Surface Meshes Smoothing

Release 0.01

Arnaud Gelas, Alexandre Gouaillard
and Sean Megason

January 24, 2009

Department of System Biology, Harvard Medical School,
Boston, MA 02115, USA

Abstract

This paper describes the implementation of a surface smoothing filter in ITK, based on the Quad Edge Mesh surface data structure [4].

Contents

1 Introduction

Extracting surface mesh models from given volume images is a common step in many applications, such as visualization, intervention planning, quantification. Coming from the acquisition, the segmentation method or the meshing method itself, the resulting surface mesh may have somehow some disturbing artefacts, or may capture too much noise. So in many cases surface mesh smoothing is required.

In VTK, one can find two filters that implement smoothing: vtkSmoothPolyDataFilter and vtkWindowedSincPolyDataFilter. This implementation is an extended version of vtkSmoothPolydataFilter for the QuadEdgeMesh data structure. This filter does not support feature edges at this time like the VTK filter does, e.g. all edges are filtered equally. However, it implements a lot of different weights taken to smooth the mesh depending on different factors (area, angles,mean curvature flow, ...) of the local surface. One unique feature that improves greatly the quality of the resulting smooth surface is the ability to use internally a Delaunay Conforming Filter [2]. The Description of the filter is highly inspired on the documentation of vtkSmoothPolyDataFilter.

2 Description

itkQuadEdgeMeshSmoothing is a filter that adjusts point coordinates using Laplacian-like smoothing. The effect is to ”relax” the mesh. Note that this filter operates only on itkQuadEdgeMesh(es) [4].

The algorithm proceeds as follows. For each point p_i^n at a given iteration n, a topological analysis (analysis of the one-ring $N(i)$) is performed to determine which points p_j^n are connected to p_i^n, and which cells are connected to the same point. Note that if no Delaunay conforming is used, the topology is constant and $\mathrm{p}_j^n = \mathrm{p}_j$. Next, an iteration phase begins over all Points p_i^n. The new coordinates of the point p_i^{n+1} are modified according to a weighted average of the connected points p_j^n.

$$\mathrm{p}_i^{n+1} = \mathrm{p}_i^n + m\_RelaxationFactor \cdot \frac{\sum_{j \in N(i)} \alpha_{ij} \cdot \left(\mathrm{p}_j^n - \mathrm{p}_i^n\right)}{\sum_{j \in N(i)} \alpha_{ij}} \tag{1}$$

(A relaxation factor, $m\_RelaxationFactor$ in $]0,1]$, is available to control the amount of displacement). The weights α_{ij} can be computed using different formula described shortly in the following section and more in details in [1].

Warning: The Laplacian operation reduces high frequency information in the geometry of the mesh. With excessive smoothing important details may be lost, and the surface may shrink towards the centroid.

3 Implementation Details

There is one main class, `itk::QuadEdgeMeshSmoothing`, that does all the job. It is templated over the Input and Output Mesh Types.

$m\_RelaxationFactor$ is given by the user, and is in $]0,1]$

$m\_NumberOfIterations$ is given by the user

$m\_DelaunayConforming$ is an option to optimize the triangle aspect ratio in the same time as smoothing, it leads to a better mesh in terms of valence, and triangle aspect ratio. Note that the resulting mesh is then an intrinsic Delaunay triangulation of the surface. Note that this operation can be quite expensive since it is applied at the end of each iteration.

