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

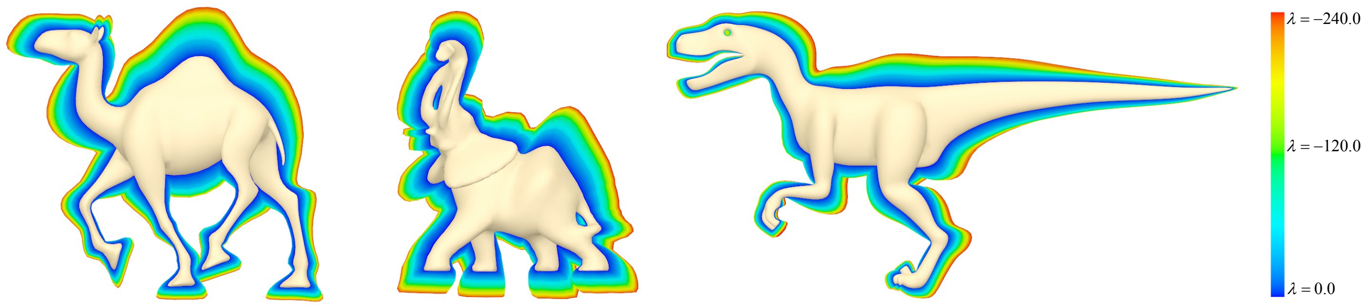

Fig. 1. A set of 48 successive shapes enhanced, from $\lambda = 0.0$ in blue to $\lambda = -240.0$ in red, with steps of -5.0.

***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**

I. Introduction

Over the last years several, modeling techniques able to generate a variety of realistic shapes, have been developed [1]. Editing techniques have evolved from affine transformations to advanced tools like sculpting [2], [3], [4], editing, creation from sketches [5], [6], and complex interpolation techniques [7], [8]. 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 [9], [10]. 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 [11], or the modification of some vertex weights to locally control the zone of influence [12]. 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 [13]. Nowadays, many artists use a manual connection of a couple of vertices to perform animation processes and interpolation [14]. 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 [4]. 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 I-A presents works related to the Laplacian mesh processing, digital sculpting, and offsetting methods for polygonal meshes; In section II , it is described the theoretical framework of the Laplacian operator for polygon meshes; In section III, 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.

A. 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 [7].

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 [15]. 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 [16].

[17] 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 [18], 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 [4]. 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 [3]. The drawback comes from the difficulty of maintaining the surface details during larger scale deformations.

II. 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.

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

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

A. Gradient of Voronoi Area

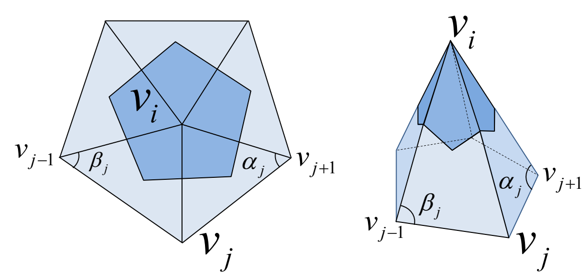

Fig. 2. Area of the Voronoi region around v_i in dark blue.
v_j belong to the first neighborhood around v_i. α_j and β_j opposite angles to edge $\overrightarrow{v_j - v_i}$.

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 2, The area change produced by the movement of v_i is called the gradient of *Voronoi region [19], [20], [21].*

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

If the gradient in equation (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 (2).

$$2\kappa\mathbf{n} = \frac{\nabla A}{A} \quad (2)$$

B. Laplace Beltrami Operator

The *Laplace Beltrami operator* LBO noted as $\triangle$ is used for measuring the mean curvature normal to the Surface S [19].

