5411 Assignment 1

Laplacian Smoothing

1. Explicit Smoothing
   1.1. Uniform Weighting Scheme

   Uniform Weighting, compared with the Cotangent one, is more stable in the sense that the weighting never becomes too large because of the instability that arises from computation of cot(α), instead, it remains 1 always. And the object being smoothed will quickly loss its original form unlike the cotangent one where the geometry features are somewhat better reserved.

   1.2. Cotangent Weighting Scheme

   From my observation, the Cotangent Weighting Scheme aligns with the geometry of the model more, thus it can be observed that it can be not "smooth" for some parts of the model even after smoothing, for example, the 6 directions normal to the faces of the "cube bumpy", and the ear of the "bunny", where the positions are all the "sharp points" of the objects, and such sharpness are mostly preserved. Also, since edge flipping has not been implemented, we still bear with the problem of precision of the double, where when an angle is large enough (~180 degrees), sin function on the angle produces a very small value, and it may be approximated to 0 by double, where the result is that the model disappears at a point when the weights go to NAN (divide by zero). For some model like the "bunny" it happens after around 60 iterations with $\lambda = 1$. For certain model like the "cube bumpy" it can make to 100+ iterations with $\lambda = 1$.

   Also, the Cotangent Scheme shrinks the models less given same amount of iterations compared with the Uniform one.

2. Implicit Smoothing
   2.1. Uniform Weighting Scheme

   This part is similar to 1.1.

   2.2. Cotangent Weighting Scheme

   This part is similar to 1.2.

   2.3. Comparison with Explicit Scheme

   Indeed, the Implicit Smoothing is slower in shrinking/smoothing the object compared with the explicit method in terms of number of iterations, possibly it is because the definition of "iteration" is quite different in the Eigen's BiCGSTAB algorithm. Another interesting thing is that when the "feline" object is concerned, the implicit scheme seems to work much better compared with

explicit one when cotangent weighting is used, since it is quite stable even after ~260 iterations.

3. Time

   All algorithms are similar in process time based on my human observation. The time measured by <chrono> on smoothing bunny is listed below:

Smoothing Scheme	Number of Iterations	Time
Explicit, Uniform	5 (I set it to execute 5 iterations for every update)	14524 microseconds
Explicit, Cotangent	5 (same as above)	19730 microseconds
Implicit, Uniform	14 (but also only one update)	12324 microseconds
Implicit, Cotangent	16 (but also only one update)	13420 microseconds

   Their performances are similar indeed, and it is reasonable that the Cotangent method generally takes longer time compared with the Uniform one. Also, since I set it to be for every update, 5 times of iterations are executed for Explicit Algorithms, they should be faster compared with the Implicit one, if we consider they are equivalent when the same entire process is only executed once, then it is 14524/5 vs 12324 and 19739/5 vs 13429.

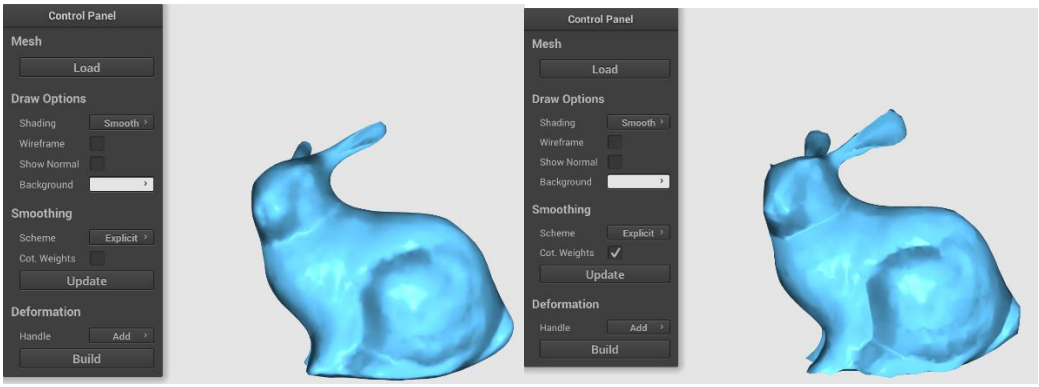
4. Additional Bug

   It can be noticed that a very interesting situation is presented in the program when switching between flat and smooth shading schemes. The trick is: Start with Flat shading, then do a couple of Smoothing Updates that the model shrinks (Suppose Taubin Smoothing is not in use) to a certain degree, and we notice that the shading of the model at the time looks not quite right (notice that the normal vectors are computed indeed after each update!), then we switch to Smooth shading, and switch back to Flat shading immediately afterwards, the appearance of the model changes completely compared with earlier before we switch to Smooth shading, which looks correct! This weird issue has been encountered by several classmates, and I am still investigating on what caused this problem.

