CG Programming - S0006E - 2016
Week 6 - Software Rasterization

Software Rasterization

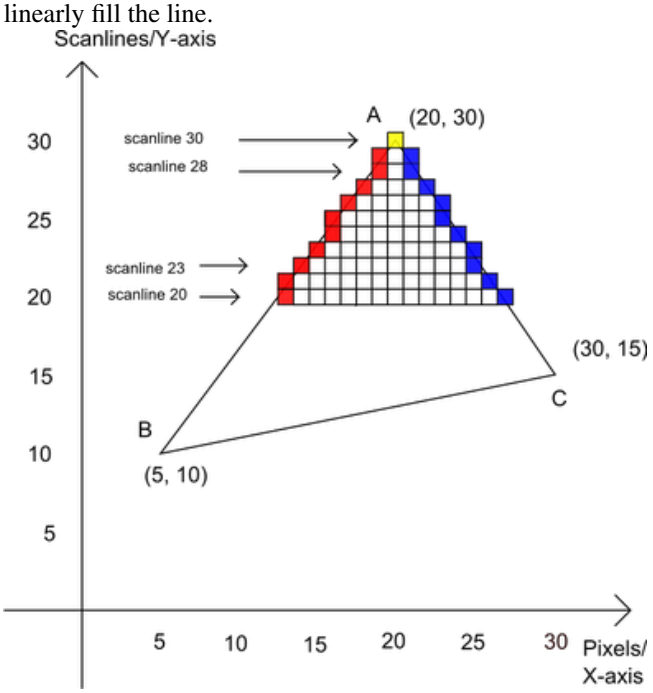
The goal is to create your own software rasterizer that can draw textured and lit triangles into an image buffer. Use a quad and the image buffer as texture to display the output.

1. Create a software rendering class that can take a vertex and index buffer and render the triangles in the index buffer. It also needs to contain a lambda function for the current per pixel shading, a model-view-projection matrix, and a texture represented by an image buffer. It needs an additional image buffer for the actual rasterization (the one the triangles will be drawn into). The image buffer dimensions should be configurable.

2. Add a raster triangle function that takes in 3 vertices with 2d positions, normals and texture coordinates. The function has to interpolate the normals and texture coordinates smoothly over the surface of the triangle and then call the currently registered lambda function for every pixel.

3. The lambda function takes uv coordinates, normals and a texture as arguments and returns a pixel colour.

4. Implement the triangle rasterizer by using a scanline technique. Start at the vertex with the lowest y value and proceed line by line downward. Use the Bresenham algorithm for finding the left and right pixels on every line and linearly fill the line.



   Call the lambda function at every pixel and store the resulting colour in the image buffer.

5. Add a draw function that will iterate through all the triangles defined by the index and vertex buffer, transform the vertices, and call the rasterize function for every transformed triangle.

6. Show the resulting image buffer using a quad in your program.

Delivery

Commit your complete project to a dedicated folder inside your SVN repository (e.g. S0006E/assignment) and upload the number of the revision to Canvas.