3D Visualization of petroleum data

Project 1: Visualization using OpenGL and C

Einar Baumann

November 4, 2013

1 Filereader

The file reading function was implemented in a separate header file. The method takes as input the path of the file to be read; a pointer to the matrix to store the data in; and the number of traces and samples contained in the file. The implementation is shown in Listing 1.

Listing 1: SEGYReader.c

```
void ReadSEGYTraceSamples (char *fileName, char *matrix, size_t traces, size_t samples)
{
  FILE* fileP = fopen(fileName, "rb");
  fseek(fileP, 3600, SEEK_SET);             // Skipping 3600 byte file header.
  for (int i = 0; i < traces; i++)
  {
    fseek(fileP, 240, SEEK_CUR);            // Skipping 240 byte trace header.
    for (int j = 0; j < samples; j++)
    {
      fread(&matrix[i*samples+j], 1, 1, fileP);
    }
  }
  fclose(fileP);
}
```

2 Color mapping

A linear color mapping from red to blue via white was used. It's illustrated in Figure **??**. To determine the color of each point the program calculates a factor determining how much each value deviates from the extremum (i.e. 128). This is then used to make the color more red or blue, depending on whether the value is negative or positive. The central parts of the implementation is shown in Listing 2.

Listing 2: Part of the color mapping implementation. The shown code calculates the factor used to determine the intensity of the color, as well as RGB color mapping of positive values.

```
element = seismic[i][j];
factor = (float) abs(element)/128.0;
if( element > 0)          // Setting positive elements to blue
{
  textureRGBArray[textureNr][3*pixelCounter]   = 255 - 255*factor;
  textureRGBArray[textureNr][3*pixelCounter+1] = 255 - 255*factor;
  textureRGBArray[textureNr][3*pixelCounter+2] = 255;
}
```

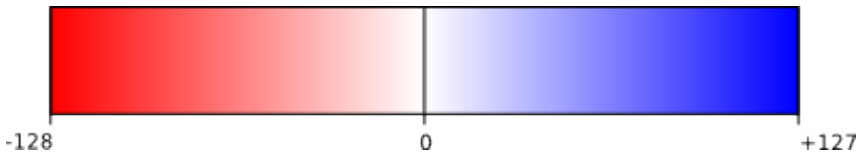

Figure 1: The scale used for the color mapping.

3 Mipmapping and filtering

Screencaps of the program with mipmapping turned on and off and with different filters and zoom levels are shown in figures 1-5.

Figures 1, 2 and 3 all show the same area of seismic data, but Figure 1 does not use mipmapping while 2 and 3 do; and 2 and 3 use different filters (linear and trilinear, respectively). We can see that the use of mipmapping and linear filtering in Figure 2 results in a much smoother picture, where the lines can be more clearly seen than in the case with no mipmapping. The use of trilinear filtering in Figure 3 smooths the picture too much, making many of the lines hard to see.

When zoomed all the way in (Figures 4 and 5), there is no difference between the picures with and without mipmapping. This is exactly as expected because mipmapping is used when the textures contains more points than what can be fitted on the screen (i.e. when zoomed out).

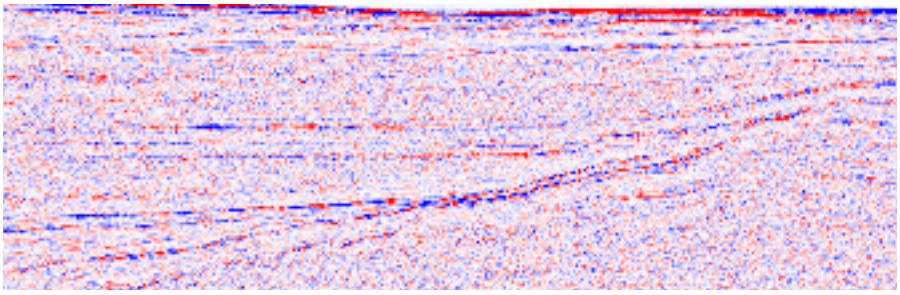

Figure 2: No mipmapping; linear filtering.