$$\triangle S = 2\kappa\mathbf{n} \quad (3)$$

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

III. 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.

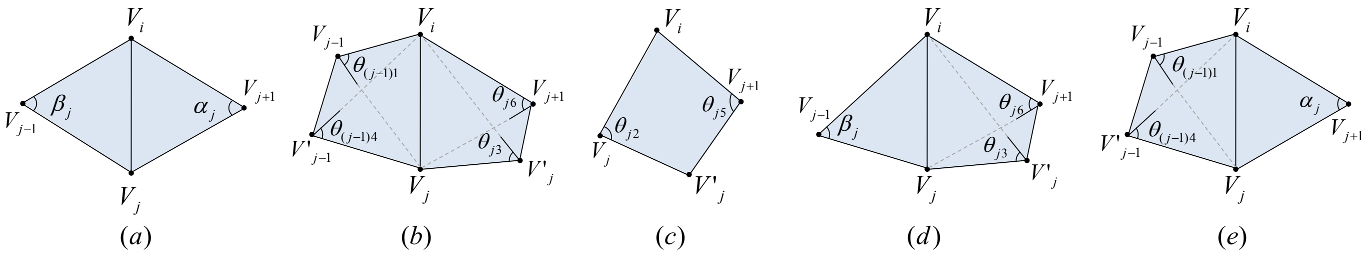

Fig. 3. The 5 basic triangle-quad cases with common vertex V_i and the relationship with V_j and V_j'. (a) Two triangles [Desbrun 1999]. (b) (c) Two quads and one quad [Xiong 2011]. (d) (e) Triangles and quads (TQLBO).

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.

A. 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

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

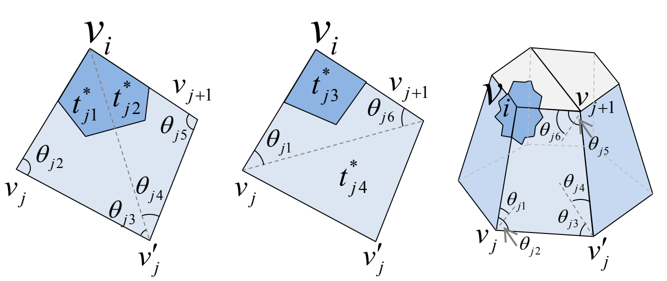

Fig. 4. $t^*_{j1} \equiv\triangle\ v_i v_j v_j'$, $t^*_{j2} \equiv\triangle\ v_i v_j' v_{j+1}$, $t^*_{j3} \equiv\triangle\ v_i v_j v_{j+1}$ Triangulations of the quad with common vertex v_i proposed by [Xiong 2011] to define Mean LBO.

Applying the mean average area, according to [22], from all possible triangulations, as show in figure 4, 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 $t_1, t_2, ..., t_k, ..., t_r \in T_{v_i}$

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

Applying the gradient operator to (4)

$$\nabla A(v_i) = \frac{1}{2}\sum_{j=1}^{m}\left[\nabla A\left(t^*_{j1}\right) + \nabla A\left(t^*_{j2}\right) + \nabla A\left(t^*_{j3}\right)\right] + \sum_{k=1}^{r}\nabla A(t_k) \tag{5}$$

According to (1), we have

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

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

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

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

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.

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

$$\triangle_S(v_i) = 2\kappa\mathbf{n} = \frac{\nabla A}{A} = \frac{1}{2A}\sum_{v_j \in N_1(v_i)} w_{ij}(v_j - v_i) \tag{6}$$

$$w_{ij} = \begin{cases} (\cot\alpha_j + \cot\beta_j) & \text{case } a. \\ \frac{1}{2}\left(\cot\theta_{(j-1)1} + \cot\theta_{(j-1)4} + \cot\theta_{j3} + \cot\theta_{j6}\right) & \text{case } b. \\ (\cot\theta_{j2} + \cot\theta_{j5}) & \text{case } c. \\ \frac{1}{2}\left(\cot\theta_{j3} + \cot\theta_{j6}\right) + \cot\beta_j & \text{case } d. \\ \frac{1}{2}\left(\cot\theta_{(j-1)1} + \cot\theta_{(j-1)4}\right) + \cot\alpha_j & \text{case } e. \end{cases} \tag{7}$$

