

Assignment 2

Andrei Bunea, Student Number: 18009149

Attempted Questions

1. Having a 3D scan, determine the Uniform Laplacian operator that is later used to determine the mean curvature H. Implement the Gaussian Curvature K by using a barycentric cells approach. Visualize the results.
2. Determine the First and Second fundamental forms for an ellipsoid.
3. Implement the Discrete Laplace-Beltrami operator and visualize the mean curvature.
4. Compute the spectral analysis for different amounts of eigenvectors.
5. Implement explicit Laplacian mesh smoothing
6. Implement implicit Laplacian mesh smoothing
7. Evaluate the performances of Laplacian denoising by adding different amounts of synthetic noise to the 3D models

Goals Achieved

I managed to successfully implement all of the tasks.

Solutions

Question 1

The Uniform Laplacian matrix was determined using the following equation:

$$\Delta_{\text{uni}} \mathbf{x}_i := \frac{1}{|\mathcal{N}_1(v_i)|} \sum_{v_j \in \mathcal{N}_1(v_i)} (\mathbf{x}_j - \mathbf{x}_i)$$

The mean curvature (H) was determined using the following equation:

$$H = \frac{1}{2} \|\Delta_S \mathbf{x}\|$$

The Gaussian Curvature (K) was determined using the following equation:

$$K = (2\pi - \sum_j \theta_j) / A$$

For the area calculus, I used the Barycentric Cells method. Therefore, for every vertex, I determined the area of every triangulated face around the vertex and the division by 3 I obtained the final A.

After computing the H and K in this manner I get the following results:

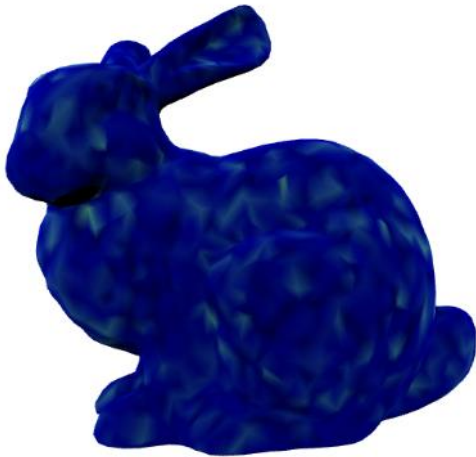


Figure 1 Mean Curvature for bunny

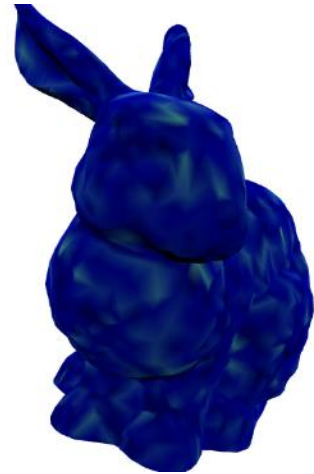


Figure 2 Mean Curvature for bunny(2)

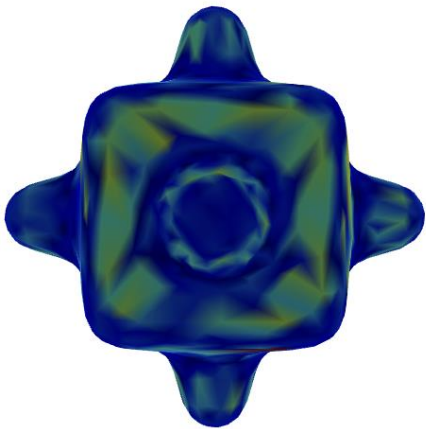


Figure 3 Mean Curvature bumpy-cube

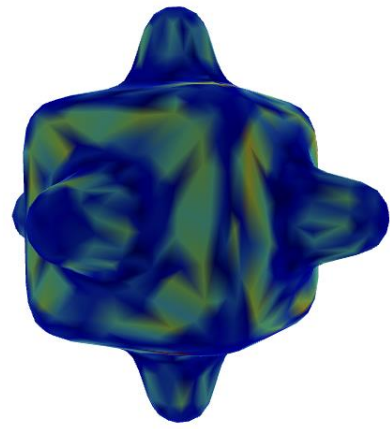


Figure 4 Mean Curvature bumpy-cube(2)

