Fundamentals of Geometry Project 2

Coloring Gaussian Curvature on Quartic Triangular Bézier Patches From Smooth Approximations of Irregular Closed Blender Meshes

Brandon Drumheller, Adam Freeman

May 19, 2016

Project Description:

The goal of this project was to color Gaussian curvature on quartic triangular Bézier Patches. These patches were generated from smooth approximations of arbitrary closed meshes, created in the open source modeling program Blender. The refinement process follows the algorithm detailed by Loop [4].

Spline Construction:

The spline construction is split into three stages, each of which is briefly detailed and illustrated:

1. Mesh Refinement
2. Quad-Net Construction
3. Quartic Triangular Bèzier Patch Construction

The output of this algorithm is a collection of quartic triangular Bèzier patches.

Mesh Refinement

The refined meshes are a "smoothing" of the the inputted closed meshes. The meshes resulting from this step have the property that each vertex in the mesh has exactly 4 edges incident on it. As the process is computationally expensive for complex meshes (as with e edges and v vertices $\mathcal{O}(ev)$), simpler meshes are included for illustration.

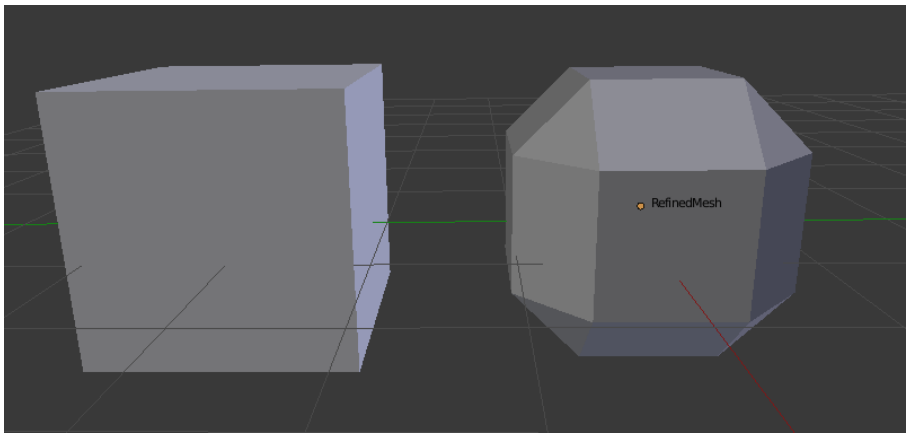

Figure 1: A cube (left) and the resulting refined mesh $\mathcal{M}^1$ (right).

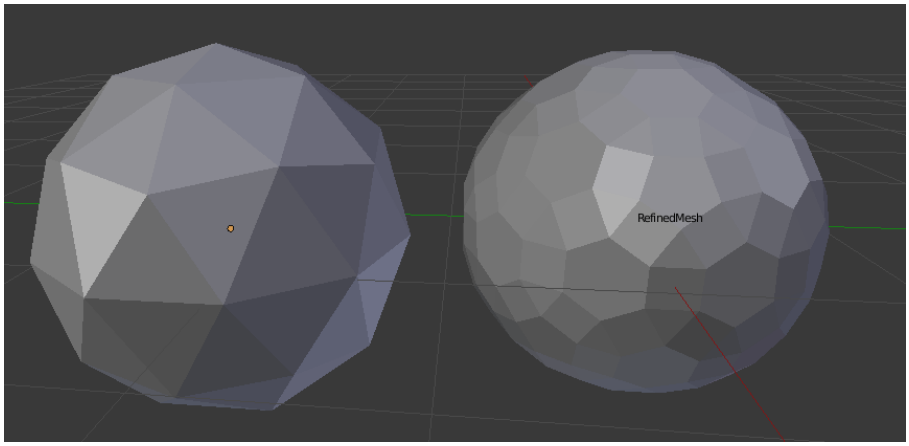

Figure 2: An icosphere (left) and the resulting refined mesh $\mathcal{M}^1$ (right).