We define a TQLBO as a matrix equation

$$L(i,j) = \begin{cases} -\frac{1}{2A_i} w_{ij} & \text{if } j \in N(v_i) \\ \frac{1}{2A_i}\sum\limits_{k \in N(v_i)} w_{ik} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

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 (7), $N(v_i)$ 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

$$L(i,j) = \begin{cases} -\frac{w_{ij}}{\sum\limits_{k \in N(v_i)} w_{ik}} & \text{if } j \in N(v_i) \\ \delta_{ij} & \text{otherwise} \end{cases} \tag{9}$$

Where δ_{ij} being the Kronecker delta function.

B. 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:

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

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

$$\left(I - \left|\lambda dt\right| W_p L\right) V' = V^t \tag{10}$$

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

The vertices V^{t+1} are inflated, along their reverse curvature direction, by solving the linear system: $Ax = b$, where $A = I - \left|\lambda dt\right| W_p L,\ L$ is the Normalized TQLBO defined in the equation (9), $x = V'$ 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 10. 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. [20], 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.

C. 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.

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

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

D. Subdivision surfaces

The Catmull-Clark subdivision transformation is used to smooth a surface, as the limit of a sequence of subdivision steps [10]. This process is governed by a B-spline curve [23], performing a recursive subdivision transformation that refines the model into a linear interpolation that approximates a smooth surface. The model smoothness is automatically guaranteed [11].

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

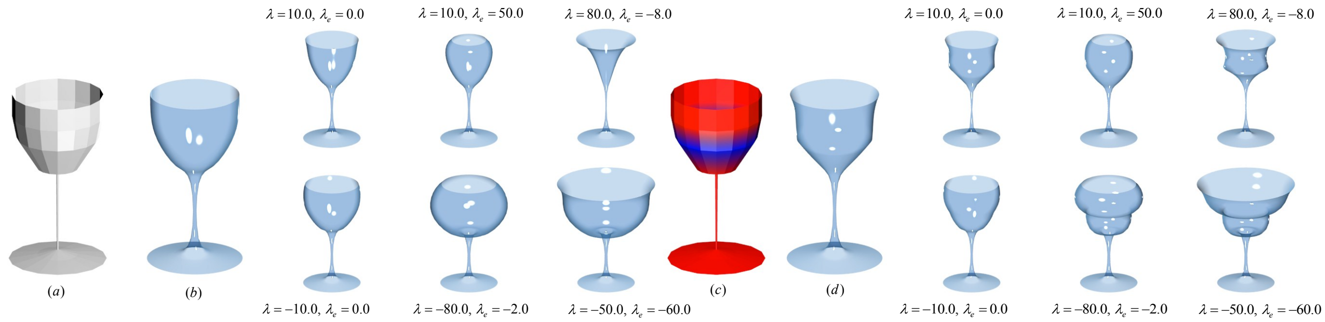

Fig. 5. Family of cups generated with our method, from a coarse model (a), (c): the shape, obtained from the Catmull-Clark Subdivision (b), (d), is inflated. Soft constraints, over the coarse model, is drawn in red and blue (c).