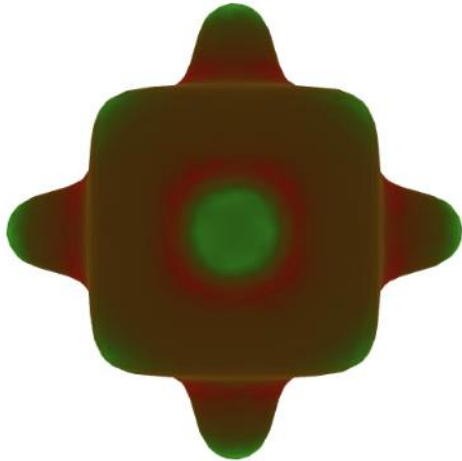


Figure 6 Gaussian Curvature bumpy-cube

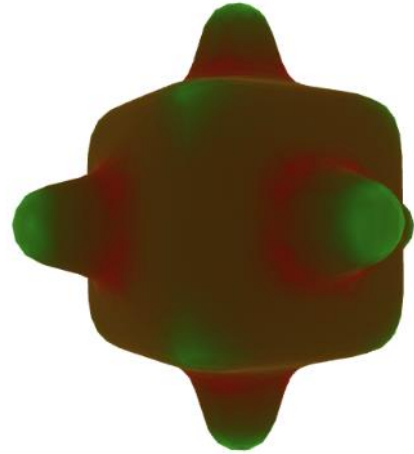


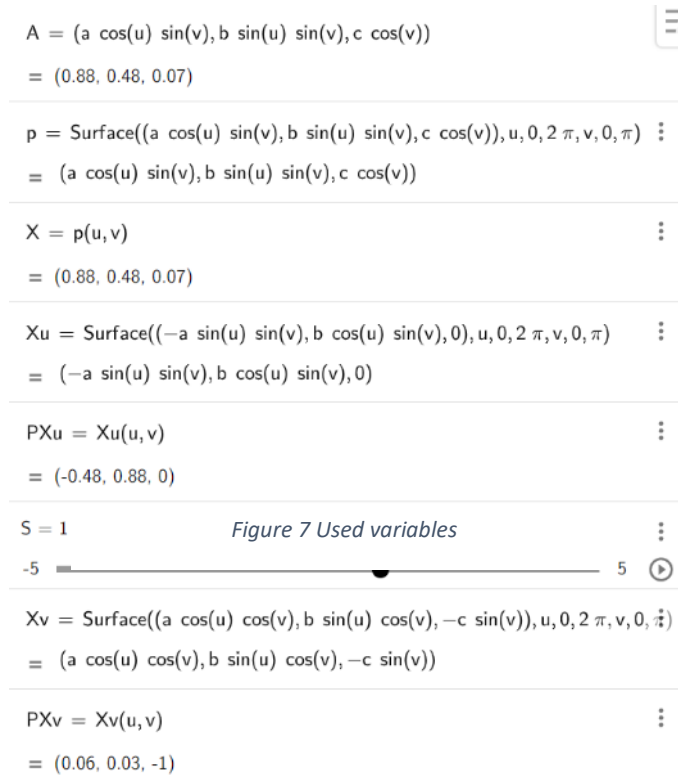
Figure 5 Gaussian Curvature bumpy-cube(2)

From the results, he can observe that Mean Curvature produces good results, especially for the bunny object. We can see the blue areas that record a higher curvature and matches the reality of the bunny model. Looking at the Mean Curvature for bumpy-cube we can observe that the algorithm is not consistent of flat surfaces. However, the Gaussian Curvature has better results for the continuous curvatures of the bumpy cube, colouring with green exactly the corners and the apexes from each face. Having these results, I would say that the Gaussian Curvature is better for the approximation of the continuous curve, the Mean Curvature produces decent results for areas with high curvatures but not so good results when applied to uniform surfaces.

