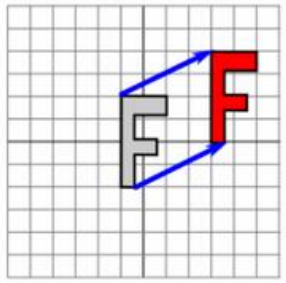
# Computação Gráfica

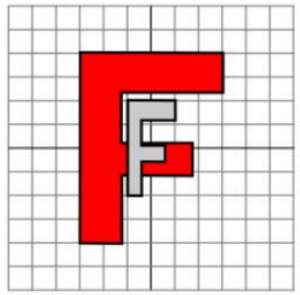
## Aula 06 – Mapeamento de Texturas

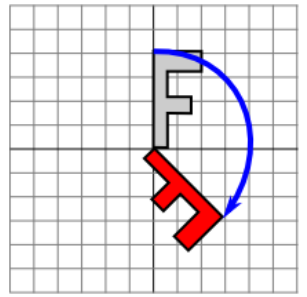
Prof. Jean R. Ponciano

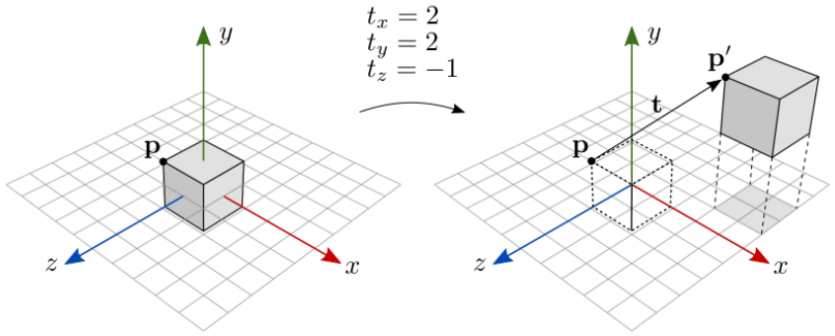
# Mapeamento de Texturas

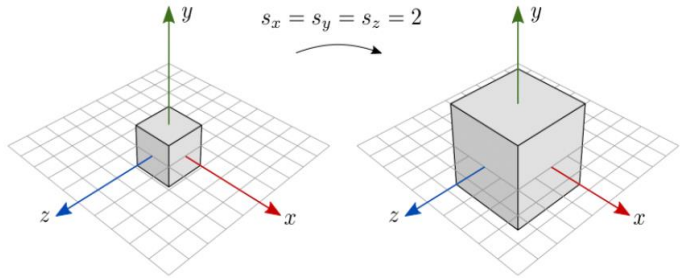
- Aprendemos a renderizar um objeto estático
- Aprendemos a fazer transformações geométricas em 2D e 3D

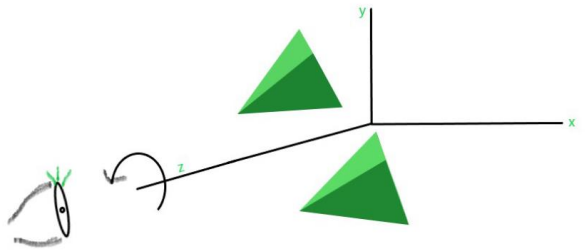






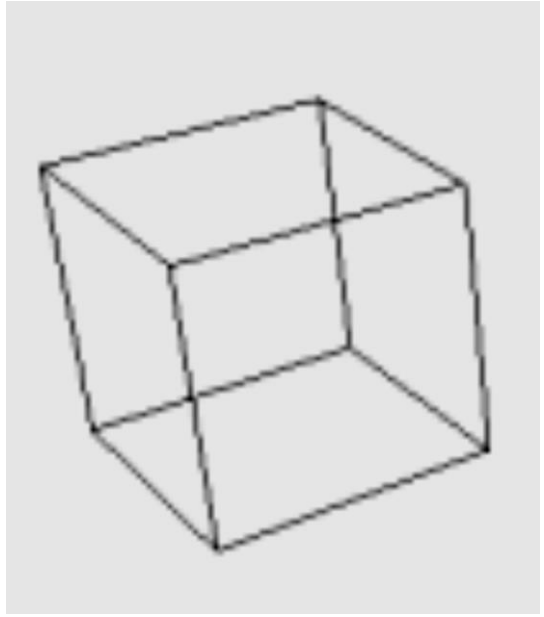


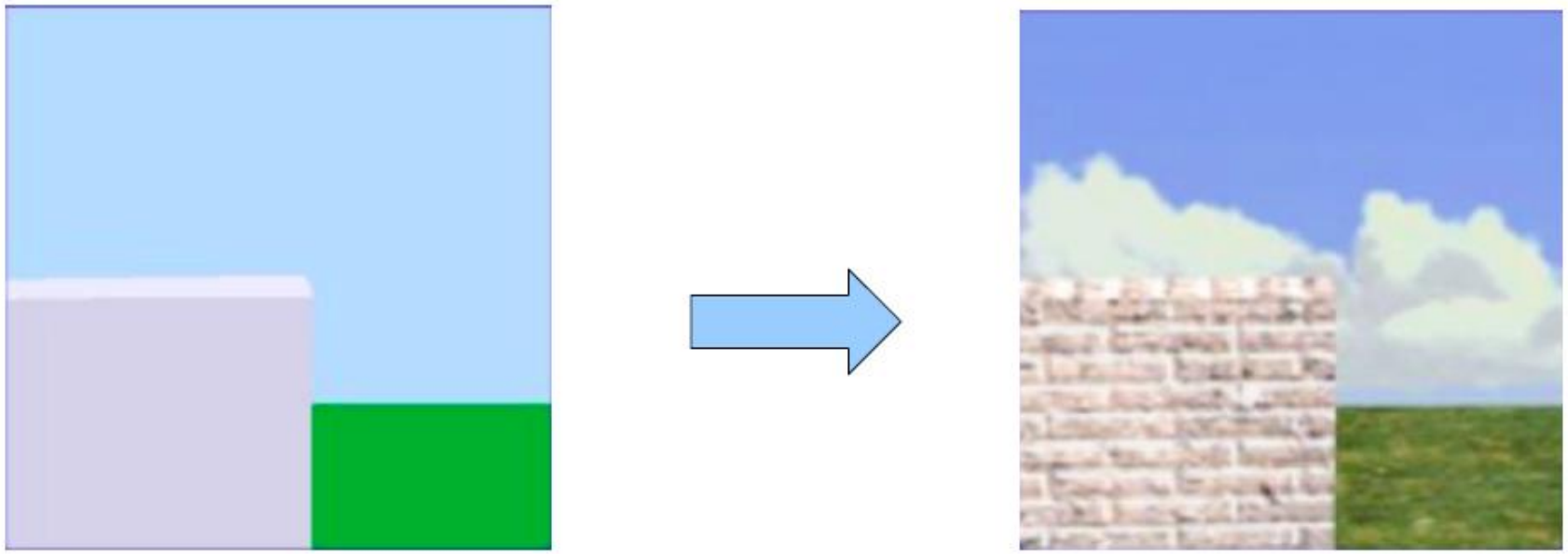
# Mapeamento de Texturas

- Agora...



