Multiresolution 3D Model Surface Mesh Deformation using Naive Laplacian Editing

Xuhui Wang

November 2021

Abstract

3D mesh deformation, widely used in computer animation and modelling, refers to as the transformation of the surface of a 3D model under some user-defined controls and settings interactively. Requiring high computing cost, mesh deformation needs to take a considerably amount of time for computation of location and rotation for almost every vertex in a model. In this project, I introduce and implement a multiresolution 3D mesh deformation method using naive Laplacian editing that effectively optimizes the computation and time cost by removing the high-frequency details of the model at first and re-transferring high-frequency details back after the backbone deformation of the model.

Introduction

In this project, we will compute a mesh deformation based on the rotations and translations applied interactively to a subset of its vertices via the user input. (Note that, in this report, the wordings that describe the algorithms are all from the course material by Dr. Schneider and I will give the appropriate citation at the end of each corresponding paragraph. On the other hand, all resulted figures, methods implementation, statistics, analysis, explanations and conclusions are all from my personal work.) Let **H** be the set of "handle" vertices that the user can manipulate (or leave fixed). We want to compute a deformation for the remaining vertices, denoted as **R**.

Let **S** be our input surface, represented as a triangle mesh. We want to compute a new surface that contains:

1. the vertices in **H** translated/rotated using the user-provided transformation t, and
2. the vertices in **R** properly deformed using the algorithm described next.

The algorithm is divided in three phases:

1. removing high-frequency details,
2. deforming the smooth mesh, and
3. transferring high-frequency details to the deformed surface [1].

The general overview of this work flow could be described as in Figure 1.

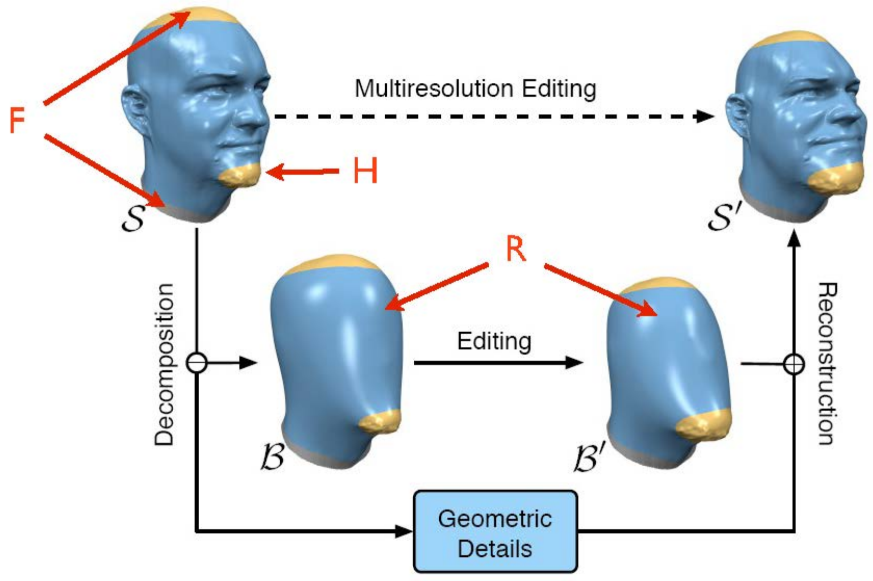


Figure 1: multiresolution mesh deformation using naive Laplacian editing [1]

Selecting the handles

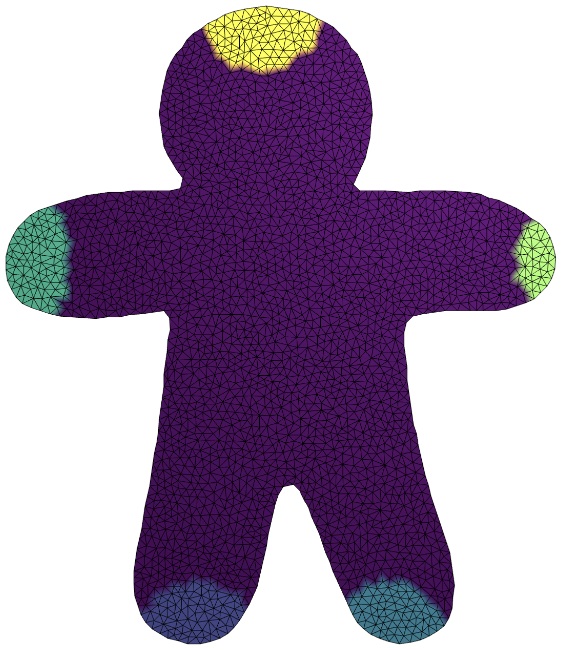
I have provided the handle vertices and corresponding segments for the ***woody*** object with which you could directly specify the displacement and rotation, and interactively deform the 3D model by running ***Deformation***. Alternately you could run ***Selection*** to specify your own user-defined handles.