Question 2

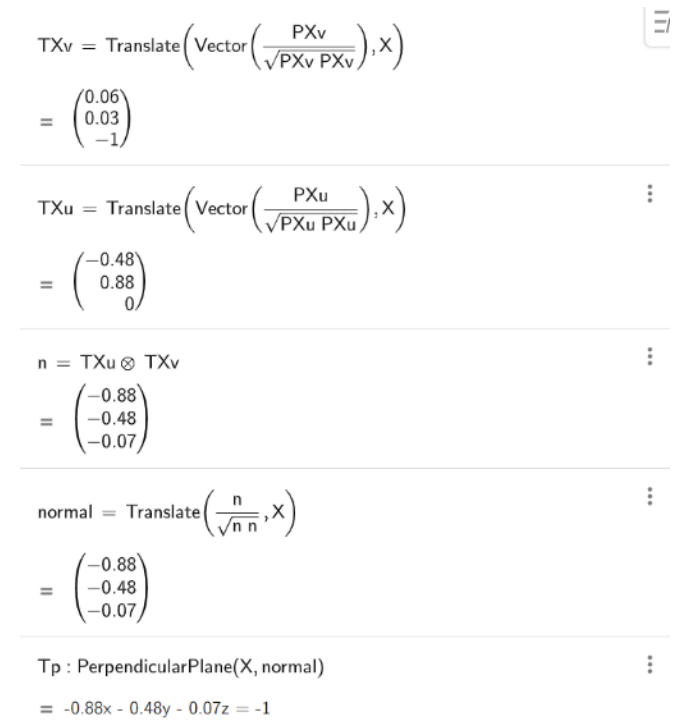
To prove my solution I am inserting screenshots from GeoGebra, while also showing the equations used.

First of all, the following screenshot presents the values of the used variables:



$$\begin{aligned}
 A &= (a \cos(u) \sin(v), b \sin(u) \sin(v), c \cos(v)) \\
 &= (0.88, 0.48, 0.07) \\
 p &= \text{Surface}((a \cos(u) \sin(v), b \sin(u) \sin(v), c \cos(v)), u, 0, 2\pi, v, 0, \pi) \\
 &= (a \cos(u) \sin(v), b \sin(u) \sin(v), c \cos(v)) \\
 X &= p(u, v) \\
 &= (0.88, 0.48, 0.07) \\
 Xu &= \text{Surface}((-a \sin(u) \sin(v), b \cos(u) \sin(v), 0), u, 0, 2\pi, v, 0, \pi) \\
 &= (-a \sin(u) \sin(v), b \cos(u) \sin(v), 0) \\
 PXu &= Xu(u, v) \\
 &= (-0.48, 0.88, 0) \\
 S &= 1 \quad \text{Figure 7 Used variables} \\
 -5 & \text{ ————— 5} \\
 Xv &= \text{Surface}(a \cos(u) \cos(v), b \sin(u) \cos(v), -c \sin(v), u, 0, 2\pi, v, 0, \pi) \\
 &= (a \cos(u) \cos(v), b \sin(u) \cos(v), -c \sin(v)) \\
 PXv &= Xv(u, v) \\
 &= (0.06, 0.03, -1)
 \end{aligned}$$

Figure 9 Equations for Ellipsoid, Xu and Xv



$$\begin{aligned}
 TXv &= \text{Translate}\left(\text{Vector}\left(\frac{PXv}{\sqrt{PXv \cdot PXv}}\right), X\right) \\
 &= \begin{pmatrix} 0.06 \\ 0.03 \\ -1 \end{pmatrix} \\
 TXu &= \text{Translate}\left(\text{Vector}\left(\frac{PXu}{\sqrt{PXu \cdot PXu}}\right), X\right) \\
 &= \begin{pmatrix} -0.48 \\ 0.88 \\ 0 \end{pmatrix} \\
 n &= TXu \otimes TXv \\
 &= \begin{pmatrix} -0.88 \\ -0.48 \\ -0.07 \end{pmatrix} \\
 \text{normal} &= \text{Translate}\left(\frac{n}{\sqrt{n \cdot n}}, X\right) \\
 &= \begin{pmatrix} -0.88 \\ -0.48 \\ -0.07 \end{pmatrix} \\
 Tp &: \text{PerpendicularPlane}(X, \text{normal}) \\
 &= -0.88x - 0.48y - 0.07z = -1
 \end{aligned}$$