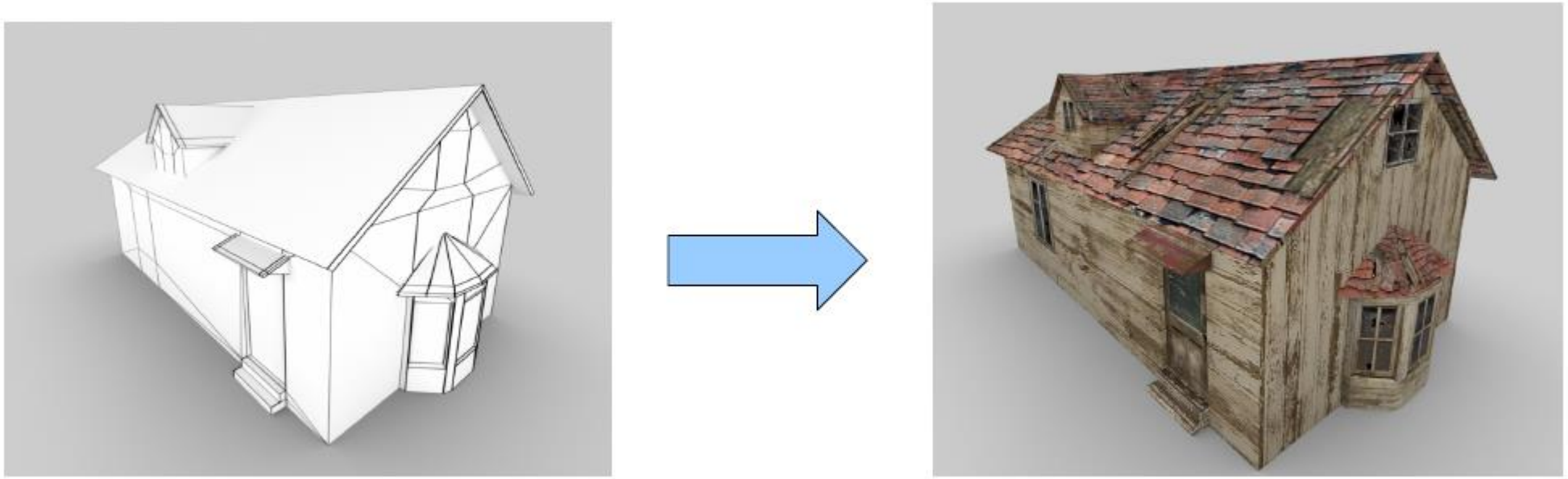
# Mapeamento de Texturas

- Agora...

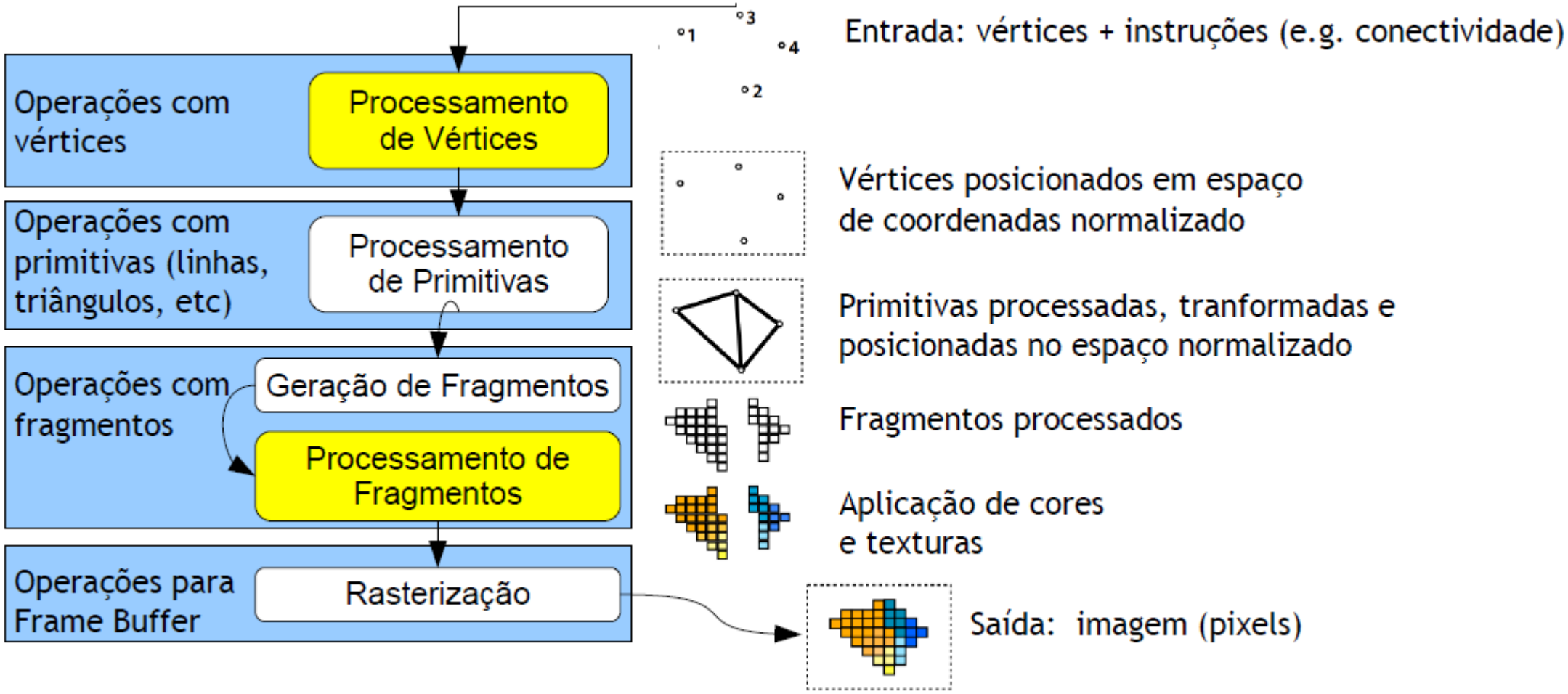


# Mapeamento de Texturas

- Agora...