Removal of high-frequency details

We remove the high-frequency details from the vertices $\mathbf{R}$ in $\mathbf{S}$ by minimizing the thin-plate energy, which involves solving a bi-Laplacian system arising from the quadratic energy minimization:

$$\underset{\mathbf{v}}{min}\{\mathbf{v^T L_\omega M^{-1} L_\omega v}\} \quad \text{s.t.} \quad \mathbf{v}_H = \mathbf{o}_H$$

where $\mathbf{o}_H$ are the handle 's vertex positions, $\mathbf{L}_\omega$ is the cotan Laplacian of $\mathbf{S}$, and $\mathbf{M}$ is the mass matrix of $\mathbf{S}$. Notice that $\mathbf{L}_\omega$ is the symmetric matrix consisting of the cotangent weights ONLY (without the division by Voronoi areas). In other words, it evaluates an "integrated" Laplacian rather than an "averaged" laplacian when applied to a vector of vertices. The inverse mass matrix appearing in the formula above then applies the appropriate rescaling so that the laplacian operator can be applied again (i.e., so that the Laplacian value computed at each vertex can be interpreted as a piecewise linear scalar field whose Laplacian can be computed). Considering the ***woody*** object as an example, this optimization will produce a mesh in Figure 2. Note that the part of the surface that we want to deform is now free of high-frequency details. We call this mesh $\mathbf{B}$ [1].


(a) Input mesh of woody object

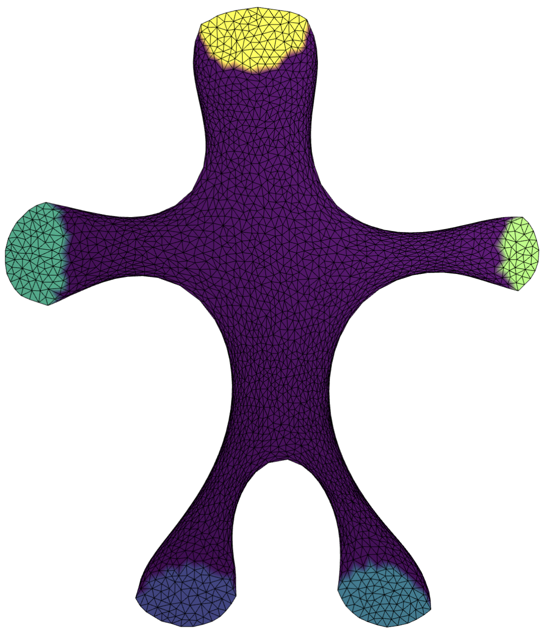

(b) Smoothed mesh of woody object

Figure 2: Input and Smoothed Meshes

Deforming the smooth mesh

The new deformed mesh is computed similarly to the previous step, by solving the minimization:

$$\underset{\mathbf{v}}{min}\{\mathbf{v^T}\mathbf{L}_\omega\mathbf{M^{-1}}\mathbf{L}_\omega\mathbf{v}\} \quad \text{s.t.} \quad \mathbf{v}_H = t(\mathbf{o}_H)$$

where $t(\mathbf{o}_H)$ are the new handle vertex positions after applying the user's transformation. We call this mesh $\mathbf{B'}$ as in Figure 3.