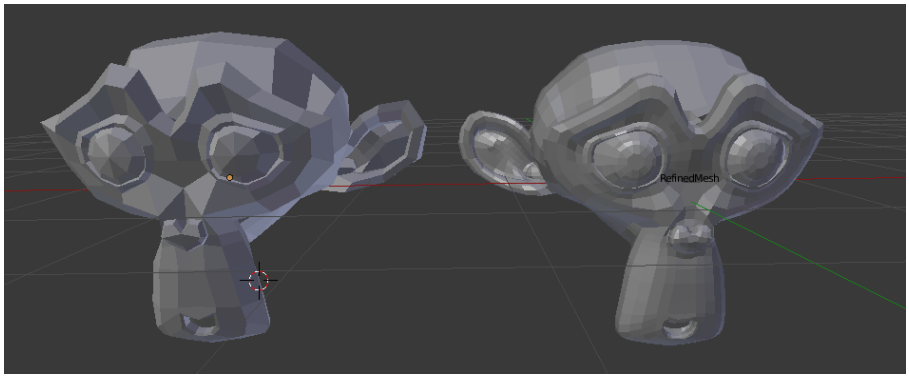

Figure 3: A monkey head (nicknamed 'Suzanne' left) and the resulting refined mesh $\mathcal{M}^1$ (right).

Quad-Net Construction

Quad-nets are sets of 16 points that surround each vertex. These points form nets that when "stitched" together cover the entire refined mesh $\mathcal{M}^1$. Ultimately, the quad-nets serve as an intermediary phase between the mesh refinement and the generation of quartic triangular Bèzier surface patches.

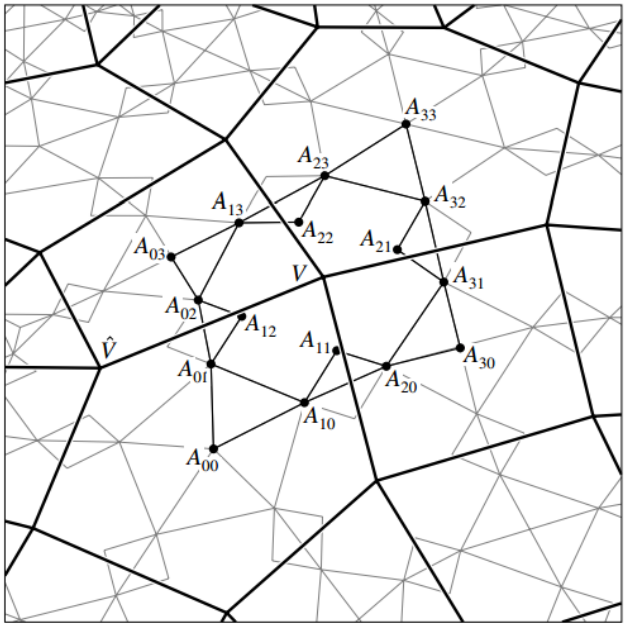

Figure 4: The quadnet corresponding to vertex V on the refined mesh $\mathcal{M}^1$ [4].

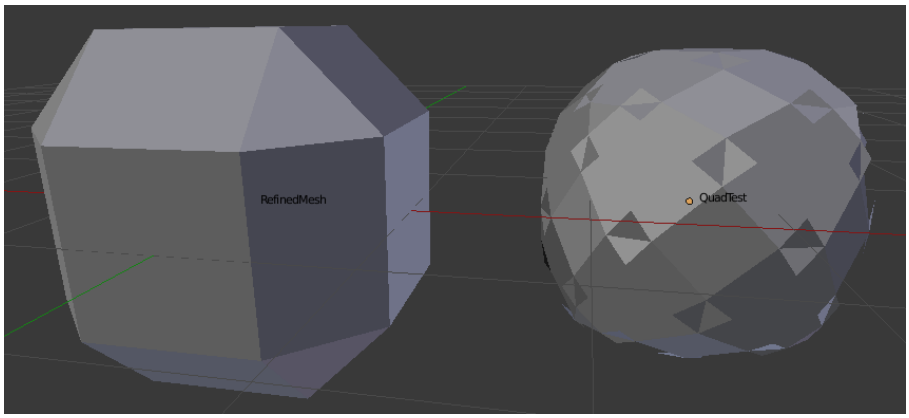

Figure 5: Quad-nets (right) corresponding to the cube's $\mathcal{M}^1$ (left)

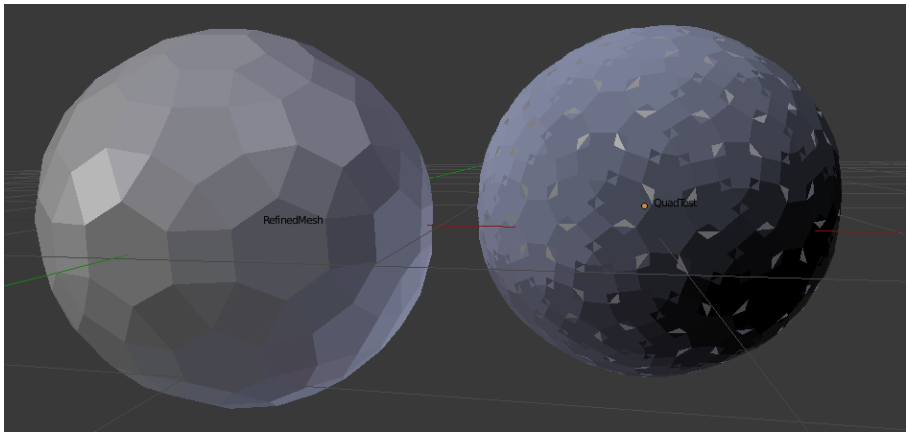

Figure 6: Quad-nets (right) corresponding to the icosphere's $\mathcal{M}^1$ (left)