# Mapeamento de Texturas



# Mapeamento de Texturas

- A malha poligonal é uma coleção de faces
  - Cada face é um conjunto de vértices
  - Formam triângulos que compõem objetos complexos

# Mapeamento de Texturas

- A malha poligonal é uma coleção de faces
  - Cada face é um conjunto de vértices
  - Formam triângulos que compõem objetos complexos

- Em muitos casos, apenas malhas e cores não são suficientes para oferecer um nível de detalhes mais realista



# Mapeamento de Texturas

- A malha poligonal é uma coleção de faces
  - Cada face é um conjunto de vértices
  - Formam triângulos que compõem objetos complexos

- Em muitos casos, apenas malhas e cores não são suficientes para oferecer um nível de detalhes mais realista

- **Texturas** ajudam nesse processo!
  - Podemos **mapear uma imagem na superfície do nosso objeto 3D**.

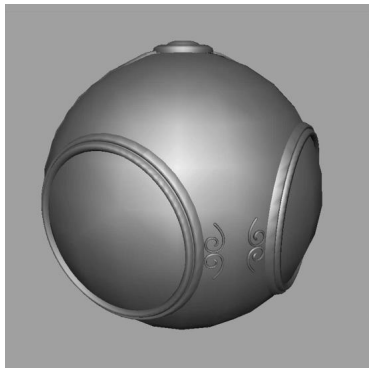
# Mapeamento de Texturas

- O mapeamento de texturas provê uma forma eficiente de lidar com as diferenças de reflectância difusa ponto-a-ponto em uma superfície (Catmull, 1975) Cofundador da Pixar

# Mapeamento de Texturas

- O mapeamento de texturas provê uma forma eficiente de lidar com as diferenças de reflectância difusa ponto-a-ponto em uma superfície (Catmull, 1975) Cofundador da Pixar

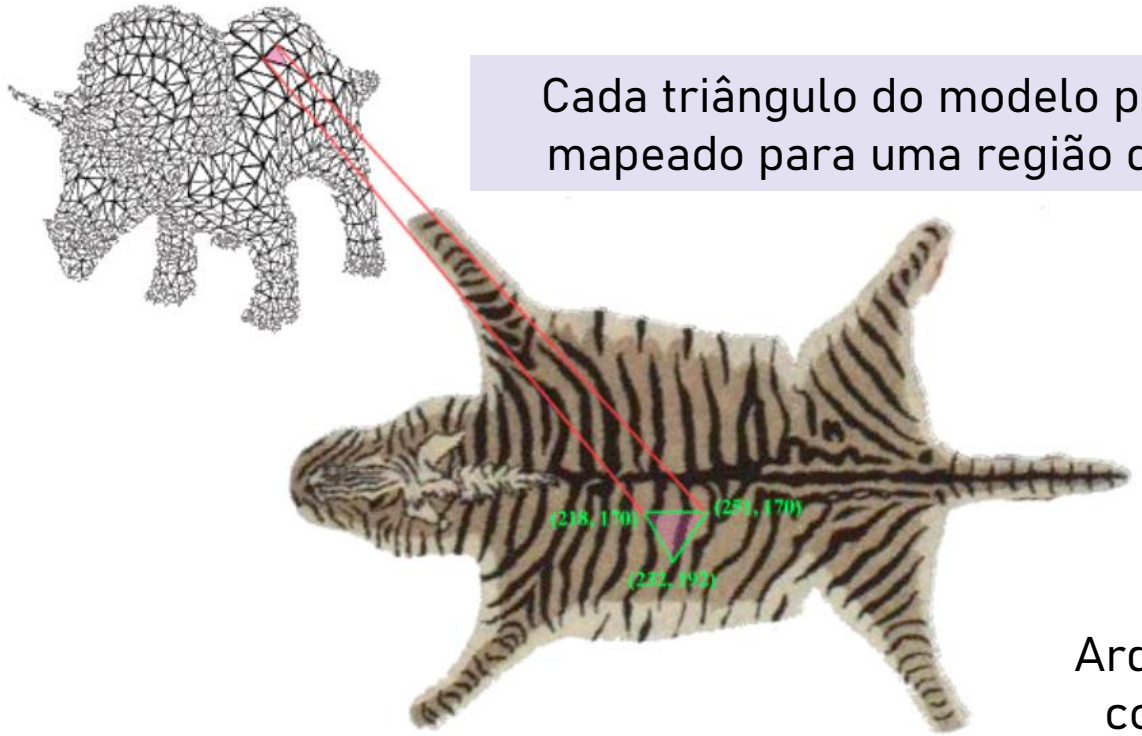
  - Mais eficiente do que tentar reproduzir o mesmo efeito apenas com iluminação e detalhes na geometria.



# Mapeamento de Texturas

- O mapeamento de texturas envolve mapear uma imagem (texturas) à superfície (triângulos) do modelo 3D

Modelo/Objeto 3D (malha)

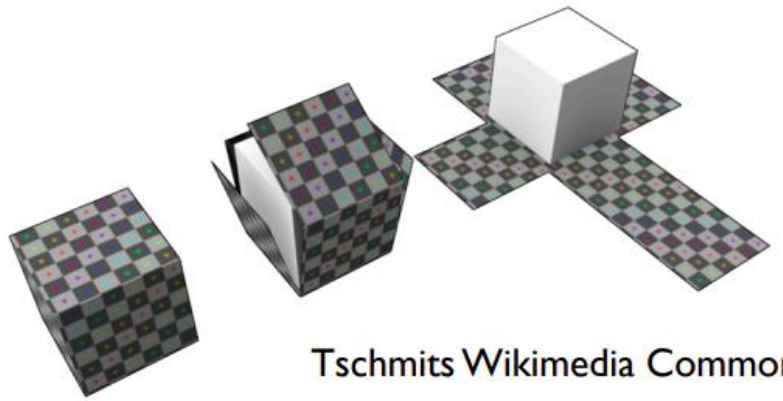


Cada triângulo do modelo precisa ser mapeado para uma região da textura

Arquivo/Imagem com a textura

# Mapeamento de Texturas

- O mapeamento de texturas envolve mapear uma imagem (texturas) à superfície (triângulos) do modelo 3D
- O arquivo/imagem de textura é gerado de forma apropriada para o modelo 3D
  - Não vamos estudar como gerar esse arquivo, mas sim em como fazer o mapeamento textura / modelos



