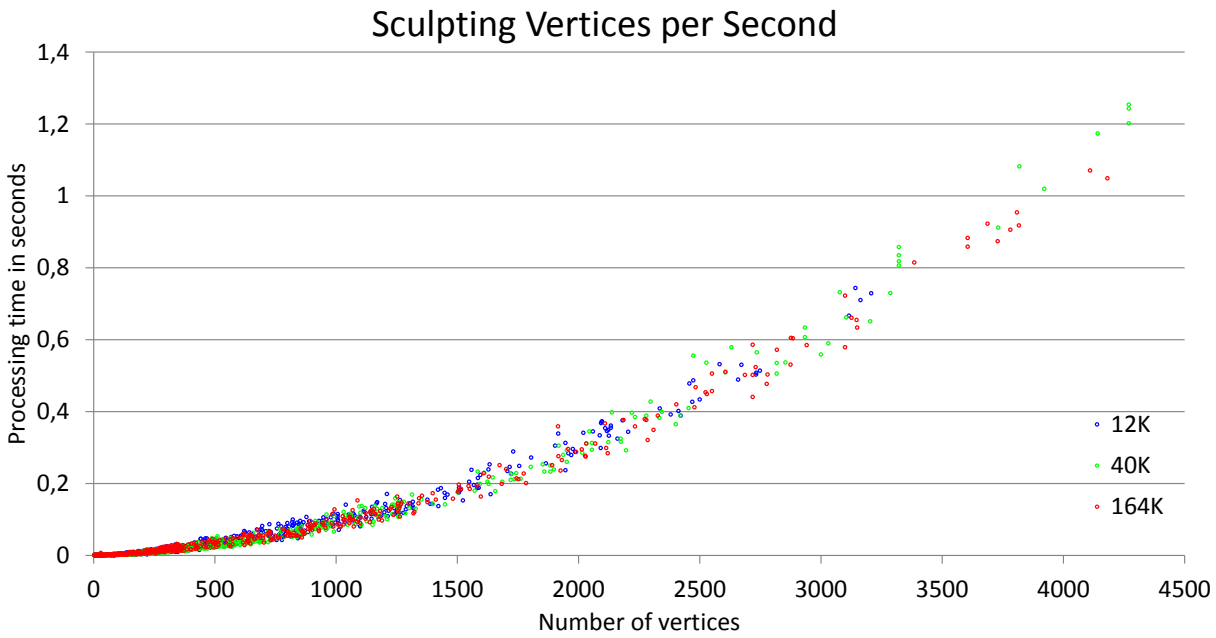


Figure **3-11**: Performance of our dynamic shape inflation brush in terms of the sculpted vertices per second. Three models with 12K, 40K, 164K vertices used for sculpting in real time.

Figure **3-10** shows the use of a shape inflation brush for sculpting in real time. One pass was used with the brush, as shown in the figure, with the blue and red radius. In figure **3-10**.b the camel foot shows the inflation intersection that looks like two bubbles, a similar pattern to what is observed to the fingers on the bottom of the same figure. The silhouette inflation is observed in figure **3-10**.c since the main shape is retained together with its finger and foot details. Similar results can be obtained by a user, however it would take several steps and require the use of several brushes, while the shape inflation took a single step. Likewise, this new method can easily inflate organic features like muscles during the sculpting process. In figure **3-11** the shape inflation brush performance is illustrated, in this experiment three models with 12K, 40K and 164K vertices, were used. These models were sculpted with the shape inflate brush, at each step the brush sphere containing a variable number of vertices for processing. The processing time for 800 vertices in the camel paw (40k model) only took 0.1 seconds, for 2600 vertices in the leg and neck (model 40k) took 0.5 seconds, these times are suitable in real applications since an artist sculpts a model for parts. %, and each part is represented by an average of about 1800 vertices.

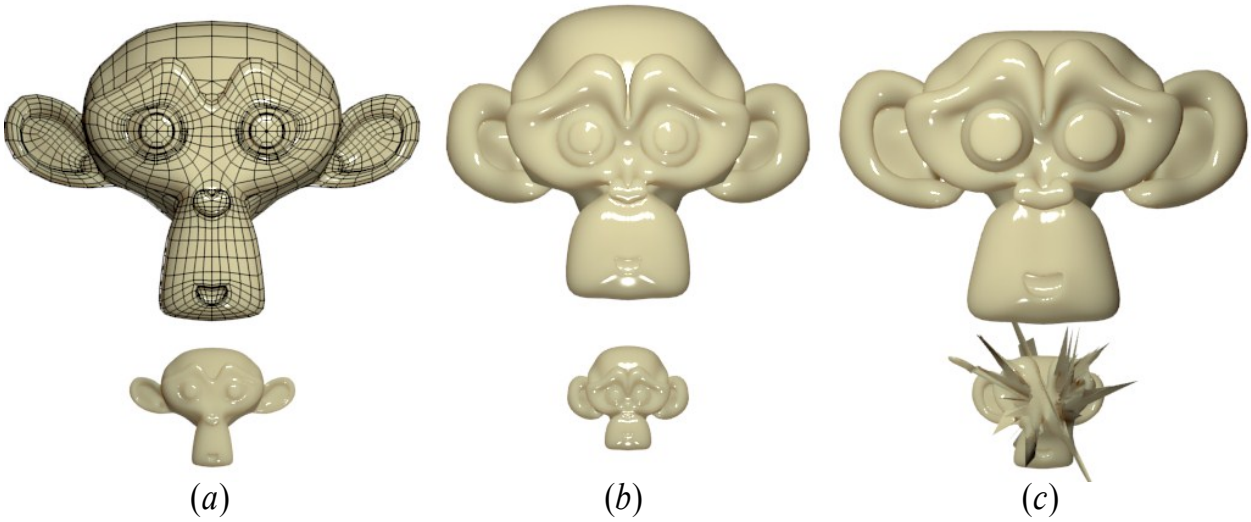


Figure **3-12**: (a) Bottom row: Original Model. Top row: Original model scaled by 4. (b) Top and bottom row: inflating with Normalized-TQLBO $\lambda = -50$ (c) Top and bottom row: inflating with TQLBO $\lambda = -50$.

Tests with the Laplacian operator (equation 3-6) and its normalized version (equation 3-7), produce similar results if the triangles or quads that compose the mesh are about the same size. The normalized version is more stable and predictable because it is not divided by the area of the ring which may be very small and causes numeric problems, as shown in figure **3-12**.c bottom row. The shape inflation of the model with the normalized Laplacian operator results in a more regular pattern. The model can be deformed with a TQLBO normalized version with large λ ($\lambda > 400$) while intersecting itself with no peaks. Figure **3-12**.c shows different results due to the quads areas in the model. Quads with larger area have smaller inflation (figure **3-12**.c skull), and smaller quads have larger inflations (figure **3-12**.c chin).

3.8 Implementation

The method was implemented as a modifier for modeling and brush for sculpting, on the Blender software [7] in C and C++. Working with the Blender allowed to test the method interactively against others, as Catmull-Clark, weight vertex groups and sculpting system in Blender.

To improve the performance, it was worked with the Blender mesh struct, visiting each triangle or quad and storing its corresponding index and the sum of the Laplacian weights of the ring in a list so that only two visits were required for the list of mesh faces and two times the edge list, if the mesh was not closed. This drastically reduced calculations, enabling

real-time processing. In the construction of the Laplacian matrix, several index were locked at vertices having face areas or edge lengths with zero value that could cause spikes and bad results.

Under these conditions, the matrix at equation 3-6 is sparse since the number of neighbors per vertex, corresponding to the number of data per row, is smaller compared to the total number of vertices in the mesh. To solve the linear system equation 3-8 was used OpenNL [9] which is a a library for solving sparse linear system.

3.9 Conclusion and future work

This work presented an extension of the Laplace Beltrami operator for hybrid quad/triangle meshes that can be used in production environments and provides results similar to those obtained by working only with triangles or quads. This paper has introduced a new way to change silhouettes in a mesh for modeling or sculpting in a few steps by means of the curvature model modification while preserving its overall shape. In addition, a new modeling method has also been presented some possible applications have been illustrated. The method works properly with isometric transformations, opening the possibility of introducing it on the process of animation.

We show that this tool may work in early modeling stages, case in which coarse models are used, allowing to modify the shape generated by the Catmull-Clark subdivision surfaces, and thereby avoiding edition of the vertices with change of a single parameters.

Future work includes the analysis of theoretical relationships between the Catmull-Clark subdivision surfaces and the Laplacian smoothing since they can produce very similar results.

Acknowledgment

We wold like to thank anonymous friends for their support of our research.

This work was supported in part by the Blender Foundation, Google Summer of code program at 2012.

Livingstone elephant model is provided courtesy of INRIA and ISTI by the AIM@SHAPE Shape Repository. Hand model is courtesy of the FarField Technology Ltd. Camel model by Valera Ivanov is licensed under a Creative Commons Attribution 3.0 Unported License. Dinosaur and Monkey models are under public domain, courtesy of Blender Foundation.

