CMSC 435/634: Introduction to Computer Graphics

Assignment 3
Rasterization
Due October 16, 2017 @ 11:59 PM

The Assignment

For this assignment, you will implement a rasterization-based renderer---a software implementation of the rendering pipeline. We will again use NFF to describe scenes. I implemented each step of the rendering pipeline as in its own method: **Vertex Processing, Clipping, Rasterization, Fragment Processing, and Blending** I also have a method that sets up the homogeneous transformation matrix.

I compute M_vp, M_per, M_cam as in the book, these are combined into a single M matrix that will be used to transform every vertex in the scene.

In Vertex Processing, I shade all vertices and apply the transformation. I store colors and transformed points in my triangle class. I do **not** perform the homogeneous divide at this point (clipping cannot be performed after the divide). Note that when vertex normals are provided in the nff file, these should be used for shading.

Clipping processes the triangles to ensure they are inside the view frustum. This is accomplished by checking whether the vertices are all inside/outside of the planes that define the frustum. If all vertices are inside the triangle is kept. If all are outside the triangle is skipped, if all are inside the triangle is passed to the rasterizer. If some are inside and some are outside the edges of the triangle are intersected with the frustum and one or two triangles are passed to the rest of the clipping planes or to the rasterizer.

The rasterizer performs the homogeneous divide and computes a 2D bounding box for the triangle. A nested for loop, loops over all the pixels in the bounding box and determines whether the pixel falls inside the triangle (this requires some care to avoid cracks, see the book). Vertex colors are also interpolated at this point.

In fragment processing we can perform phong shading (image forthcoming).

Blending, for each pixel loop over all fragments and determine which is closest, use the color of this fragment in the image.

Extra Credit

For up to 25 pts of extra credit, implement clipping

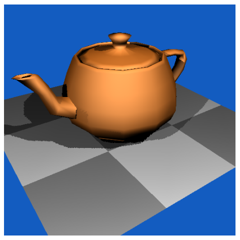
For up to 15 pts of extra credit, implement fragment shading. During rasterization interpolate position and surface normal and use these to shade the fragment during fragment processing. You can run my program as

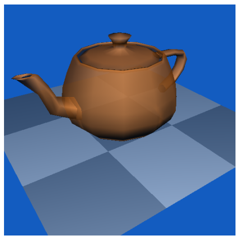
```
~adamb/public/rasterize/rasterize -f teapot-3.nff output.ppm
```

For up to 20 pts of extra credit, implement shadow maps. Instead of renderig the image from the camera location, set up the transform from the light position (use the other camera data, e.g. angle, hither, at) and instead of recording the color, record the nearest z coordinate. Then, when rendering tranform the point you are shading using the light transformation and compare the z value to the value in the shadow map. If they are the same, the surface can see the light, if the value in the shadow buffer is closer, hten the surface is in shadow and the light should be ignored. You'll need a shadowbias, just as in raytracing (I used 2e-3). This works better with fragment shading, at least on the teapot.

You can run my program as

```
~adamb/public/rasterize/rasterize -s -f teapot-3.nff output.ppm
```





For up to 15 pts of extra credit, implement transparency. During blending sort fragments in reverse z order (farthest away first). Then for each fragment, set the pixel color to

```
alpha * fragment_color + (1.0 - alpha) * current_pixel_color
```

You can run my program as

```
~adamb/public/rasterize/rasterize -t 0.65 teapot-3.nff output.ppm
```

What to turn in

Turn in this assignment electronically by pushing your source code to your `assn2` GIT directory by 11:59 PM on the day of the deadline. We will be looking for multiple checkins documenting your development process.

As always, double check that you have submitted **everything** we need to build and run your submission, but **no** generated files (.o's, executables, or images). Be sure to include a Makefile that will build your project when we run '`make`', and a `readme.txt` file telling us about your assignment. Do not forget to tell us what (if any) help did you receive from books, web sites or people other than the instructor and TA.

(This page: www.csee.umbc.edu/~adamb/435/proj3.html)