Texturas

Texturas

O uso de texturas pode eliminar a necessidade de uma geometria mais pormenorizada

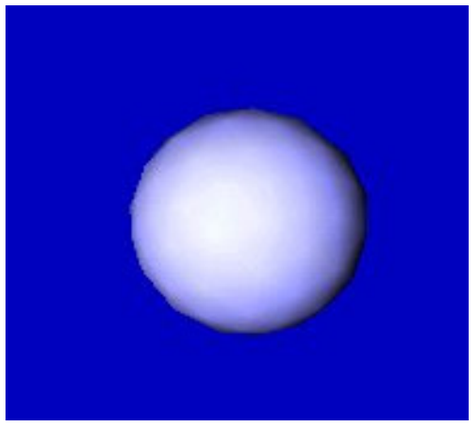
S/ texturas

C/ texturas

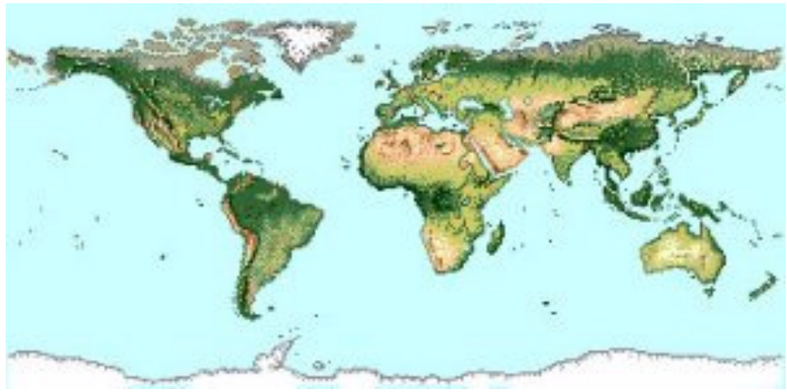
M.Próspero

Mapeamento de Texturas

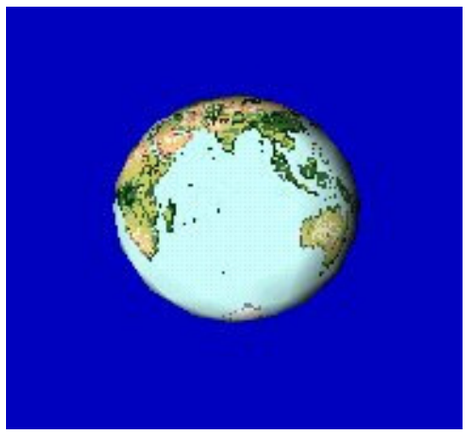
(TEXTURE MAPPING)



+

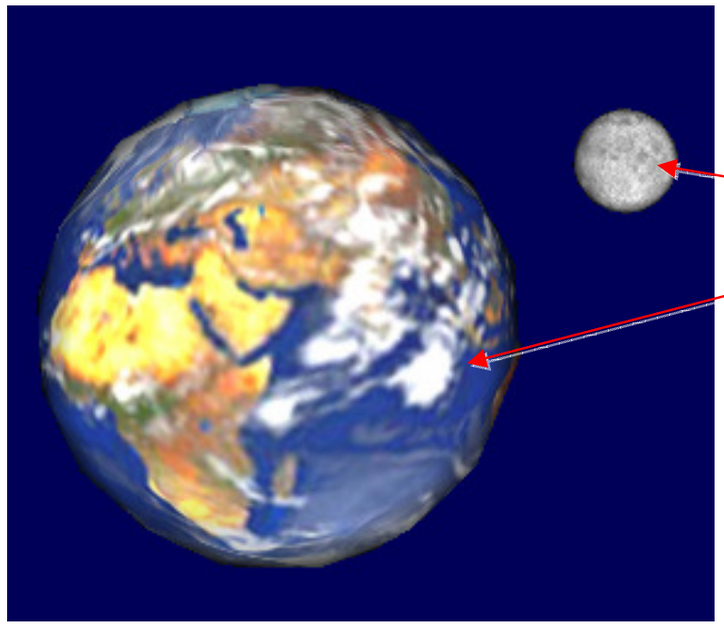


=



M.Próspero

Obtenção das Texturas

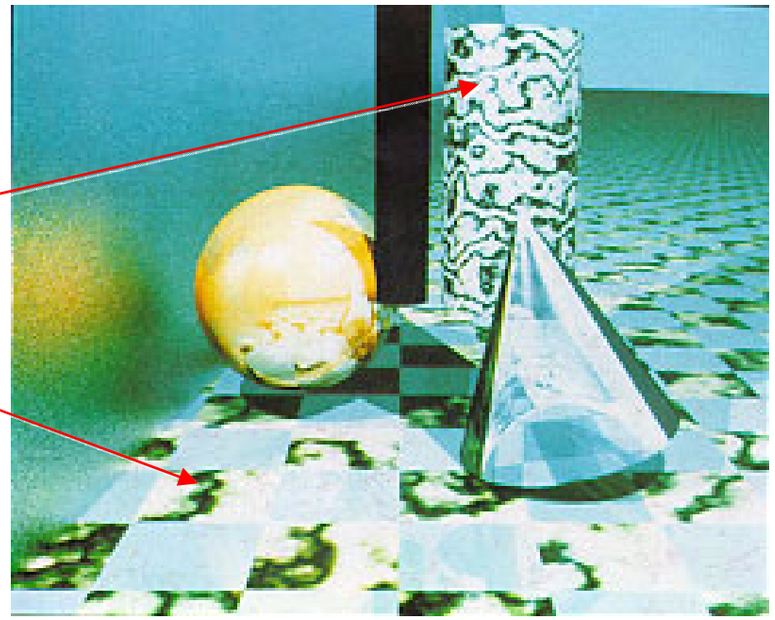


Uma textura pode ser armazenada num quadro, quer proveniente de imagem digital…

Para se utilizar um quadro como textura, em OpenGL:

`glTexImage2D()`

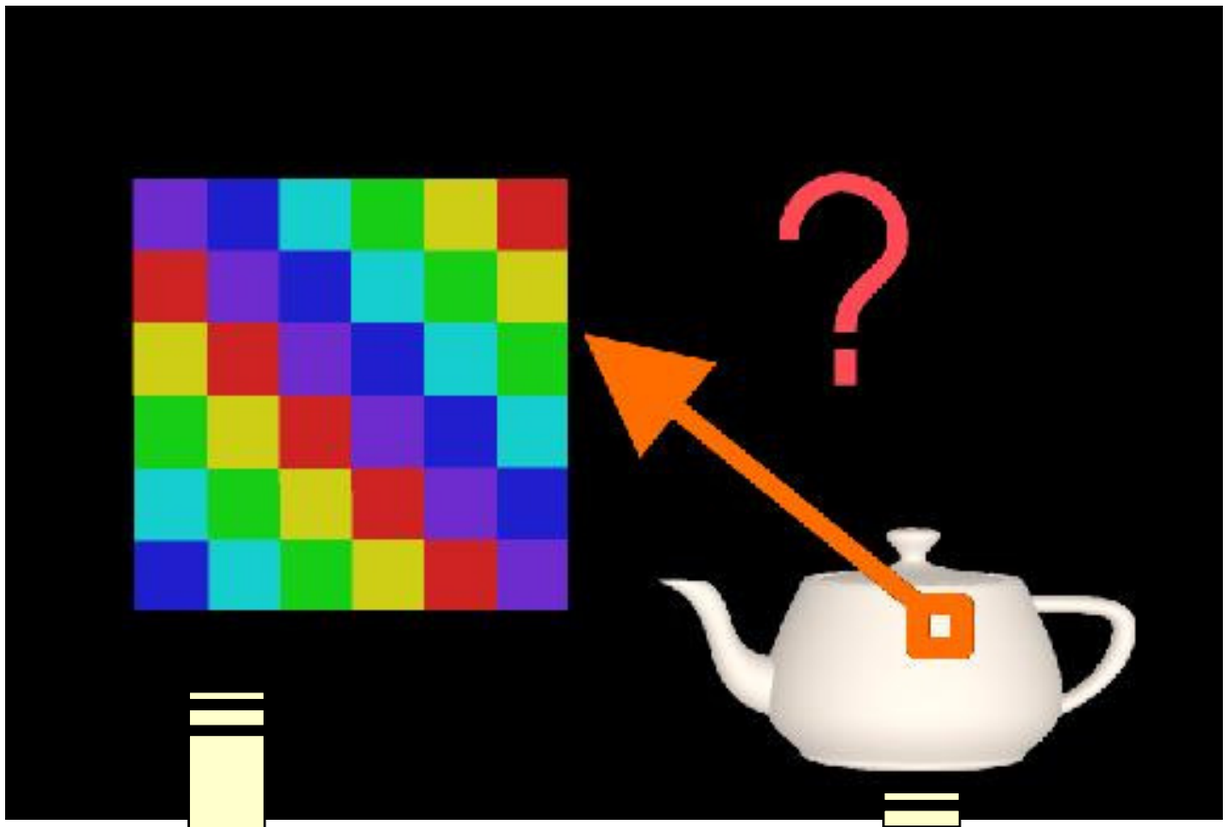
… quer calculada por um procedimento em tempo de execução



M.Próspero

Texturas

O problema da correspondência no mapeamento



Entidades:

Textura (formada por **texels**)

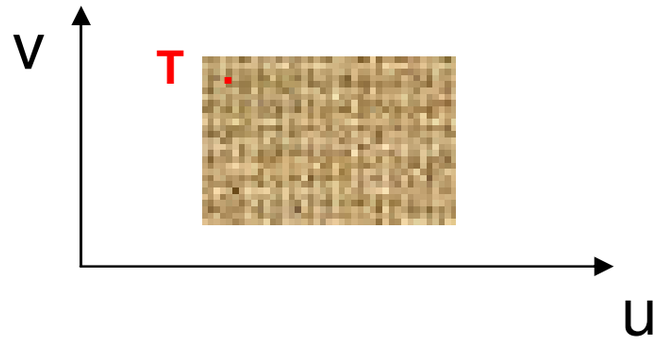
Imagem (formada por **pixels**)

