Assignment 3:
Image compression and vectorization by QEM

NAME: CHENQIHE
STUDENT NUMBER: 2020533088
EMAIL: CHENQH@SHANGHAITECH.EDU.CN

1 INTRODUCTION

In this assignment, we implement 2D mesh on the image, simplify the image mesh based on QEM algorithm and compare the results under different target QEM number.

It takes us 4 days to complete the coding part all ourself and 1 day to write the report.

2 RELATED WORK

Mesh simplification was first proposed to produce levels of detail by subsequently removing elements from a complex object. In this assignment, we only target at the image mesh simplification and thus we focus on maintaining the original shape and color of the object.

However, existing QEMs are not ideal for images. QEMs are mostly defined as a summation of a spatial quadric that measures volume shrinkage and a colour quadric. Since the spatial domain of a raster image is planar, the volume shrinkage during simplifying an image mesh is zero everywhere except along the boundaries. Here, we replace the gemometry measurement with new metrics to implement mesh implification on 2D image mesh.

3 METHODS

3.1 Image triangulation

Given an RGB image, we transform it into a 2D mesh. For the simple and convenient purpose, we write a python script to make a image mesh.

Due to the complexity of our algorithm, we downsample the original image into (128×128) shape by using the third library function skimage.transform.resize. Then a downsampling raster image is converted to an initial dense triangular mesh; each pixel is represented by a vertex and the quadgirds are triangulated using one of the two diagonals.

3.2 New QEM algorithm

We denote a triangular mesh as a pair $(\mathcal{P}, \mathcal{K})$, where $\mathcal{P}$ is a set of N regular indices of vertices; while the topology is represented as an abstract simplicial complex $\mathcal{K}$, set of singles(vertices), couples(edges) and triples(facets) of indices in $\mathcal{P}$. Each index $i \in \mathcal{P}$ is realized as a five-dimensional point $v_i = (p_i, s_i)^T = (x_i, y_i, L_i, a_i, b_i)^T \in R^5$ with $1 \le i \le N$, where p represents the geometry components, s represents the attribute components and N is the number of vertices

Fig. 1. Initial dense triangular mesh

in the mesh. We confine the attributes to colour in this assignment. We choose CIELAB colour space for its ability to approximate perceptual colour distances with Euclidean distances. Two vertices i and j are neighbours if $(i, j) \in \mathcal{K}$.

We choose edge collapse as our basic operation in simplification. We prioritize the edges based on approximated error introduced by collapsing them. For each iteration, we picks the edge with the minimal error value at the start of each iteration as well updates the error value of the indices and vertices. To collapse an edge, we merge the two endpoints to one point, assign to it a new set of features(position as well as colour) and remove appropriate faces and edges during the process.

Since we update all face and vertices for each iteration, the total time complexity of our algorithm is $O(n^2)$. However, we notice that one collapse removes one vertex, at most two faces, and at most three edges. Therefore, we can simplify our algorithm to more tolerable time complexity in future work(at least after this assignment)...

3.3 Novel quadric error metric

We propose a novel flattened QEM to deal with five-dimensional image meshes that have flat geometry domain.

Each face f of the original mesh defines a quadric as the sum:

$$Q^f(v = (p, s)^T) = Q_p^f(v) + Q_s^f(v)$$

where $Q_p^f(v)$ represents the geometric error while $Q_s^f(v)$ represent the attribute error i.e. the colour error. We define the geometry quadric to be the squared distance from p to the geometric centroid

student number: 2020533088
email: chenqh@shanghaitech.edu.cn

$t = (x_t, y_t)^T \in R^2$ of f and the colour quadric is the same as geometry quadric.

Each vertex v of the original mesh is assigned the sum of quadrics on its adjacent faces weighted by face area: $Q^v(v) = \sum_{v \in f} area(f) \cdot Q^f(v)$.

Each edge e is assigned a quadric that is the sum of vetex quadrics of its two endpoints.