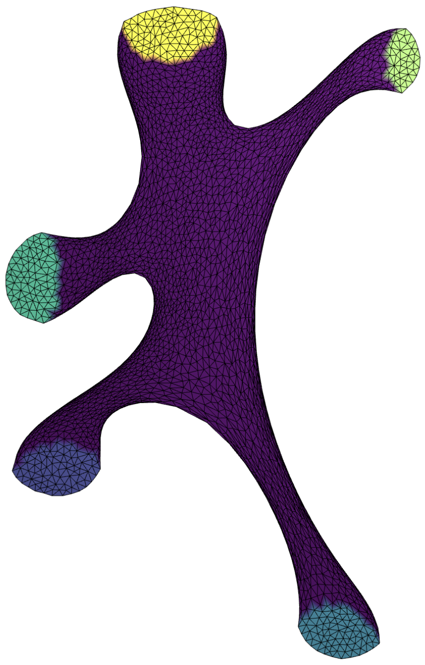


Figure 3: Deformed/Smoothed Mesh of woody object

Transferring high-frequency details to the deformed surface

The high-frequency details on the original surface are extracted from $\mathbf{S}$ and transferred to $\mathbf{B}'$. We first encode the high-frequency details of $\mathbf{S}$ as displacements w.r.t. $\mathbf{B}$. We define an orthogonal reference frame on every vertex v of $\mathbf{B}$ using:

1. The unit vertex normal
2. The normalized projection of one of v's outgoing edges onto the tangent plane defined by the vertex normal. A stable choice is the edge whose projection onto the tangent plane is longest.
3. The cross-product between (1) and (2) [1].

For every vertex v, we compute the displacement vector that takes v from $\mathbf{B}$ to $\mathbf{S}$ and represent it as a vector in v's reference frame. For every vertex of $\mathbf{B}'$, we also construct a reference frame using the normal and the SAME outgoing edge we selected for $\mathbf{B}$ (not the longest in $\mathbf{B}'$; it is important that the edges used to build both reference frames are the same). We can now use the displacement vector components computed in the previous paragraph to define transferred displacement vectors in the new reference frames of $\mathbf{B}'$. See Figure 3 for an example in the case of ***woody*** object. Applying the transferred displacements to the vertices of $\mathbf{B}'$ generates the final deformed mesh $\mathbf{S}'$ [1]. See Figure 4 for an example.

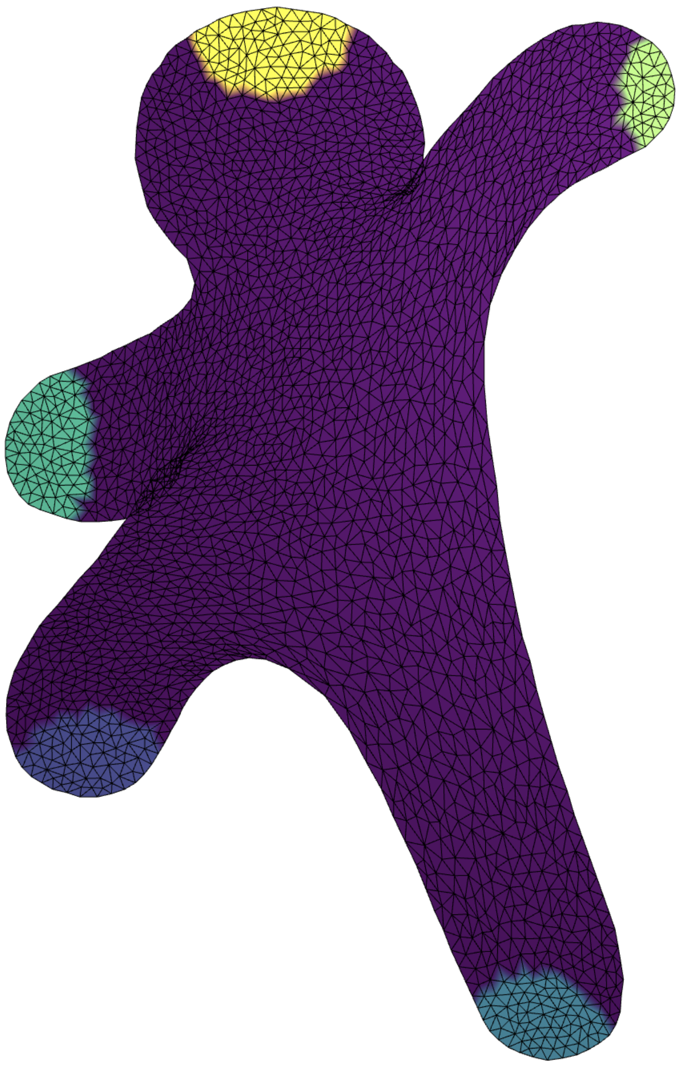


Figure 4: Final Deformation Results of woody object

Mesh Deformation in more objects

In this section, as another example, we interactively deform a ***hand*** 3D model with the above algorithm and show the rendering of different deformed meshes.