Figure 8 Equations for normal and le

$$\begin{aligned}
 If &= PX_u PX_v \\
 &= 0 \\
 Ig &= PX_v PX_v \\
 &= 1 \\
 P &= -4.9 \\
 text1 &= "I = \begin{vmatrix} 0.99 & 0 \\ 0 & 1 \end{vmatrix}" \\
 Xu_u &= Surface((-a \cos(u) \sin(v), -b \sin(u) \sin(v), 0), u, 0, 2\pi, v, 0, \pi) \\
 &= (-a \cos(u) \sin(v), -b \sin(u) \sin(v), 0) \\
 PX_{uu} &= Xu_u(u, v) \\
 &= (-0.88, -0.48, 0) \\
 Xv_v &= Surface((-a \cos(u) \sin(v), -b \sin(u) \sin(v), -c \cos(c)), u, 0, 2\pi, v, 0, \pi) \\
 &= (-a \cos(u) \sin(v), -b \sin(u) \sin(v), -c \cos(c)) \\
 PX_{vv} &= Xv_v(u, v) \\
 &= (-0.88, -0.48, -0.54)
 \end{aligned}$$

Figure 10 Equations for If , Ig , Xu_u , Xv_v

$$\begin{aligned}
 Xu_v &= Surface((-a \sin(u) \cos(v), b \cos(u) \cos(v), 0), u, 0, 2\pi, v, 0, \pi) \\
 &= (-a \sin(u) \cos(v), b \cos(u) \cos(v), 0) \\
 PX_{uv} &= Xu_v(u, v) \\
 &= (-0.03, 0.06, 0) \\
 L &= PX_{uu} \text{ normal} \\
 &= 0.99 \\
 M &= PX_{uv} \text{ normal} \\
 &= 0 \\
 N &= PX_{vv} \text{ normal} \\
 &= 1.03 \\
 text2 &= "II = \begin{vmatrix} |cc| \end{vmatrix}" + (FormulaText(L)) + "&" + \\
 &+ (FormulaText(M)) + "\\" + (FormulaText(M)) + "&" + \\
 &+ (FormulaText(N)) + " \end{vmatrix}"
 \end{aligned}$$