M.Próspero

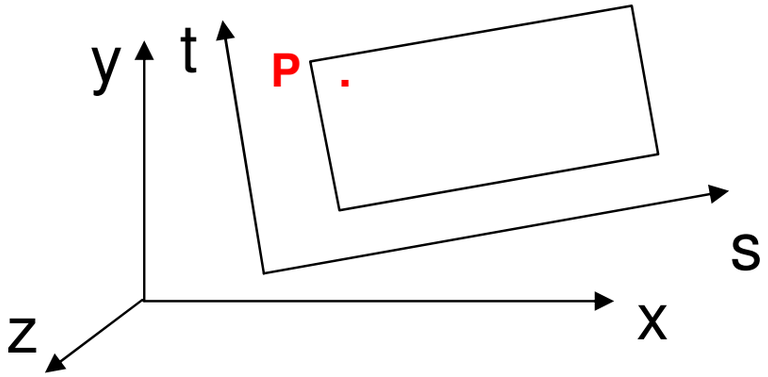
Mapeamento de uma Textura

Tratamento Matemático

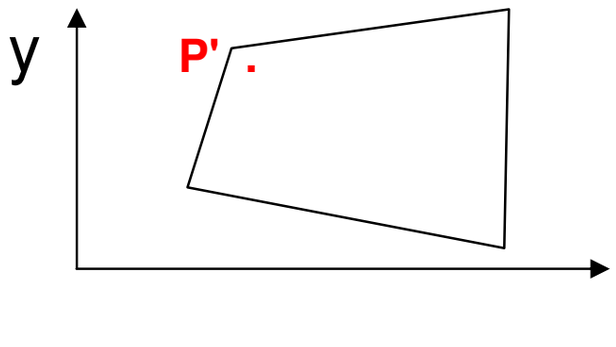
ESPAÇO DA TEXTURA

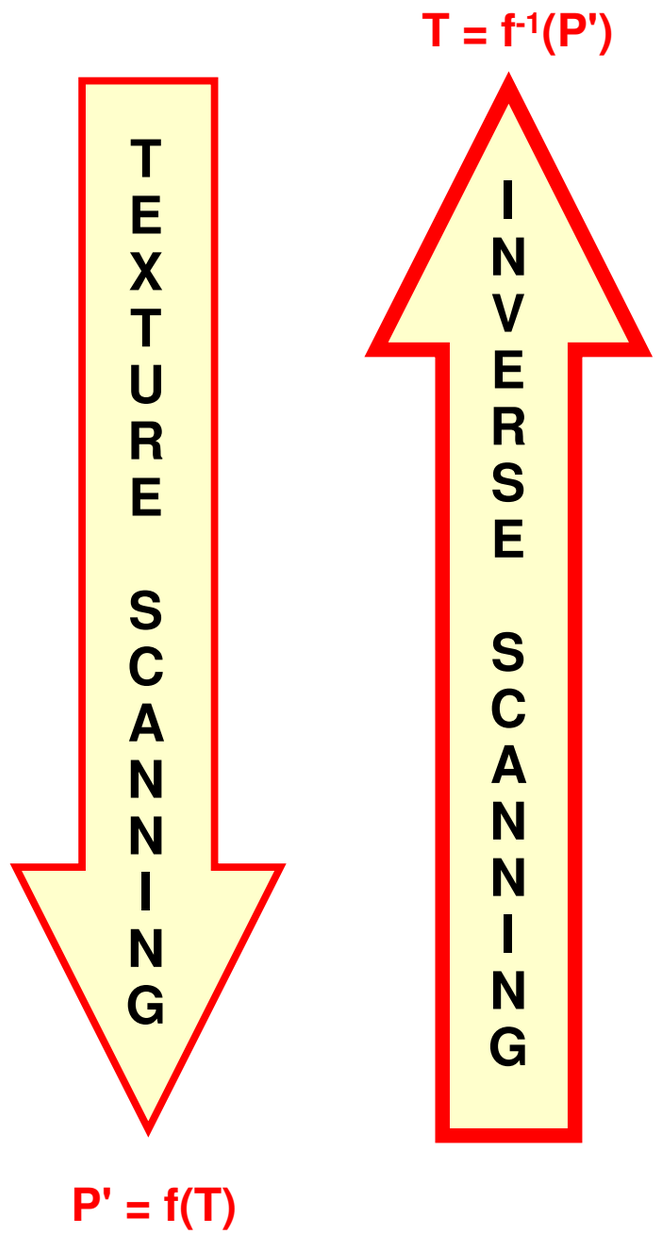


ESPAÇO DO OBJETO
(Superfície)



ESPAÇO DA IMAGEM





scanning ≡ varrimento

M.Próspero

Mapeamento de uma Textura

COMPARAÇÃO DOS MÉTODOS RELATIVOS À ORDEM DE EXECUÇÃO

TEXTURE SCANNING (varrimento da textura)

☹ Em geral a textura não coincide com um número inteiro de pixels, implicando cálculos para a subdivisão da área de um pixel.

☺ As transformações de visualização seguem-se no sentido mais natural.

INVERSE SCANNING (varrimento inverso)

☹ Exige o cálculo das transformações de visualização inversas.

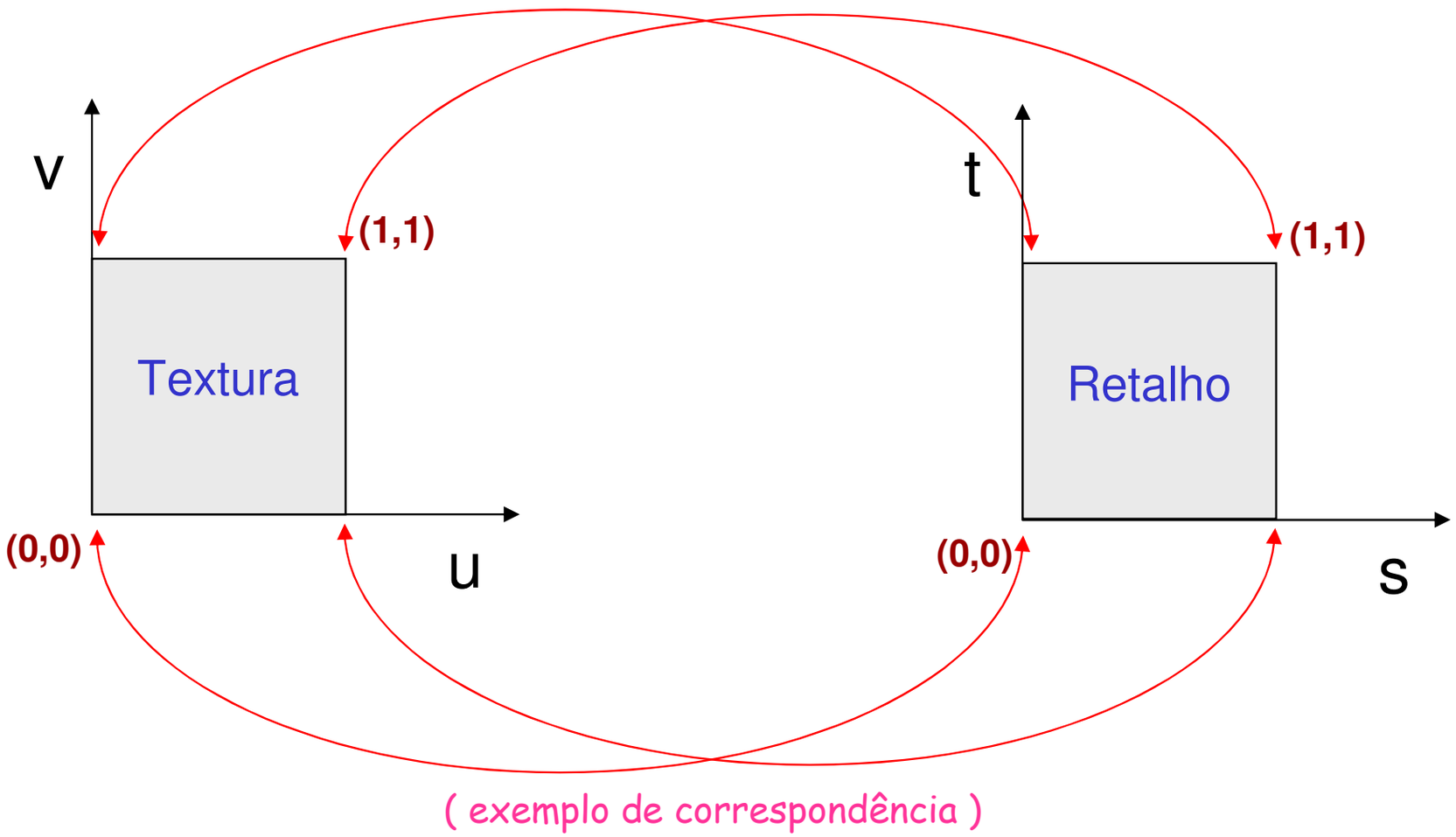
☺ Evita a subdivisão de pixels e permite, além disso, o uso de filtros para *antialiasing* (matéria posterior do curso).

Conclusão: o método usualmente preferido é o de INVERSE SCANNING.

M.Próspero

Mapeamento de uma Textura

SUPERFÍCIE DEFINIDA PARAMETRICAMENTE



(exemplo de correspondência)

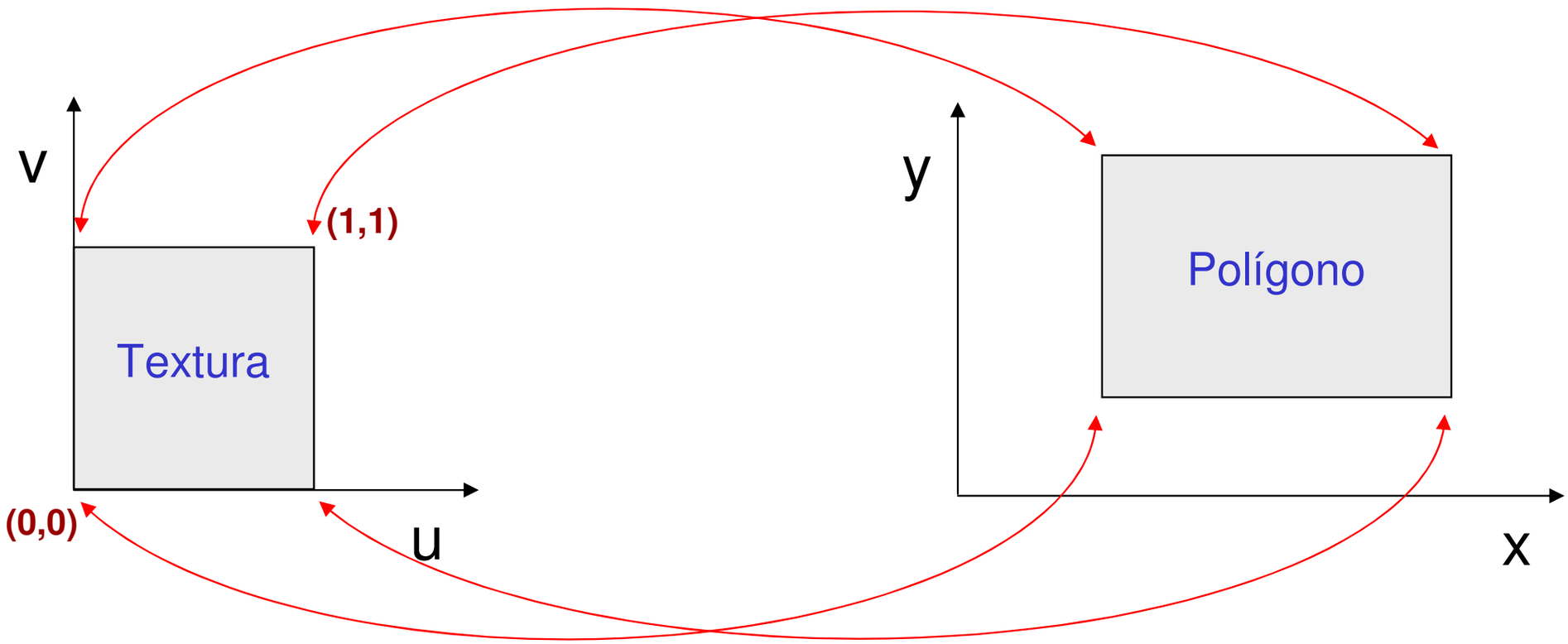
É habitual fazer-se com que este tipo de correspondência seja uma interpolação linear:

$$\begin{bmatrix} s \\ t \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

M.Próspero

Mapeamento de uma Textura

SUPERFÍCIE DEFINIDA POR UM POLÍGONO



Também é habitual que a correspondência seja uma interpolação linear
(estar sempre ciente das possíveis distorções!)