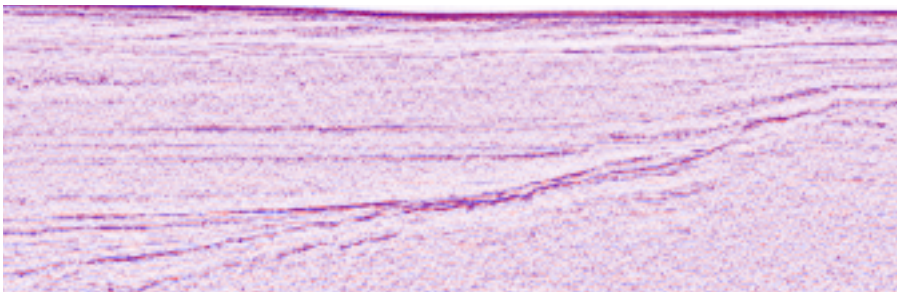

Figure 3: Mipmapping; linear filtering.

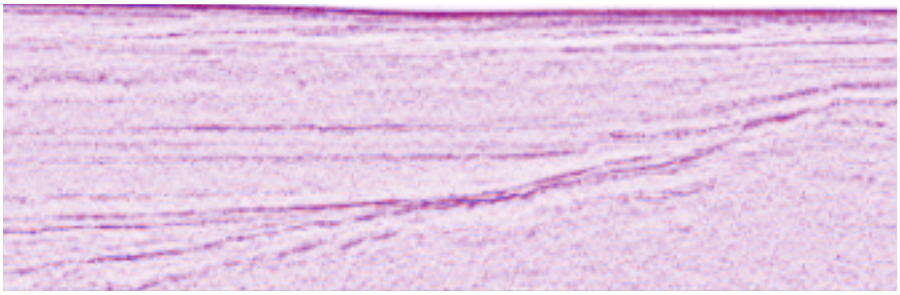

Figure 4: Mipmapping; trilinear filtering.

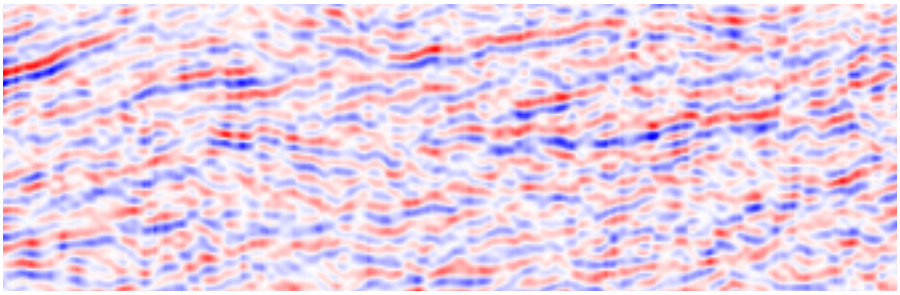

Figure 5: No mipmapping; linear filtering. Zoomed in.

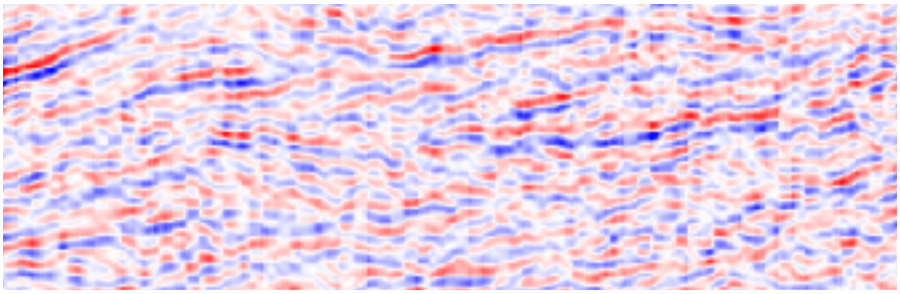

Figure 6: Mipmapping; linear filtering. Zommed in.