Othello! It's Snowing!

Samuel Grant Dawson Williams

May 17, 2007

1 Introduction

Image based rendering is interesting because of the number of effects possible in real-time. In particular, the recent addition of programmable graphics hardware has increased the ease and depth at which image based techniques can be utilised.

2 Particle Effects

A snow effect is implemented using point sprites (see figure 1). Initially, this effect was implemented using single points, and then with textured quads[1]. To avoid the processing cost of orienting every quad, the point sprites extension was used to allow each vertex to represent a textured quad.

Figure 1: Snow falling from the sky, using a quadratic falloff of size, and eventually alpha blended to nothing.

Particles have a lifetime of 35 seconds, and various other parameters used for simulation. They are drawn last and do not contribute to the depth buffer. A maximum particle count helps to avoid resource over-use.

This particular implementation allows particles to land on the playing surface (see figure 2), to give a more realistic physical model.
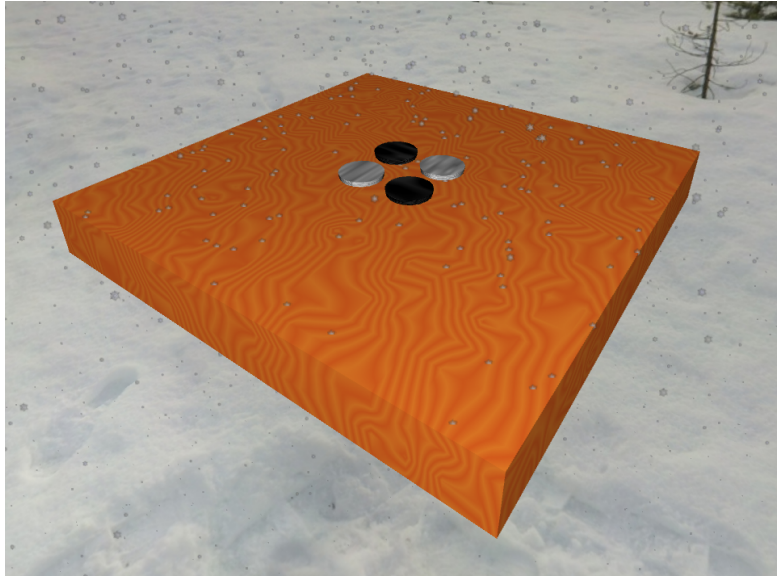

Figure 2: Collision detection allows particles to land on the playing surface.

3   Sky Box

The sky box[2] is a clever trick using textures projected on to the sides of a cube to simulate a complete surrounding environment. The effect of a surrounding environment is quickly lost if the viewer moves far from the centre. To minimise this problem, the box is typically several magnitudes bigger than the 'objects' contained within (see figure 3).

In this implementation, the sky box has no lighting equations applied to it, and is rendered with a slight overlap of each edge, to avoid any visual problems at the edges of the cube. The overall result is fairly convincing.

4   Text Rendering

Text rendering is a useful tool to present real-time information. Font rendering involves compositing each character one after the other, taking into account kerning and various other typographic information (see figure 4). This particular example is also anti-aliased.

Text is composited to the screen using an orthographic quad textured using the TEXTURE_RECTANGLE
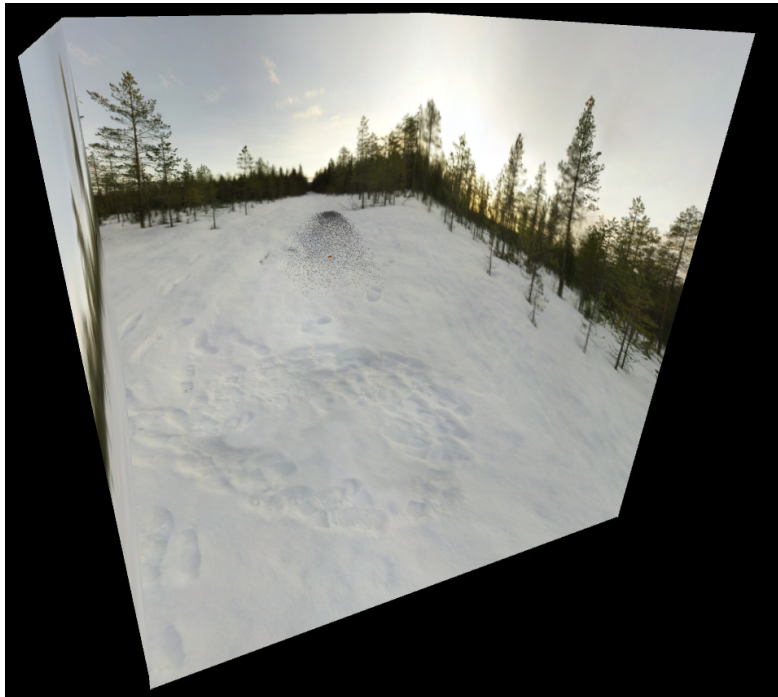
Figure 3: The sky box is actually a cube, and as can be seen here, the effect only works when the eye origin is positioned near the centre.