α_{ij} are spring coefficients taken from `itk::QuadEdgeMeshParamMatrixCoefficients`

1. *Graph theory based*: `class itk::OnesMatrixCoefficients` [6,7]

$$D_{ij} = 1 \tag{2}$$

2. *Chord Length*: `class itk::InverseEuclideanDistanceMatrixCoefficients`

$$D_{ij} = \frac{1}{\|\mathrm{p}_i - \mathrm{p}_j\|^2} \tag{3}$$

3. *Discrete Conformal*: `class itk::ConformalMatrixCoefficients` [5]

$$D_{ij} = \cot \alpha_{ij} + \cot \beta_{ij} \tag{4}$$

4. *Discrete Authalic*: `class itk::AuthalicMatrixCoefficients` [3]

$$D_{ij} = \frac{\cot \gamma_{ij} + \cot \delta_{ij}}{\|\mathrm{p}_i - \mathrm{p}_j\|^2} \tag{5}$$

5. *Intrinsic*: `class itk::HarmonicMatrixCoefficients` [3]

$$D_{ij} = \mu * \frac{\cot \gamma_{ij} + \cot \delta_{ij}}{\|\mathrm{p}_i - \mathrm{p}_j\|^2} + (1-\mu) * \cot \alpha_{ij} + \cot \beta_{ij} \tag{6}$$

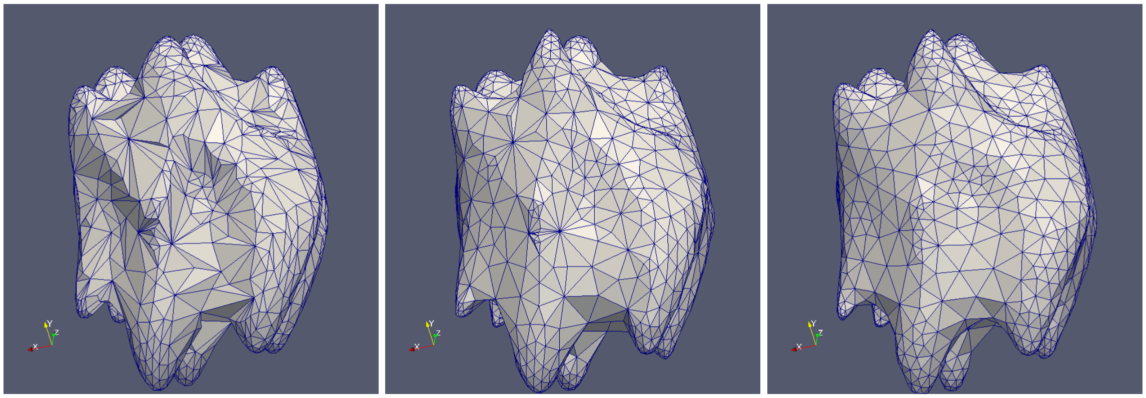
4 Usage

```
typedef double Coord;
const unsigned int Dimension = 3;

typedef itk::QuadEdgeMesh< Coord, Dimension > MeshType;

...

// Note that any other coefficients could have been used
itk::OnesMatrixCoefficients< MeshType > coeff0;

typedef itk::QuadEdgeMeshSmoothing< MeshType, MeshType > SmoothingType;
SmoothingType::Pointer filter = SmoothingType::New( );
filter->SetInput( input );
filter->SetNumberOfIterations( 10 );
filter->SetRelaxationFactor( 0.5 );
filter->SetDelaunayConforming( true );
filter->SetCoefficientsMethod( &coeff0 );
filter->Update();
```

5 Results

We first ran the algorithm on a relatively simple dataset already included in ITK which represent a tooth (/Insight/Testing/Data/Input/geusZeroSurface01.vtk). We ran it using only 5 iterations, once with and once without Delaunay conforming. This dataset is perfect example of what you obtain after a surface extraction algorithm followed by a decimation. The geometry is well captured, up to the artifacts caused by the surface extraction algorithm, but the connectivity is really bad. Lots of vertices are connected to way more than 6 edges and that is known to cause numerical instabilities for mesh processing. We can see on figure 1 that using the Delaunay conforming during the smoothing improves the connectivity and the triangle aspect ratio.

