Ficha IV – Texturas

Exercício 1

You need to render them back to front relative to the camera in order for transparency to correctly work, check painters algorithm, and depth buffering.

The rational behind this, is that because how depth tests work. The "usual" case for OpenGL is to test fragments for depth, once a fragment passes the depth test it will be written to the frame buffer overwriting any fragment that was written in the same frame buffer position, in other words when your scene have only opaque objects the depth buffer is enough to sort your rendered objects because it only keeps fragments that are near the camera.

When transparency comes into play, this is no more the case, and you need more info than the only nearest fragment. The depth test will still make near fragments overwrite far ones in the frame buffer, even though transparency says otherwise.

GL_BLEND_FUNC() (deprecated btw you should switch to shaders) can only blend fragments using a certain fixed equation, has nothing to do with the order pixels are written and have no control over the depth buffer. So it will apply the blending equation regardless of how the depth buffer sorted them. So the only practical option is to draw far then near objects.

As a side note opaque only objects (unlike transparent) should be draw front to back to avoid over-draw.

Exercício 2

<u>The alpha channel:</u>

The concept of the alpha channel is pretty simple. Instead of a writing an RGB result, you write an RGBA :

// Ouput data : it's now a vec4

out vec4 color;

the first 3 components are still accessed with the .xyz swizzle operator, while the last one is accessed with .a :

color.a = 0.3;

Unintuitively, alpha = opaqueness, so alpha = 1 means fully opaque while alpha = 0 means fully transparent.

Here, we simply hardcode the alpha channel at 0.3, but you probably want to use a uniform, or read it from a RGBA texture (TGA supports the alpha channel, and GLFW supports TGA).

Here's the result. Make sure to turn backface culling off (glDisable(GL_CULL_FACE)) because since we can look through the mesh, we could see that it has no "back" face.

total transparency:

For total transparency the alpha channel test is an appropriate solution.

– The test is performed before the Z-buffer is written and eliminates every pixel which

fails the test

– Hence, these pixels do not affect the Z buffer

Exercício 3

Exercício 4

O parâmetro GL_LINEAR , devolve para cada pixél a média ponderada dos 4 pixéis da textura que se encontram mais próximos do centro do pixel a ser texturado, dando uma aparência mais suave à textura.

No caso do GL_NEAREST , este devolve o valor da textura que se encontra mais próximo do centro do pixel a ser texturado, ficando com uma aparência mais bruta ou "pixelizada".

Exercício 5

- Uma textura com mipmapping é uma sequência de imagens gradualmente mais pequenas, sendo a dimensão da imagem seguinte metade da anterior.

- Na aplicação é escolhida a imagem cuja resolução mais se aproxima da resolução indicada para o pixel em questão, ou uma combinação linear entre as duas imagens mais próximas.

Vantagens:

- melhor aspecto visual devido à consistência entre a zona a texturizar e a imagem de onde são recolhidas as amostras;

- potencialmente mais rápido quando as imagens selecionadas são as mais pequenas devido à cache de leitura.

Desvantagens:

- memória ocupada (< 1 + 1/3 da versão original)

- Setup inicial

Exercício 6

```
glBindTexture(GL_TEXTURE_2D,texID);
glBegin(GL_QUADS);
            glTexCoord2f(0,0);glVertex3f(-1.0f, -1.0f, 0.0f);
            glTexCoord2f(1,0);glVertex3f( 1.0f, -1.0f, 0.0f);
            glTexCoord2f(1,1);glVertex3f( 1.0f, 1.0f,  0.0f);
            glTexCoord2f(0,1);glVertex3f(-1.0f, 1.0f,  0.0f);
    glEnd();
```