4 Mesh smoothing based on curvature flow operator in a diffusion equation

This work was accepted and completed for the software Blender [7] that is an open source 3D application for modeling, animation, rendering, compositing, video editing and game creation. In the Google Summer of Code 2012 program which was administered for Google Inc.

4.1 Synopsis

Computer graphics objects reconstructed from real world contain undesirable noise. A Mesh smoothing removes undesirable noise while still preserving desirable geometric and shape of the original model.

This project improving the mesh smoothing tools in blender, based on curvature flow operator in a diffusion equation.

This project allow to work with hybrid meshes composed by triangles and quads based on Laplacian operator proposed by Pinzon and Romero [34].

4.2 Benefits to Blender

This project proposes a new and robust mesh smoothing tool for blender user that require improves the appearance of surfaces models.

New methods to scan computer graphics objects using the Kinect ZCam within Blender, need to remove the noise present at the time of capture.

This mesh smoothing method produce higher quality results without shrinkage. The smoothing tool current collapses the mesh after several iterations.

This mesh smoothing method permit uses a hard and soft constraints on the positions of the points in the mesh to maintain control over the shape.

This mesh smoothing method can help to remove noise generated during the sculpting, without removal the desired details of the model.

4.3 Deliverables

A new and robust mesh smoothing tool for Blender.

Some pages of documentation to be included in the manual.

A technical document for developers to improve the method in the future.

A tutorial explaining the use of the tool.

4.4 Project Details

The project would divide into four parts:

To implement mesh smoothing algorithm based on curvature flow operator in a diffusion equation for blender geometric structures.

1. Initialize data and necessary structures.
2. Compute the Laplacian Matrix.
3. Define the sparse linear system.
4. Solving the sparse linear system, we can use a preconditioned bi-conjugated gradient numerical library.

Integrate or use numerical library present in Blender to solve sparse linear system

Generation of the documentation and tutorials.

4.5 Project Schedule

- 3 weeks: Understanding the Blender source code and identify the key points for the project.
- 1 week: Define the data structures necessary to work with the architecture of blender.
- 1 week: Implement methods for the initial configuration needed for the smoothing algorithm. Implement the method that calculates the Laplacian matrix.
- 2 week: Integrate or use numerical library.

- 2 week: Define the sparse linear system and implement the method to solve sparse linear system.
- 3 weeks: Define and implement graphical user integration.
- 2 weeks: Testing the tool.
- 3 weeks: Generation of the documentation and tutorials

4.6 Mesh Smoothing

A common way to attenueate noise in a polygonal mesh is through a diffusion process [43, 17]. Laplacian smooth techniques over a diffusion process allow a proper noise reduction on the mesh surface with minimal shape changes, while still preserving a desirable geometry as well as the original shape. The simple idea is that the vertices are moved in the direction of the Laplacian when we use the contangent version the vertices are moved in the direction of the curvature flow. The complexity of Laplacian smoothing can be linear in time and space with a fast convergence and the diffussion proccess can attenuate noise with only one iteration due the sparseness of the laplacian operator.

$$\frac{\partial V}{\partial t} = \lambda L\left(V\right) \tag{4-1}$$

Where L is the Laplacian matrix defined in equation 4-3 for meshes composed by triangles or quads with differents size or irregular sampling. λ is a scalar that control the diffussion proccess, and smoothing factor.

The equation 4-1 can be linearly approximated using implicit integration with a Laplacian Operator version of TQLBO, the use of implicit integration permit the system to be more stability.

$$\left(I - \lambda dt L\right) V^{n+1} = V^{n} \tag{4-2}$$

To permit the user to define the region of interest where the laplacian smooth to be applied, we add a diagonal matrix W_p to equation 4-2 where the every element in the diagonal correspond to the weight for every vertex.

$$\left(I - \lambda dt W_p L\right) V^{n+1} = V^{n}$$

For non-close meshes or meshes with holes is not posibble to compute the curvature flow. For this reason the system smooths the edges differently within the diffusion process. The bondaries are treated as a one-dimensional curve. In a curve the Laplacian is defined as the weighted difference between the vertex and the two immediate neighbors, thus ensures

the curve maintained its original form as much as possible. We define a Laplacian for mesh smoothing as a matrix equation.

$$L(i,j) = \begin{cases} -\frac{1}{2A_i} w_{ij} & \text{if } j \in N(v_i) \wedge v_i \notin \text{Boundary} \\ \frac{1}{2A_i} \sum\limits_{j \in N(v_i)} w_{ij} & \text{if } i = j \wedge v_i \notin \text{Boundary} \\ -e_{ij} & \text{if } j \in N(v_i) \wedge \{v_i, v_j\} \in \text{Boundary} \\ \frac{2}{E_i} \sum\limits_{j \in N(v_i)} e_{ij} & \text{if } i = j \wedge \{v_i, v_j\} \in \text{Boundary} \\ 0 & \text{otherwise} \end{cases} \tag{4-3}$$

Where L is a $n \times n$ matrix, n is the number of vertices of a given mesh M, w_{ij} is the TQLBO defined in equation (3-5), $N(v_i)$ is the 1-ring neighborhood with shared face to v_i, $e_{ij} = \frac{1}{\|v_i - v_j\|}$ is the inverse length of the edge between vertices $\{v_i, v_j\}$, $E_i = \sum\limits_{j \in N(v_i)} e_{ij}$. A_i is the ring area around v_i.

4.7 Results and Conclusions

The user interface developed to this tool for the software Blender can be seen in figure **4-1**. This tool allows to set the parameters of λ for interior points and boundaries. Allows to configure soft constraints using weights defined by vertices in "Vertex Group", and allows you to set strong constraints applying the algorithm independently of axis X, Y or Z.

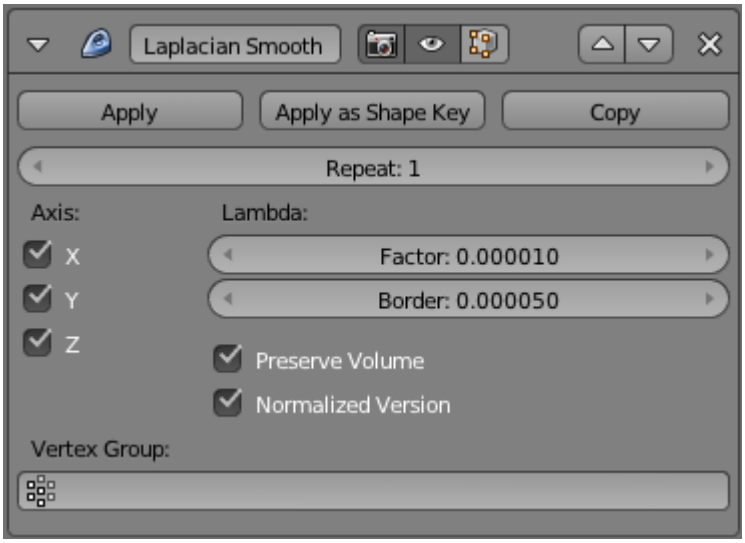


Figure **4-1**: Panel inside blender user interface of the Laplacian Smooth modifier tool.

The tool developed can set the parameter λdt of equation 4-2. Using a small Lambda factor ($\lambda < 1.0$), you can remove noise from the shape without affecting desirable geometry (see figure **4-2**.b). Using a large Lambda factor ($\lambda > 1.0$) you get smoothed versions of the shape at the cost of losing fine geometry details (see figure **4-2**.c and **4-2**.d).

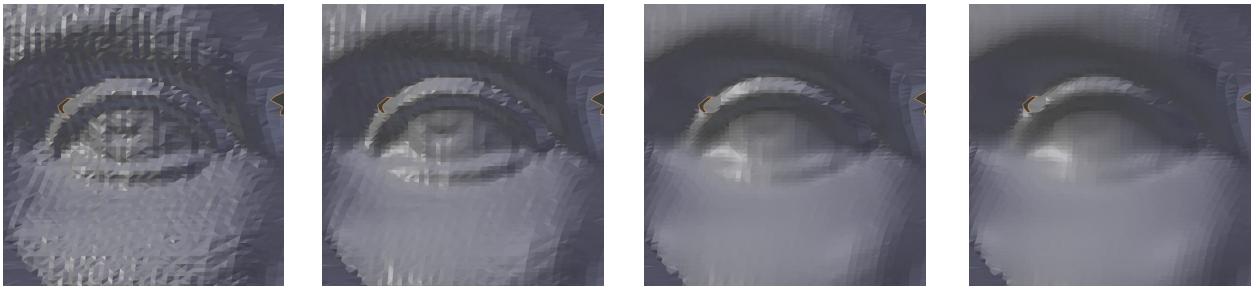


Figure **4-2**: Noise attenuation in face model with Laplacian smoothing tool using only one iteration and changing λ. (a) Original Model. (b) Smoothing $\lambda = 0.5$. (c) Smoothing $\lambda = 2.5$ (d) Smoothing with $\lambda = 5.0$.

