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

This work was accepted as part of the software Blender [?], an open source 3D application for modeling, animation, rendering, compositing, video editing and game creation. The work was supported by an awarded internship of the Google Summer of Code 2012 program, administered for Google Inc.

1.1 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 allows working with hybrid meshes composed by triangles and quads based on Laplacian operator proposed by Pinzón and Romero [?].

1.2 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.

1.3 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.

1.4 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.

1.5 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

1.6 Mesh Smoothing

A common way to attenuate noise in a polygonal mesh is through a diffusion process [?, ?]. 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 cotangent 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 diffusion process can attenuate noise with only one iteration due the sparseness of the laplacian operator.

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

Where L is the Laplacian matrix defined in equation 1-3 for meshes composed by triangles or quads with different size or irregular sampling. λ is a scalar that control the diffusion process, and smoothing factor.

The equation 1-1 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.

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

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 1-2 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 possible to compute the curvature flow. For this reason the system smooth the edges differently within the diffusion process. The boundaries 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.

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

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 (**??**), $N(v_i)$ 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.

1.7 Results and Conclusions

The user interface developed to this tool for the software Blender can be seen in figure **1-1**. 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.

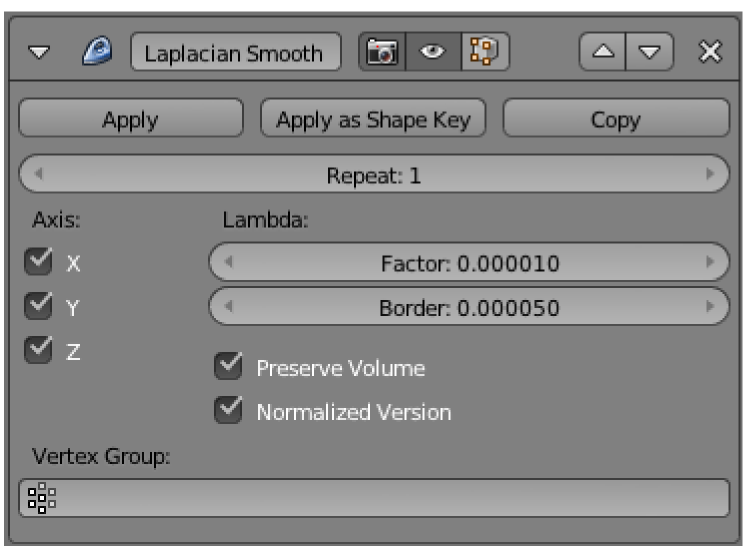

Figure **1-1**: Panel inside blender user interface of the Laplacian Smooth modifier tool.

The tool developed can set the parameter λdt of equation 1-2. Using a small Lambda factor ($\lambda < 1.0$), you can remove noise from the shape without affecting desirable geometry (see figure **1-2**.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 **1-2**.c and **1-2**.d).

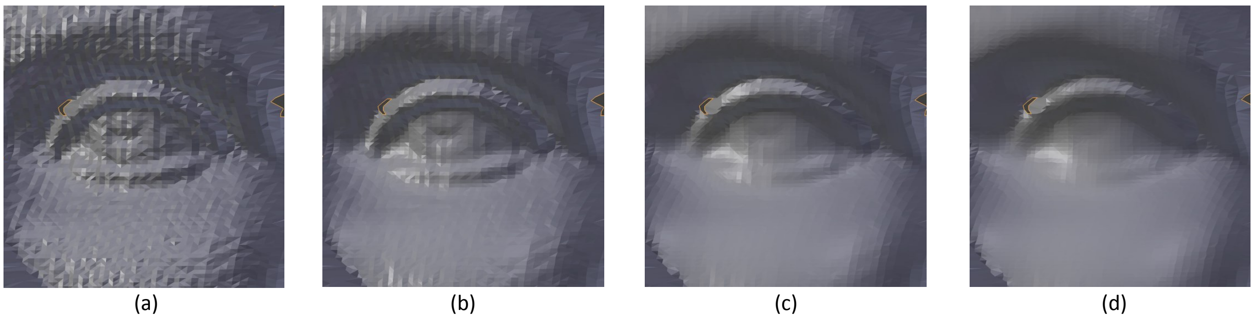

Figure **1-2**: Noise attenuation in face model with Laplacian smoothing tool using only one iteration and changing λ. (a) Original Model. (b) Smoothing $\lambda = 0.5$. (c) Smoothing $\lambda = 2.5$ (d) Smoothing with $\lambda = 5.0$.