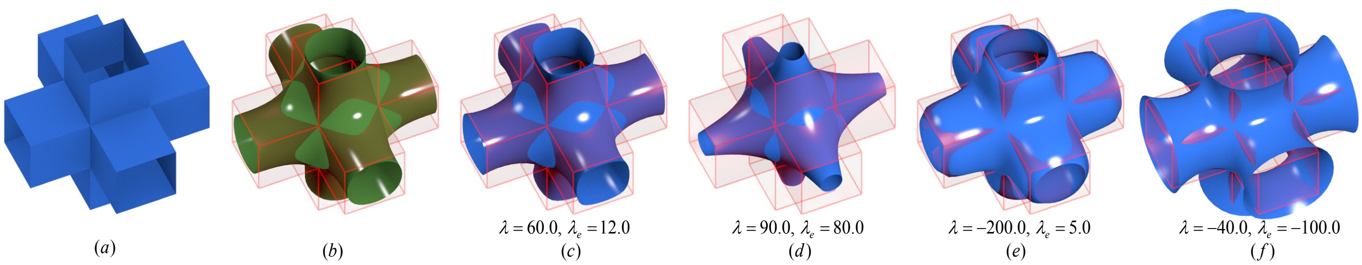

Fig. 6. (a) Original Model, (b) Model with Catmull-Clark Subdivision. Models with Laplacian smoothing: (c) and (d). Models with a first Laplacian filtering $\lambda = 60.0$, $\lambda_e = 12.0$ and before applying shape inflation: (e) and (f).

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 5.c.

In equation 10, 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.

IV. 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 1, 5, 6, 7, 8, 9, 10, 11, 12. The shape inflation was assessed with TQLBO method on a PC with AMD Quad-Core Processor @ 2.40 GHz and 8 GB RAM.

Figure 7 shows the results when applying the Laplace Beltrami Operator TQLBO of equation (8) 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.

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 (5). This test used a coarse cup model, in which the subdivision was performed, followed by a Laplacian smoothing and inflation. In figure (5).c, (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 (5).d.

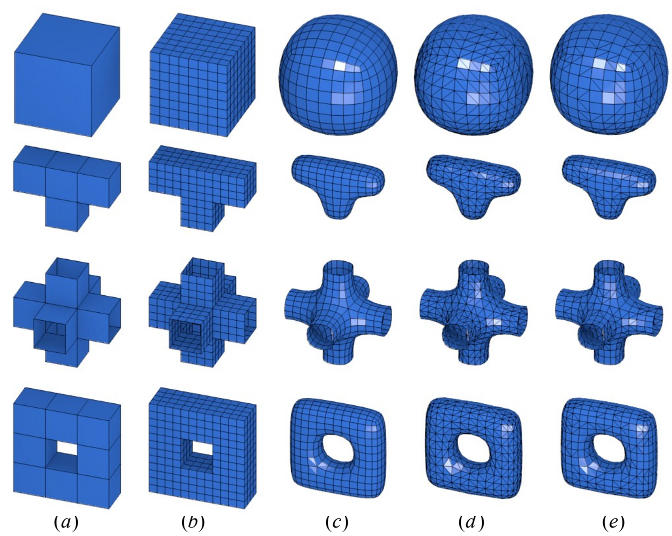

Fig. 7. (a) Original Model. (b) Simple subdivision. (c), (d) (e) Laplacian smoothing with $\lambda = 7$ and 2 iterations: (c) for quads, (d) for triangles, (e) for triangles and quads random chosen.

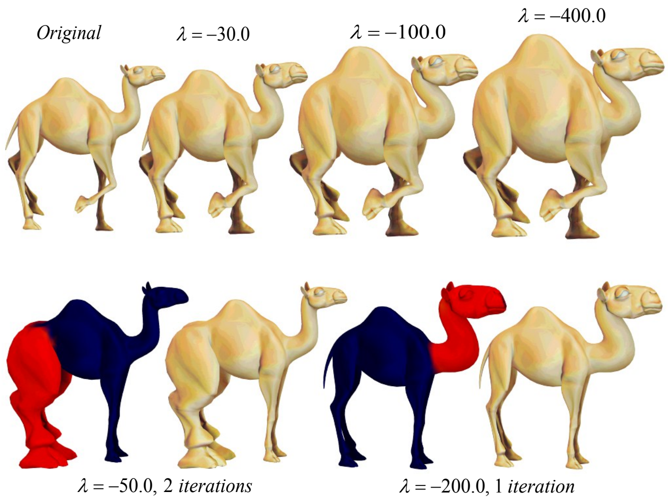

Fig. 8. Top row: Original camel model in left. Shape inflation with $\lambda = -30.0$, $\lambda = -100.0$, $\lambda = -400.0$. Bottom row: Shape inflation with weight vertex group, $\lambda = -50.0$ and 2 iterations for the legs, $\lambda = -200.0$ and 1 iteration for the head and neck.