The user can smooth the boundaries configuring the parameter "*Border*", seen in figure **4-1**. Boundaries are treated differently. There is no way to calculate the curvature flow on them. For this reason the Lambda factor "*Border*" just smooths them. The change of this parameter and the results seen in figure **4-3**, in the figure you can see how the boundary inside the red circle is smoothing.

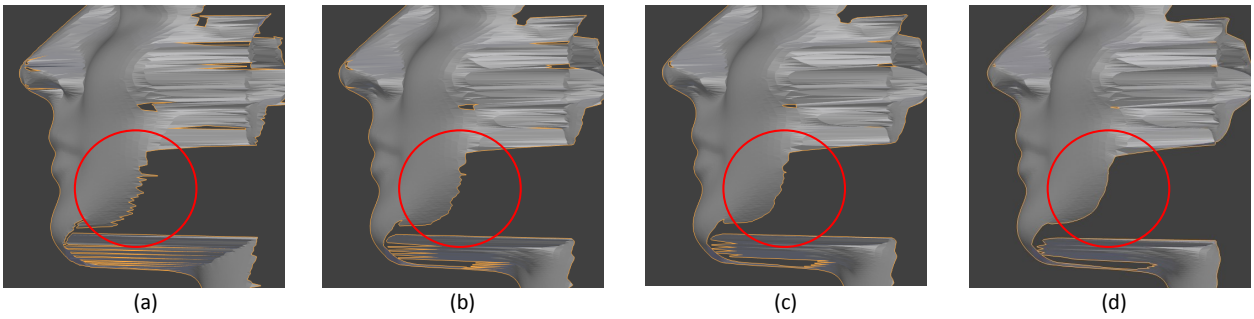


Figure **4-3**: Smoothing boundary changing λ_{Border} factor. (a) Original Model. (b) Smoothing $\lambda_{Border} = 1.0$. (c) Smoothing $\lambda_{Border} = 2.5$ (d) Smoothing with $\lambda_{Border} = 10.0$.

The tool allows the user to add soft constraints using weights for each vertex, this allows to define regions of interest where you want to apply the algorithm in the figure **4-4**.c defined in red region where you want it applied the algorithm the results are shown in figure **4-4**.d where only were smoothed vertices in the red region.

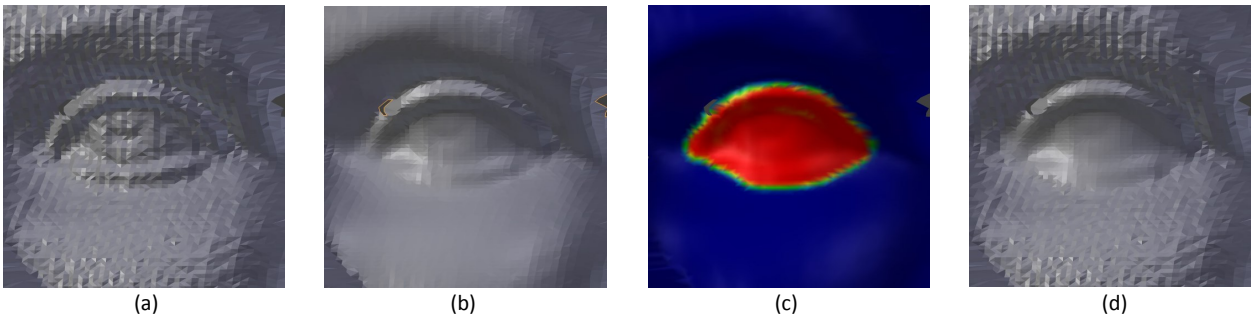


Figure **4-4**: Use of weights per vertex to constrain the effect of mesh smoothing. (a) Original Model. (b) Smoothing with $\lambda = 1.5$ (c) red vertices $weight = 1.0$, blue vertices $weight = 0.0$. (d) Smoothing with $\lambda = 2.5$. Red vertices were only smoothing.

Module was developed as a tool for Blender software to remove noise in the most efficient way as they had been doing.

The methods proposed in the art to remove noise, they could only be applied if the mesh was composed only by triangles, with the developed tool artists can now remove the noise of their models composed of triangles and quadrangles.

5 Mesh Editing with Laplacian Deform

This work was accepted and completed for the software Blender [7] that is an open source 3D application for modeling, animation, rendering, compositing, video editing and game creation. In the Google Summer of Code 2013 program which was administered for Google Inc.

5.1 Synopsis

The mesh editing is generally done with affine transformations, Blender3D offers some tools to transform the vertices as "proportional editing object mode" with which the transformation of some vertices is interpolated to the other vertices connected with the use of simple distance functions.

This project proposes to implement a method for mesh editing based on sketching lines defines by user and preserving the geometric details of the surface.

This method captures the geometric details using a differential coordinates representations. The differential coordinates captures the local geometric information (curvature and direction) of the vertex based on its neighbors. This method allows you to retrieve the best possible original model after changing the positions of some vertices using the differential coordinates of the original model.

5.2 Benefits to Blender

This project proposes a new tool for blender user that requires preserve the geometric details of the surface during a modeling, transformation, definition of the shape keys of the mesh vertices.

The method will allow novice users to edit any polygon mesh preserving the surface details.

This method allows the user to define new shape keys in a most fast and intuitive way.

5.3 Deliverables

- A new mesh editing tool for Blender.
- Some pages of documentation to be included in the manual
- A technical document for developers to improve the method in the future.
- A tutorial explaining the use of the tool.

5.4 Project Details

The project would divide into several parts:

1. Calculate the differential coordinates.
2. Store the fixed vertices (Hard constraints).
3. Store positions of the edited vertices.
4. Store the more representative vertex for retrieve rotation of every differential coordinate.
5. Solve the initial solution – in least-squares sense.
6. Rotate the differential coordinates with base on initial solution and more representative vertex.
7. Reconstruct the surface – in least-squares sense.
8. Generation of the documentation and tutorials.

5.5 Project Schedule

- 2 Weeks: Calculate the differential coordinates.
- 2 Weeks: Store the fixed vertexes (Hard constraints).
- 2 Weeks: Store positions of the edited vertexes.
- 2 Weeks: Compute initial solution.
- 2 Weeks: Rotate differential coordinates.
- 2 Weeks: Reconstruct the surface – in least-squares sense.
- 1 Weeks: Testing the tool and Define and implement graphical user integration.
- 2 Weeks: Generation of the documentation and tutorials.

5.6 Laplacian Deform

The Laplacian deform allows to pose a mesh while preserving geometric details of the surface. This method allows to defines a set of anchor vertices, and then moves some of them around. The system keeps the rest of the anchor vertices in fixed positions, and calculates the best possible locations of all the remaining vertices to preserve the original geometric details.

This work adapt the method proposed by Sorkine [38] for mesh deformations, delete the use of static vertices and allow application of this system in hybrid meshes composes by triangles and quads with the TQLBO proposed.

This method captures the geometric details using a differential coordinates representations. The differential coordinates captures the local geometric information (curvature and direction) of the vertex based on its neighbors how show in figure **5-1**.

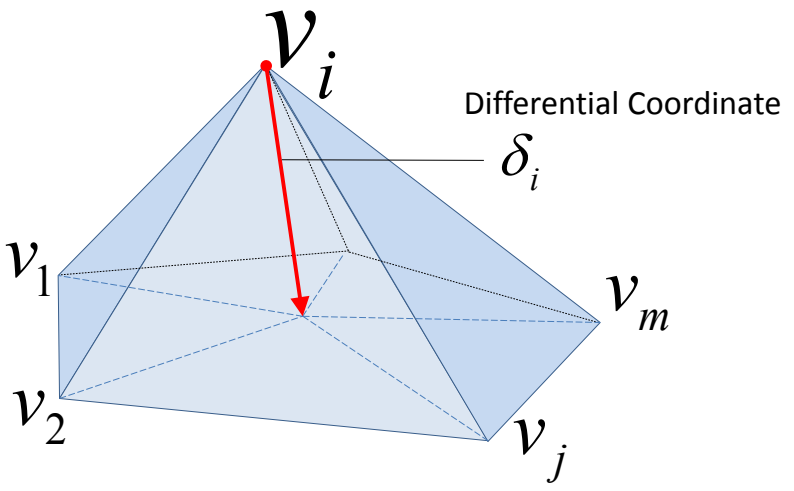


Figure **5-1**: Difference between v_i and the center of mass of its neighboors $v_1, ..., v$.

$$\delta_i = \sum_{j=1}^{m} w_{ij} \left(v_i - v_j \right) \tag{5-1}$$

Where δ_i is the differential coordinate for vertex v_i. The v_j are the inmediate neighoors of v_i, and w_{ij} is the weigth betwen vertex v_i and v_j defined in equation 3-5 that is TQLBO.

