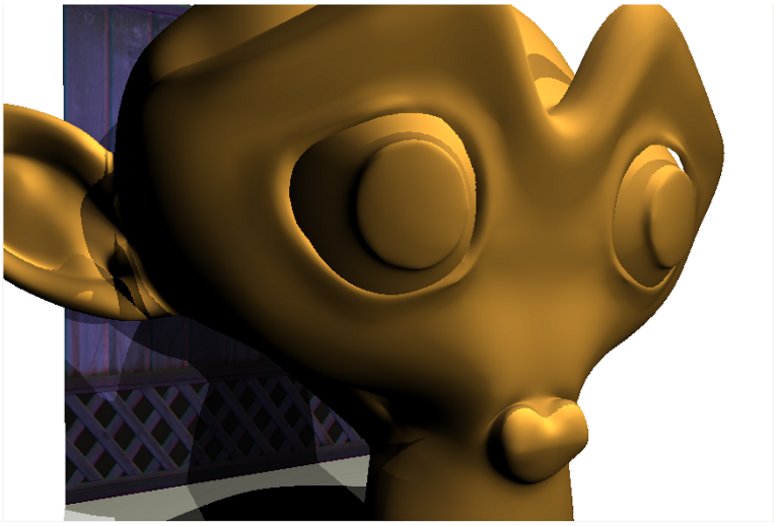
Loop Subdivision Practical Work

IGR202 - Computer Graphics & Virtual Reality



Lais Isabelle ALVES DOS SANTOS

January 2023

1 Introduction

The main purpose of this report is to briefly explain the implementation of Loop Subdivision, one of the most common subdivision schemes to smooth a surface. To compile and work as expected, the project files and directories were refactored from the provided ones. The following commands from the *src* directory should be done to properly set the compilation and running.

```
$ mkdir build
$ cmake -B build
$ cd build; make
$ cd ..; ./tpSubdiv
```

2 The Algorithm

The algorithm is divided in two parts. In general, it is needed to update the even vertices (the vertices that already exist in a nth iteration) and the odd vertices (the new created vertices according to neighbor even ones). The first part is responsible for updating the even vertices and the second one creates and updates the odd vertices.

The mesh is computed in triangle faces, so for each triangle it is computed its vertices and the vector *neighboringVertices* stores the set of neighbor vertices for each vertex. This is important to soon evaluate the valence of each vertex and use it to update the even vertices with the β calculation (in the code called α).

The even vertex is updated to smooth the mesh and avoid corners. Intuitively, the new value cannot be stored on the current *_vertexPositions* vector, because the current one will be needed to obtain the odd vertices. Hence, for each vertex, the following formula is applied with n being the number of triangles surrounding a vertex and k the valence:

$$v_{new} = v(1 - k\beta) + \sum_i u_i\beta$$

$$\beta = \begin{cases} \frac{3}{8n}, & \text{if } n > 3 \\ \frac{3}{16}, & \text{if } n = 3 \end{cases}$$

Subsequently, a structure *Edge* is used to assist the new edges created from two vertices. One odd vertex is placed in the half of a newly edge and the value of the third vertex is stored in the *oddVertices* vector, to be used in the next iteration and update the odd vertex. In case the edge already exists, the odd vertex will be updated according to the following equation, considering v_{odd} the vertex created from the mean of a newly edge in a previous iteration, v_c being the vertex stored in *oddVertices* from the first face on the edge and, finally, v_d being the third vertex of the current face on the edge:

$$v_{new} = \frac{3}{4}v_{odd} + \frac{1}{8}v_c + \frac{1}{8}v_d$$

The same procedure is applied to all edges in a triangle face and, at the end of the loop, the *_vertexPositions* is updated with the new vertices.

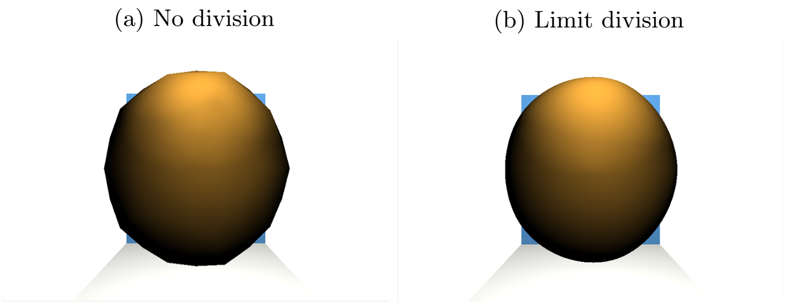
3 Results

The first sketch implements only the linear subdivision method, which does not smooth the mesh correctly. The Loop Subdivision method was then implemented and tested on a sphere and, after a good result, on the final shape, the monkey. Figure 1 and 2 show the results.