Tschmits Wikimedia Commons

# Conceitos básicos

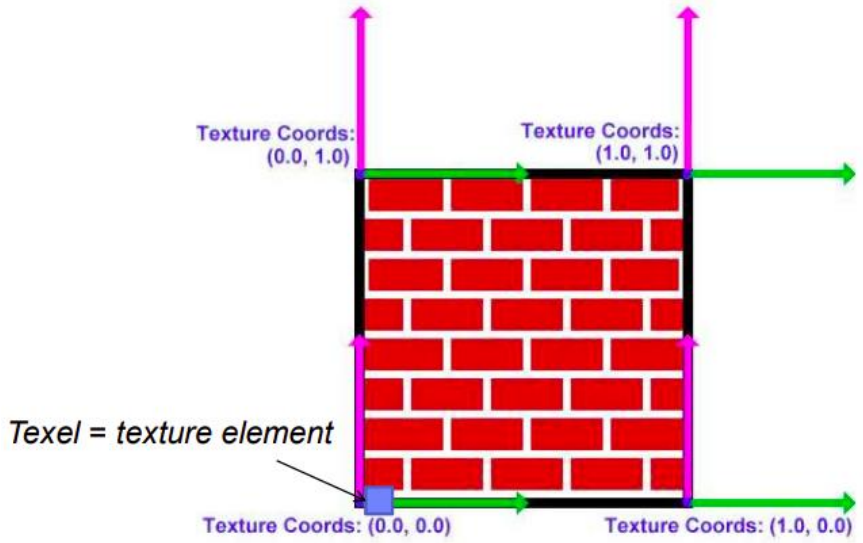
- A textura é um arranjo 2D de tamanho $u$ x $v$

Também existem texturas:
1D (caracterizam curvas)
3D (descrevem material sólido a partir do qual um objeto será esculpido)

Nosso foco é na textura 2D, a mais comumente utilizada.

# Conceitos básicos

- A textura é um arranjo 2D de tamanho u x *v*
- Cada elemento desse arranjo é chamado de *texel (texture element)*



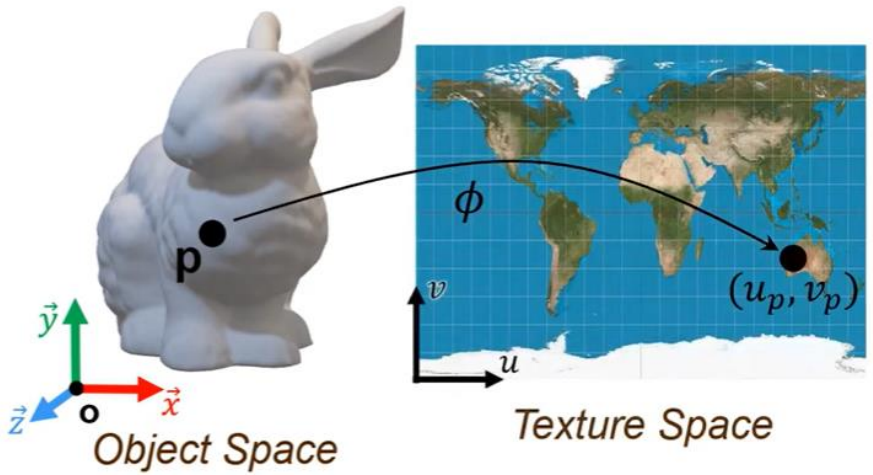

As coordenadas de textura são definidas no intervalo [0,1]

# Mapeamento

- Problema: nosso modelo é uma malha (triangular) 3D
- As texturas são mapas/arranjos 2D
- Como mapeá-los?

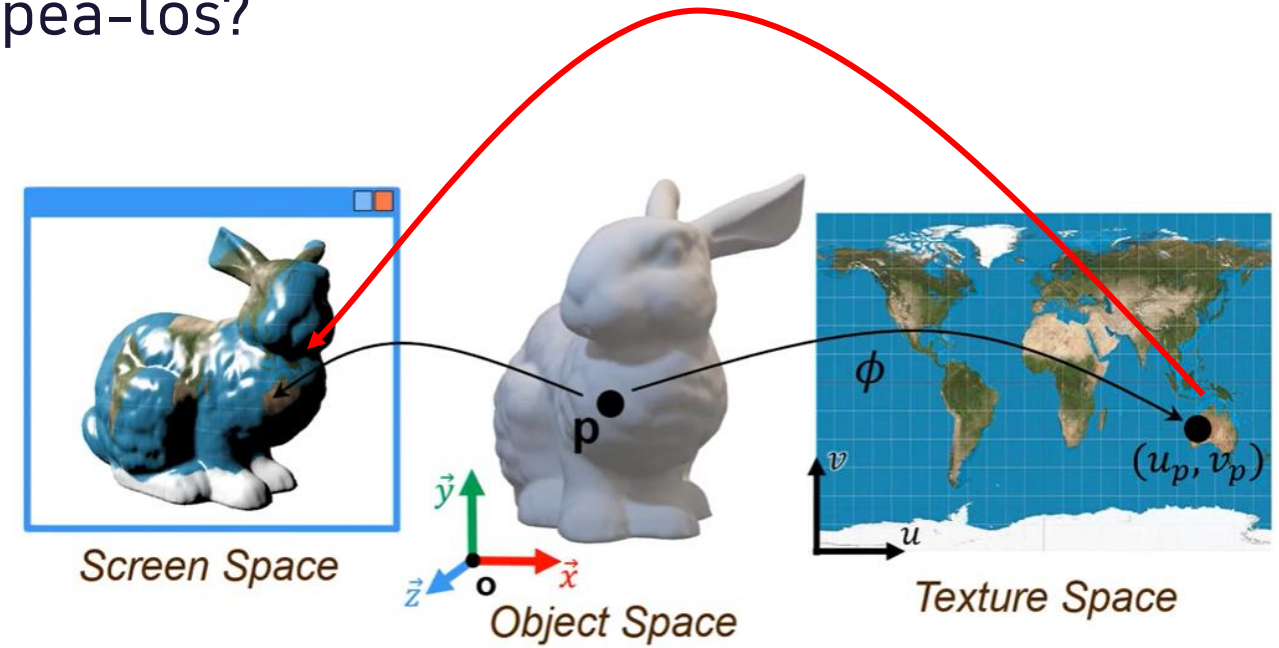
# Mapeamento

- Problema: nosso modelo é uma malha (triangular) 3D
- As texturas são mapas/arranjos 2D
- Como mapeá-los?



# Mapeamento

- Problema: nosso modelo é uma malha (triangular) 3D
- As texturas são mapas/arranjos 2D
- Como mapeá-los?



# Mapeamento

- Qualquer função $f : \mathbb{R}^3 \mapsto \mathbb{R}^2$ que mapeie pontos (x,y,z) do espaço 3D para pontos (u,v) do espaço 2D é uma função de mapeamento de textura.

  - E se (u,v) obtidos estiverem fora do intervalo [0,1]? Veremos mais tarde.