Then the linear systems for find the new pose of a mesh is.

$$\begin{bmatrix} w_l L \\ W_c \end{bmatrix} X = \begin{bmatrix} \delta \\ W_c C \end{bmatrix} \tag{5-2}$$

Where w_l is the weigth for Laplacian Matrix L, the Laplacian matrix L was defined in equation 3-6 . Wc is a matrix that containts ones int the indexes of anchors vertices. C is a

vector with coordinates of anchors vertices after several transformatios. δ are the differential coordinates defined in equation 5-1.

5.7 Performance Testing of Solvers

Fot this project we choose a numerical solver to be included in Blender software with base in a performance evaluation of computation of the initial factorization of the Laplacian deform system.

Linear equation system to solve

$$\begin{bmatrix} w_l L \\ W_c \end{bmatrix} X = \begin{bmatrix} \delta \\ W_c C \end{bmatrix}$$

Solving the sparse linear system

$$Ax = b$$

Where:

$$A = \begin{bmatrix} w_l L \\ W_c \end{bmatrix}$$

$$x = V$$

$$b = \begin{bmatrix} \delta \\ W_c C \end{bmatrix}$$

5.7.1 Hardware Specification

- Processor: AMD Quad-Core 2.40 GHz
- RAM: 8.0 GB
- OS: Windows 7 Professional
- Graphics controller: NVIDIA Quadro FX 570

5.7.2 Software Specification

CGAL Computational Geometry Algorithms Library

Graphite Research platform for computer graphics

5.7.3 Numeric Solvers Used

CG: Conjugate gradient method.

BICGSTAB: Biconjugate gradient stabilized method.

GMRES: Generalized minimal residual method.

SUPERLU: Sparse Direct Solver, LU decomposition with partial pivoting.

TAUCS_LDLT: A library of sparse linear solvers with LDLT factorization.

CHOLMOD: Supernodal sparse Cholesky factorization.

LU factorization is a numerical method that works with large, sparse, nonsymmetric systems of linear equations [24]. We choose the implementacion of LU factorizacion in a OpenNL-SuperLU library, because this method show the better performance for the computacion of a solution for a Laplacian Deform linear system of equations presented in equation 5-2 how see in the table and plot **5-1**, **5-2**. OpenNL SuperLU allow works with the Graphics Unit Proccesor GPU, for exploit the capacity of GPU to work with parallel structures, more fast that traditional CPU.

Model	Vertices	CG	BICGSTAB	GMRES	SUPERLU	TAUCS	CHOLMOD
Cross	24	0.05	0.05	0.04	0.04	0.05	0.05
King	538	0.83	0.63	0.71	0.61	0.68	0.79
YModel	4770	19.60	16.44	16.93	16.06	16.88	17.95
Man	10002	33.43	27.76	29.91	28.54	29.53	30.80
Neptune	28052	133.97	136.46	136.39	133.21	142.87	142.76
Armadillo	34594	194.48	174.88	175.80	169.92	181.70	183.49

Table **5-1**: Vertices Vs Seconds, Laplacian Deform initial factorization performance.

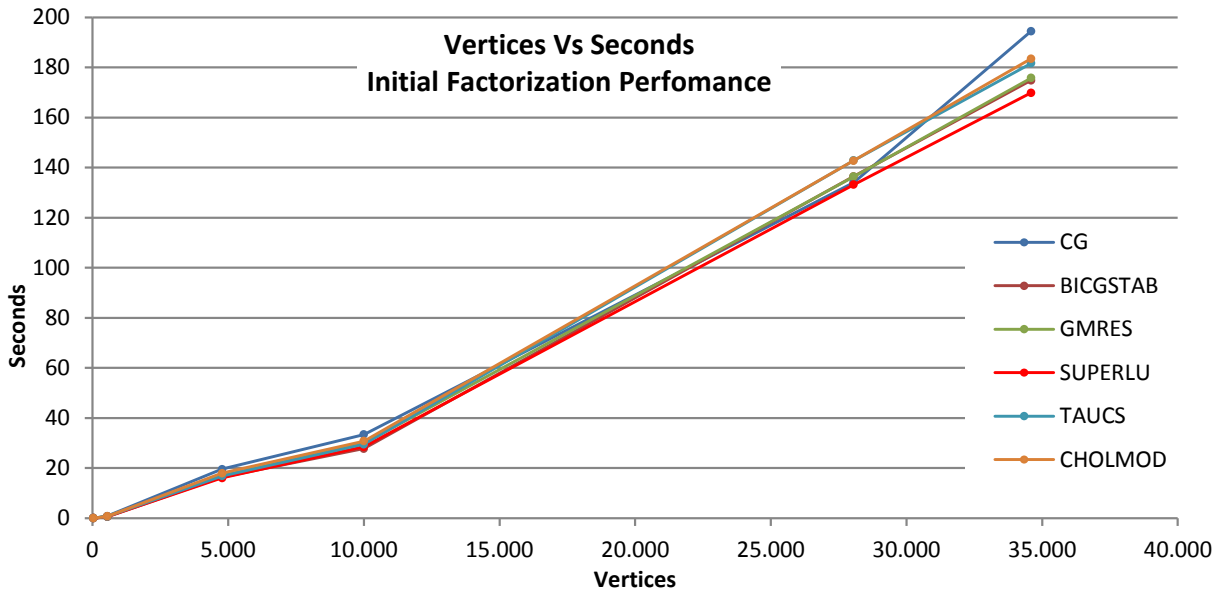


Figure **5-2**: Plot of Vertices Vs Seconds, Initial factorization performance.

5.8 Results

The user interface for software Blender can be seen in figure (**5-3**). In this tool the user define the anchor vertices with the use of vertex groups a feature offered by Blender, the user configure this name in the field *Anchors Vertex Group*. The *Bind* option init the system and capture the geometry details in form of differential coordinates and compute the factorization of the linear system, after that the system is ready to repose meshes in real-time user interaction session.

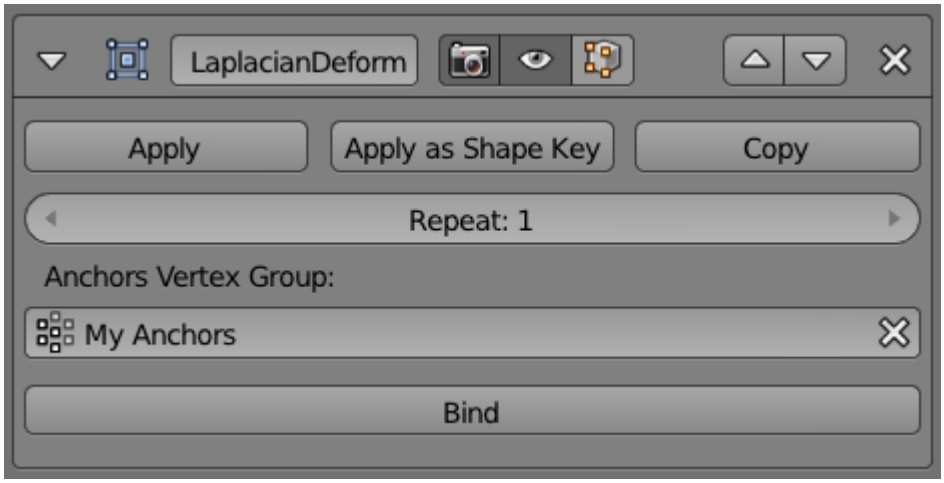


Figure **5-3**: Panel inside blender user interface of the Laplacian Deform modifier tool.

Figure (**5-4**) shows the Laplacian Deform applied in a model with 173K vertices, only anchor vertices was used and are represented in blue color, when the user apply several transformation (location, rotation, scale) over this anchors vertices, the system find a solution and estimate the position of the vetices in yellow color. This method work in real time, to achieve this the matrix $\begin{bmatrix} w_l L \\ W_c \end{bmatrix}$ in the equation (5-2) is decomposed in matrix LU with LU factorization only one time when the system init. Once the matrix is factorized the system can solve the unknowns in a fast way in term of miliseconds. For get better results the method permits solve several times the system of equations, and not need factorize matriz LU could be that at every iteration, only the differential coordinates are adjust since the differential coordinates can be rotated at every iteration and get bettter results. In the figure only four iterations were used, but the system find good solutions with only one iteration when the angle of rotations are less that π.

