Università della Svizzera italiana
Facoltà di scienze informatiche

Project Proposal

Barycentric Data Visualization for Triangle Meshes

Costanza **Volpini**
costanza.volpini@usi.ch

Professor: Kai **Hormann**
Assistant: Jan **Svoboda**

Spring Semester 2018

1 Introduction and Motivation

Using the power of barycentric coordinates we can figure out to which area a pixel belongs and which color it should be painted with.

Passing barycentric coordinates to the fragment shader will clearly demonstrate that we can get different results from the classic color interpolation (fig.1).

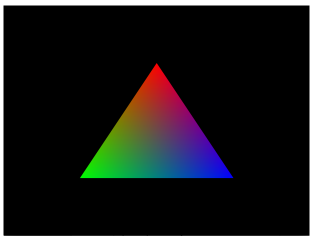


Figure 1: Interpolation.

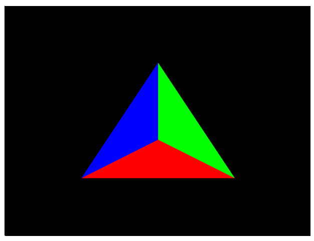


Figure 2: Min Diagram.

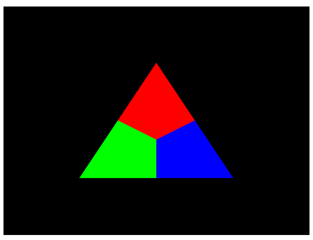


Figure 3: Max Diagram.

Some achievable approaches can make out focusing on the distance from vertices. Undoubtedly for each vertex, we can recognize some closer points to it and some not. One approach can be represented by the following.

Let's consider a triangle in which in the top vertex we have the red color, bottom left the green and in the bottom right the blue. The triangle barycentre divides each median into two pieces that have a 2:1 ratio. In fig.2 the farthest region from a vertex is colored. Also, we can notice that regions are divided by the "longer" part of medians. In fig.3 we have colored the closest region from a vertex. These approaches can be expressed in relation to barycentric coordinates.

2 Project description and Goals

Alternative data visualization techniques can be found using the power of barycentric coordinates and GPU programming. The usual way to visualize data for a triangle mesh is to associate data to vertices and then interpolating over the mesh triangles, that does not work in case of edges and triangles.

As a **first main goal**, we want to expand the idea of *Flat Shading*[1] also on vertices and edges. A next step could be to split the surface of triangle meshes into regions around vertex (fig.4) and edges (fig.5) and color them.

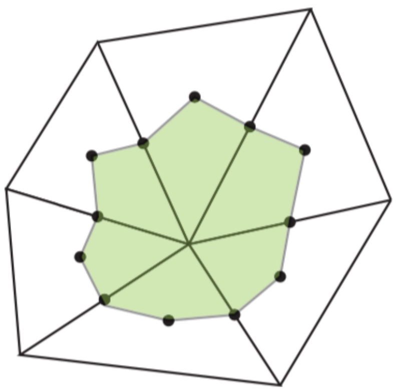


Figure 4: Region around a vertex.

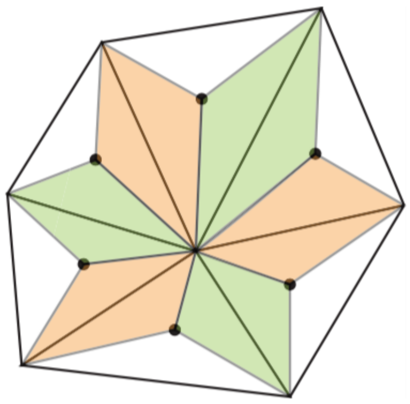


Figure 5: Region around an edge.

These regions can be determined using barycentric coordinates and GPU fragment program. Visualizing data given at the vertices or edges of the mesh in a piecewise constant simulates the classical triangle flat shading.

Examples of these edge and vertex data are the discrete Gaussian and discrete mean value curvature.

The next step will be to analyze the results obtained. A **second main goal** would be to extend the project set up a *Gouraud Shading*[2] for these new mesh tessellations. Using the barycentric coordinates and the fragment shader we can spell out a similar piecewise linear continuous interpolation over vertex and edge of regions (fig.4 and fig.5). The final step would be to evaluate these data.

3 Milestones and Project Timeline

3.1 Milestones

M1) Environment setup.

[1]Flat Shading: the idea is to draw all the pixels of a triangle with the same color.

[2]Gouraud Shading: linear interpolate data given at the vertices over the triangles of the mesh, resulting in a piecewise linear and continuous visualization of the data.

M2) Flat shading on vertices and edges.

M3) GPU fragment program.

M4) Extension of Gouraud shading.

M5) Project thesis and report.

3.2 Tasks

T1) GPU vertex program.

T2) Extend Flat Shading.

T3) Split surface triangle mesh into regions.

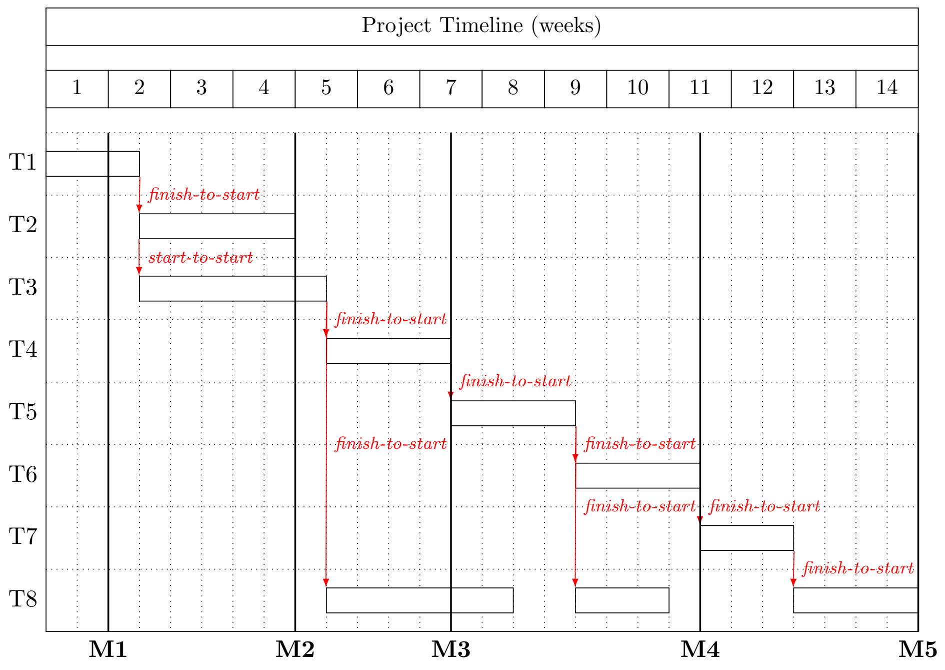
T4) GPU fragment program.

T5) Evaluation and comparison.

T6) Extend Gouraud Shading.

T7) Evaluation and comparison.

T8) Project thesis and report.


Project Timeline (weeks)
T1
T2
T3
T4
T5
T6
T7
T8
finish-to-start
start-to-start
finish-to-start
finish-to-start
finish-to-start
finish-to-start
finish-to-start
finish-to-start
finish-to-start
M1
M2
M3
M4
M5