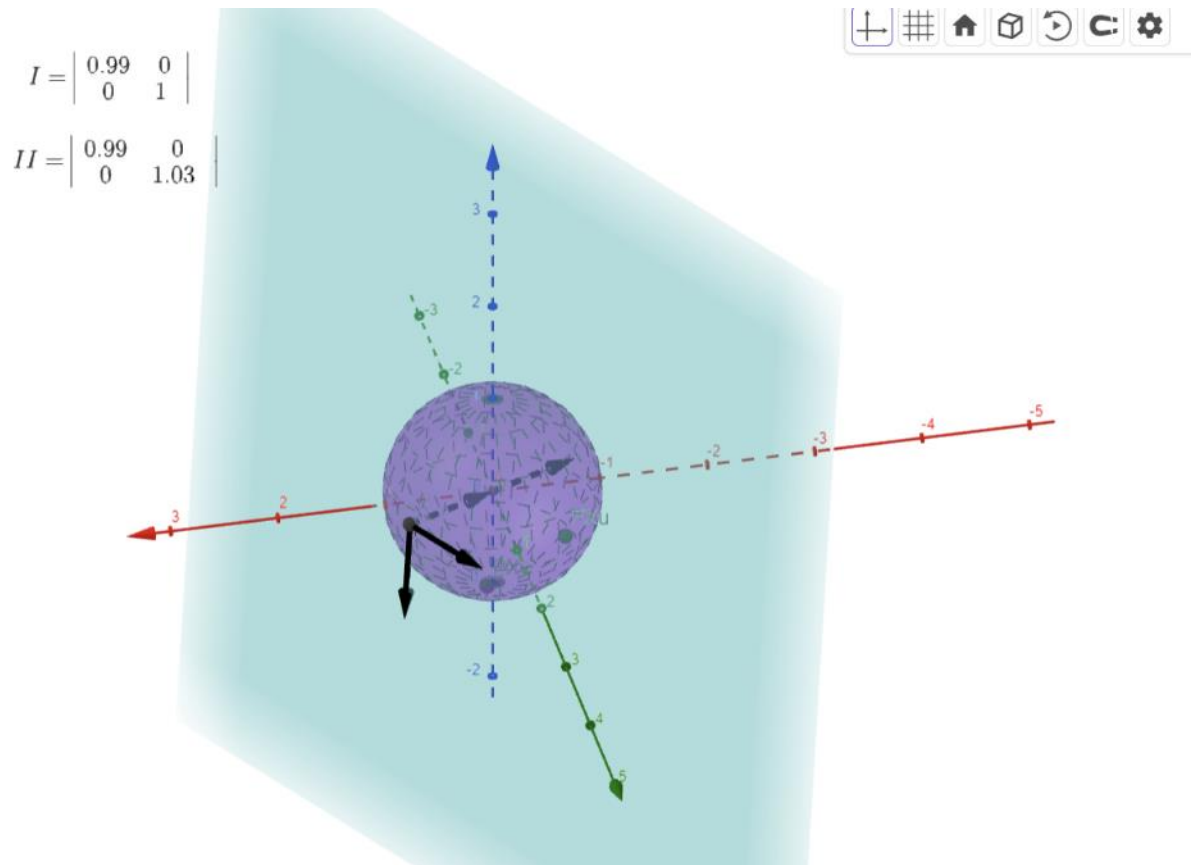
Figure 11 Equations for L , M , N 

Figure 12 Representation and values for the First and Second Fundamental Forms

Question 3

To compute the Discrete Laplace-Beltrami equation I used the following equations:

$$\mathbf{L} = \mathbf{M}^{-1} \mathbf{C} \in \mathbb{R}^{n \times n}$$

$$\mathbf{C}_{ij} = \begin{cases} \cot \alpha_{ij} + \cot \beta_{ij}, \\ -\sum_{v_j \in \mathcal{N}_1(v_i)} (\cot \alpha_{ij} + \cot \beta_{ij}) \\ 0 \end{cases}$$

$$\mathbf{M}^{-1} = \text{diag} \left(\dots, \frac{1}{2A_i}, \dots \right)$$

The area is determined using Barycentric Cells as previously described. The Mean Curvature H is determined as previously described for Question 1.

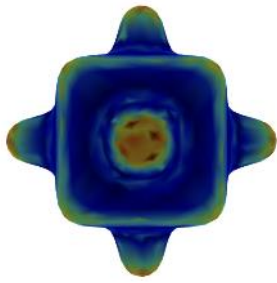


Figure 14 DLB results for bumpy-cube

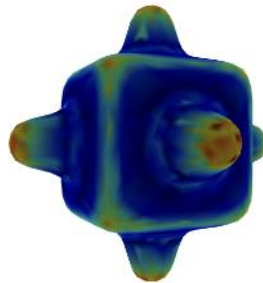


Figure 13 DLB results for bunny

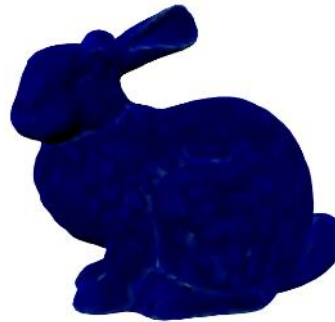


Figure 15 DLB results for lillium



Figure 16 DLB results for plane

Analysing these results I consider this method to have better performances than the Uniform Laplace method, as the resulting meshes are smoother in showing the curvature change. It is more stable in uniform areas.

Question 4

In order to achieve Modal Analysis I determine the Discrete Laplace Beltrami operator. Having this as a matrix I extract the “k” smallest eigenvalues and I get their corresponding eigenvectors. I use this vectors to reconstruct the mesh as shown in the equations below:

$$\begin{aligned} \mathbf{x} &:= [x_1, \dots, x_n] & \mathbf{y} &:= [y_1, \dots, y_n] & \mathbf{z} &:= [z_1, \dots, z_n] \\ \mathbf{x} &\leftarrow \sum_{i=1}^k (\mathbf{x}^T \mathbf{e}_i) \mathbf{e}_i & \mathbf{y} &\leftarrow \sum_{i=1}^k (\mathbf{y}^T \mathbf{e}_i) \mathbf{e}_i & \mathbf{z} &\leftarrow \sum_{i=1}^k (\mathbf{z}^T \mathbf{e}_i) \mathbf{e}_i \end{aligned}$$

Therefore I achieve the following results:

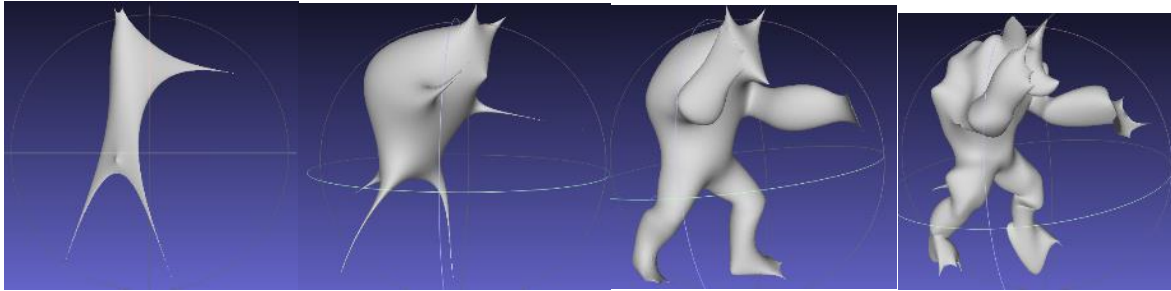


Figure 17 Spectral Analysis using a) $k=5$ b) $k=15$ c) $k=50$ d) $k=200$

It is clear that as the number of eigenvectors increases more details start to appear, therefore showing better results, results that get closer to the original object.

Question 5

In order to achieve Diffusion Flow on Meshes, first, I determine the Uniform Laplacian. After that, at each iteration the following equation is applied:

$$\mathbf{p}_i \leftarrow \mathbf{p}_i + \lambda \Delta \mathbf{p}_i$$

The algorithm produces the following results:

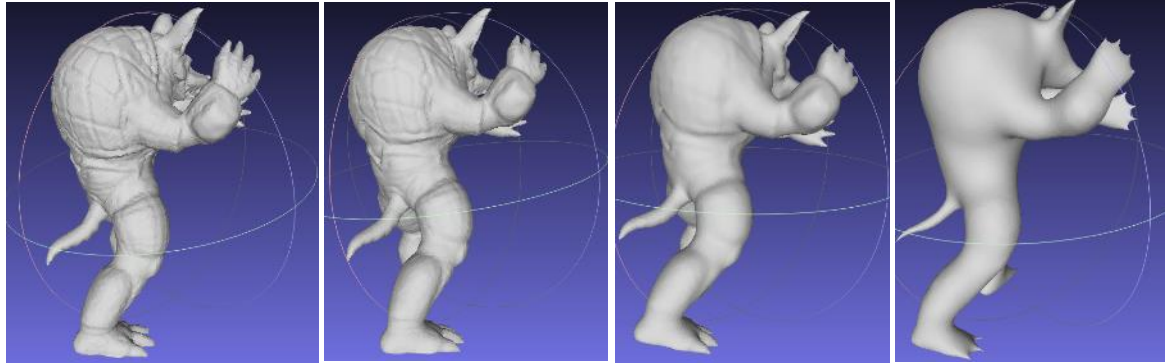


Figure 18 ($\lambda=0.01$, $iter=5$), ($\lambda=0.01$, $iter=50$), ($\lambda=0.5$, $iter=5$), ($\lambda=0.5$, $iter=50$)

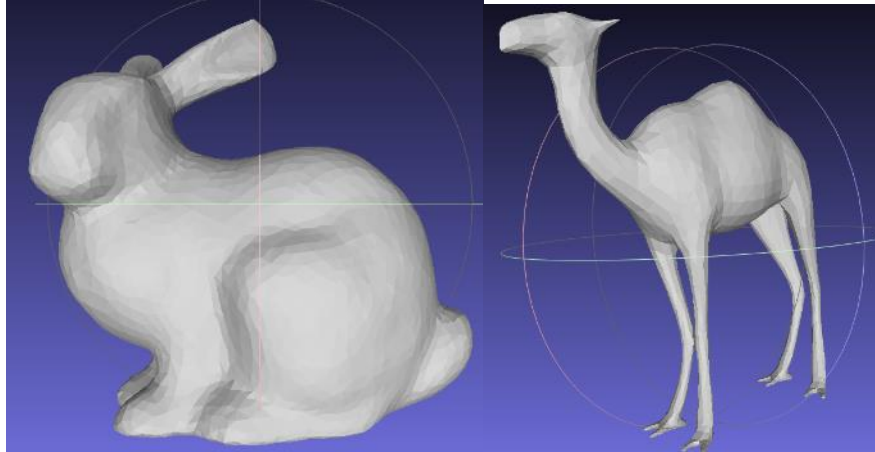


Figure 19 ($\lambda=0.1$, $iter=0.25$)

Using a higher λ or a higher number of iterations causes a higher smoothing effect. This is clear in the figure showing the armadillo results, as 0.5 λ produces smoother effects than 0.1 λ , as well as 50 iterations compared to 5 iterations. However, in the last armadillo picture, we can observe that using $\lambda=0.5$ and 50 iterations leads to losing the details as the mesh is too smooth. A good λ value that I found experimentally is 0.1 and the results can be seen in the figure above that depicts the bunny and camel meshes.