As previously stated, the implementation is contingent on the fact that the $\mathcal{M}^0$ is a closed mesh. Upon closer examination one finds that the monkey head 'Suzanne' is an open mesh. Although the smoothing process succeeds, quad-net resolution fails for this open surface.

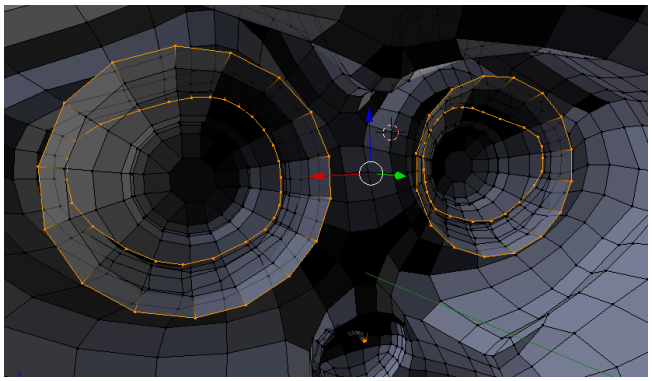

Figure 7: Inside of 'Suzanne'. Upon close examination one recognizes the mesh is open.

Quartic Triangular Bèzier Patch Construction

The final step of the construction creates a collection of triangular patch control points, from which unique equations for quartic triangular Bèzier patches are calculated. From these patches the Gaussian curvature is calculated and colored.

Quartic triangular Bèzier patches are calculated using the equation ($n = 4$):

$$\boldsymbol{\sigma}(s,t,u) = \sum_{\substack{i+j+k=n \\ i,j,k\geq 0}} \frac{n!}{i!j!k!} s^i t^j u^k \alpha^i \beta^j \gamma^k \quad s+t+u=1$$

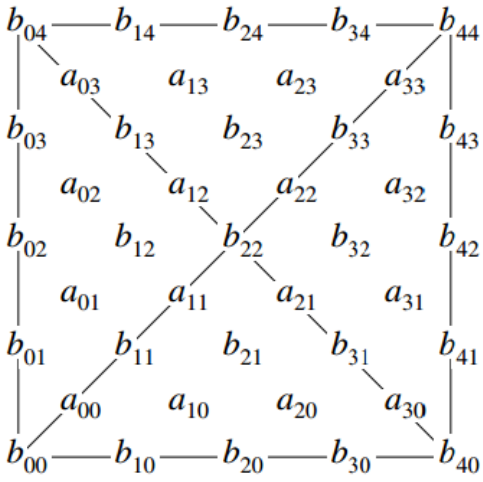

Figure 8: The 4 triangular Bèzier patches on a quad-net. [4]

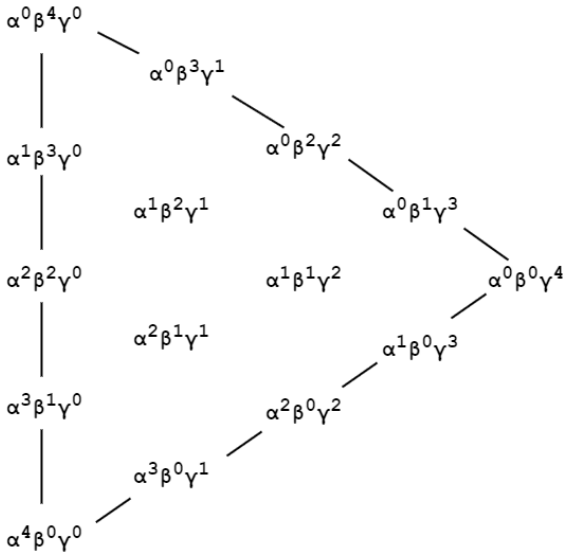

Figure 9: Application of the above formula to the left triangular Bèzier patch in Figure 8