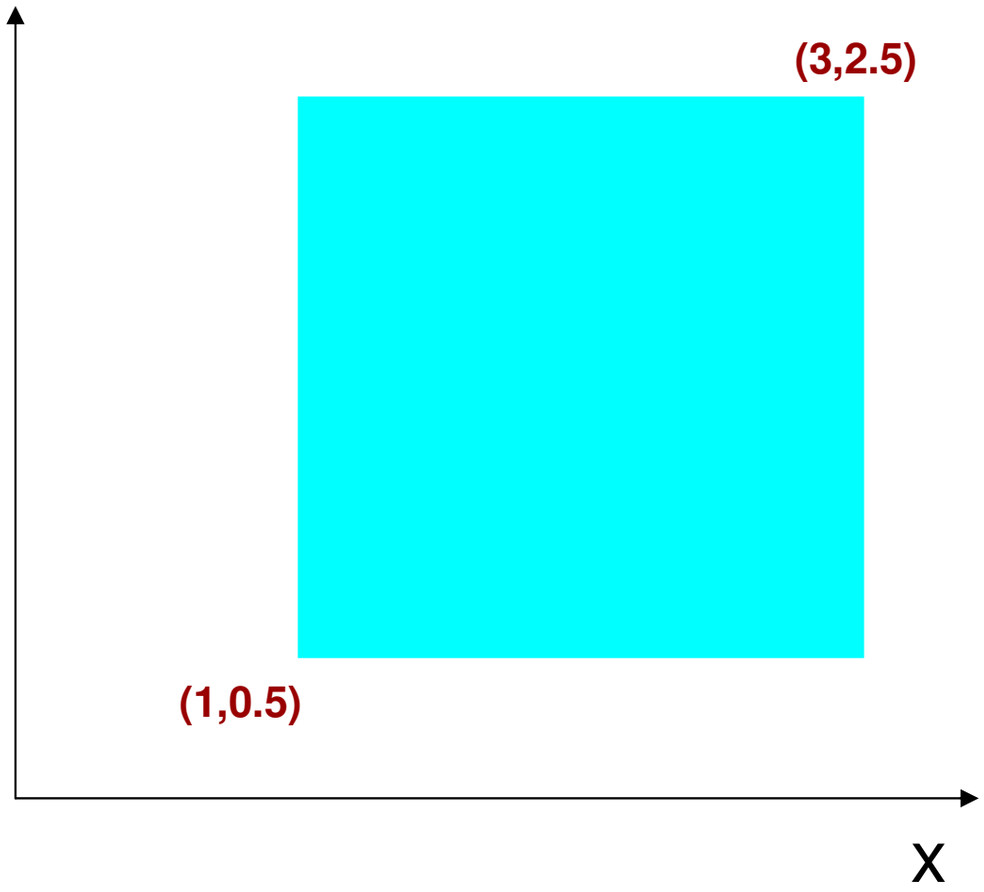
Exemplos com OpenGL

M.Próspero

Exemplos

Nota:

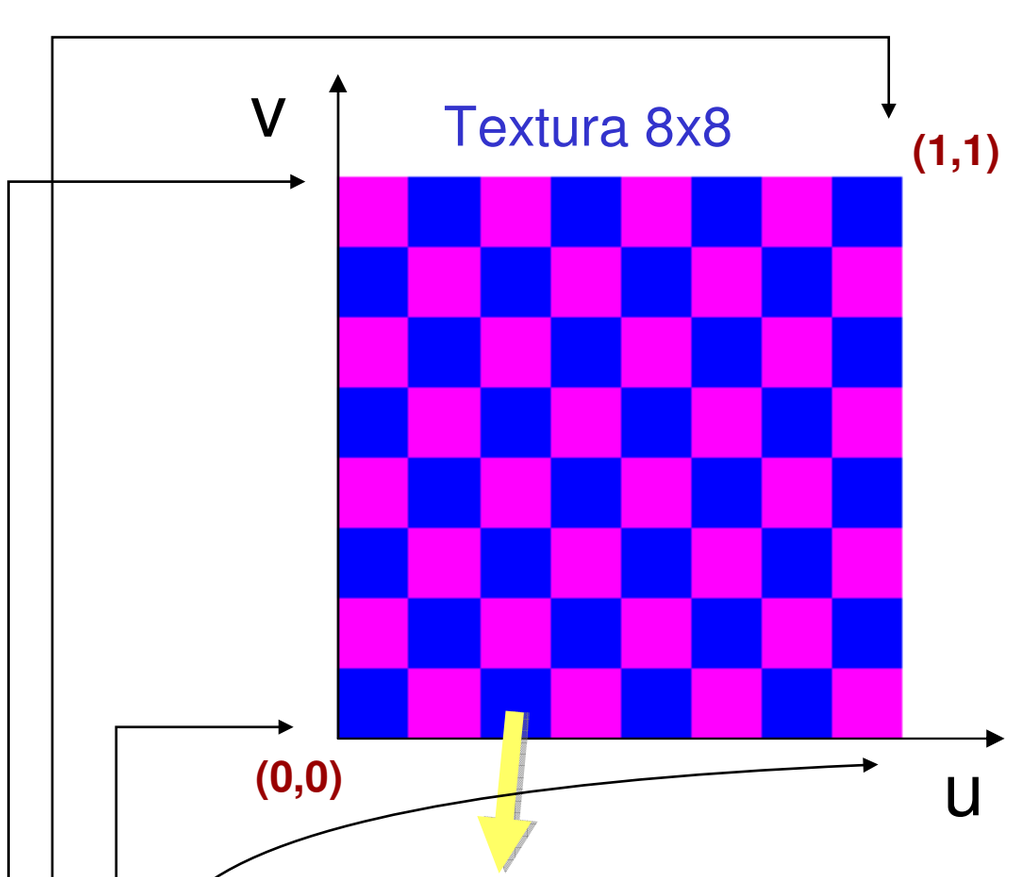
Nos exemplos seguintes está a usar-se a API do OpenGL, por comodidade. Presume-se que cada pixel seja apenas pintado com a cor da textura.

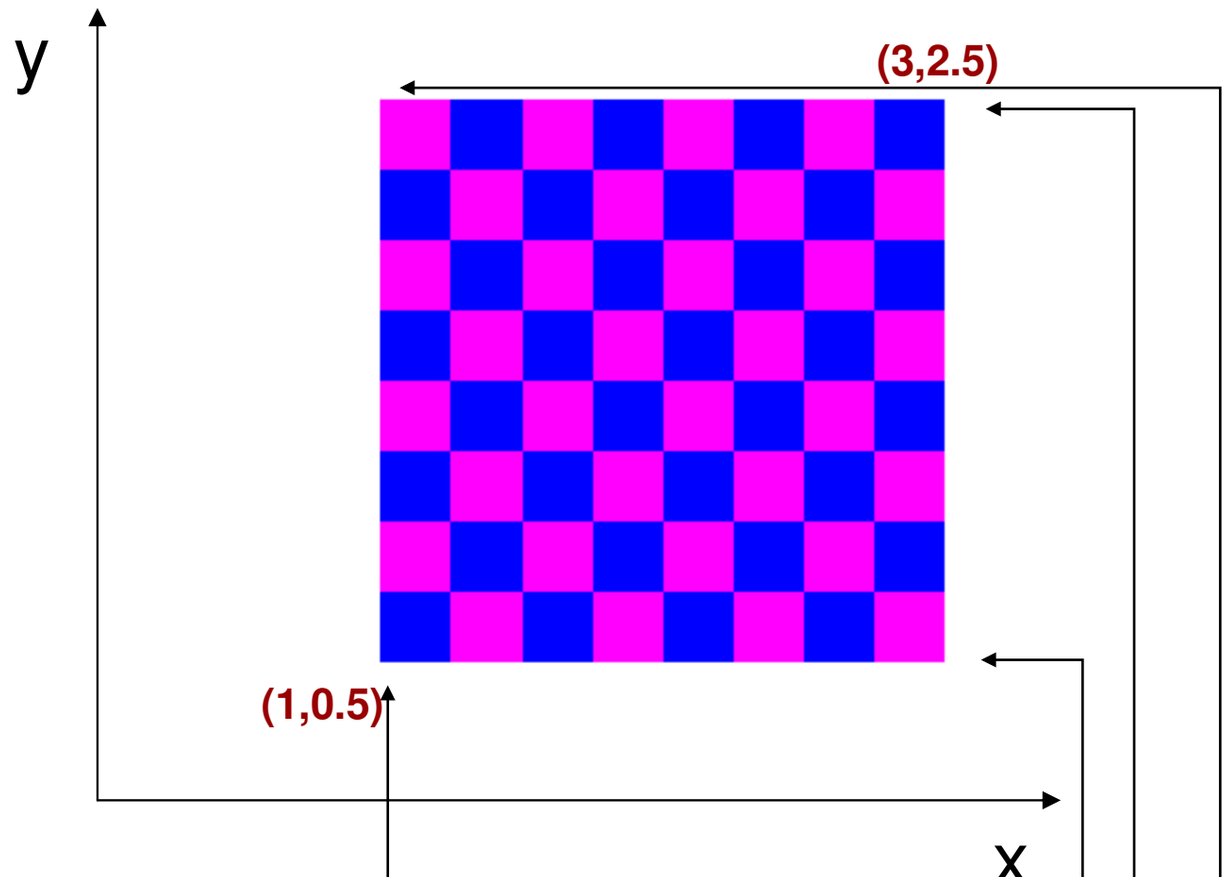



```
glColor3d(0.0, 1.0, 1.0);
glBegin(GL_QUADS);
   glVertex3d(1.0, 0.5, 0.0);
   glVertex3d(1.0, 2.5, 0.0);
   glVertex3d(3.0, 2.5, 0.0);
   glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

M.Próspero

Exemplos





Definiu-se cada quadrícula como um único texel:

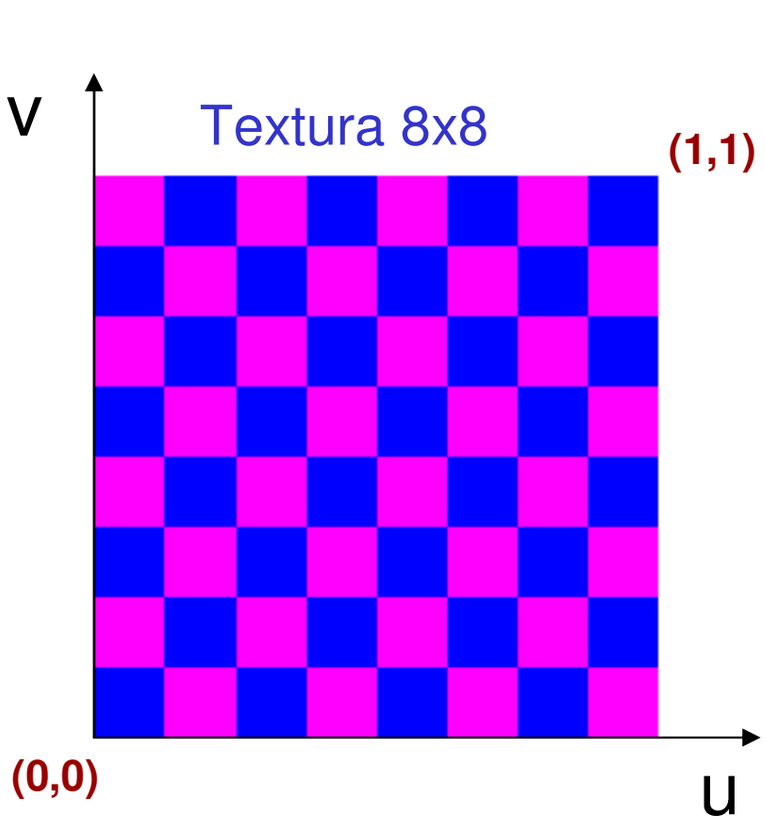
```
GLubyte textura[8][8][3];
```

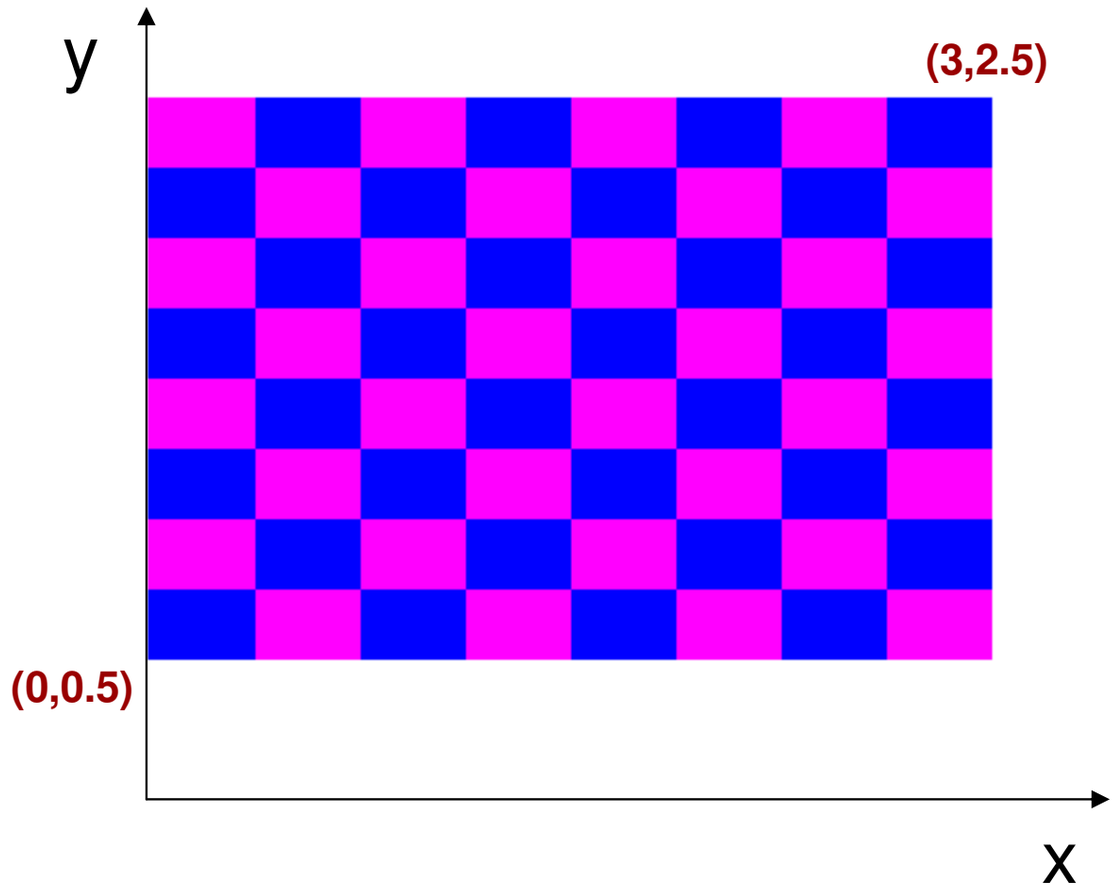
Cor RGB 24 bits