Question 6

To implement Implicit Laplacian Smoothing, the Discrete Laplace-Beltrami is needed. The results are determined using the following equation:

$$(\mathbf{I} - \lambda \mathbf{L})\mathbf{P}^{(t+1)} = \mathbf{P}^{(t)}$$

The algorithm produces the following results:

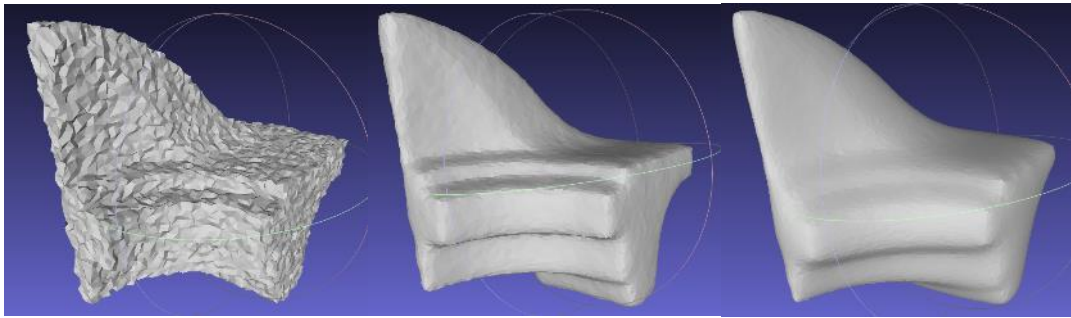


Figure 20 Fansidk with lambda 0.01, 0.1 and 0.5

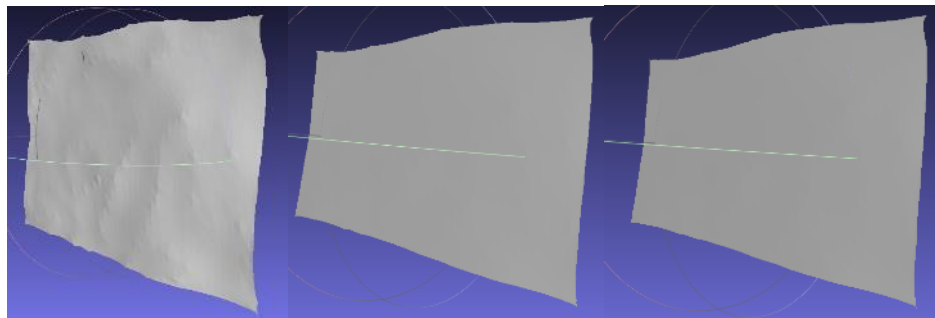


Figure 21 Plane with lambda 0.01, 0.1 and 0.5

Regarding the fandisk, the best results are obtained with lambda 0.1, as the 0.01 lambda doesn't remove the noise and the 0.5 lambda smooths the mesh too much eliminating important details.

Observing the plane results, as there are no visible important features that risk being removed following a higher lambda, we observe that the results for 0.1 and 0.5 lambdas are similar. Lambda 0.01 is smoothing too little the mesh. Therefore I consider the best value for lambda to be 0.1.

Question 7

To present the results I will present in parallel the results produced by both explicit and implicit methods.

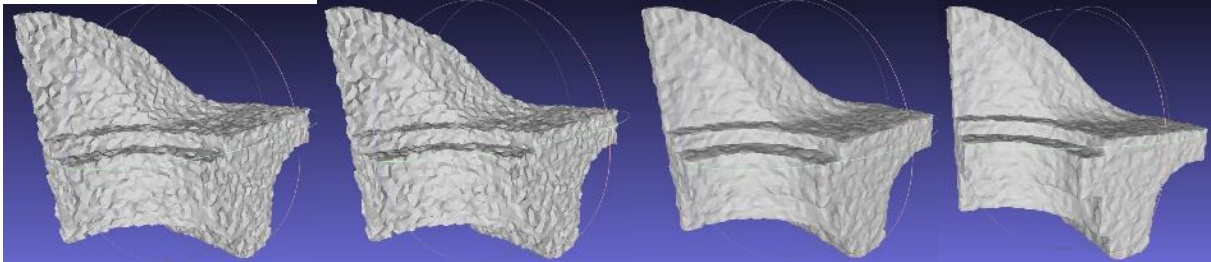


Figure 22 Explicit Smoothing: a) $\lambda=0.01$, $\text{noise}=0.0001$ b) $\lambda=0.1$, $\text{noise}=0.001$ c) $\lambda=0.5$, $\text{noise}=0.0001$ d) $\lambda=0.5$, $\text{noise}=0.001$