The new vertex introduced by edge collapse (after two vertices are removed) is assigned the position and attribute that minimizes the edge quadric(the minimizer). The minimum value of the quadric is defined as the error for edges.

$$v^* = \min_v Q^e(v) = \min_v Q^{v_1}(v) + Q^{v_2}(v)$$

Furthermore, we get that

$$\begin{aligned} v^* &= \min_v \sum_{v_1|v_2 \in f} area(f) \cdot Q^f(v) \\ &= \min_v \sum_{v_1|v_2 \in f} area(f) \cdot (\|p - p_f\|_2^2 + \|c - c_f\|_2^2) \end{aligned}$$

By taking the derivative of the right-hand side, we get

$$v^* = \frac{1}{\sum_{v_1|v_2 \in f} area(f)} \sum_{v_1|v_2 \in f} area(f) \cdot (p_f, c_f)$$

3.4 Implement details

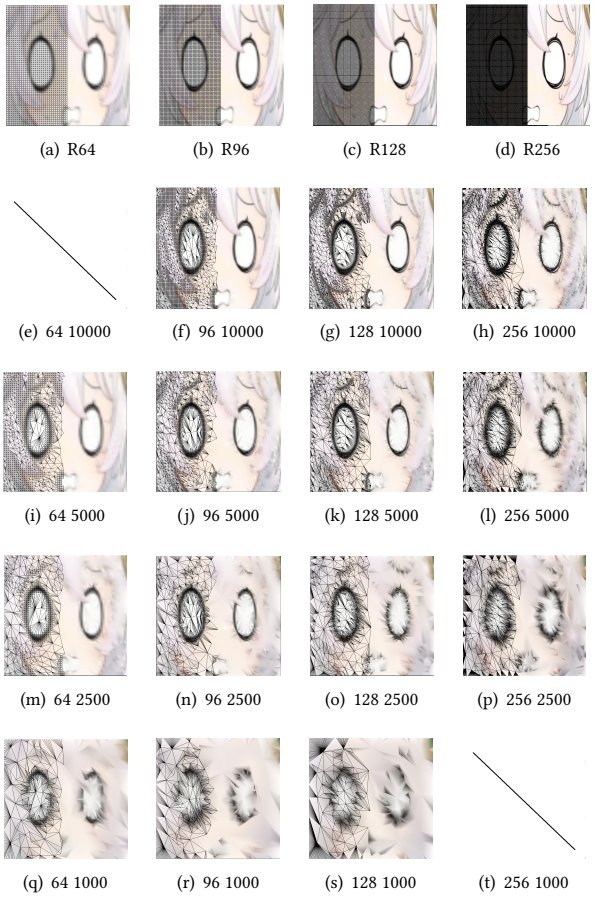
The basic pipeline is that we transform colour from RGB space to CIELAB space, perform New QEM algorithm for several iterations and transform colour back to RGB space.

Algorithm 1 New QEM algorithm

```
colors ← DoRGBtoLABConversion(colors)
while # face > QEM target num do
    // 1.construct facets
    faces ← getFaces(vertices, indice)
    // 2.construct QEM of each vertex
    QEMs(v) ← Q^v(v)
    // 3.construct QEM of each edge
    QEMs(e) ← Q^{v1}(v1), Q^{v2}(v2)
    // 4.find the edge with minimum QEM
    MinEdge ← min_e QEMs(e)
    v* ← calculate new point
    // 5.collapse the edge
    vertices, indices ← remove v1, v2, add v*
end while
```

4 RESULTS

Here we compare the results of our New QEM algorithm with different resolution 2D image and QEM target number.



(a) R64 (b) R96 (c) R128 (d) R256
(e) 64 10000 (f) 96 10000 (g) 128 10000 (h) 256 10000
(i) 64 5000 (j) 96 5000 (k) 128 5000 (l) 256 5000
(m) 64 2500 (n) 96 2500 (o) 128 2500 (p) 256 2500
(q) 64 1000 (r) 96 1000 (s) 128 1000 (t) 256 1000

Fig. 2. Results: different resolution from left to right, different QEM target number from top to bottom.

(a) Original (b) Simplified

Fig. 3. Results: QEM target number = 50000 for resolution=(256× 256)

...

(a) Original (b) Simplified

Fig. 4. Results: Zoom in with QEM target number = 50000 for resolution=(256× 256)

Image resolution QEM target number	(Original)	10000	5000	2500	1000
64	0ms(7936)	-	944ms	1574ms	1768ms
96	0ms(18050)	5093ms	7747ms	8203ms	7607ms
128	0ms(32258)	20832ms	20787ms	23429ms	22977ms
256	0ms(130050)	332776ms	339969ms	337034ms	-

Table 1. Running time for different resolution and QEM target number