

1- Se desenharmos um objeto A atrás e depois de ter desenhado um objeto transparente B, não iremos conseguir visualizar o objeto A através do B. Para ser possível é necessário obedecer à ordem de desenho, ou seja, primeiro o objeto A e depois o B. Mesmo entre objetos transparentes é necessário decidir a ordem de desenho, pois vai definir o lado em que o observador pode ver os objetos atrás.

2- O canal alfa é uma imagem em que cada pixel tem um valor de 0 a 1. Após aplicar o teste do canal alfa, os pixels que tiverem um alfa menor que um threshold estes não são desenhados(não são incluídos no z-buffer). Este mecanismo garante a transparência sem nos preocuparmos com a ordem de desenho.

5- Para resolver o problema de flickering temos o mecanismo de mipmapping que permite otimizar a renderização de texturas em diferentes escalas. O processo começa com a textura original com resolução máxima. Depois vão-se criando texturas reduzidas da anterior até atingir 1*1 pixel. As versões são armazenadas numa pirâmide de mipmapping.

-Vantagens
Redução de aliasing
Melhora desempenho

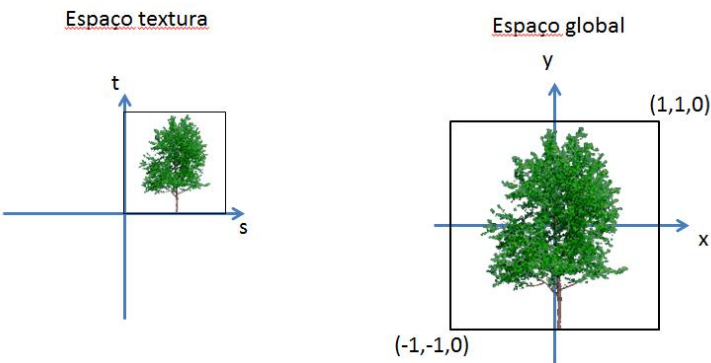
-Desvantagens
Armazenamento adicional para as versões de texturas
Overhead de pré-processamento

Ficha de Consolidação IV

Texturas

1. Para obter transparências parciais é necessário ordenar os triângulos de modo a que os triângulos transparentes sejam desenhados só no final, ordenados por distância decrescente à câmara. Justifique esta necessidade.
2. Descreva como funciona o mecanismo de transparências totais utilizando o teste do canal alfa.
3. Descreva o problema de amostragem resultante de se projectar uma textura no ecrã numa área com um número de pixels muito inferior à dimensão da textura.
4. Descreva o processo de amostragem utilizando o filtro GL_LINEAR e GL_NEAREST.
5. Descreva o mecanismo de mipmapping, indicando as suas vantagens e desvantagens.
6. Considere que se pretende mapear uma textura num QUAD. Complete o código seguinte para obter o resultado da figura.

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



3. Caso a textura tenha uma dimensão superior à área de desenho da tela ocorre o processo de flickering. Neste caso uma movimentação da câmara para percorrer um determinado número de pixels, corresponde a percorrer um número superior de pixels na textura, o que acaba por ser apresentado um pixel diferente.