extension. Not only does this avoid any problems with size, it allows pixel-perfect composition.
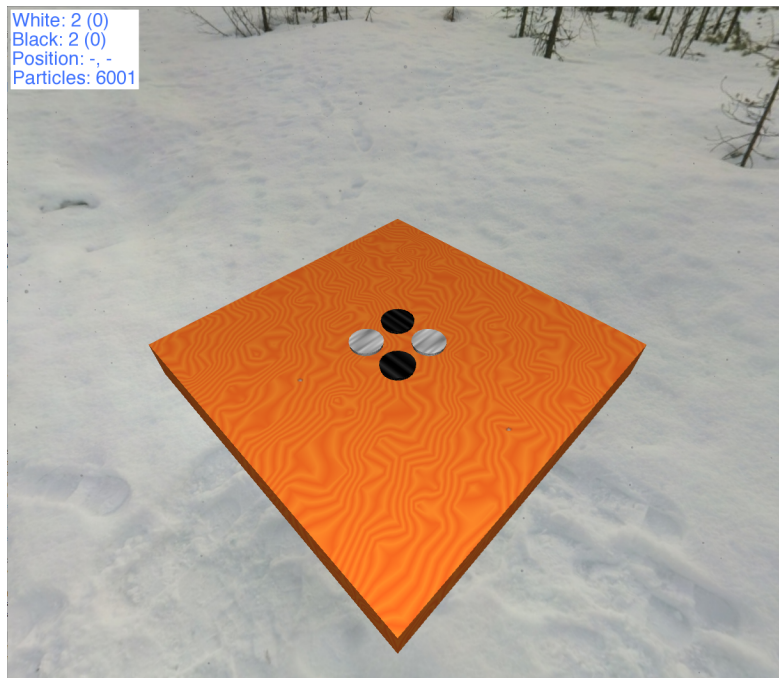

Figure 4: Real-time feedback is very useful for debugging.

5   Wooden Shader

The wooden texture is generated by a vertex+fragment shader. This shader utilises a 3D texture of sampled perlin noise to perturb a circular wood grain (see figure 4).

6   Implementation Notes

Included in the source code are cross-platform image loaders for PNG[3] and JPEG[4] using their respective open source image loading libraries. These have been coded for this particular visualisation engine.

A resource locating framework which loads various resources based on path and file names has also been implemented using the boost framework[5] for cross-platform filesystem compatibility.

Freetype2[6] was used to provide font rendering capabilities.

A composite shader file format (and loader) that allows vertex and fragment shaders to be combined for ease of use and simplicity.

References

[1] Game Programming Wiki. Opengl particle tutorial - game programming wiki, 2007. [Online; accessed May-2007]. Available from: `http://gpwiki.org/index.php/OpenGL:Tutorials:Tutorial_Framework:Particles`.

[2] Game Programming Wiki. Sky box - game programming wiki, 2007. [Online; accessed May-2007]. Available from: `http://gpwiki.org/index.php/Sky_Box`.

[3] Greg Roelofs. Portable network graphics image format, 2007. [Online; accessed May-2007]. Available from: `http://www.libpng.org/`.

[4] Independent JPEG Group. jpeglib, 2007. [Online; accessed May-2007]. Available from: `http://www.ijg.org/`.

[5] Boost c++ libraries, 2007. [Online; accessed May-2007]. Available from: `http://www.boost.org/`.

[6] The freetype project, 2007. [Online; accessed May-2007]. Available from: `http://freetype.sourceforge.net/`.