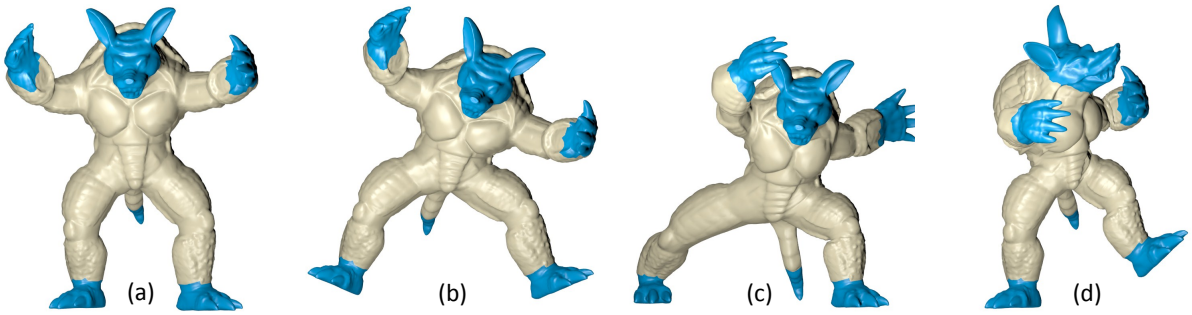


Figure **5-4**: Anchor vertices in blue color. (a) Original Model, (b,c,d) new poses change only anchor vertices, system find positions for vertices in yellow color.

In figure **5-5** show a comparison after made a one simple transformation that consist in rotate 70^{o} to right the parts in blue color. between traditional method to interpolate the changes made in some parts of the mesh to entire model. The results for simple interpolation is show in **5-5**.b and show how the main trunk loss your shape and is rotated. The propagation of changes made with Laplacian deform (figure **5-5**.c) for the same transformation show better results and how the details and shape of the original model are preserve as best as possible.

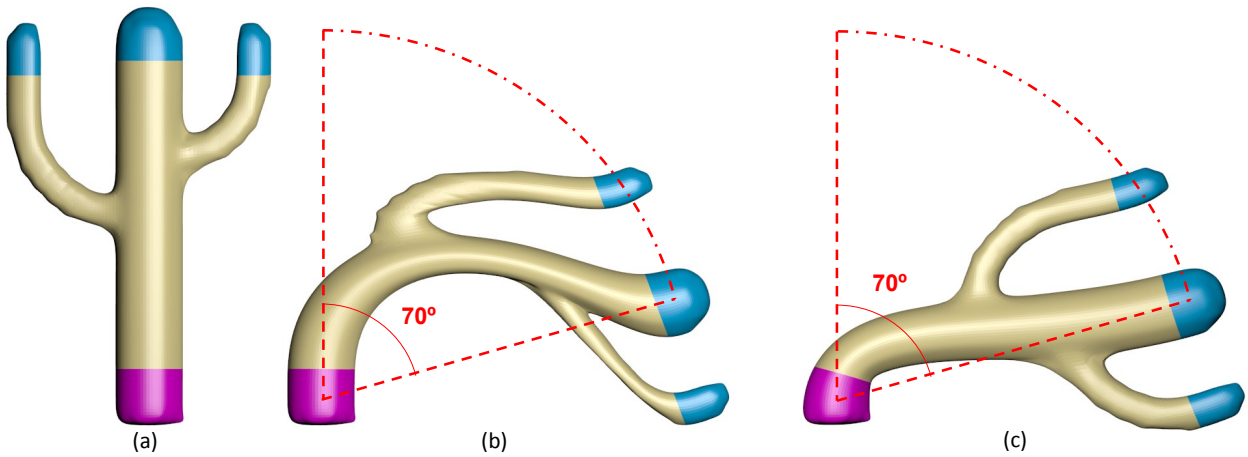


Figure **5-5**: (a) Original cactus model. (b) Rotate 70º to right the blue segments with basic interpolation (c) Rotate 70º to right the blue segments with Laplacian deform tool.

The Laplacian Deform tool allow a user to repose a model while preserve the geometry details in the figure **5-6**.b y **5-6**c you can see the new pose of he horse after five transformations and rotation of the head, in figure **5-6**.b you was use basic interpolation and the shape and details are lost with every change made. In figure **5-6**..c yo can see how the new pose of the horse look more natural, and how the body and neck, of the horse preserve the original shape, this comparison show that method allow to apply any number of transformations without lost details, could be the system try to recover geometry details with base in the original pose of the model.

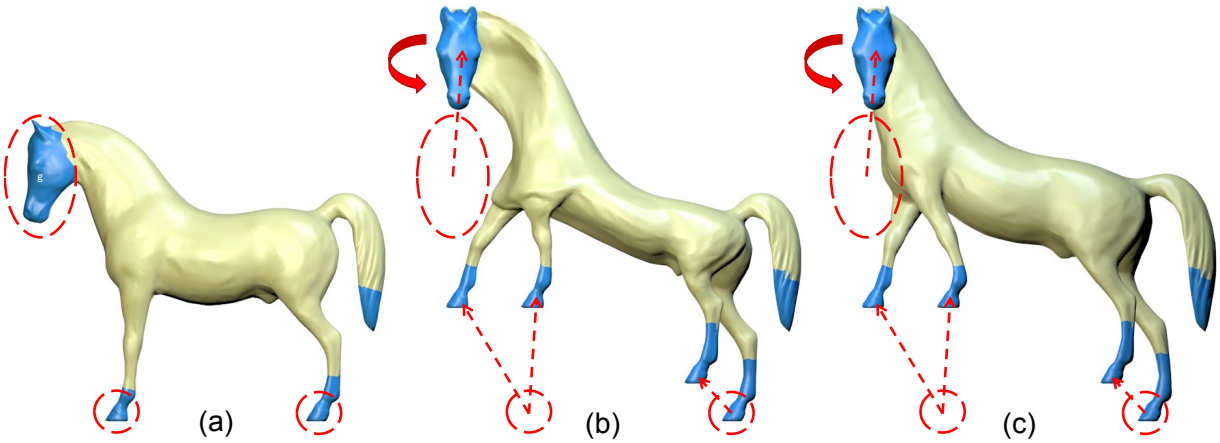


Figure **5-6**: (a) Original Horse model. (b) Translate and rotate the blue segments with basic interpolation (c) Translate and rotate the blue segments with Laplacian deform tool.

6 Skeleton Extraction

Part of this work was accepted and presented as a poster with name *Software para la Extracción del Esqueleto por Contracción y Suavizado* at 7th International Seminar on Medical Image Processing and Analysis SIPAIM 2011. The conference was held on December 6-8 of 2011, in the city of Bucaramanga, Colombia [33], the thumbnail of the poster is show in figure **6-1**.

Part of this work was accepted and presented as a poster with name *Análisis Experimental de la Extracción del Esqueleto por Contracción con Suavizado Laplaciano* at 6th International Seminar on Medical Image Processing and Analysis SIPAIM 2010. The conference was held on December 1-4 of 2011, in the city of Bogotá, Colombia [32], the thumbnail of the poster is show in figure **6-2**.

Skelton extraction reduces the dimensionality and represents a three-dimensional object as a uni-dimensional structure [14].

The skeleton extraction use the natural shrink produced by the laplacian smoothing [27], to contratc iteratively the mesh until the volume inside the surface is near to zero (see figure **6-3**). The method stretch the shrink mesh to warranty that object preservers the best at posibble the original topology with the use of attraction constraint [4].

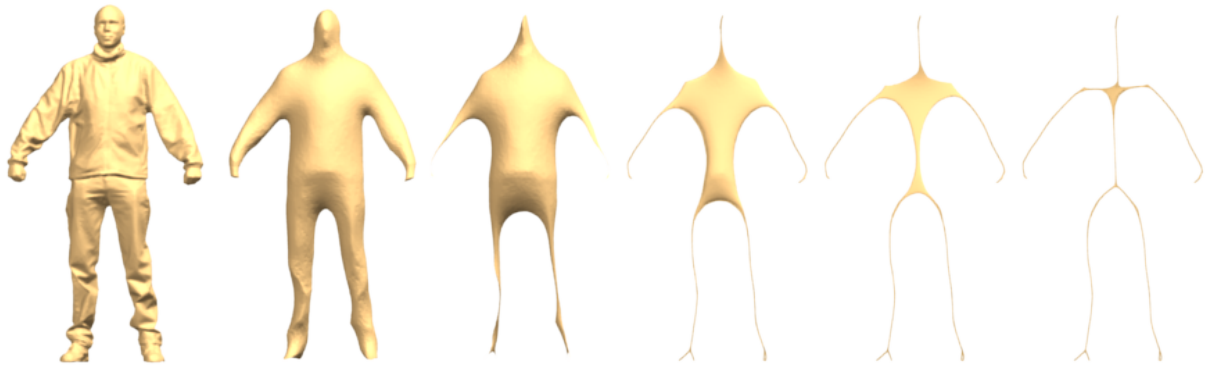


Figure **6-3**: From left to right iterative mesh contraction.

The laplacian smooth proccess move the vertices in the direction of the Laplacian, if cotangent laplacian operator is used, the vertices move in the direction of minimal surface [31]

Software para la Extracción del Esqueleto por Contracción y Suavizado

Alexander Pinzón, Eduardo Romero

Abstract

Este articulo presenta un software para el procesamiento, visualización, y extracción del esqueleto desde mallas de polígonos. El software se diseño con base en un sistema de plugins y filtros, se implemento un plugin que contenía un filtro para la extracción del esqueleto por contracción en dirección gradiente con suavizado Laplaciano. El software producido proporciona una plataforma flexible para el diseño e implementación de plugins.