```
glBegin(GL_QUADS);
    glTexCoord2d(0.0, 0.0); glVertex3d(1.0, 0.5, 0.0);
    glTexCoord2d(0.0, 1.0); glVertex3d(1.0, 2.5, 0.0);
    glTexCoord2d(1.0, 1.0); glVertex3d(3.0, 2.5, 0.0);
    glTexCoord2d(1.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

M.Próspero

Exemplos

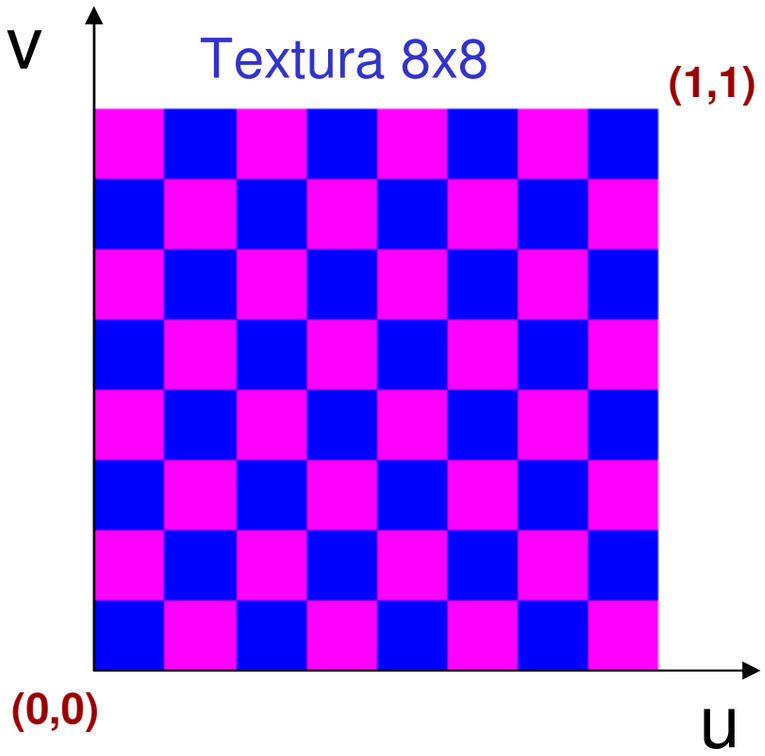


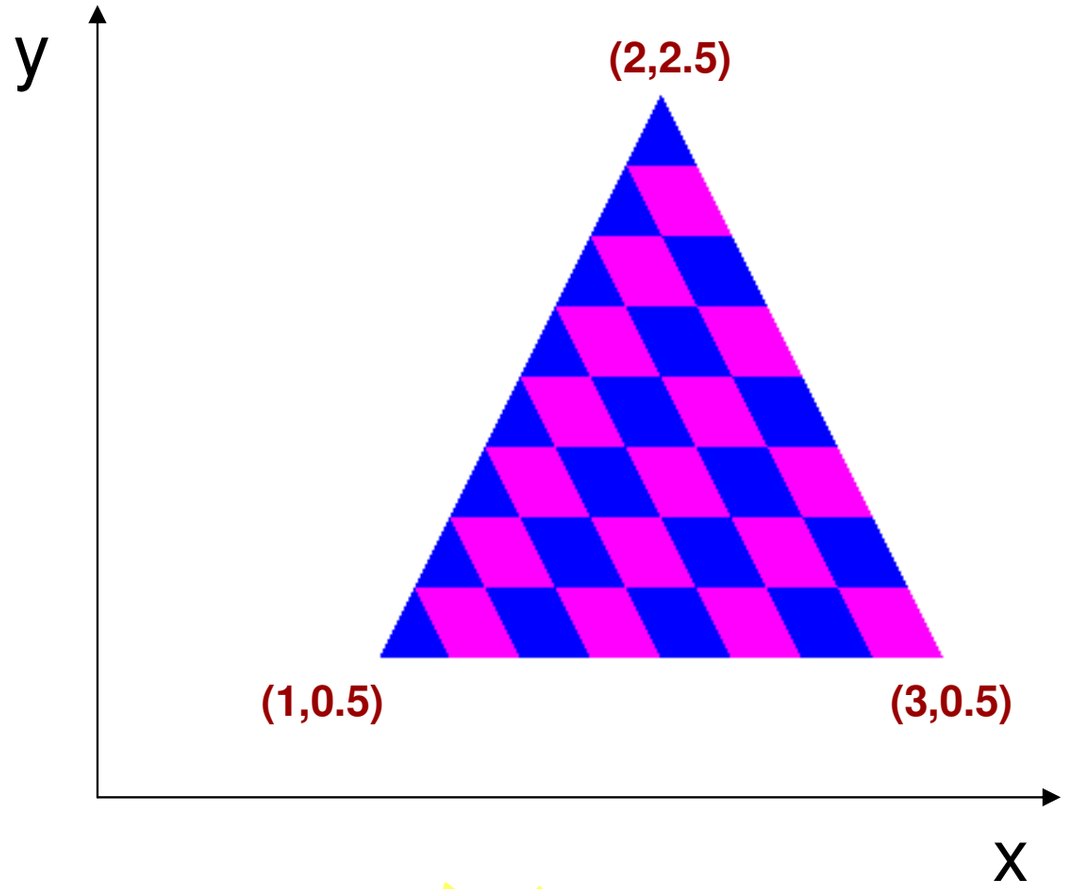



```
glBegin(GL_QUADS);
   glTexCoord2d(0.0, 0.0); glVertex3d(0.0, 0.5, 0.0);
   glTexCoord2d(0.0, 1.0); glVertex3d(0.0, 2.5, 0.0);
   glTexCoord2d(1.0, 1.0); glVertex3d(3.0, 2.5, 0.0);
   glTexCoord2d(1.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

M.Próspero

Exemplos



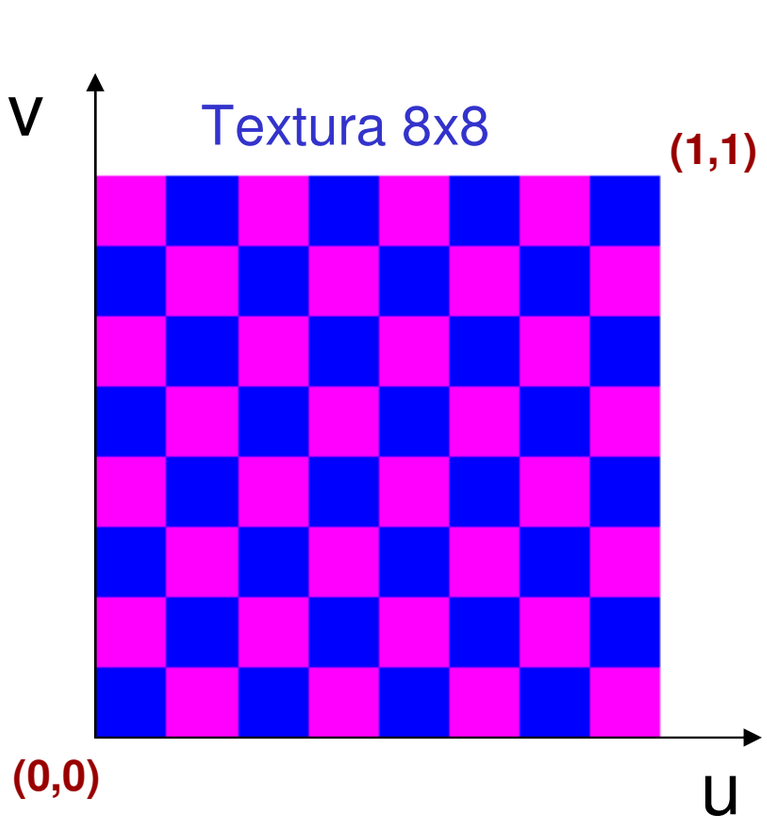


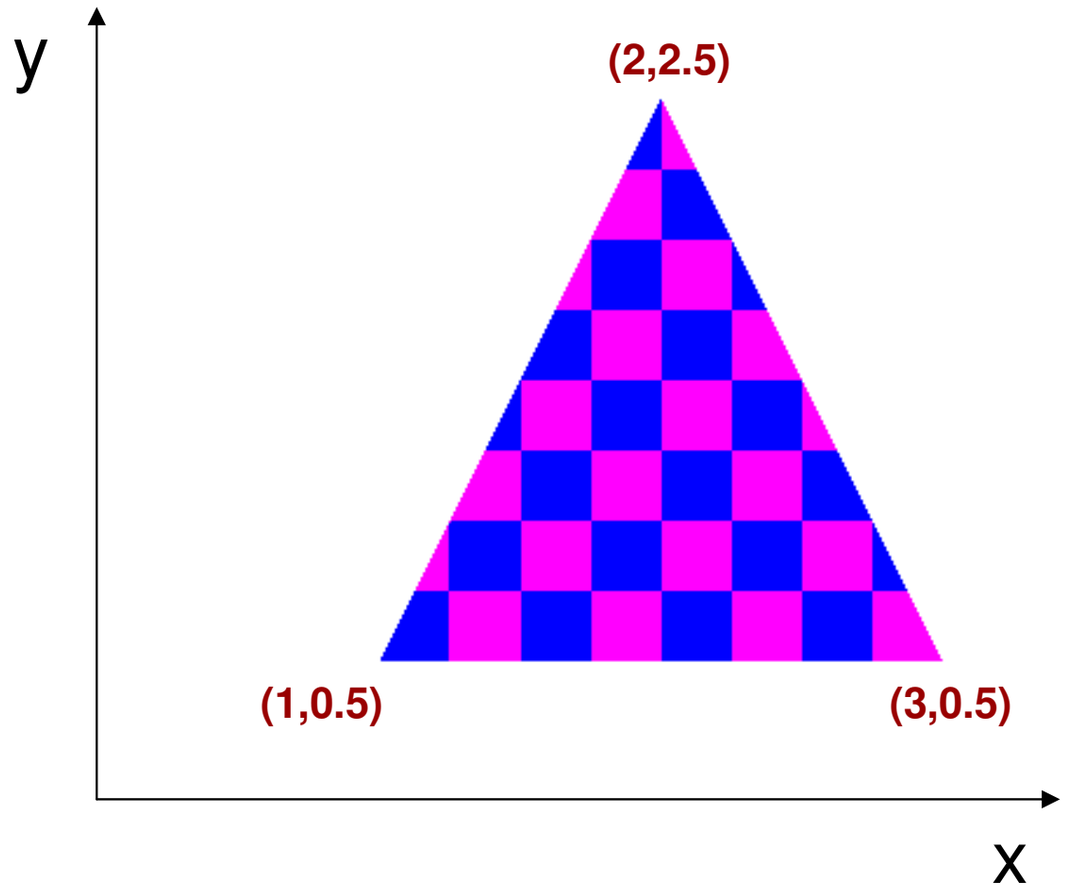
ATENÇÃO!