- Alguns tipos de mapeamento:

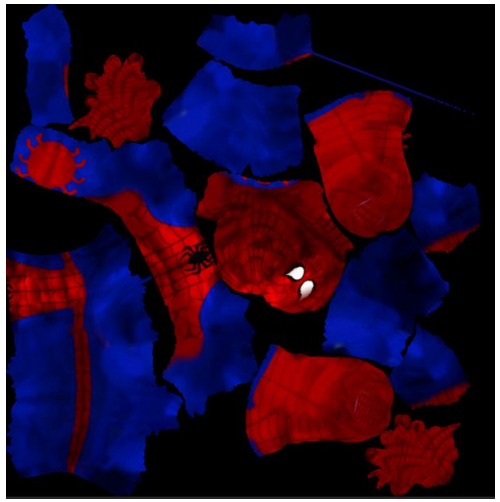
  - Desdobramento UV
  - Mapeamento planar
  - Mapeamento cúbico
  - Mapeamento esférico
  - Mapeamento cilíndrico
  - Mapeamento em duas fases

# Desdobramento UV

- Ao invés de usar uma função para mapeamento de textura, define diretamente qual é o ponto (u,v) associado a cada vértice da malha geométrica
  - (u,v) é dado como um atributo adicional de vértice

# Desdobramento UV

- Ao invés de usar uma função para mapeamento de textura, define diretamente qual é o ponto (u,v) associado a cada vértice da malha geométrica
  - (u,v) é dado como um atributo adicional de vértice
- A ideia é “desembrulhar” a superfície (por ex., vinda de scanner 3D), cortando e esticando até ela estar contida em um plano 2D.

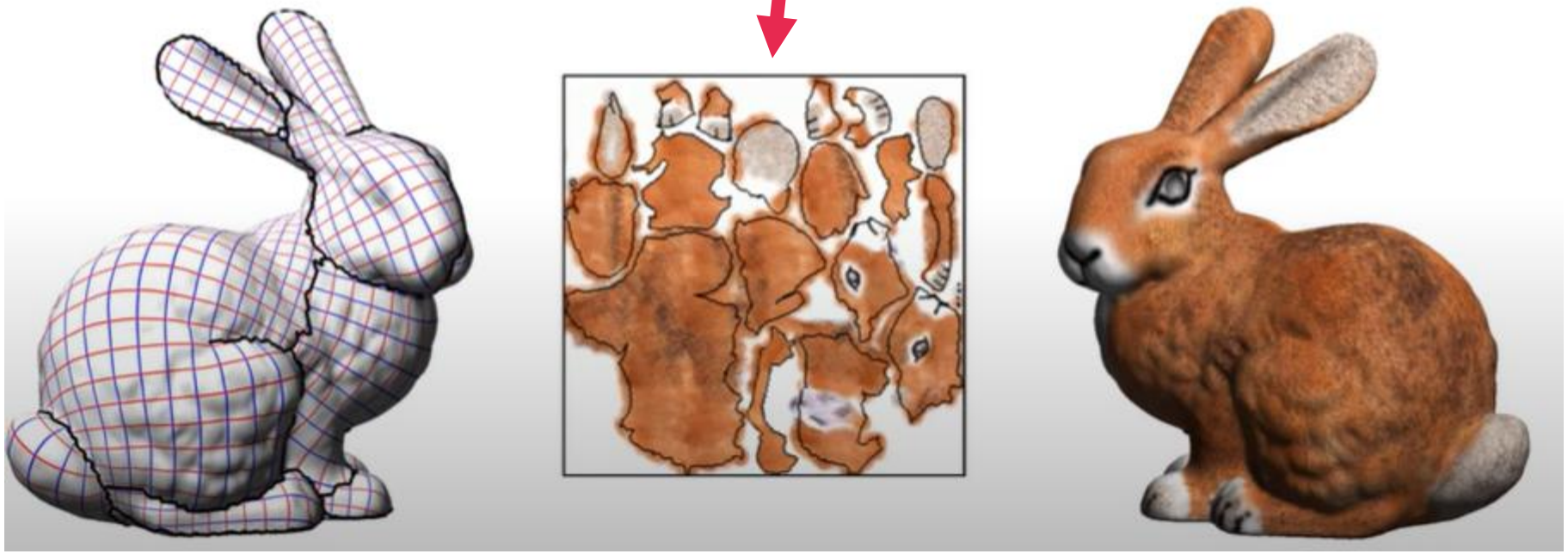


# Desdobramento UV

**Least Squares Conformal Maps for Automatic Texture Atlas Generation**

Bruno Lévy Sylvain Petitjean Nicolas Ray Jérome Maillot*

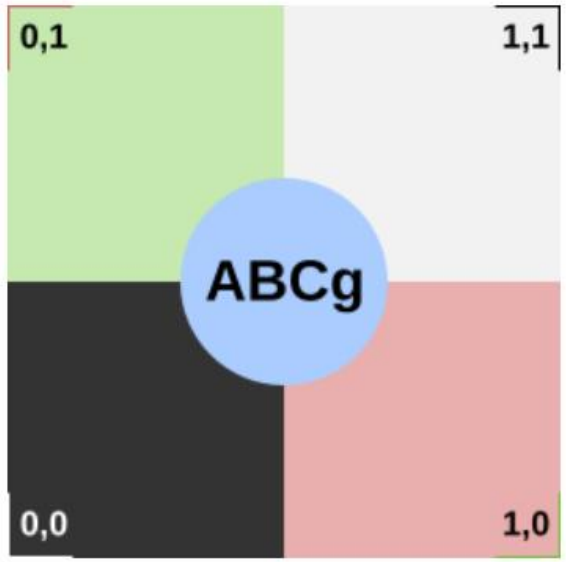
ISA (Inria Lorraine and CNRS), France



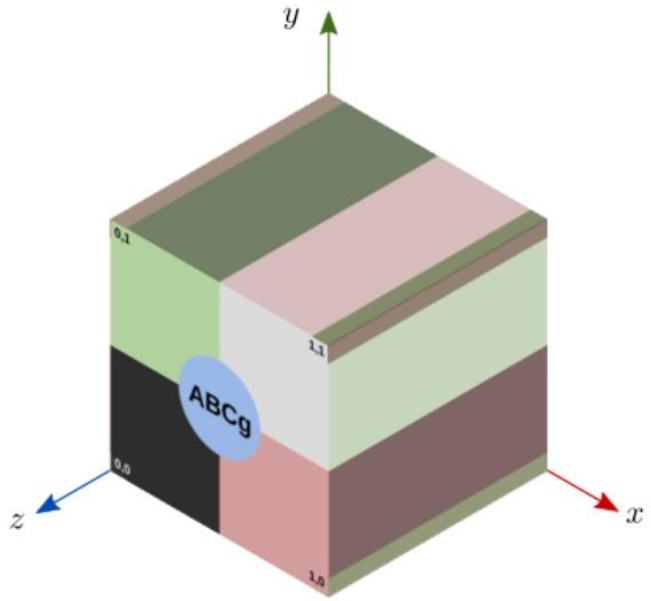
https://www.cs.jhu.edu/~misha/Fall09/Levy02.pdf