I have provided the handle vertices and corresponding segments for the ***hand*** object with which you could directly specify the displacement and rotation, and interactively deform the 3D model by running ***Deformation***. Alternately you could run ***Selection*** to specify your own user-defined handles.

Firstly, we show the original mesh in Figure 5.

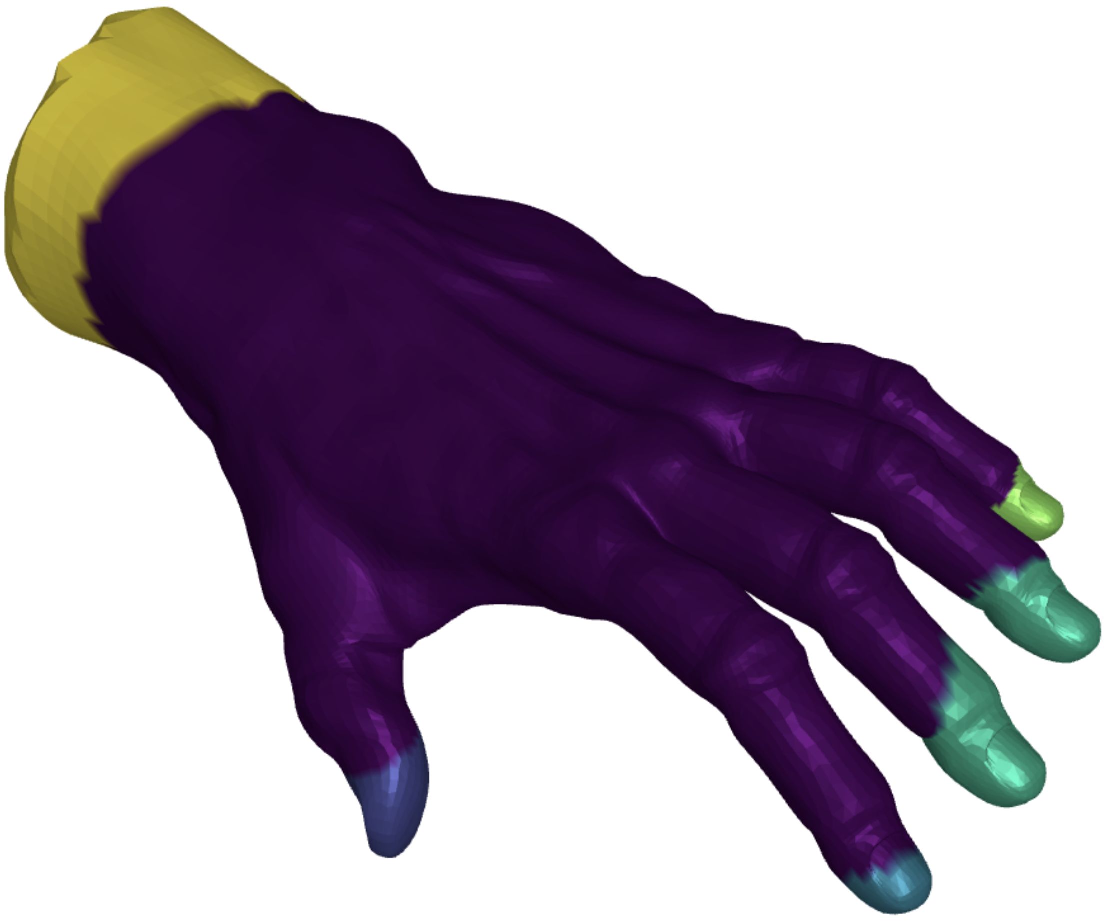


Figure 5: original mesh of hand object

After removing the high frequency features, we could obtain the smoothed mesh in Figure 6.