One notes that the above formula is equivalently expressed in the usual u, v parametrization as:

$$\boldsymbol{\sigma}(u,v) = \sum_{\substack{i+j+k=n \\ i,j,k\geq 0}} \frac{n!}{i!j!k!}(1-u-v)^i u^j v^k \alpha^i \beta^j \gamma^k \quad 0 \leq u, v \quad u+v \leq 1$$

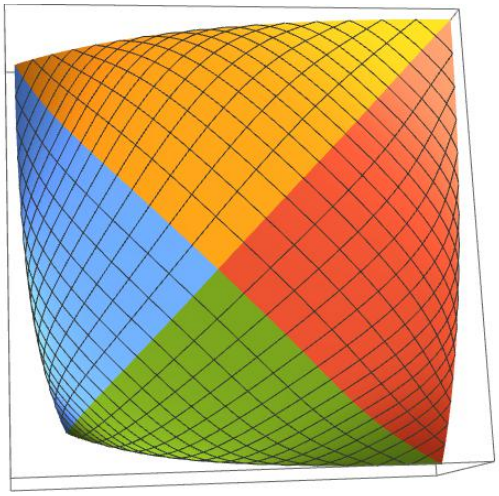

Figure 10: Mathematica Plot of Quartic Triangular Bèzier patches from a cubic Blender mesh

Coloring Quartic Bèzier Patches:

⟨add explanation of coloring process here⟩

Software:

Blender, Python 3.5.1

Images/animations produced:

⟨Add images and animations along with explanations here.⟩

Relevant concepts/equations/formulas covered in class:

1. The unit normal to a surface patch $\boldsymbol{\sigma}$ used to calculate the Gaussian curvature.

$$\vec{n} = \frac{\boldsymbol{\sigma}_u \times \boldsymbol{\sigma}_v}{\|\boldsymbol{\sigma}_u \times \boldsymbol{\sigma}_v\|}$$

2. The first fundamental form coefficient used to calculate the Gaussian curvature.

$$E = \boldsymbol{\sigma}_u \cdot \boldsymbol{\sigma}_u$$

3. The first fundamental form coefficient used to calculate the Gaussian curvature.

$$F = \boldsymbol{\sigma}_u \cdot \boldsymbol{\sigma}_v$$

4. The first fundamental form coefficient used to calculate the Gaussian curvature.

$$G = \boldsymbol{\sigma}_v \cdot \boldsymbol{\sigma}_v$$

5. The second fundamental form coefficient used to calculate the Gaussian curvature.

$$L = \boldsymbol{\sigma}_{uu} \cdot \vec{n}$$

6. The second fundamental form coefficient used to calculate the Gaussian curvature.

$$M = \boldsymbol{\sigma}_{uv} \cdot \vec{n}$$

7. The second fundamental form coefficient used to calculate the Gaussian curvature.

$$N = \boldsymbol{\sigma}_{vv} \cdot \vec{n}$$

8. The Gaussian curvature.

$$K = \frac{LN - M^2}{EG - F^2}$$

9. The formulation of quartic Bèzier patches for control points $\alpha^i, \beta^j, \gamma^k$

$$\boldsymbol{\sigma}(u,v) = \sum_{\substack{i+j+k=n \\ i,j,k \geq 0}} \frac{n!}{i!j!k!}(1-u-v)^i u^j v^k \alpha^i \beta^j \gamma^k \quad 0 \leq u, v \quad u+v \leq 1$$

10. For each quartic patch this is in the form:

$$\begin{aligned}\boldsymbol{\sigma}(u,v) = {} & b_{00}(1-u-v)^4 + 4b_{01}(1-u-v)^3 u + 6b_{02}(1-u-v)^2 u^2 + 4b_{03}(1-u-v)u^3 \\ & + b_{04}u^4 + 4a_{03}u^3 v + 6b_{13}u^2 v^2 + 4a_{12}uv^3 + b_{22}v^4 + 4a_{11}(1-u-v)v^3 \\ & + 6b_{11}(1-u-v)^2 v^2 + 4a_{00}(1-u-v)^3 v + 12a_{02}(1-u-v)u^2 v \\ & + 12a_{01}(1-u-v)^2 uv + 12b_{12}(1-u-v)uv^2\end{aligned}$$

where $0 \leq u, v \quad u+v \leq 1$

References

[1] *Blender*, 2016. `https://www.blender.org`.

[2] *Draw.io*, 2016. `https://www.draw.io/`.

[3] *Python 3.5.1*, 2016. `https://www.python.org/downloads/release/python-351/`.

[4] Charles Loop. Smooth spline surfaces over irregular meshes. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 303–310. ACM, 1994.