For the second dataset we used a pelvis model, courtesy of of the Visualization Research Group, University of Magdeburg (http://isgwww.cs.uni-magdeburg.de/ rbade/meshsmoothing/data/index.html) This dataset



(a) (b) (c)

Figure 1: Illustration of the smoothing filter output with or without Delaunay conforming. (a) the original surface. One can see the degenerated points. (b) simply smoothed surface. The smoothing operator was performing less well on the degenerated points. (c) smoothing and Delaunay conforming. The sampling of the surface is much more regular, the triangle aspect ratio improved, and the geometry correctly smoothed.

is the output of a surface extraction algorithm and need serious smoothing. Figures 2 show the great improvement in the geometry. No Delaunay conforming as been used here, as the surface extraction algorithm over-sample the surface.

Acknowledgment

This work was funded by a grant from the NHGRI (P50HG004071-02) to found the Center for in toto genomic analysis of vertebrate development.

For the second dataset we used a pelvis model, courtesy of of the Visualization Research Group, University of Magdeburg (http://isgwww.cs.uni-magdeburg.de/ rbade/meshsmoothing/data/index.html).

References

[1] A Gouaillard A. Gelas. Parameterization of discrete surfaces. In *Insight Journal - July - december*, 2007.

[2] A Gouaillard A. Gelas and S. Megason. Triangular meshes delaunay conforming filter. In *Insight Journal - July - december*, 2008.

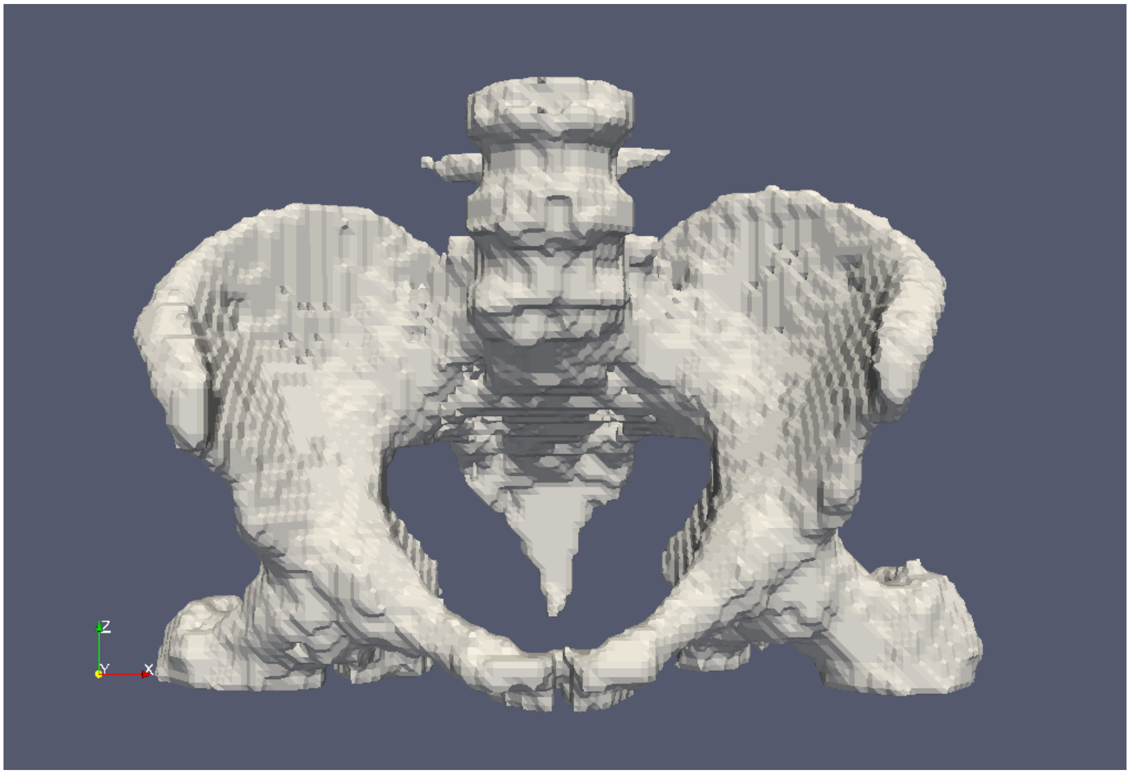
[3] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 21:209–218, 2002. Eurographics conference proceedings.