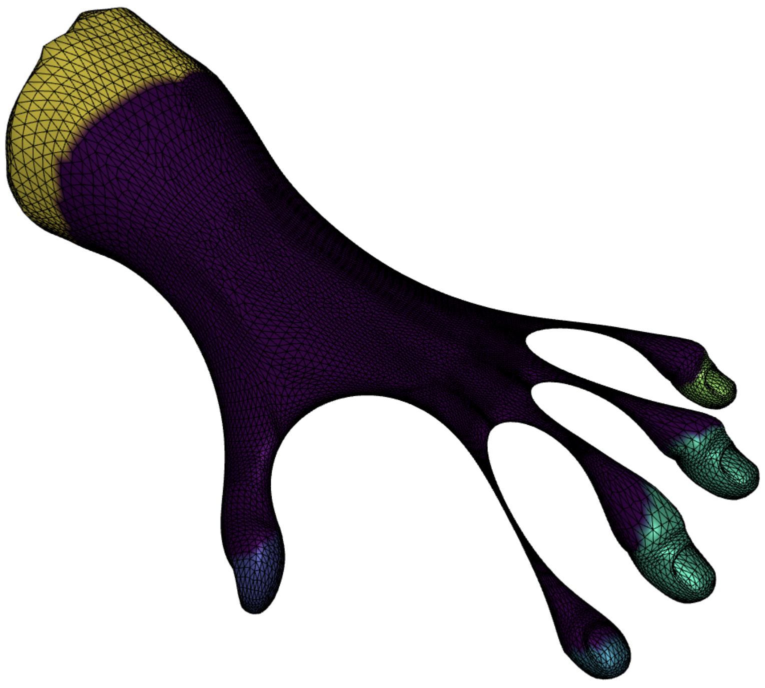


Figure 6: smoothed mesh of hand object without high frequency features

Then we could specify the translation and rotation given the user-defined vertex handles as shown in Figure 7. Here we try to deform the mesh to pose a ”rock hand”.

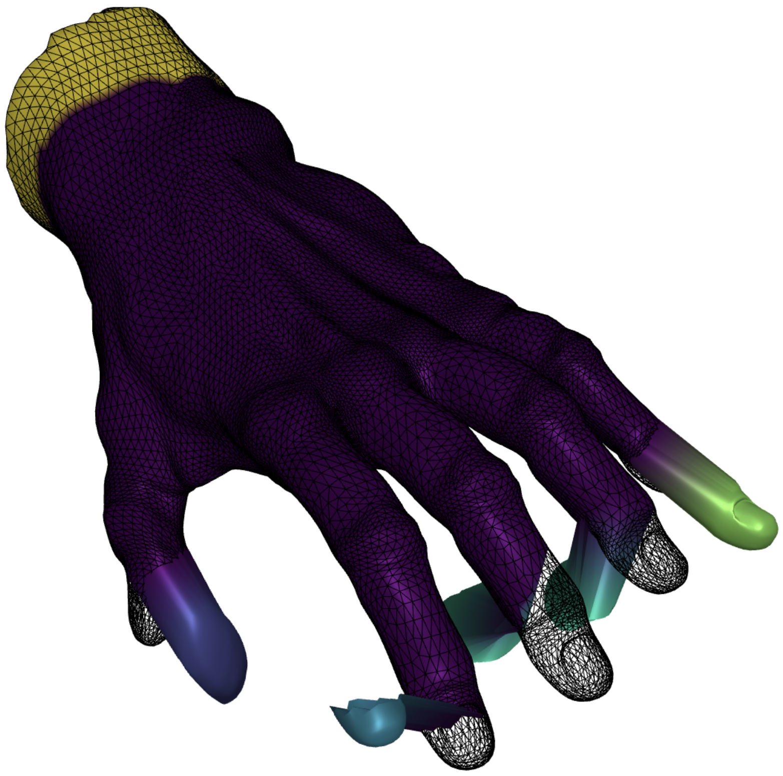


Figure 7: the deformed original mesh of hand object based on user defined constraints

In Figure 8, we show the smoothed mesh deformed without the high frequency features.