```
glBegin(GL_TRIANGLES);
    glTexCoord2d(0.0, 0.0); glVertex3d(1.0, 0.5, 0.0);
    glTexCoord2d(1.0, 1.0); glVertex3d(2.0, 2.5, 0.0);
    glTexCoord2d(1.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

M.Próspero

Exemplos

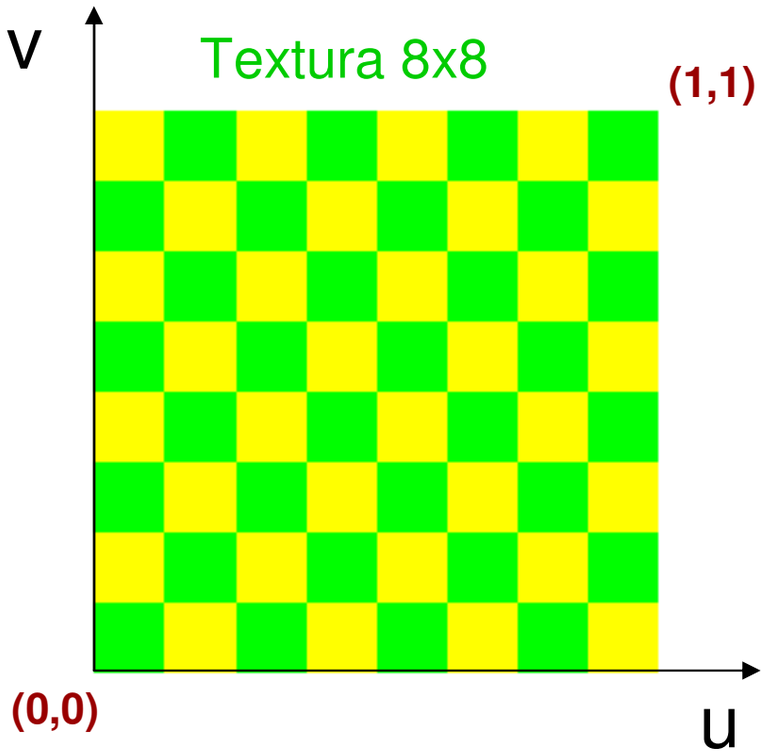


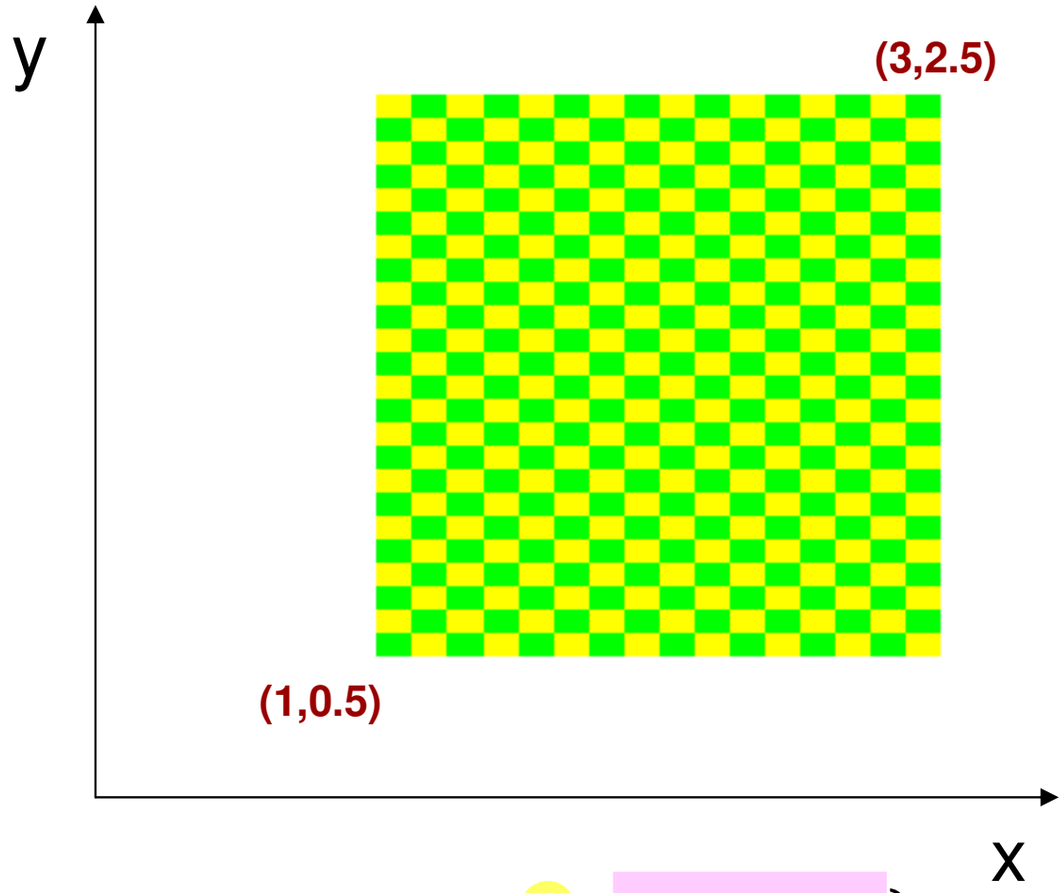



```
glBegin(GL_TRIANGLES);
    glTexCoord2d(0.0, 0.0); glVertex3d(1.0, 0.5, 0.0);
    glTexCoord2d(0.5, 1.0); glVertex3d(2.0, 2.5, 0.0);
    glTexCoord2d(1.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

M.Próspero

Exemplos





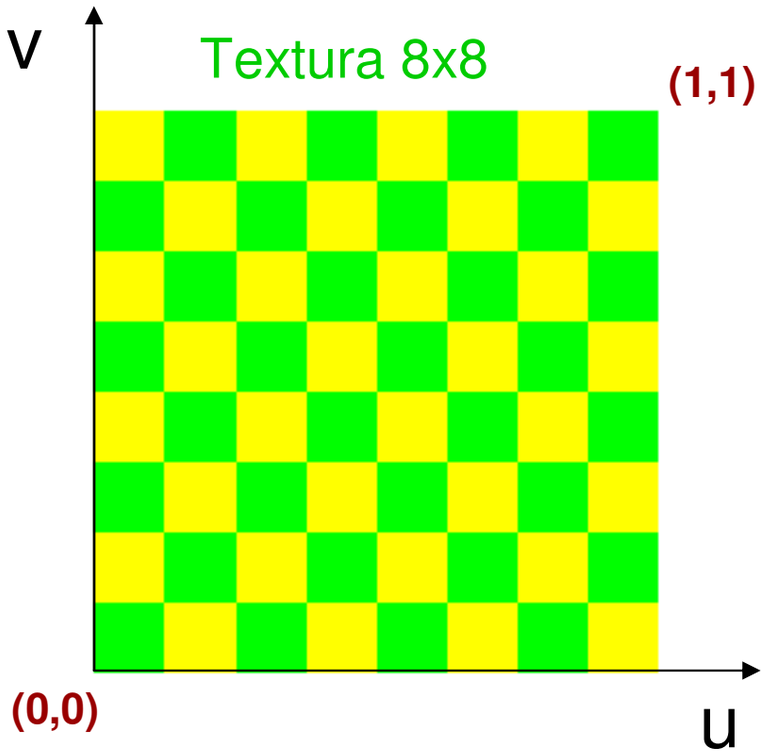

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);

glBegin(GL_QUADS);
    glTexCoord2d(0.0, 0.0); glVertex3d(1.0, 0.5, 0.0);
    glTexCoord2d(0.0, 3.0); glVertex3d(1.0, 2.5, 0.0);
    glTexCoord2d(2.0, 3.0); glVertex3d(3.0, 2.5, 0.0);
    glTexCoord2d(2.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

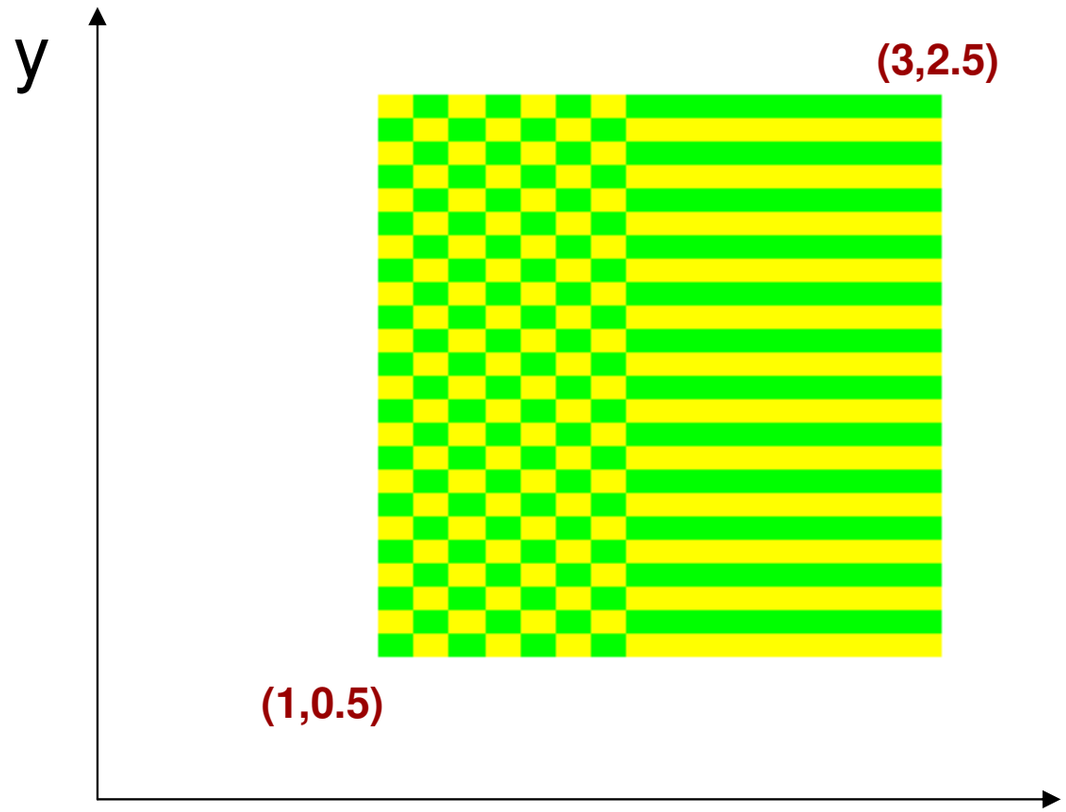
Notação das coordenadas de textura em OpenGL !

Repetição da textura se o parâmetro cair fora do intervalo

M.Próspero

Exemplos





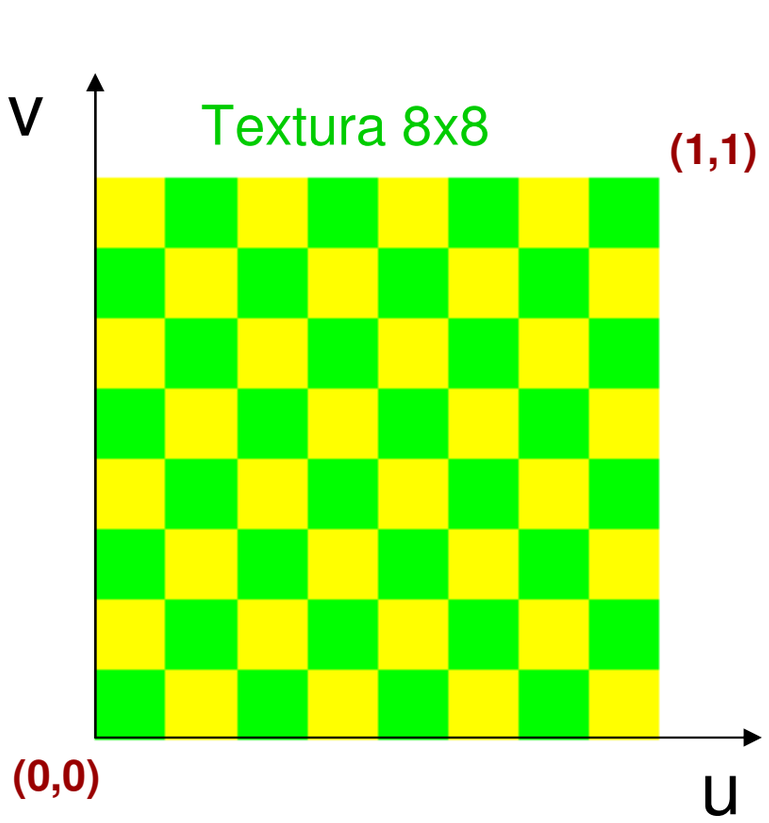