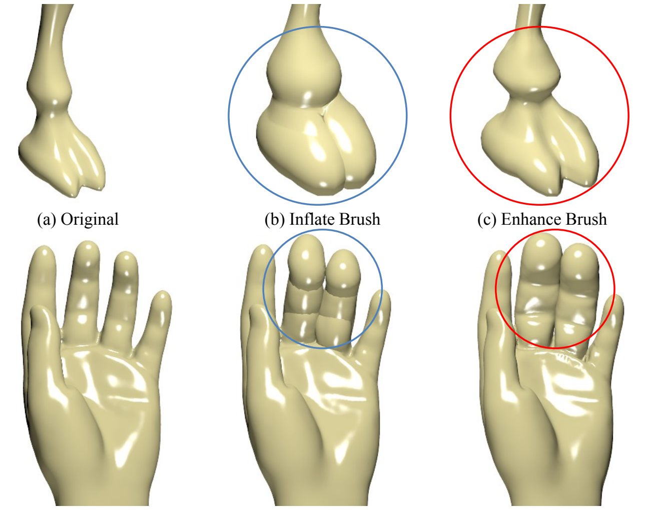

Fig. 9. Top row: (a) Leg Camel, (b) Traditional inflate brush for leg into blue circle, (c) Shape inflation brush for leg into red circle. Bottom row: (a) Hand, (b) Traditional inflate brush for fingers into blue circle, (c) Shape inflation brush for fingers in red circle.

Laplacian smoothing applied with simple subdivision (see figure 6.c.) may produce similar results to those obtained with Catmull-Clark (see figure 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 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 1. The larger the λ parameter the larger the model feature inflations. The bottom row of figure 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.

The inflation of the silhouette features is predictable and invariant under isometric transformations, as those classically used in some animations (see Figure 12). 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 12. 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 9 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 9.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 9.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 10 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.

Tests with the Laplacian operator (equation 8) and its normalized version (equation 9), 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 11.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 11.c shows different results due to the quads areas in the model. Quads with larger area have smaller inflation (figure 11.c skull), and smaller quads have larger inflations (figure 11.c chin).

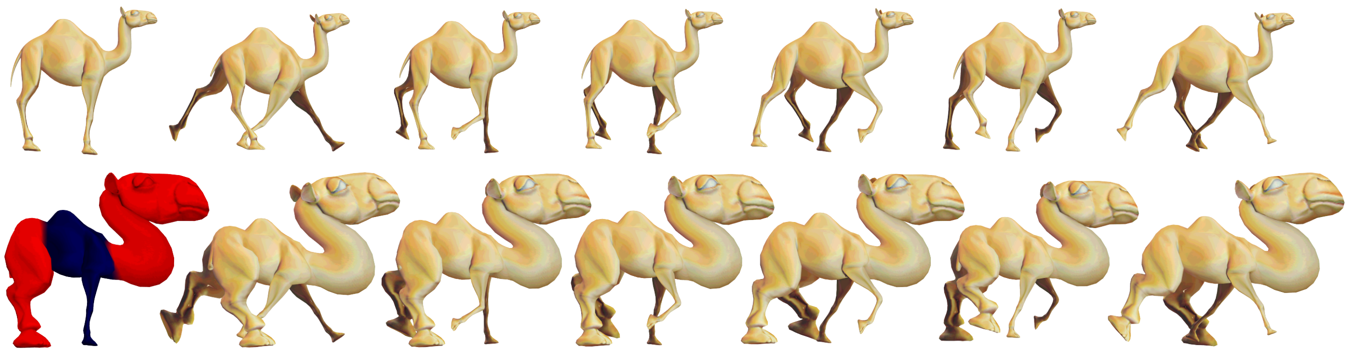

Fig. 12. The method is pose insensitive. The inflation for the different poses are similar in terms of shape. Top row: Original walk cycle camel model. Bottom row: Shape inflation with weight vertex group, $\lambda = -400$ and 2 iterations.