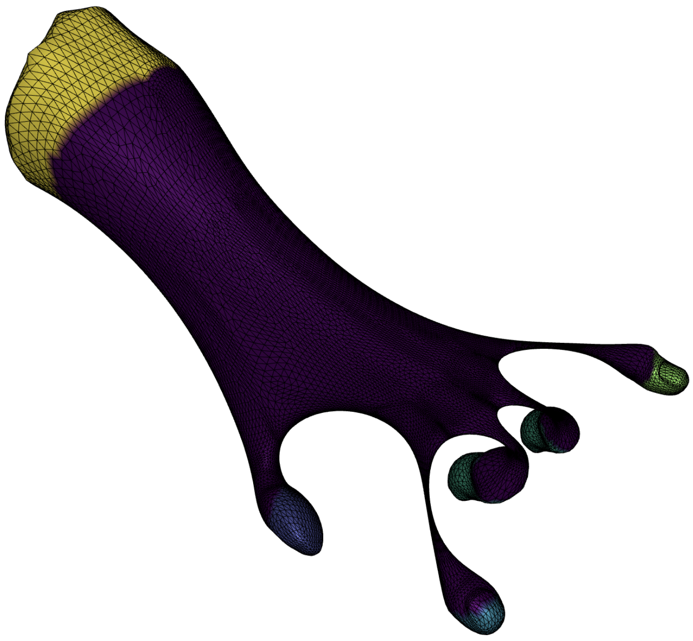


Figure 8: the smoothed mesh of hand object deformed without the high frequency features

Having transferred high-frequency details to the deformed surface, finally we could obtain the ultimate reconstructed mesh of the ***hand*** object. In Figure 9, we could clearly see the "rock hand" deformed from the original mesh.

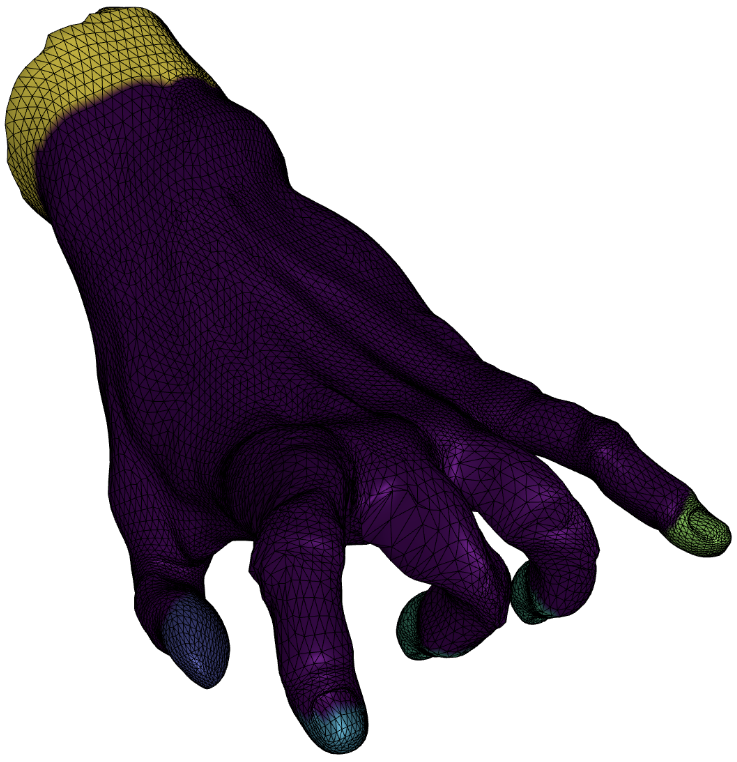


Figure 9: the ultimate reconstructed mesh of hand object

We view the final mesh from another perspective to see our deformation performance in Figure 10.