```
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);

glBegin(GL_QUADS);
    glTexCoord2d(0.0, 0.0); glVertex3d(1.0, 0.5, 0.0);
    glTexCoord2d(0.0, 3.0); glVertex3d(1.0, 2.5, 0.0);
    glTexCoord2d(2.0, 3.0); glVertex3d(3.0, 2.5, 0.0);
    glTexCoord2d(2.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

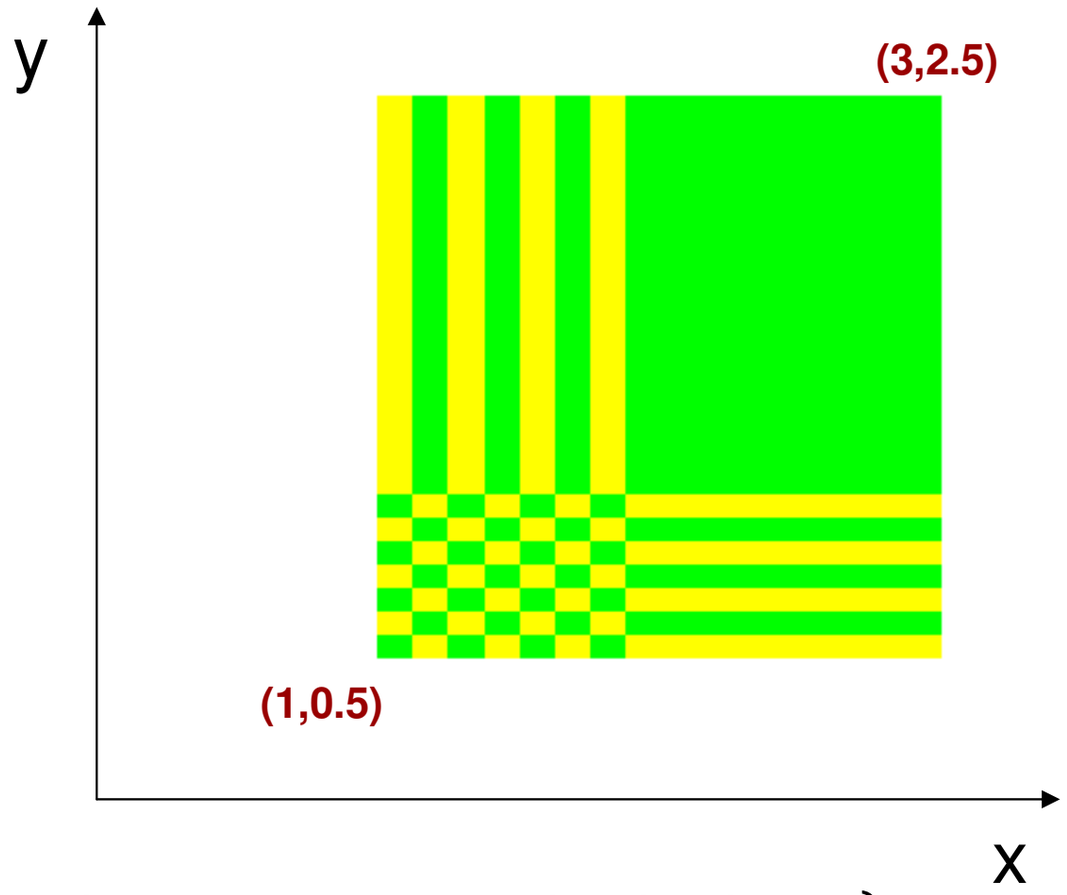
u>1 → u=1

u<0 → u=0

M.Próspero

Exemplos






```
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP);

glBegin(GL_QUADS);
    glTexCoord2d(0.0, 0.0); glVertex3d(1.0, 0.5, 0.0);
    glTexCoord2d(0.0, 3.0); glVertex3d(1.0, 2.5, 0.0);
    glTexCoord2d(2.0, 3.0); glVertex3d(3.0, 2.5, 0.0);
    glTexCoord2d(2.0, 0.0); glVertex3d(3.0, 0.5, 0.0);
glEnd();
```

M.Próspero

Adaptação para WebGL

- As coordenadas de textura são enviadas como atributos dos vértices
- A função glTexParameter (OpenGL) tem uma equivalência directa gl.texParameter (em WebGL)