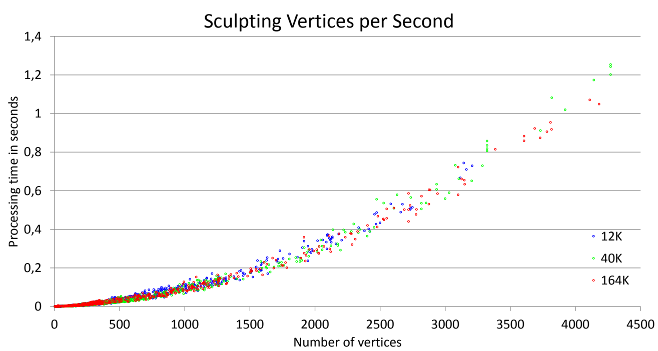

Fig. 10. Performance of our dynamic shape inflation brush in terms of the sculpted vertices per second. Three models with 12K, 40K, 164K vertices used for sculpting in real time.

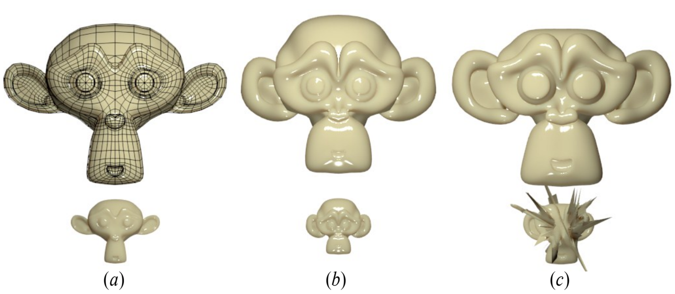

Fig. 11. (a) Bottom row: Original Model. Top row: Original model scaled by 4. (b) Top and bottom row: inflating with Normalized-TQLBO $\lambda = -50$ (c) Top and bottom row: inflating with TQLBO $\lambda = -50$.

A. *Implementation*

The method was implemented as a modifier for modeling and brush for sculpting, on the Blender software [13] 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 8 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 10 was used OpenNL [24] which is a a library for solving sparse linear system.

V. 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.

REFERENCES

[1] M. Botsch, M. Pauly, C. Rossl, S. Bischoff, and L. Kobbelt, "Geometric modeling based on triangle meshes," in *ACM SIGGRAPH 2006 Courses*, ser. SIGGRAPH '06. New York, NY, USA: ACM, 2006. [Online]. Available: http://doi.acm.org/10.1145/1185657.1185839

[2] S. Coquillart, "Extended free-form deformation: a sculpturing tool for 3d geometric modeling," *SIGGRAPH Comput. Graph.*, vol. 24, no. 4, pp. 187–196, Sep. 1990. [Online]. Available: http://doi.acm.org/10.1145/97880.97900

[3] T. A. Galyean and J. F. Hughes, "Sculpting: an interactive volumetric modeling technique," *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 267–274, Jul. 1991. [Online]. Available: http://doi.acm.org/10.1145/127719.122747

[4] L. Stanculescu, R. Chaine, and M.-P. Cani, "Freestyle: Sculpting meshes with self-adaptive topology," *Computers & Graphics*, vol. 35, no. 3, pp. 614 – 622, 2011, shape Modeling International (SMI) Conference 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0097849311000720

[5] T. Igarashi, S. Matsuoka, and H. Tanaka, "Teddy: a sketching interface for 3d freeform design," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '99. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 409–416. [Online]. Available: http://dx.doi.org/10.1145/311535.311602

