B657 In-class activity #2

Spring 2017

In this activity you'll get some experience using image projections and transformations. We recommend using C/C++ for this activity with the CImg library provided for Assignment 2, although you can feel free to use Python or another language. Please work in your Assignment 2 groups and commit your code to a separate directory in your A2 repository called activity2.

Creating stereograms

We will distribute glasses having a red lens (to be worn on your left eye, which is your right eye from the perspective of someone looking at you) and a cyan lens (for your right eye). These are a simple form of 3d viewing technology, and the mechanism is straightforward: since cyan and red are complementary colors, pixels in the red channel will not be (very) visible to the right eye, and vice-versa. To create the illusion of 3d, we can create special images that encode two separate images, each of which is visible to one eye. To see some examples, type "anaglyph" into Google Image Search.

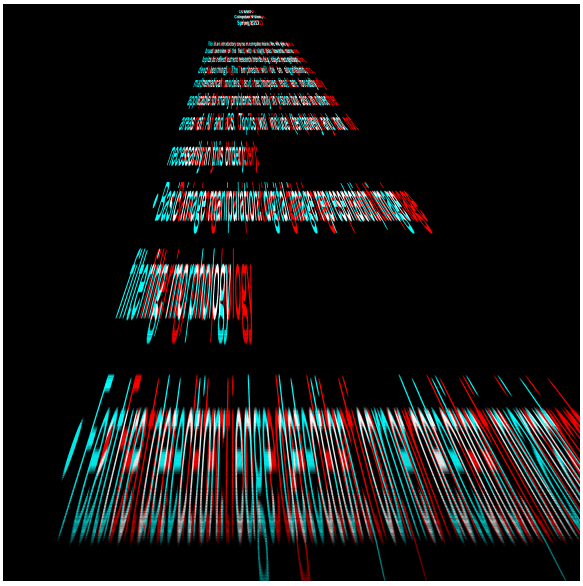
On canvas, under Activity 2, you'll find a file containing 3d point counts for an object. These are in a very simple file format. Each line corresponds to a single point on the object, and in the format:

```
x y z
```

where x, y, and z are the coordinates of the point in the 3d world.

Try the following steps:

1. Write some code that loads in and parses the file.
2. Write some code that projects the 3d object onto the plane of a camera stationed a bit away from the object. For example, you can do this by using a transformation and projection matrix where the 3d rotation part is the identity matrix, and the translation part has zero x and y translation and a small z.
3. Now write some code that projects the 3d object into a more interesting configuration by introducing a 3d rotation.
4. Finally, create the stereoscopic image by projecting onto 2 different image plans, each separated by a small amount of translation (corresponding to the difference in position of your two eyes). Combine these images together, encoding one on the red color plan and the other on the cyan channel (i.e. blue and green planes). Here's an example of what you might end up with:



What to turn in

Commit and push your code and sample output files to the Assignment 2 repository, under a directory called activity2.