Métodos de Suavizado de Mallas

Los métodos para suavizar mallas reducen el ruido, o permiten iterativamente eliminar frecuencias altas presentes en el muestreo tridimensional de los modelos.

Métodos Laplacianos

La idea básica consiste en mover un vértice en la misma dirección del Laplaciano .

$$Eq(1) \quad \frac{\partial X}{\partial t} = \lambda L(X)$$

La ecuación 1 se implementa como la ecuación de diferencias hacia adelante así:

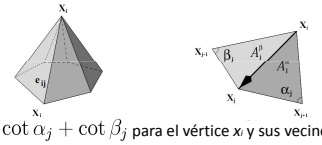
$$Eq(2) \quad X_{t+1} = (I + \lambda L) X_t$$

Donde ***X*** es el conjunto de vértices, *L* es el Laplaciano, y λ ∈ $\Re$ es la velocidad de difusión.

Y la aproximación discreta de la ecuación 2 es:

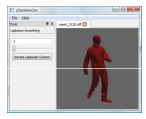
$$Eq(3) \quad L(x_i) = \sum w_{ij}(x_j - x_i), \quad x_j \in Vecinos(x_i)$$

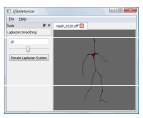
Aproximación del Laplaciano mediante la Curvatura normal



Con $w_{ij} = \cot \alpha_j + \cot \beta_j$ para el vértice x_i y sus vecinos x_j

Software Skeletonizer





sjSkeletonizer es el software desarrollado en el grupo Bioingenium para el procesamiento, visualización y extracción del esqueleto desde mallas de polígonos.

- Usa **CGAL** (Computational Geometry Algorithms Library)
- Usa **Graphite** (Software de Geometría Numérica y Computación Grafica)
- Se integraron las siguientes librerías de procesamiento numérico: ACE, AMD, ARPACK, ARPACK_UTIL, CBLAS, CCOLAMD, CHOLMOD, CLAPACK, COLAMD, F2CLIBS, METIS, MISC, NL, SUPERLU, TAUCS

Implementación para la extracción del esqueleto

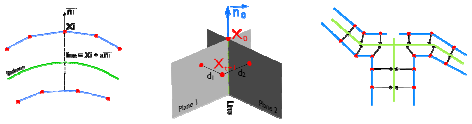
La Esqueletonización reduce la dimensionalidad y representa un cuerpo como un estructura uní-dimensional.

El esqueleto puede ser obtenido suavizando la malla pero bajo dos restricciones, W_L que da peso al Laplaciano y W_H que mantiene los vértices en su localización original.

Extracción del esqueleto. $$\begin{bmatrix} W_L L \\ W_H \end{bmatrix} X_{t+1} = \begin{bmatrix} 0 \\ W_H X_t \end{bmatrix}$$

Donde *L(X)* = Suavizado Laplaciano con w_{ij} basado en la curvatura de flujo $w_{ij} = \cot \alpha_j + \cot \beta_j$

Y la nueva restricción propuesta en este trabajo



Tratar de suavizar los vértices a lo largo de la línea
La distancia del punto a la linea

$$line = \overrightarrow{P_1} + t\overrightarrow{P_2},\ point\ P_0 \Longrightarrow distance(line, P_0) = \frac{|(P_2 - P_1) \times (P_1 - P_0)|}{|P_2 - P_1|}$$

Cada punto en un plano satisface esta ecuación

$$P_0 : a\mathbf{x} + b_0\mathbf{y} + c_0\mathbf{z} + d_0 = 0.$$

Resultados

- Los vértices se pueden mover a lo largo de la línea.
- El esqueleto tiene muchas ramas.
- Muchas mas ecuaciones que incógnitas.
- La solución debe ser restringida a una región particular de la línea.

Referencias y Agradecimientos

• Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. **Skeleton extraction by mesh contraction**. *ACM Transactions on Graphics*, 27(3):10, 2008. Skeleton Extraction.
•Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. **Articulated mesh animation from multi-view silhouettes**. *ACM Trans. Graph.*,27(3):1–9, 2008. 3D Reconstruction.
•Nicu D. Cornea and Patrick Min. **Curve-skeleton properties, applications, and algorithms**. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548, 2007. Skeleton Extraction Survey Member-Silver, Deborah.

Agradecimientos: The captured performance data were provided courtesy of the Computer Graphics Group of the MIT CSAIL Vision Research (Cambirdge, USA).

Contacto

Alexander Pinzón Fernández apinzonf@gmail.com
Grupo de Investigación Bioingenium www.bioingenium.unal.edu.co
Universidad Nacional de Colombia www.unal.edu.co
Facultad de Medicina, Edificio 471 Primer Piso

Figure **6-1**: Poster *Software para la Extracción del Esqueleto por Contracción y Suavizado* presented at 7th International Seminar on Medical Image Processing and Analysis SIPAIM 2011.

Análisis Experimental de la Extracción del Esqueleto por Contracción con Suavizado Laplaciano

Alexander Pinzón, Fabio Martínez, Eduardo Romero

Abstract

Este articulo presenta un análisis sistemático del método de extracción del esqueleto por medio de la contracción de un volumen con suavizado Laplaciano. El trabajo realiza una aproximación experimental al problema de la extracción del esqueleto, para encontrar el rendimiento del método evaluado frente a cambios isométricos, y durante la fase de simplificación.

Esta evaluación utilizo el modelo tridimensional animado de una persona que realizaba una caminata, a este modelo se le extrajo el esqueleto y se compararon las diferencias en distintos instantes de la animación, y distintas configuraciones del proceso de simplificación. Los resultados muestran un óptimo rendimiento del método frente a las transformaciones isométricas, y múltiples problemas en la fase de simplificación de mallas.

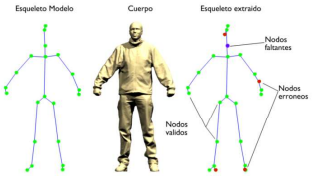


Fig. 2. Proceso de extracción del esqueleto

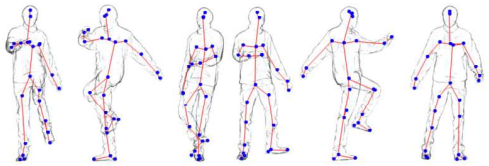


Fig. 3. Variación del numero de nodos

Contracción con suavizado Laplaciano

Este método contrae iterativamente una malla de polígonos por medio del suavizado laplaciano hasta tener un volumen de cero ver figura 1.
La contracción es tomada como un problema de minimización de energía, con los siguientes términos.

$$\|W_L L V'\|^2 + \sum_i W_{H,i}^2 \|V'_i - V_i\|^2$$

• ***L***: Operador Lapaciano para remover las frecuencias altas, es decir suavizar los detalles de la geometría
• W_H: Fuerza de atracción que usa los vértices, para mantener información clave de la geometría durante la contracción.
• W_L: Fuerza de contracción que hace que la forma tridimensional pierda volumen.

Fig. 1. Proceso de extracción del esqueleto

Experimentación

Se uso la implementación hecha en Oscar Kin-Chung Au et al. Por los autores, para realizar la extracción del esqueleto de un modelo tridimensional que fue obtenido mediante el método de recuperación de forma desde las siluetas realizado por Vlasic et al. De este modelo se registro una caminata durante 240 cuadros.
Para evaluar el proceso de simplificación se variaron el numero de nodos usados para describir uniones en el esqueleto (ver figura 3) y se clasificaron las uniones así ver figura 2.

En el segundo experimento se seleccionaron diferentes poses para observar la correspondencia topológica entre los esqueletos extraídos y analizar el comportamiento del método frente a transformaciones isométricas de la geometría de un cuerpo.

Resultados

El método recupera de forma óptima el esqueleto de un cuerpo bajo transformaciones isométricas, ver la tabla 1, se recuperaron en promedio 19.71 nodos de los 21 necesarios para reconstruir el esqueleto en diferentes poses.
El método no pudo recuperar 1.86 nodos de los necesarios para reconstruir totalmente el esqueleto. La línea verde en la figura 3 describe el numero de nodos que hacen falta para recuperar el modelo.

	Validos	Erróneos	Faltantes
Media	19.71	4.83	1.86
SD	1.47	1.47	1.47
Min	17	3	0
Max	21	7	4

Tabla 1. Resultados obtenidos al realizar transformaciones isométricas

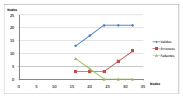


Fig. 3. Gráfico de la variación de nodos validos, erróneos y faltantes durante la fase de simplificación

Durante la fase de simplificación (ver figura 3) el método no sufre perdida ni es sensible al uso de mas de 24 nodos (línea azul). El método tiene un numero mínimo de 3 nodos recuperados erróneamente como se observa el la línea roja de la figura 3.