[6] O. Gonen and E. Akleman, "Smi 2012: Short paper: Sketch based 3d modeling with curvature classification," *Comput. Graph.*, vol. 36, no. 5, pp. 521–525, Aug. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.cag.2012.03.019

[7] 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*, ser. SGP '04. New York, NY, USA: ACM, 2004, pp. 175–184. [Online]. Available: http://doi.acm.org/10.1145/1057432.1057456

[8] K. Zhou, J. Huang, J. Snyder, X. Liu, H. Bao, B. Guo, and H.-Y. Shum, "Large mesh deformation using the volumetric graph laplacian," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 496–503, Jul. 2005. [Online]. Available: http://doi.acm.org/10.1145/1073204.1073219

[9] E. Catmull and J. Clark, "Recursively generated b-spline surfaces on arbitrary topological meshes," *Computer-Aided Design*, vol. 10, no. 6, pp. 350–355, Nov. 1978. [Online]. Available: http://dx.doi.org/10.1016/0010-4485(78)90110-0

[10] J. 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*, ser. SIGGRAPH '98. New York, NY, USA: ACM, 1998, pp. 395–404. [Online]. Available: http://doi.acm.org/10.1145/280814.280945

[11] T. DeRose, M. Kass, and T. Truong, "Subdivision surfaces in character animation," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '98. New York, NY, USA: ACM, 1998, pp. 85–94. [Online]. Available: http://doi.acm.org/10.1145/280814.280826

[12] H. Biermann, A. Levin, and D. Zorin, "Piecewise smooth subdivision surfaces with normal control," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 113–120. [Online]. Available: http://dx.doi.org/10.1145/344779.344841

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

[14] T. Mullen, *Introducing character animation with Blender*. Indianapolis, Ind. Wiley Pub. cop., 2007. [Online]. Available: http://opac.inria.fr/record=b1122208

[15] W. Zhuo and J. Rossignac, "Curvature-based offset distance: Implementations and applications," *Computers & Graphics*, vol. 36, no. 5, pp. 445 – 454, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0097849312000520

[16] Y. Chen and C. C. L. Wang, "Uniform offsetting of polygonal model based on layered depth-normal images," *Comput. Aided Des.*, vol. 43, no. 1, pp. 31–46, Jan. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.cad.2010.09.002

[17] R. Gal, O. Sorkine, N. J. Mitra, and D. Cohen-Or, "iwires: An analyze-and-edit approach to shape manipulation," *ACM Transactions on Graphics (Siggraph)*, vol. 28, no. 3, pp. #33, 1–10, 2009.

[18] Y. Ohtake, A. Belyaev, and H.-P. Seidel, "Ridge-valley lines on meshes via implicit surface fitting," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 609–612, Aug. 2004. [Online]. Available: http://doi.acm.org/10.1145/1015706.1015768

[19] U. Pinkall, S. D. Juni, and K. Polthier, "Computing discrete minimal surfaces and their conjugates," *Experimental Mathematics*, vol. 2, pp. 15–36, 1993.

[20] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '99. New York, NY, USA: ACM Press Addison-Wesley Publishing Co., 1999, pp. 317–324. [Online]. Available: http://dx.doi.org/10.1145/311535.311576

[21] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," in *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Heidelberg: Springer-Verlag, 2003, pp. 35–57.

[22] Y. Xiong, G. Li, and G. Han, "Mean laplace-beltrami operator for quadrilateral meshes," in *Transactions on Edutainment V*, ser. Lecture Notes in Computer Science, Z. Pan, A. Cheok, W. Muller, and X. Yang, Eds. Springer Berlin / Heidelberg, 2011, vol. 6530, pp. 189–201.

[23] C. Loop, "Smooth subdivision surfaces based on triangles," Department of Mathematics, University of Utah, Utah, USA, Aug. 1987.

[24] L. Buatois, G. Caumon, and B. 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*, ser. HPCC'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 358–371. [Online]. Available: http://dl.acm.org/citation.cfm?id=2401945.2401989