CMSI 371-01

COMPUTER GRAPHICS

Spring 2012

Assignment 0306

This assignment is your opportunity to engage in some low-level graphics thinking.

Outcomes

This assignment will affect your proficiency measures for outcomes *1a*, *2b*, *3a*, *3b*, and *4a–4f*.

Not for Submission

By February 21

Read the following sections in Angel: 2.5 (pages 67–73) and 6.8 to 6.10 (pages 323–331).

By February 23

Depending on how far we go before spring break, you may get some overlapping work — so, if you finish the work for submission by today (and it *is* possible), you will set yourself up better for what is to come.

By March 6

Read the following sections in Angel: 1.1–1.9 (pages 1–40).

For Submission

A Few Good Filters

Copy the *nanoshop* sample code and modify the `Nanoshop` module so that it has a "library" of pixel filter functions (the way our `KeyframeTweener` had a set of easing functions). Come up with three (3), and of course modify the accompanying demo page to show them off. Be creative, have fun!

Commit and push your work to your `git` repository under *homework/nanoshop-filters*.

Primitive Behavior

Copy the *primitives* sample code and make the following modifications to it:

- Modify the `lineBresenham` function so that it accepts a `dash` parameter. This parameter is expected to be an integer that draws a dashed line. A `dash` argument of 5, for example, would draw 5 pixels first, then skip a pixel, then another 5, then skip, etc. (like this: — — — —)
- Modify the `plotCirclePoints` function so that, instead of plotting the outline of a circle, it *fills* the circle.
- This last one is known to be difficult, and you are not expected to finish it. However, you are expected to give it a fair shot — modify the `fillPolygon` function and its accompanying code (`Edge`, helper functions) so that the `color` parameter may be an *array* of colors, one for each vertex in the given `polygon`. The resulting fill should then set each vertex to its designated color, with the pixels in between transitioning gradually to the color of the adjacent vertices.

  Look at the `fillRectFourColors` function inside `fillRect` to see how this can be done — what you want is a generalization of that code.

Commit and push your work to your `git` repository under *homework/primitives-plus*.

Primitive Questions

After studying the provided code and reading the first set of Angel sections, answer the following:

1. The implemented Bresenham line algorithms in the `Primitives` module are hardcoded for a specific configuration of line segment.
   a. Why is this an acceptable starting point?
   b. What would need to be done so that they can handle the general case?
2. For the special case implementation that is provided, list three types of line segments that would look wrong.
3. While you might not be able to completely code up the `fillPolygon` enhancement listed in the previous section, you should be able to describe what the algorithm would do. Write up that plain English description.

Commit and push your answer in some widely readable format under *homework/primitives-plus*.