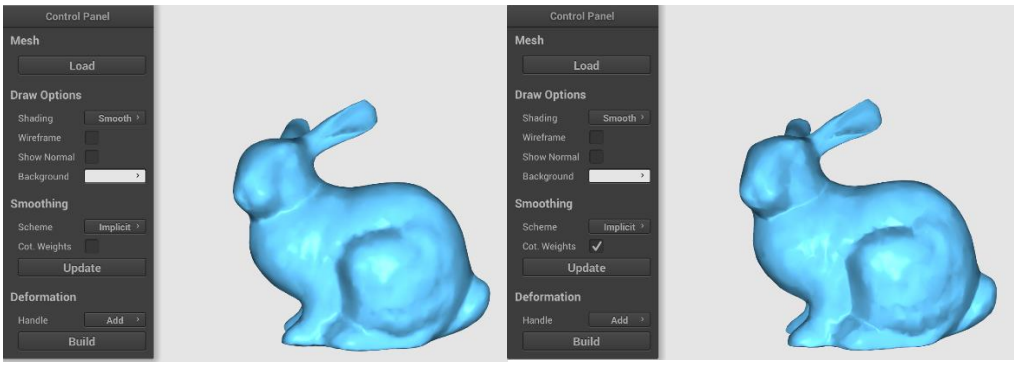
   On lambda, one special artifact appears for Implicit Smoothing, that is once the lambda value is smaller than 1, the entire object gets bigger on scale after each iteration, even if the smoothing is still applied. And the smaller the value, the bigger it gets after each iteration. And vice versa when the lambda is larger than 1, the object shrinks in contrary. This phenomena only appears for Implicit Smoothing, while the Explicit method stays normal.

5. Pictures

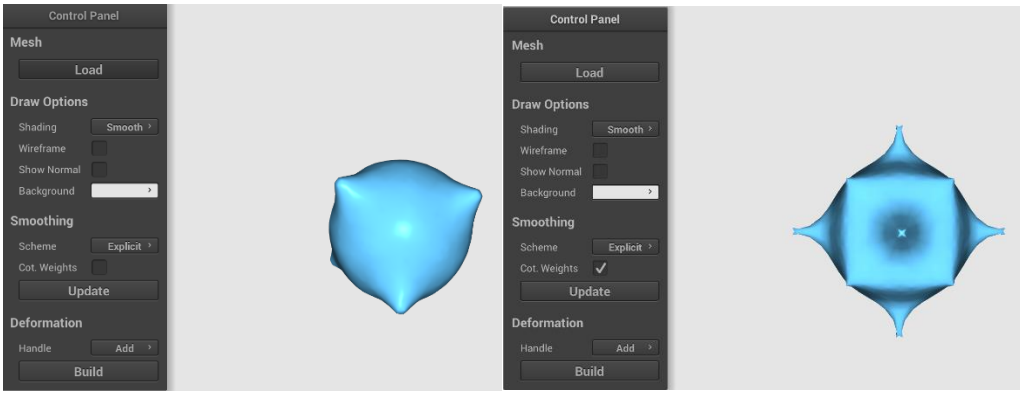
   Below are a few pictures of the three objects under different situations, more pictures are included in the zip file, as there are too many of them, the pictures that follows are using methods with λ=1:



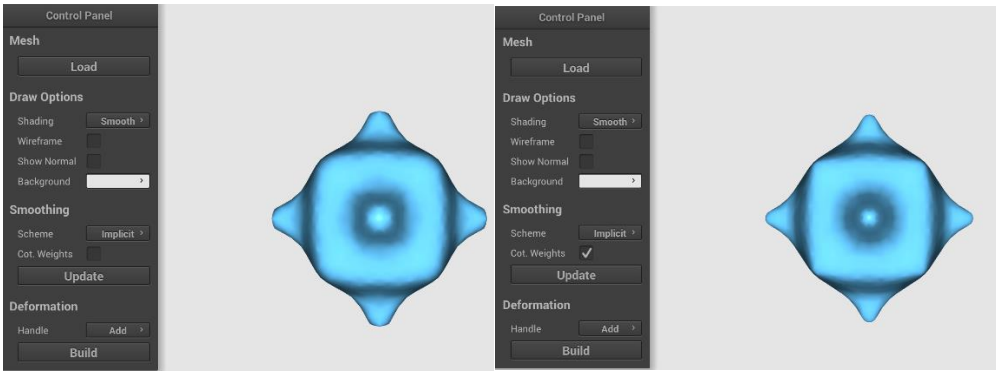
Left: Explicit Uniform(EU)-25 iterations. Right: Explicit Cotangent(EC)-25 iterations.



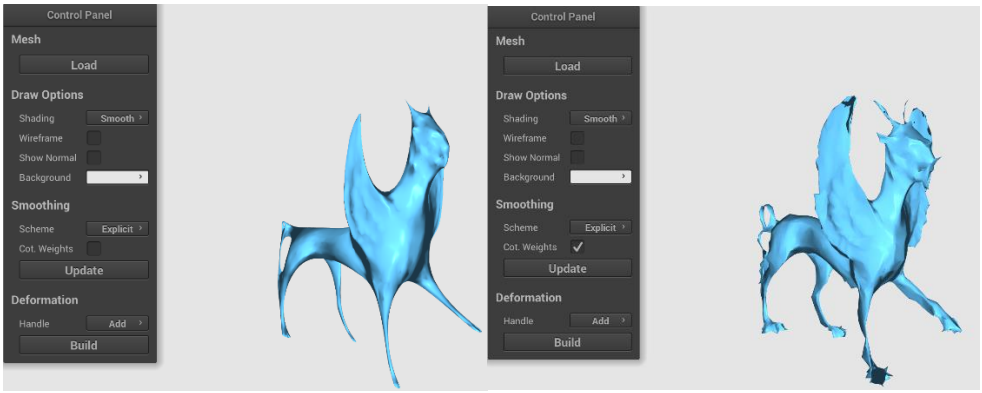
Left: Implicit Uniform(IU)-80 iterations. Right: Implicit Cotangent(IC)-93 iterations



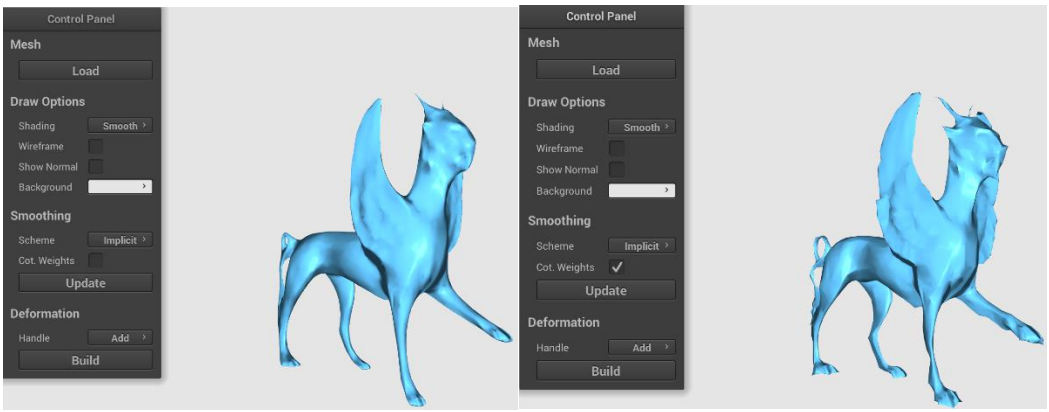
Left: EU-40 iterations. Right: EC-40 iterations



Left: IU-100 iterations. Right: IC-240 iterations



Left: EU-40 iterations. Right: EC-30 iterations(Which looks very bad, and the values at its sharp feet are starting to get out of control)



Left: IU-about 260 iterations. Right: IC-about 260 iterations.