Referencias y Agradecimientos

• Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. **Skeleton extraction by mesh contraction**. *ACM Transactions on Graphics*, 27(3):10, 2008. Skeleton Extraction.
•Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. **Articulated mesh animation from multi-view silhouettes**. *ACM Trans. Graph.*,27(3):1–9, 2008. 3D Reconstruction.
•Nicu D. Cornea and Patrick Min. **Curve-skeleton properties, applications, and algorithms**. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548, 2007. Skeleton Extraction Survey Member-Silver, Deborah.

Agradecimientos: The captured performance data were provided courtesy of the Computer Graphics Group of the MIT CSAIL Vision Research (Cambirdge, USA).

Conclusiones y Trabajo Futuro

El método de extracción mostró ser robusto y tener baja sensibilidad frente a cambios isométricos de la geometría, el método puede trabajar de forma automática a o largo de todos los cuadros. El método recupera de forma eficiente el esqueleto sin hacer uso de un modelo, eliminando la necesidad de estimar la pose. El método permite realizar de forma sencilla y automática el seguimiento del esqueleto a lo largo del vídeo.

Como trabajo futuro es posible mejorar la recuperación de información haciendo uso de la coherencia espacio temporal no presente en la técnica de extracción, para superar la perdida de información entre cuadros de vídeo. Es posible automatizar el proceso de simplificación para encontrar el numero óptimo de nodos con lo cual puede ser representado el esqueleto, haciendo uso de algoritmos de partición de mallas.

Contacto
Alexander Pinzón Fernández apinzonf@gmail.com
Grupo de Investigación Bioingenium www.bioingenium.unal.edu.co
Universidad Nacional de Colombia www.unal.edu.co
Facultad de Medicina, Edificio 471 Primer Piso

Figure **6-2**: Poster *Análisis Experimental de la Extracción del Esqueleto por Contracción con Suavizado Laplaciano* presented at 6th International Seminar on Medical Image Processing and Analysis SIPAIM 2010.

following the curvature flow of the mesh surface.

In the work of Au [4] is proposed the next system of equation to iteratively contratc the mesh until a skeleton.

$$\begin{bmatrix} W_L L \\ W_H \end{bmatrix} X_{t+1} = \begin{bmatrix} 0 \\ W_H X_t \end{bmatrix} \tag{6-1}$$

where L is the Laplacian matrix describe in equation 3-6, W_L is the factor for smoothing, W_H is the factor for attraction constraint.

W_L and W_H are changing in every iteration. The contraction and smoothing constraint $W_L^{t+1} = S_L W_L^t$ with $S_L = 2.0$. The attraction constraint $W_{H,i}^{t+1} = W_{H,i}^0 \sqrt{\frac{A_i^0}{A_i^t}}$.

A_i^t y A_i^0 are the current area and initial aerea of the ring surrounding x_i.

In this thesis an additional constraint is proposed which seeks the vertices and the skeleton does not shrink, to eliminate the need to adjust the final skeleton taking each skeleton node and move to the center of your local mesh region.

The basic idea is move the vertices along of the normal line estimated on every vertex based on the avergage of the faces normals.

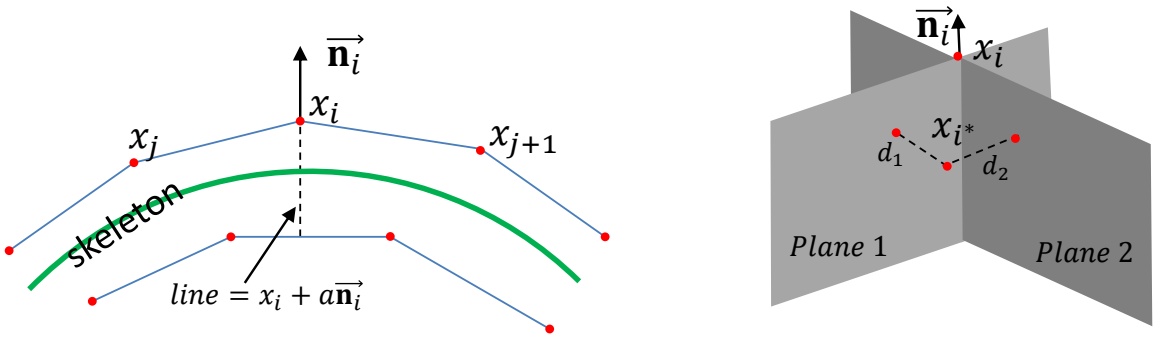


Figure **6-4**: Left: The vertex x_i move along line constraint. Right: the distance of vertex x_i to plane 1 and plane 2 when change the position in every iteration.

This line can be see as intersection of two planes (see figure **6-4**) created with the information of the vertex x_i, the normal $\overrightarrow{\mathbf{n}_i}$ and neighboors x_j of x_i.

The equation of the plane.

$$a\mathbf{x} + b_0\mathbf{y} + c_0\mathbf{z} + d_0 = 0$$

There is varios methods to build a equation of the plane based on three non-collinear points, we choose this three points based on vertex x_i, the normal $\overrightarrow{\mathbf{n}_i}$ and neighboors x_j of x_i.

$P_1 = x_i$, $P_2 = x_j$ when $x_j \in Neighboors\,(x_i)$ and $P_3 = P_1 - \overrightarrow{\mathbf{n}_i}$.

To build a equation plane only need three non-collinear points $\{P_1, P_2, P_3\}$ and solve the next systems of equations for variables a, b, c.

$$\begin{aligned} ax_1 + by_1 + cz_1 + d = 0 \\ ax_2 + by_2 + cz_2 + d = 0 \\ ax_3 + by_3 + cz_3 + d = 0 \end{aligned}$$

The distance of point $P_0 = \{x_0, y_0, z_0\}$ to a plane $\Pi = a\mathbf{x} + b_0\mathbf{y} + c_0\mathbf{z} + d_0$.

$$|\Pi - P_0| = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}}$$

In the equation above, every point $P_0 = \{x_0, y_0, z_0\}$ that belong to plane made the equation zero. If the point P_0 is not in the plane the value of the equation of the plane change, and the absolute value is increased if P_0 is more distant of the plane. Let us use this simple relation to build a valid constraint that can be put in the form of linear equation $Ax = B$. Where A is the matrix with values $\begin{vmatrix} a & b & c \end{vmatrix}$ the x value correspond to every vertex in mesh $\{x_0, y_0, z_0\}$ and B correspond to d value of the plane equation. How the linear system presented in equation 6-1 don't has a unique solution, the system solve it in the least-squares sense. With the new constraint the system then try to minimize the distance betwen the new vertex positions and the two planes.

The new system of equations proposed

$$\begin{bmatrix} W_L L \\ W_H \\ W_D \Pi_1 \\ W_D \Pi_2 \end{bmatrix} X_{t+1} = \begin{bmatrix} 0 \\ W_H X_t \\ -D_1 \\ -D_2 \end{bmatrix} \tag{6-2}$$

where W_D is the weigth to new constraints, that forced the vertices to move along of the normal line of every vertex. Π_1 and Π_2 are matrix that contains a, b, c values of the equation of the plane for every vertex. D_1 and D_2 are the vector with d values of the equations of the planes for every vertex.

6.1 Results

As a result of this work was developing the software Skeletonizer that permits the proccessing, visualization and extraction of the skeleton from polygonal hybrid meshes composed by triangles and quads. The software was designed with base in a plugin system and pipe-filters

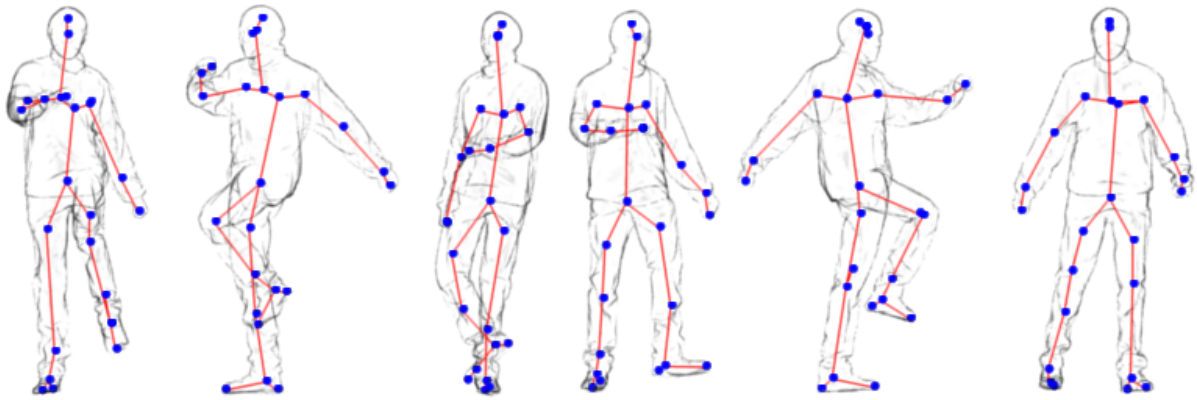


Figure **6-5**: Model with different poses and skeleton obtained with our skeleton extraction software.

paradigm. This software implement the method for skeleton extraction describe in equation 6-2.

