IMPLEMENTING MESH SIMPLIFICATION

HAOCHEN QIU

Abstract. This is the c++ code for mesh simplification using quadratic error metric. Not refer to mesh framework, I construct my own Mesh, Vertice and many other essential data structure for fitting the tailed purposes. It is a capstone of my data structure ans c++ learning. Furthermore, I tried some improvement for better visual effects and do experiments on various mesh models for showing its robustness.

1. Parameters & environment

I tested code on :

- Ubuntu 20 64bit
- gcc 9.1
- 0.5GB RAM

Extra parameter **t**: the threshold for select valid pair, if the Eculidian distance for two vertice is less than t, it counts as valid pair.

2. Details

I construct Mesh for loading, writing and simplifying purpose. In Mesh, it contains Vertice and Face structs. Each Vertice store its one-ring neigbourhood faces in fas.

I also construct Heap, but using two map to maintain timestamp and existence of each element in Heap. When deletion happens, we change its timestamp to -1 and erase it in in_que_pairs. This solve the key bottle neck in each iteration , making deletion O(1).

3. Improvement Idea

First, I do dfs for search valid pair, because of direct edge search is O(n2), I implement IDA search , if the search distance is greater than t, we stop the recursion.

Second, in some sense, the smoother area needa less faces, more faces should contribute to sharp topological details , I add penalty to the original cost:

$$cost = \lambda v^T Q v + (1-\lambda) e^{-(v1 \cdot v2)}$$

where latter term is basically derived from the dot product for two normal vectors between the edge.

4. Experiments

The speed of this algorithm tested on 50000 dragon model for simplify ratio 0.001 needs 10s , here is the comparison for other input meshes:

Date: 2021 .09 .10.

TABLE 1. Comparison between meshes

	time	#vertices
Dinason2k	5s	40k
Bunnyfine	3s	50k
Dragon	10s	50k