# Mapeamento planar

- Usa duas coordenadas do 3D para definir as coordenadas (u,v).
  - Exemplo:
    - (u,v) = (x, y) [Mapeamento na direção z]



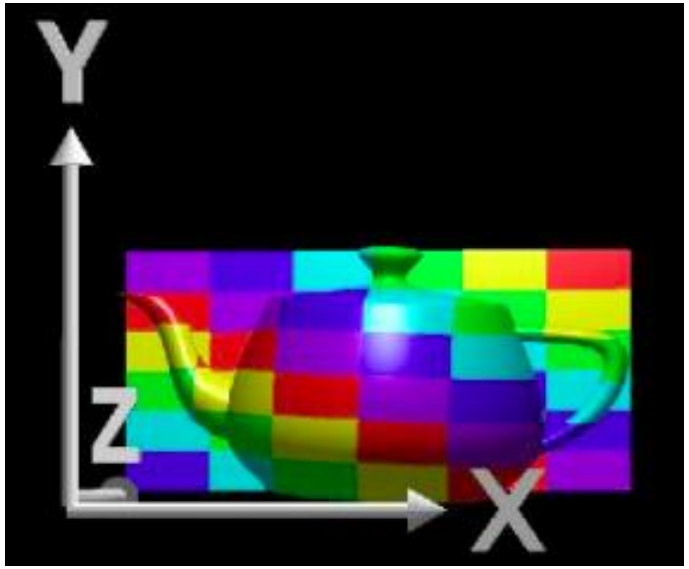

Textura

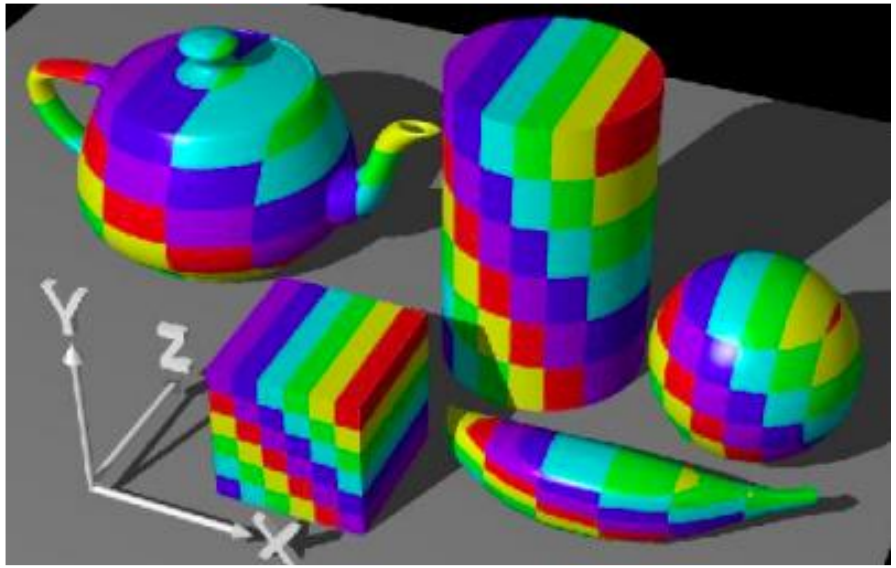



Cubo unitário (a mudança de tom é por causa de iluminação)

# Mapeamento planar

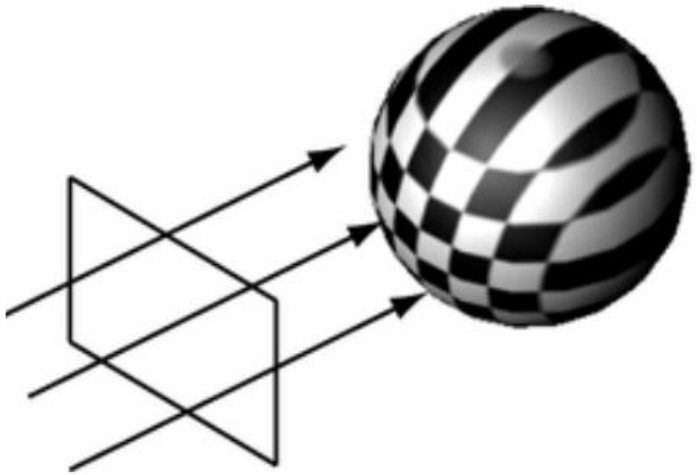
- Usa duas coordenadas do 3D para definir as coordenadas (u,v).
  - Outros exemplos:



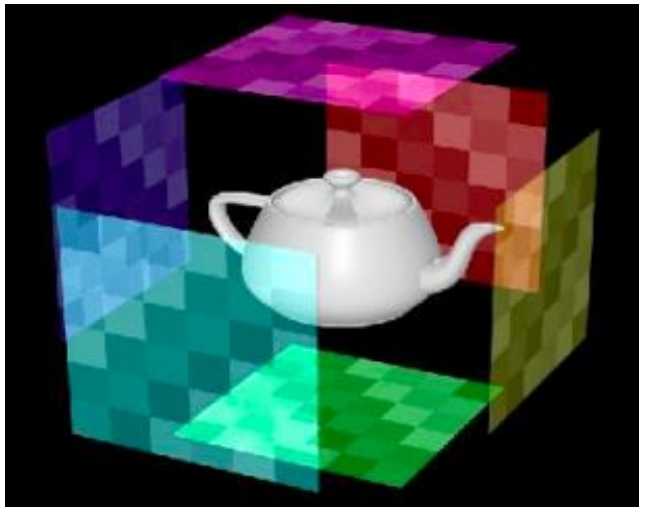
# Mapeamento planar

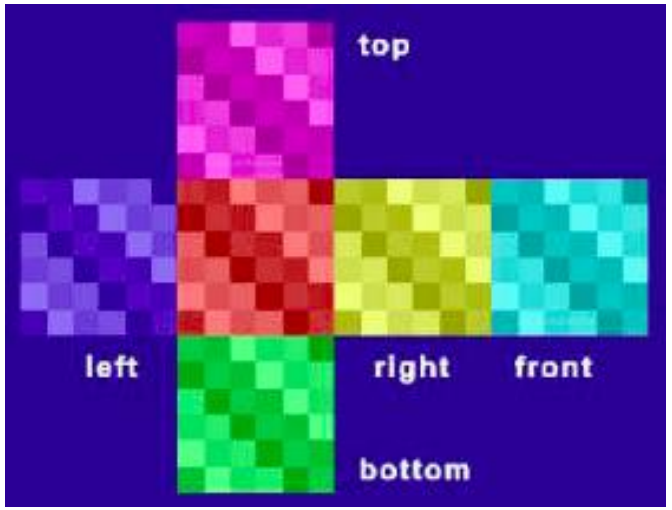
- Simples, mas apresenta efeito ruim em normais perpendiculares



# Mapeamento cúbico

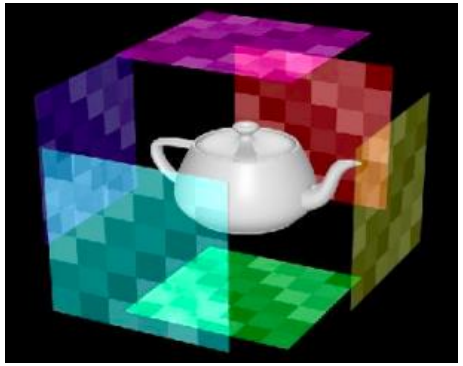
- Faz o mesmo que o planar, mas agora para as seis faces de um cubo.