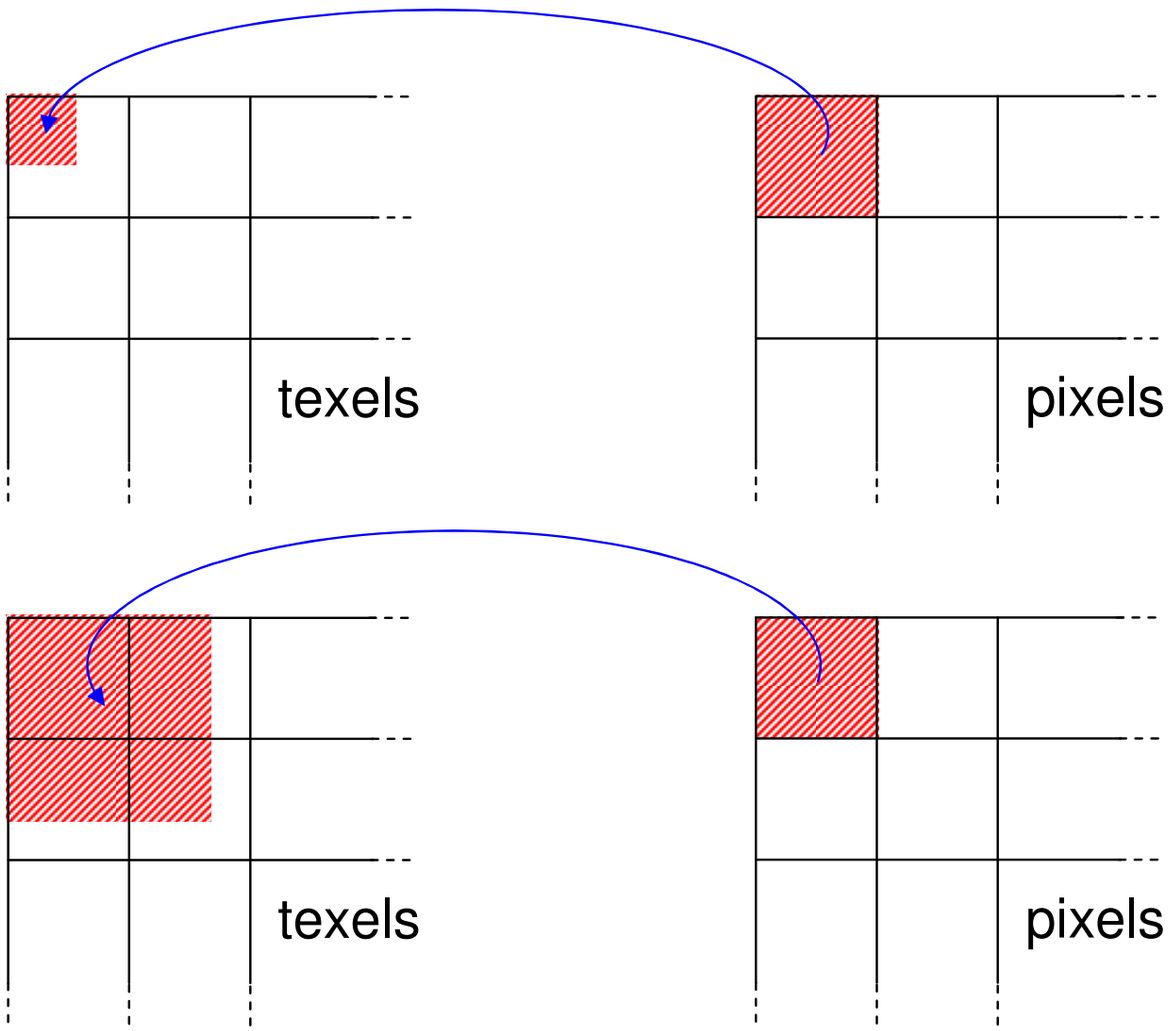
- exemplos:

```
// Prevents s-coordinate wrapping (repeating).

gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_S, gl.CLAMP_TO_EDGE);

// Prevents t-coordinate wrapping (repeating).

gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_T, gl.CLAMP_TO_EDGE);
```

Texturas

CORRESPONDÊNCIA ENTRE PIXELS E TEXELS (1)

Um pixel tem uma determinada área. Quando se faz a correspondência com a textura, a área aí obtida poderá ser diferente da área de um texel.



Pixel pintado com **ampliação** (MAGNIFICATION) da textura

Pixel pintado com **redução** (MINIFICATION) da textura

Nestes casos, qual o valor com que se deve pintar o pixel ?

Obs.: Havendo distorção, a área na textura não será um quadrado.

M.Próspero

Texturas

CORRESPONDÊNCIA ENTRE PIXELS E TEXELS (2)

Soluções em alternativa (que se podem misturar), tanto para **ampliação** como para **redução**:

Em WebGL:

- Usar-se o texel mais próximo do ponto calculado pela correspondência com o (centro do) pixel.

```
gl.texParameteri(gl.TEXTURE_2D,
                 gl.TEXTURE_MAG_FILTER, gl.NEAREST);
gl.texParameteri(gl.TEXTURE_2D,
                 gl.TEXTURE_MIN_FILTER, gl.NEAREST);
```

- Fazer-se uma média ponderada com os 2x2 texels mais próximos do ponto calculado pela correspondência com o pixel (**filtragem**).

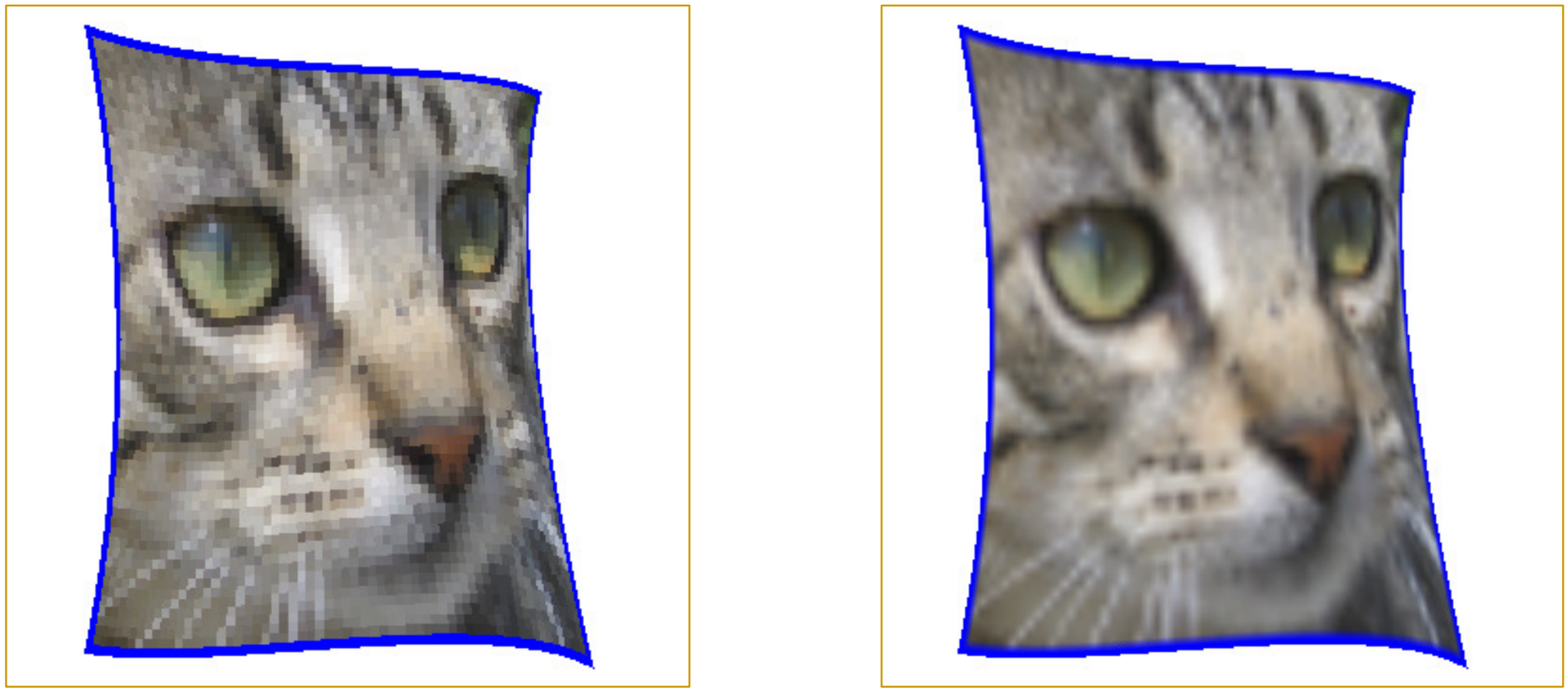
```
gl.texParameteri(gl.TEXTURE_2D,
                 gl.TEXTURE_MAG_FILTER, gl.LINEAR);
gl.texParameteri(gl.TEXTURE_2D,
                 gl.TEXTURE_MIN_FILTER, gl.LINEAR);
```

M.Próspero

CORRESPONDÊNCIA ENTRE PIXELS E TEXELS (3)

EXEMPLOS DE RESULTADOS DE APLICAÇÃO

64x64 texels



360x360 pixels

GL_NEAREST

GL_LINEAR

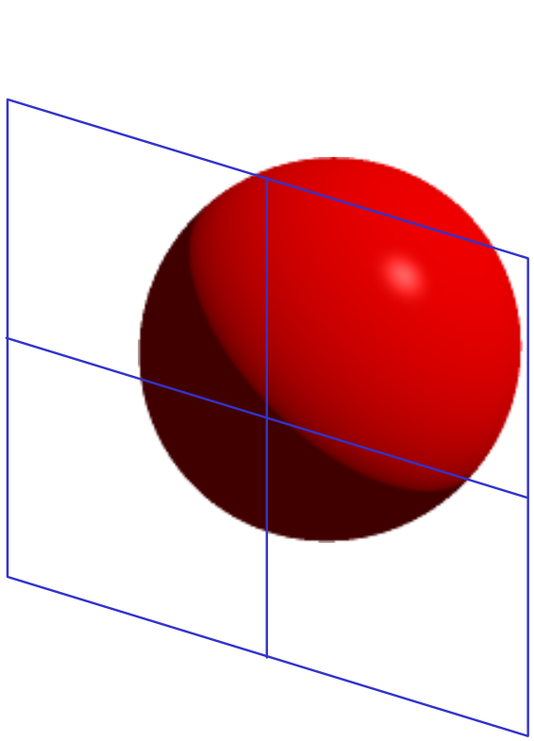
mais lento,
melhor resultado

M.Próspero

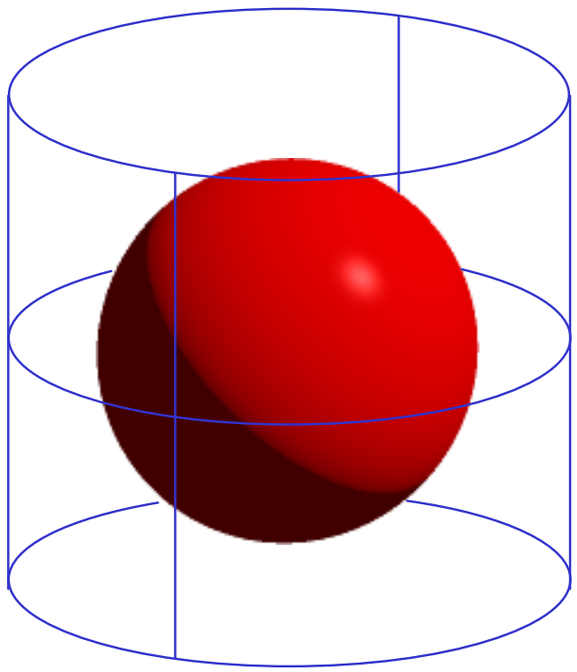
Exemplos de Mapeamentos Clássicos

TEXTURAS MAPEADAS

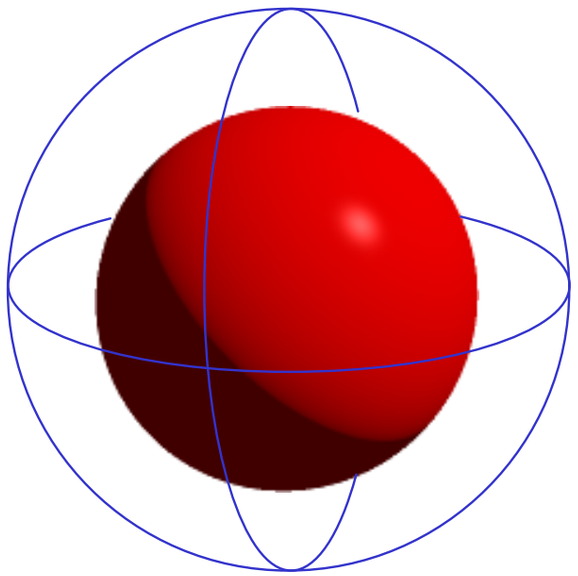
TIPOS CLÁSSICOS DE MAPEAMENTO:



Ortogonal



Cilíndrico



Esférico

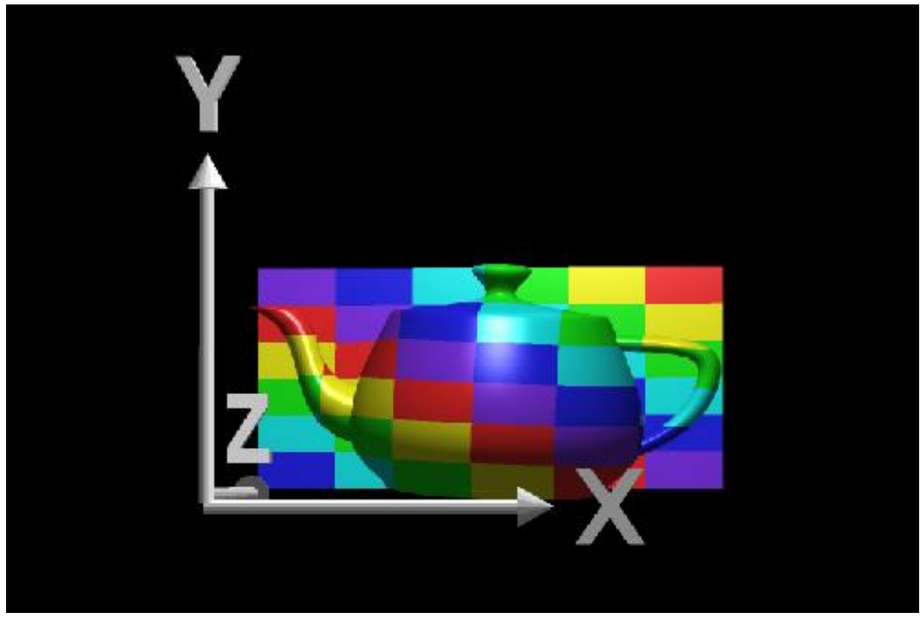
Os slides seguintes, num referencial segundo a regra da mão esquerda, são da autoria de

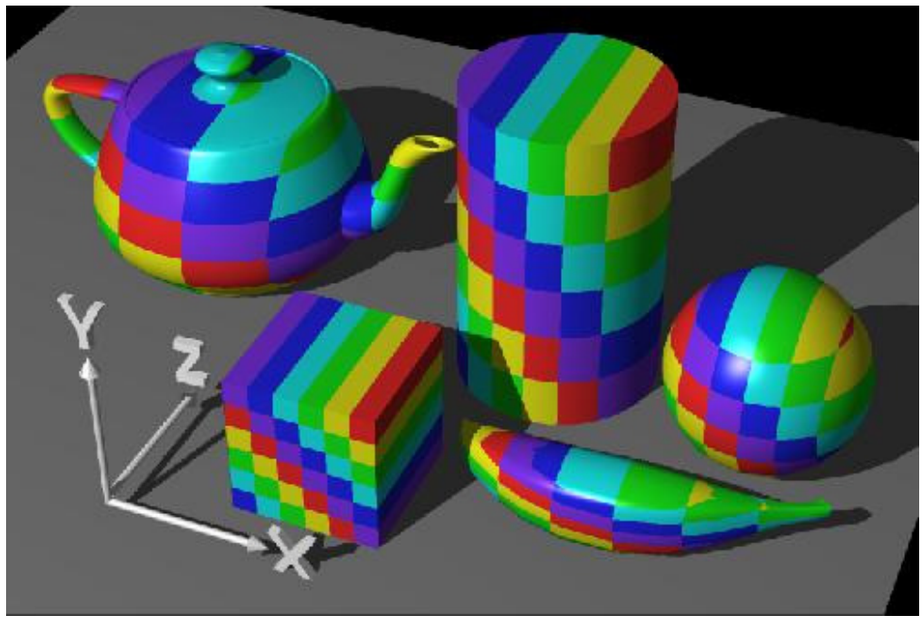
Rosalee Wolfe
SIGGRAPH 97 Education Slide set

M.Próspero

Exemplos de Mapeamentos Clássicos

Mapeamento Ortogonal



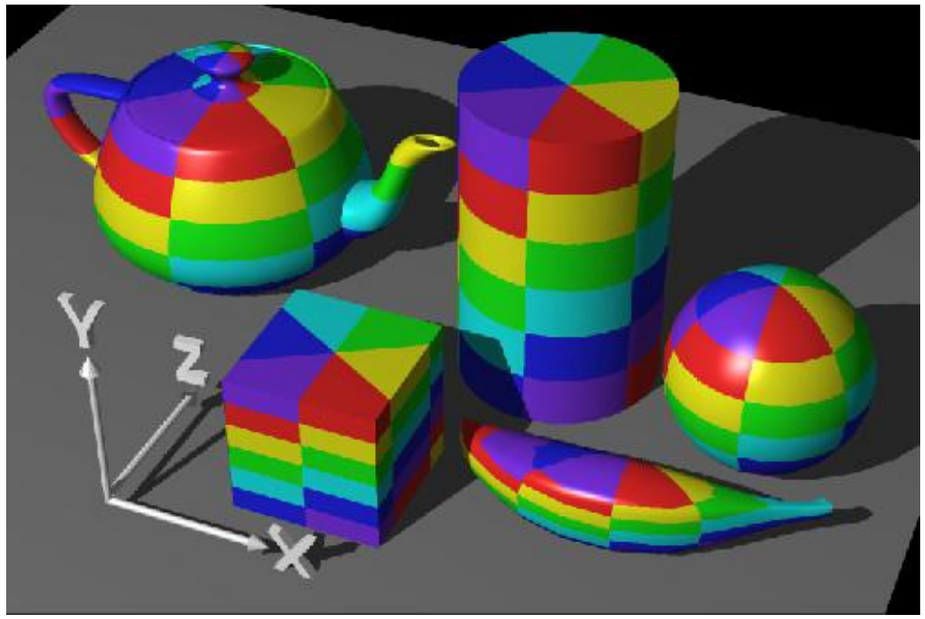


Qual a correspondência entre o espaço da textura e o espaço do objeto ?

M.Próspero

Exemplos de Mapeamentos Clássicos

Mapeamento Cilíndrico

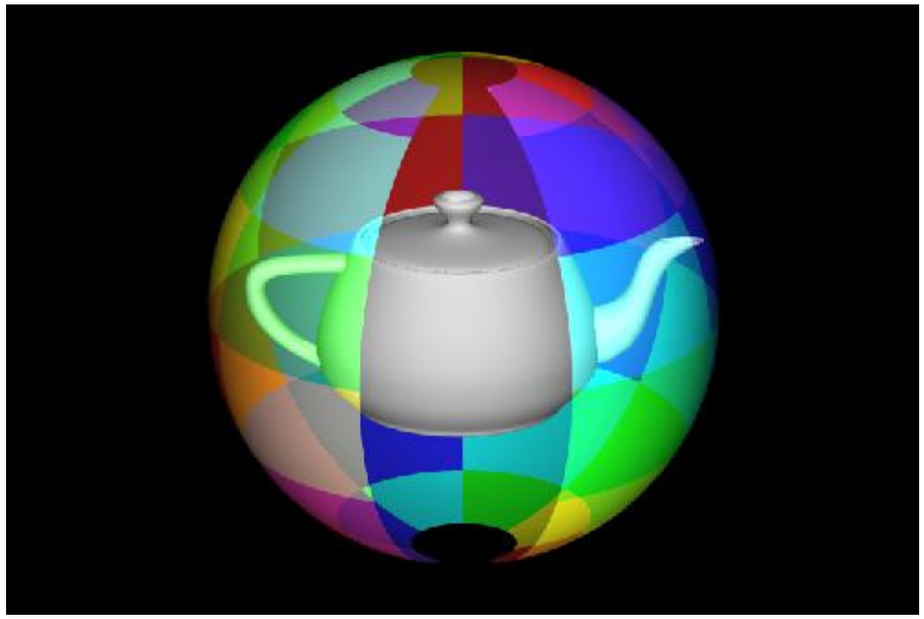


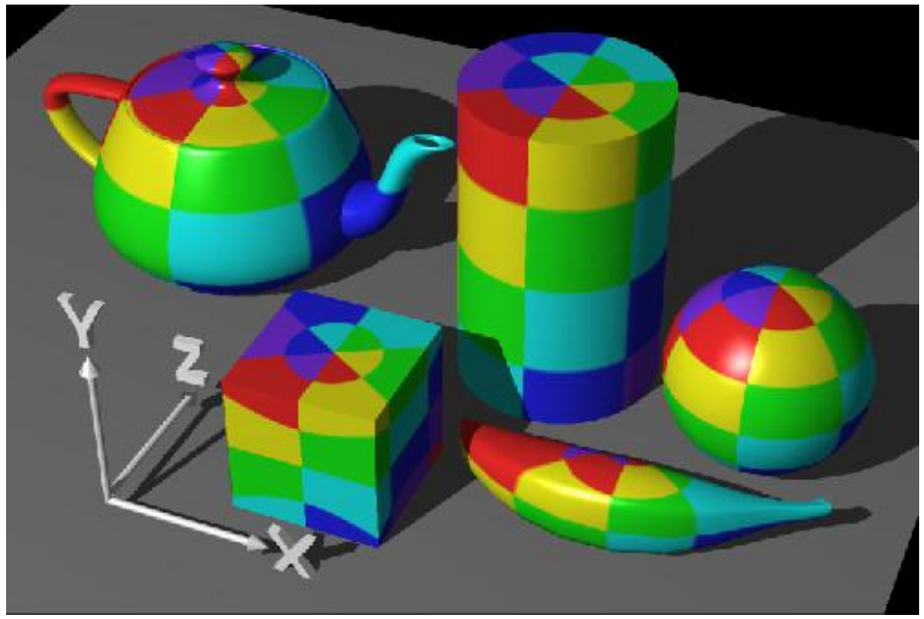
Qual a correspondência entre o espaço da textura e o espaço do objeto ?

M.Próspero

Exemplos de Mapeamentos Clássicos

Mapeamento Esférico





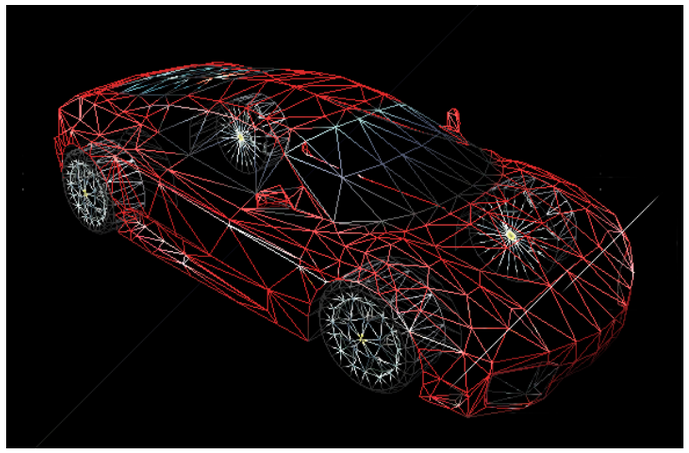
Qual a correspondência entre o espaço da textura e o espaço do objeto ?

M.Próspero

Texture Atlas

512x512

+



From Roland's VRML97 Site

M.Próspero