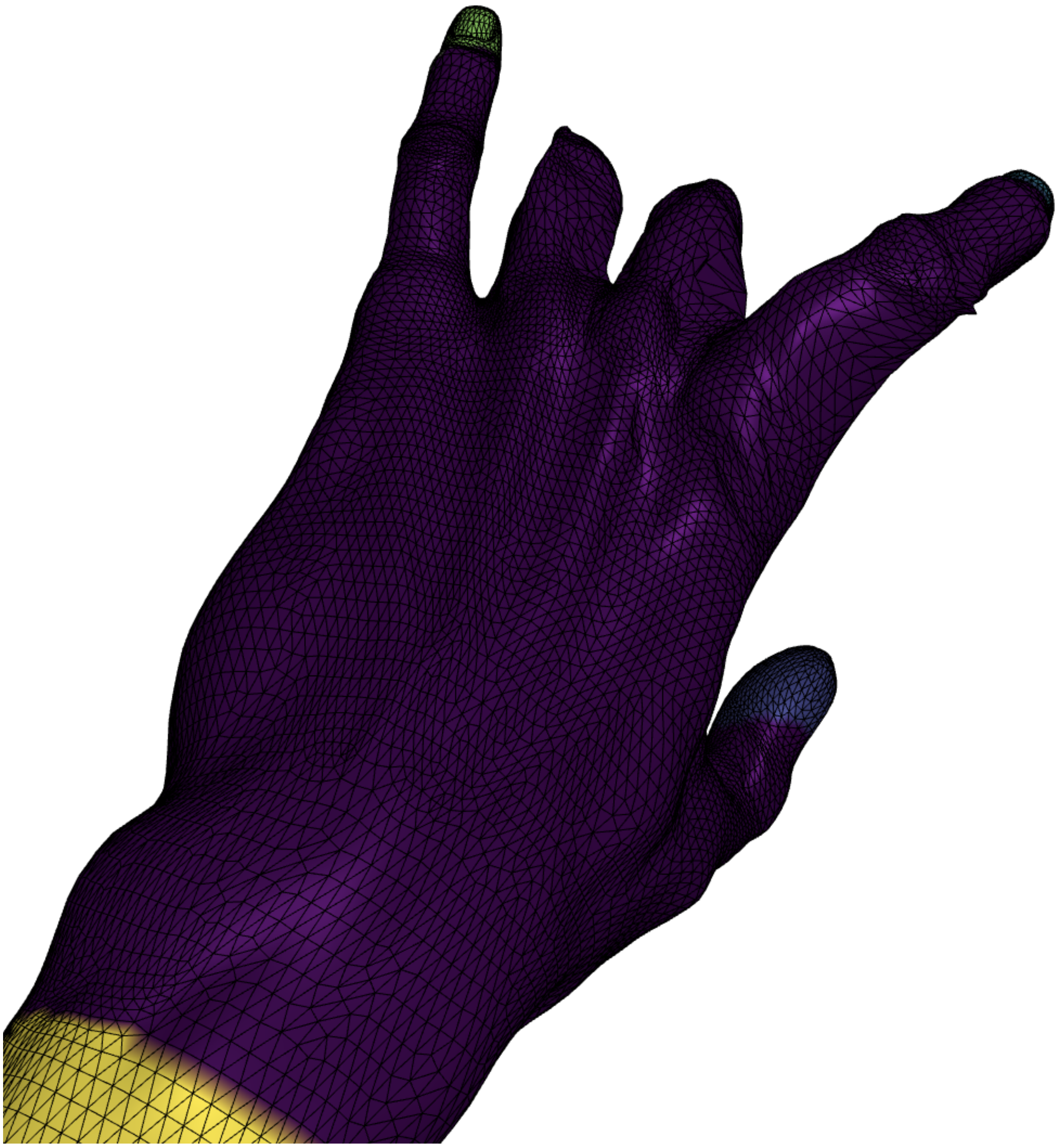


Figure 10: the ultimate reconstructed mesh of hand object

Conclusion

In this project, I introduced and implemented a multiresolution 3D mesh deformation method that effectively optimizes the computation and time cost by removing the high-frequencydetails of the model at first and re-transferring high-frequency details back after the backbone deformation ofthe model. In addition, I used two different models as examples to visually demonstrate the effectiveness of the method. The smooth and looking-natural models resulted after mesh deformation show that the multiresolution 3D mesh deformation method is an efficient and effective method to accomplish the tasks of computer model surface deformation in application.

References

[1] T. Schneider, *Csc486b: Geometric modelling course materials.*