[4] A Gouaillard, L Florez-Valencia, and E Boix. itkquadedgemesh: A discrete orientable 2-manifold data structure for image processing. In *Insight Journal - July - december*, 2006.

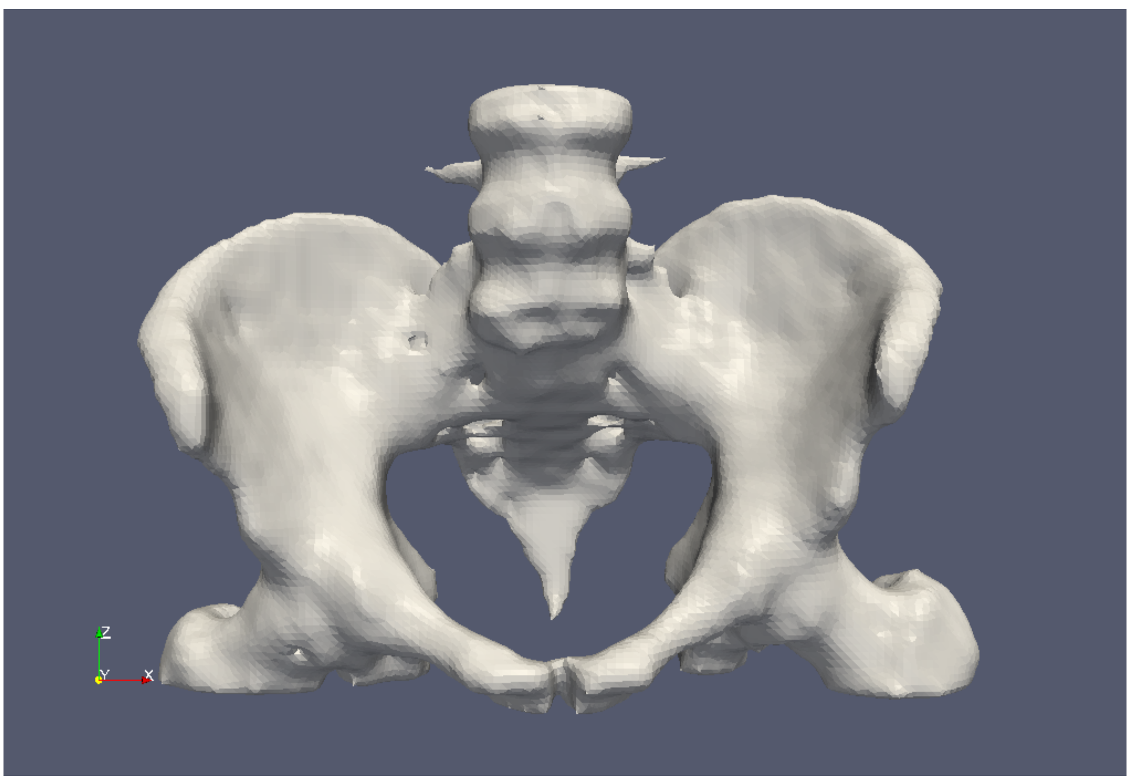
[5] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[6] W. T. Tutte. Convex representations of graphs. In *Proceedings of the London Mathematical Society*, volume 10, pages 304–320, 1960.

[7] W. T. Tutte. How to draw a graph. In *Proceedings of the London Mathematical Society*, volume 13, pages 743–767, 1963.



(a)



(b)

Figure 2: Illustration of the smoothing. (a) original surface. Usual ”staircase” artifact is visible. (b) smoothed surface.