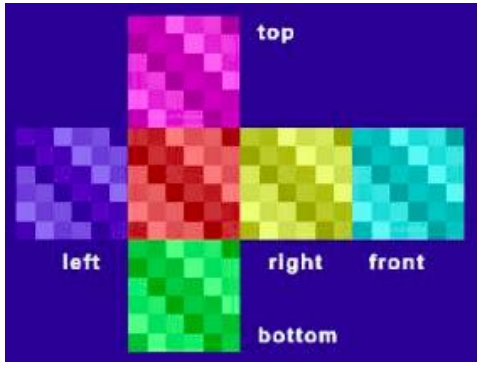


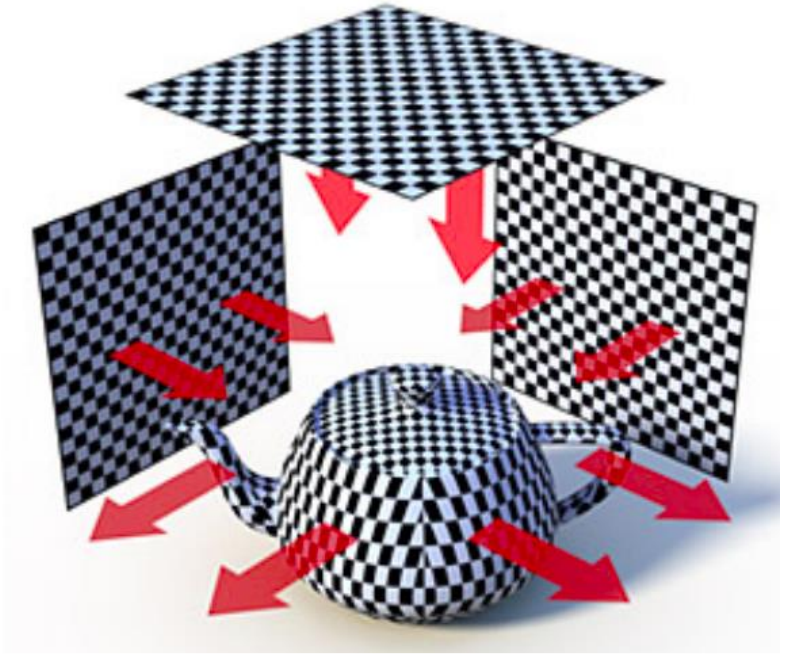


# Mapeamento cúbico

- Faz o mesmo que o planar, mas agora para as seis faces de um cubo.



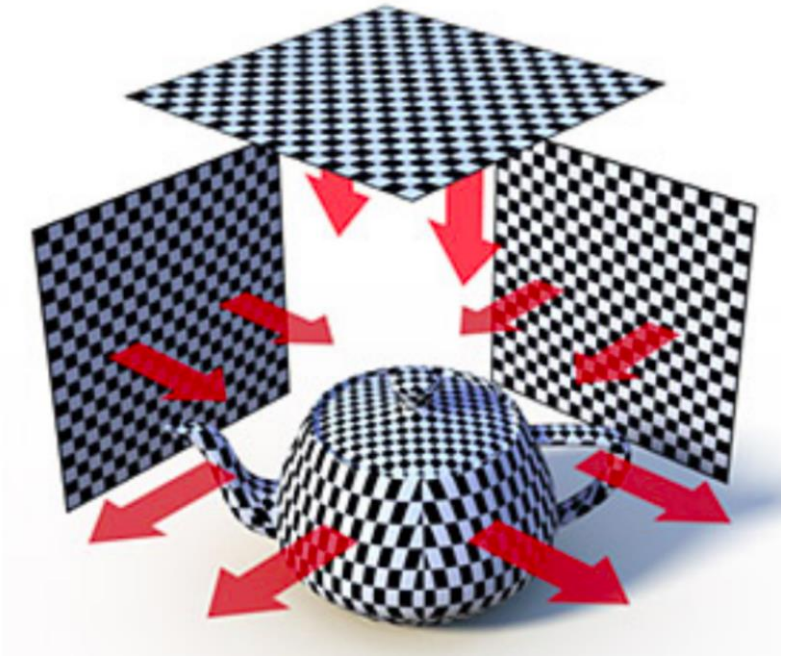


# Mapeamento cúbico

- Faz o mesmo que o planar, mas agora para as seis faces de um cubo.

Resultado melhor que o planar, mas pode haver distorção na fronteira entre os planos