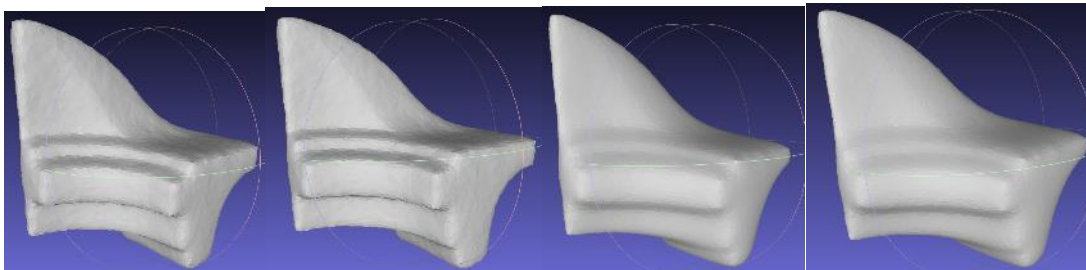


Figure 23 Implicit Smoothing: a) $\lambda=0.01$, $\text{noise}=0.0001$ b) $\lambda=0.1$, $\text{noise}=0.001$ c) $\lambda=0.5$, $\text{noise}=0.0001$ d) $\lambda=0.5$, $\text{noise}=0.001$

From the results, we can observe that implicit smoothing generates better results for every value of λ . However, using a bigger λ or a greater amount of noise leads to over-smooth important details when looking at the implicit smoothing results.

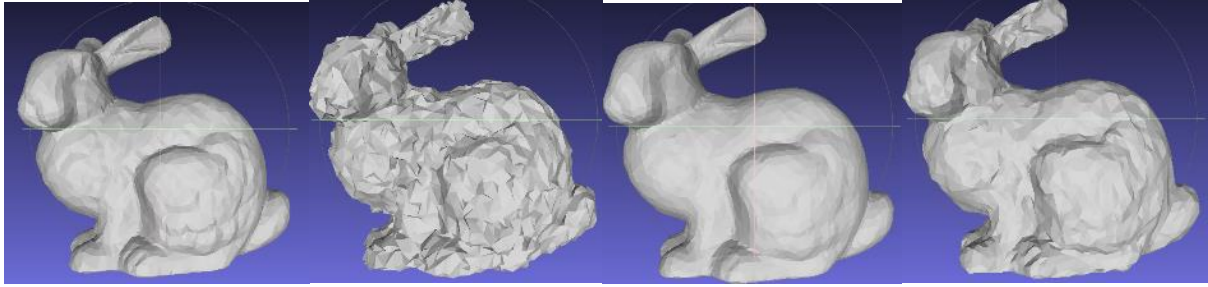


Figure 24 Explicit Smoothing: a) $\lambda=0.01$, $\text{noise}=0.0001$ b) $\lambda=0.1$, $\text{noise}=0.001$ c) $\lambda=0.5$, $\text{noise}=0.0001$ d) $\lambda=0.5$, $\text{noise}=0.001$

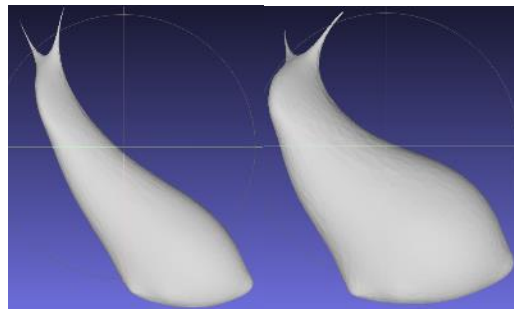


Figure 25 Implicit Smoothing: a) $\lambda=0.01$, $\text{noise}=0.0001$ b) $\lambda=0.01$, $\text{noise}=0.0001$

However, when the noise greatly affects the structure of the mesh, as presented in the results above, implicit smoothing, because it has a bigger smoothing effect, over-smooths the meshes leading to incorrect results.