The software is called sjSkeletonizer and use the following computacional frameworks and libraries.

CGAL Computational Geometry Algorithms Library [11].

Graphite is a research platform for computer graphics, 3D modeling and numerical geometry [23].

QT is a cross-platform application and UI framework [1].

7 Conclusion and future work

This work presented an extension of the Laplace Beltrami operator for hybrid quad/triangle meshes that can be used in production environments and provides results similar to those obtained by working only with triangles or quads. This paper has introduced a new way to change silhouettes in a mesh for modeling or sculpting in a few steps by means of the curvature model modification while preserving its overall shape. In addition, a new modeling method has also been presented some possible applications have been illustrated. The method works properly with isometric transformations, opening the possibility of introducing it on the process of animation.

We show that this tool may work in early modeling stages, case in which coarse models are used, allowing to modify the shape generated by the Catmull-Clark subdivision surfaces, and thereby avoiding edition of the vertices with change of a single parameters.

Future work includes the analysis of theoretical relationships between the Catmull-Clark subdivision surfaces and the Laplacian smoothing since they can produce very similar results.

Bibliography

[1] Qt. Cross-platform application and UI framework. http://qt-project.org/.

[2] Marc Alexa and Max Wardetzky. Discrete laplacians on general polygonal meshes. *ACM Trans. Graph.*, 30(4):102:1–102:10, July 2011.

[3] D. Attali, J.-D. Boissonnat, and H. Edelsbrunner. Stability and computation of the medial axis — a state-of-the-art report. *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, 1:–, 2007.

[4] Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. Skeleton extraction by mesh contraction. *ACM Transactions on Graphics*, 27(3):10, 2008.

[5] Mikhail Belkin, Jian Sun, and Yusu Wang. Discrete laplace operator on meshed surfaces. In *Proceedings of the twenty-fourth annual symposium on Computational geometry*, SCG '08, pages 278–287, New York, NY, USA, 2008. ACM.

[6] Henning Biermann, Adi Levin, and Denis Zorin. Piecewise smooth subdivision surfaces with normal control. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 113–120, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[7] Blender-Foundation. Blender open source 3d application for modeling, animation, rendering, compositing, video editing and game creation. http://www.blender.org/, 2012.

[8] Mario Botsch, Mark Pauly, Christian Rossl, Stephan Bischoff, and Leif Kobbelt. Geometric modeling based on triangle meshes. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[9] Luc Buatois, Guillaume Caumon, and Bruno Lévy. Concurrent number cruncher: an efficient sparse linear solver on the gpu. In *Proceedings of the Third international conference on High Performance Computing and Communications*, HPCC'07, pages 358–371, Berlin, Heidelberg, 2007. Springer-Verlag.

[10] E. Catmull and J. Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, November 1978.

[11] Cgal, Computational Geometry Algorithms Library. http://www.cgal.org.

[12] Yong Chen and Charlie C. L. Wang. Uniform offsetting of polygonal model based on layered depth-normal images. *Comput. Aided Des.*, 43(1):31–46, January 2011.

[13] Sabine Coquillart. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. *SIGGRAPH Comput. Graph.*, 24(4):187–196, September 1990.

[14] Nicu D. Cornea and Patrick Min. Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548, 2007.

[15] Tony DeRose, Michael Kass, and Tien Truong. Subdivision surfaces in character animation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 85–94, New York, NY, USA, 1998. ACM.

[16] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H. Barr. Anisotropic feature-preserving denoising of height fields and bivariate data. In *In Graphics Interface*, pages 145–152, 2000.

[17] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 317–324, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[18] Ran Gal, Olga Sorkine, Niloy J. Mitra, and Daniel Cohen-Or. iwires: An analyze-and-edit approach to shape manipulation. *ACM Transactions on Graphics (Siggraph)*, 28(3):#33, 1–10, 2009.

[19] Tinsley A. Galyean and John F. Hughes. Sculpting: an interactive volumetric modeling technique. *SIGGRAPH Comput. Graph.*, 25(4):267–274, July 1991.

[20] Ozgur Gonen and Ergun Akleman. Smi 2012: Short paper: Sketch based 3d modeling with curvature classification. *Comput. Graph.*, 36(5):521–525, August 2012.

[21] Uwe Hahne. *Weighting in Laplacian Mesh Editing.* PhD thesis, Bauhaus-Universitat Weimar, 2006.

[22] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: a sketching interface for 3d freeform design. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[23] ALICE Team in INRIA Nancy Grand-Est / Loria. Graphite. Research platform for computer graphics, 3D modeling and numerical geometry. http://alice.loria.fr/WIKI/index.php/Graphite.

[24] Bruno Levy. Numerical methods for digital geometry processing. In *2005 Israel-Korea Bi-national Conference on New Technologies and Visualization Methods for Product Development on Design and Reverse Engineering*, Haifa/Israel, Nov 2005.

[25] Liu, L., Bajaj, C., Deasy, J. O., Low, D. A., Ju, and T. Surface reconstruction from non-parallel curve networks. *Computer Graphics Forum*, 27(2):155–163, April 2008.

[26] C. Loop. Smooth subdivision surfaces based on triangles. Department of mathematics, University of Utah, Utah, USA, August 1987.

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

[28] Tony Mullen. *Introducing character animation with Blender*. Indianapolis, Ind. Wiley Pub. cop., 2007.

[29] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, GRAPHITE '06, pages 381–389, New York, NY, USA, 2006. ACM.

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

[31] Ulrich Pinkall, Strasse Des Juni, and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2:15–36, 1993.

[32] Alexander Pinzon, Fabio Martinez, and Eduardo Romero. Análisis experimental de la extracción del esqueleto por contracción con suavizado laplaciano, December 2010. Poster presented at 6th International Seminar on Medical Image Processing and Analysis (SIPAIM 2010), December 1–4, Universidad Nacional de Colombia, Bogota, Colombia.

[33] Alexander Pinzon and Eduardo Romero. Software para la extracción del esqueleto por contracción y suavizado, December 2011. Poster presented at 7th International Seminar on Medical Image Processing and Analysis (SIPAIM 2011), December 6–8, Universidad Industrial de Santander, Bucaramanga, Colombia.

[34] Alexander Pinzon and Eduardo Romero. Shape inflation with an adapted laplacian operator for hybrid quad/triangle meshes. *2013 26th SIBGRAPI Conference on Graphics, Patterns and Images*, 0:179–186, 2013.

[35] Pawas Ranjan. *Discrete Laplace Operator: Theory and Applications*. PhD thesis, The Ohio State University, Columbus, OH, USA, 2012.

[36] Steven Rosenberg. *The Laplacian on a Riemannian manifold: an introduction to analysis on manifolds*. Number 31. Cambridge University Press, 1997.

[37] Robert Schneider, Leif Kobbelt, and Hans-Peter Seidel. Improved bi-laplacian mesh fairing. In Tom Lyche and Larry L. Schumaker, editors, *Mathematical Methods for*

Curves and Surfaces: Oslo 2000, Innovations in Applied Mathematics, pages 445–454, Oslo, Norway, 2001. Vanderbilt University.

[38] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '04, pages 175–184, New York, NY, USA, 2004. ACM.

[39] Olga Sorkine. Differential representations for mesh processing. *Comput. Graph. Forum*, 1:789–807, 2006.

[40] Michael Spivak. *A Comprehensive Introduction to Differential Geometry*, volume 1. Publish or Perish, Inc, 3rd edition, January 1999.

[41] Jos Stam. Exact evaluation of catmull-clark subdivision surfaces at arbitrary parameter values. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 395–404, New York, NY, USA, 1998. ACM.

[42] Lucian Stanculescu, Raphalle Chaine, and Marie-Paule Cani. Freestyle: Sculpting meshes with self-adaptive topology. *Computers & Graphics*, 35(3):614 – 622, 2011. Shape Modeling International (SMI) Conference 2011.

[43] Gabriel Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, pages 351–358, New York, NY, USA, 1995. ACM.

[44] Yunhui Xiong, Guiqing Li, and Guoqiang Han. Mean laplace-beltrami operator for quadrilateral meshes. In Zhigeng Pan, Adrian Cheok, Wolfgang Muller, and Xubo Yang, editors, *Transactions on Edutainment V*, volume 6530 of *Lecture Notes in Computer Science*, pages 189–201. Springer Berlin / Heidelberg, 2011.

[45] Guoliang Xu. Discrete laplace-beltrami operators and their convergence. *Comput. Aided Geom. Des.*, 21(8):767–784, 2004.

[46] Kun Zhou, Jin Huang, John Snyder, Xinguo Liu, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Large mesh deformation using the volumetric graph laplacian. *ACM Trans. Graph.*, 24(3):496–503, July 2005.

[47] Wei Zhuo and Jarek Rossignac. Curvature-based offset distance: Implementations and applications. *Computers & Graphics*, 36(5):445 – 454, 2012.