The user can smooth the boundaries configuring the parameter "*Border*", seen in figure **1-1**. 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 **1-3**, in the figure you can see how the boundary inside the red circle is smoothing.

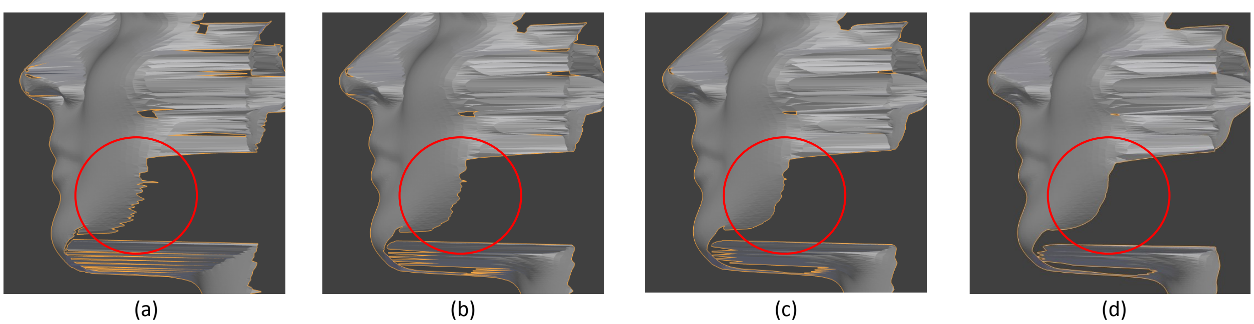

Figure **1-3**: Smoothing boundary changing λ_{Border} factor. (a) Original Model. (b) Smoothing $\lambda_{Border} = 1.0$. (c) Smoothing $\lambda_{Border} = 2.5$ (d) Smoothing with $\lambda_{Border} = 10.0$.

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 **1-4**.c defined in red region where you want it applied the algorithm the results are shown in figure **1-4**.d where only were smoothed vertices in the red region.

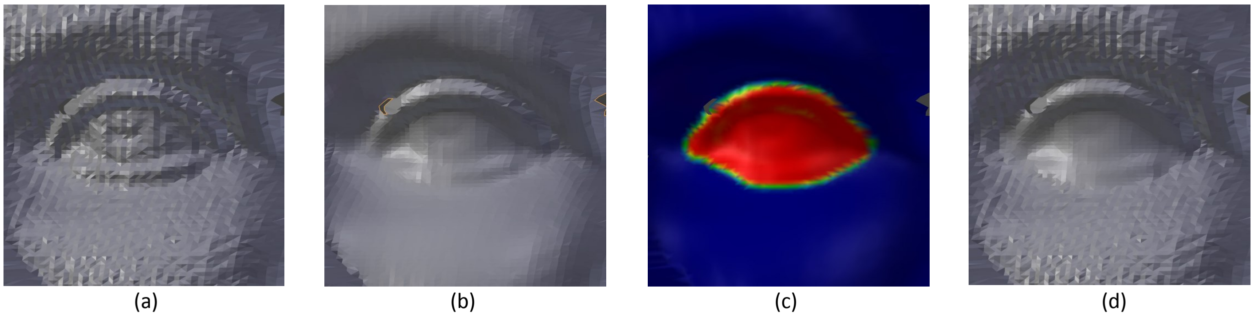

Figure **1-4**: Use of weights per vertex to constrain the effect of mesh smoothing. (a) Original Model. (b) Smoothing with $\lambda = 1.5$ (c) red vertices $weight = 1.0$, blue vertices $weight = 0.0$. (d) Smoothing with $\lambda = 2.5$. Red vertices were only smoothing.

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.