Assignment 6 for csci580

Accumulation Buffer and Multipass Antialiasing

In this assignment you will create an application that performs multipass antialiasing. The algorithm computes multiple samples distributed around each display pixel and weights (filters) these into one final value. A screen space offset is added to the transformed vertex coordinates of each triangle to produce the sample distribution.

The offset is specified by two new Renderer parameters
(GZ_AASHIFTX, GZ_AASHIFTY). (The default offsets = zero)

A table, such as the one below, specifies the filter offsets and weights. Feel free to build and test your own filters. Offsets are fractional pixel shifts and the sum of all the weights = 1.0. Note that for convenience, we define the number of samples/offsets at compile time.

```
#define AAKERNEL_SIZE       6

float AAFilter[AAKERNEL_SIZE][3]                    /* X, Y, coef */
{
-0.52, 0.38, 0.128,
0.41, 0.56, 0.119,
0.27, 0.08, 0.294,
-0.17, -0.29, 0.249,
0.58, -0.55, 0.104,
-0.31, -0.71, 0.106
};
```

Your application sets up multiple Renderers and Displays -- one pair for each sample in the table. Each Renderer is initialized with a different offset and display to write into. All triangles are sent to all renderers.

After rendering all the triangles, the application combines the multiple display images by weighting their pixel values. The filtered (result) image is stored in one of the displays (or a new one) and then written to disk. Also the filtered image is stored in the frame buffer to window draw.