As it is possible to notice, applying the method to the mesh and, after a reasonable number of iterations (in a limit), there is an infinite number of triangles on the mesh. For that reason, the monkey face geometry and the sphere appear smoother.

4 Conclusion

In conclusion, with this practical assignment, it was possible to understand the Loop Subdivision method with a practical approach. As an idea for future projects, the Catmull-Clark Subdivision used for Pixar's Geri's Game can be implemented by way of comparison.



(a) No division

(b) Limit division

Figure 1: Loop Subdivision applied on Sphere

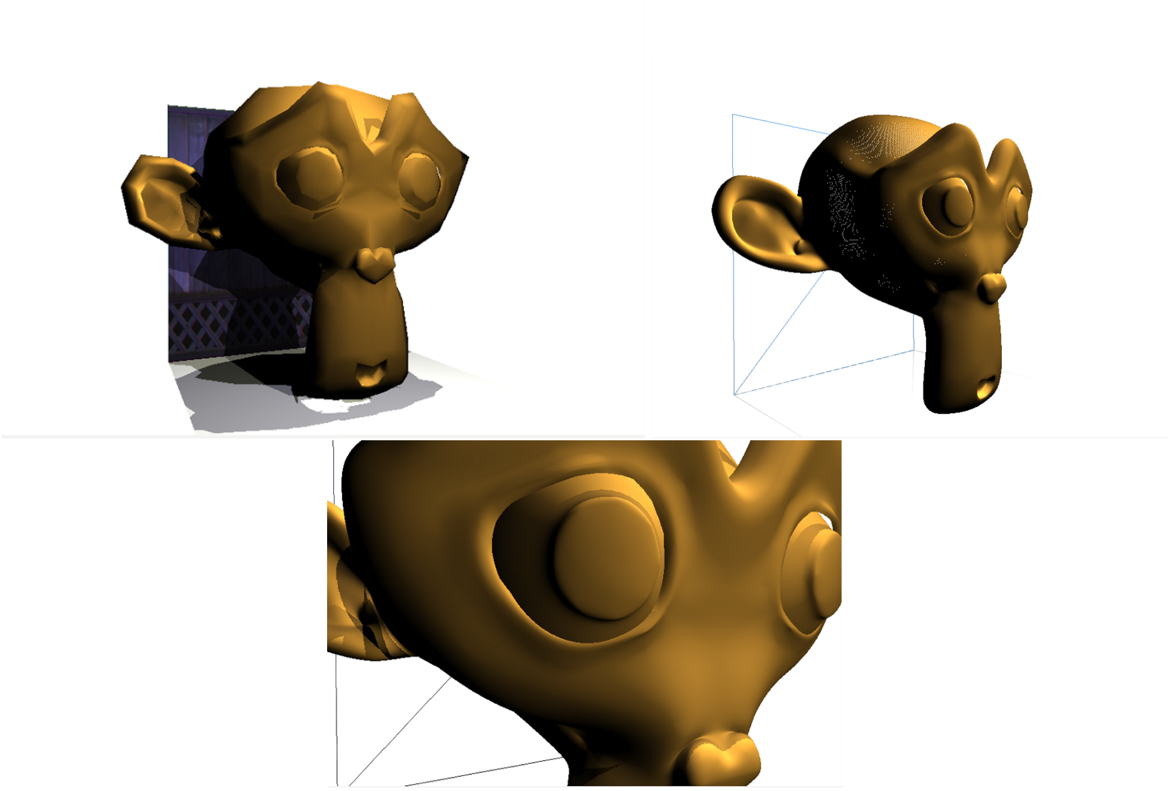


Figure 2: Loop Subdivision applied on a Monkey

5 References

1. **Curves and Surfaces in Computer Graphics** Dev Parakkat, Amal.
2. **Subdivision Surfaces** Heide, Felix. Princeton University, 2020.