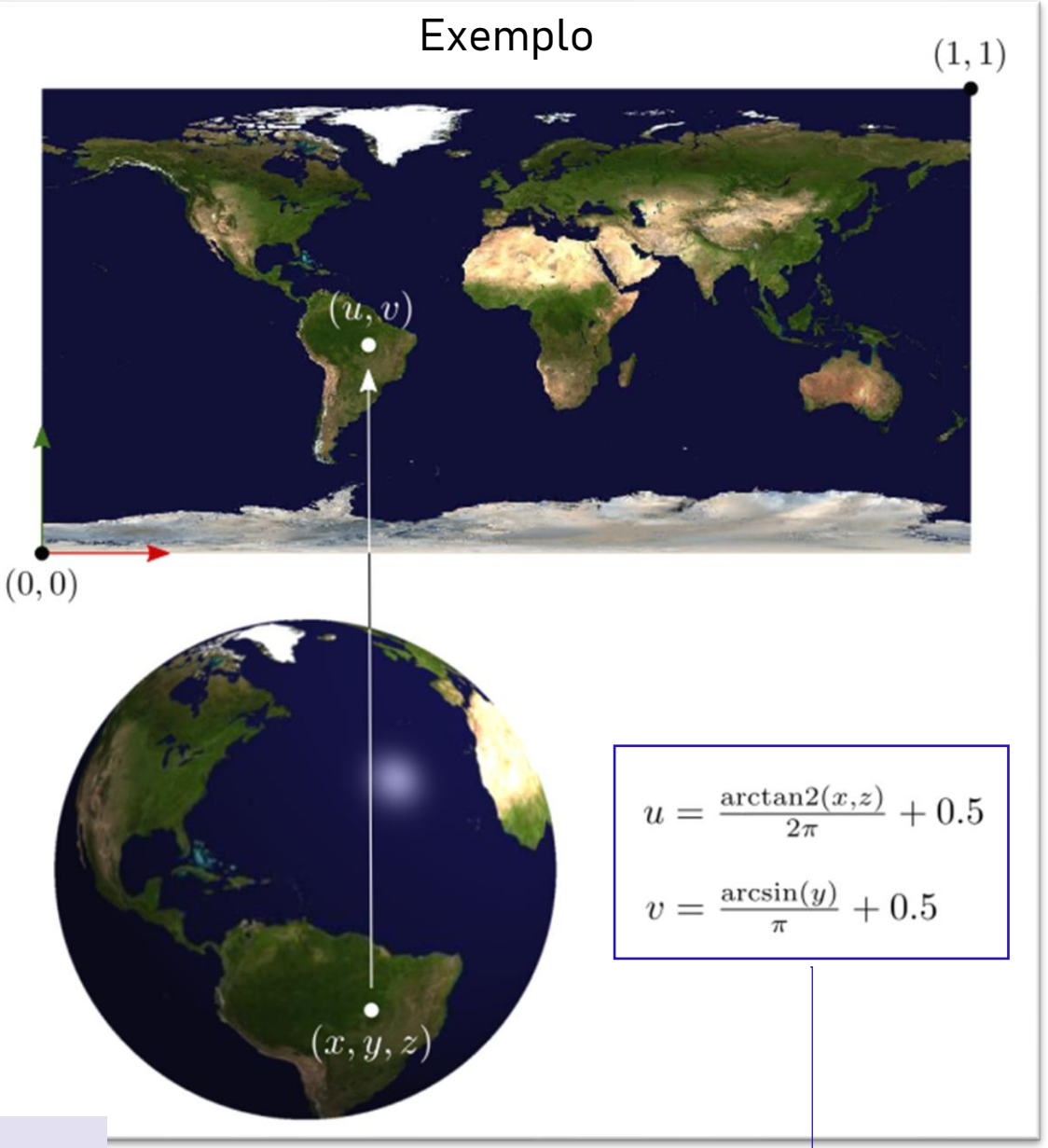
# Mapeamento esférico

- Coordenadas uv são mapeadas segundo coordenadas polares esféricas

# Mapeamento esférico

- Coordenadas uv são mapeadas segundo coordenadas polares esféricas

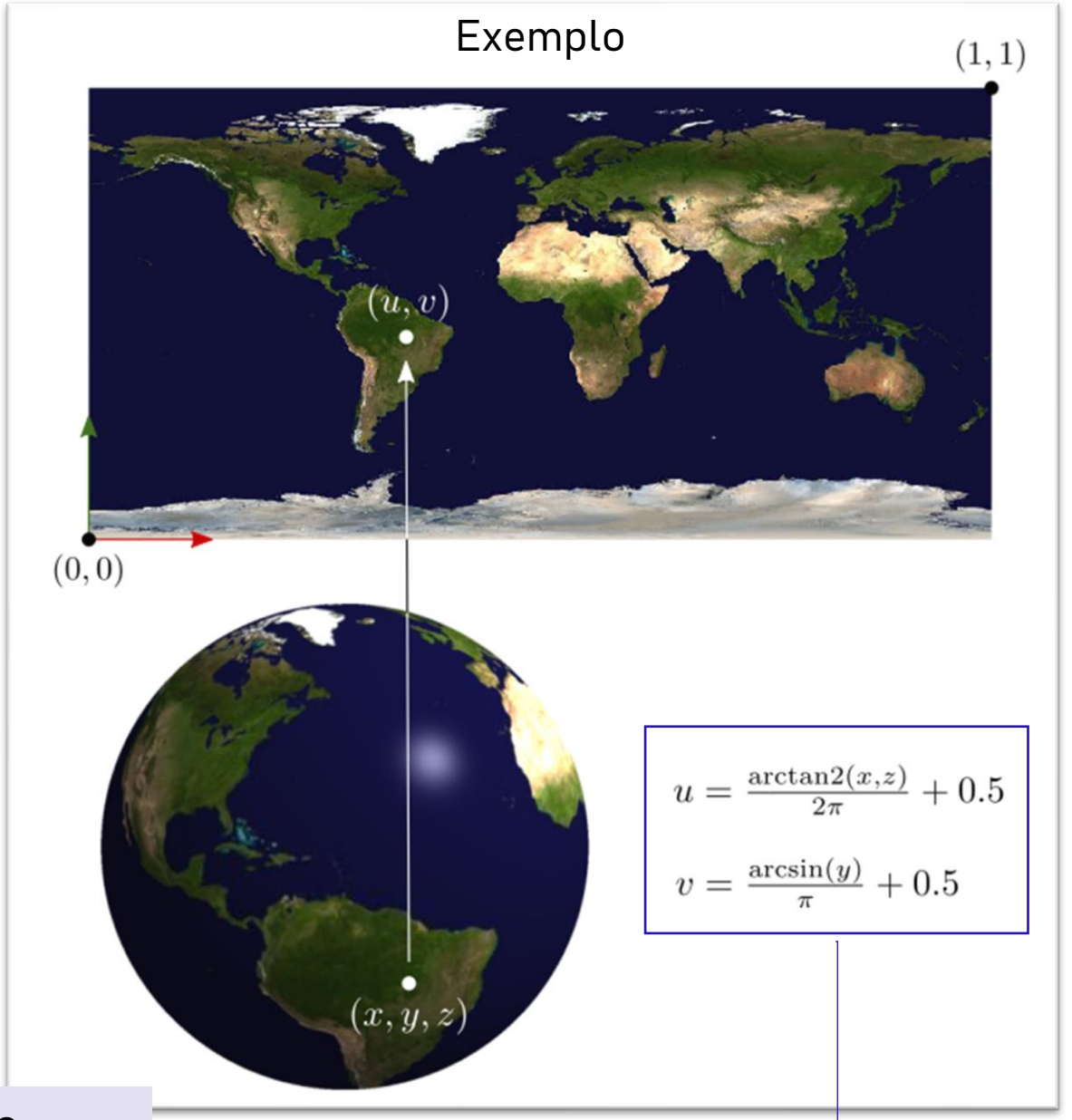
Exemplo




Mapeamento esférico
(coordenadas polares esféricas)
Veja detalhes aqui

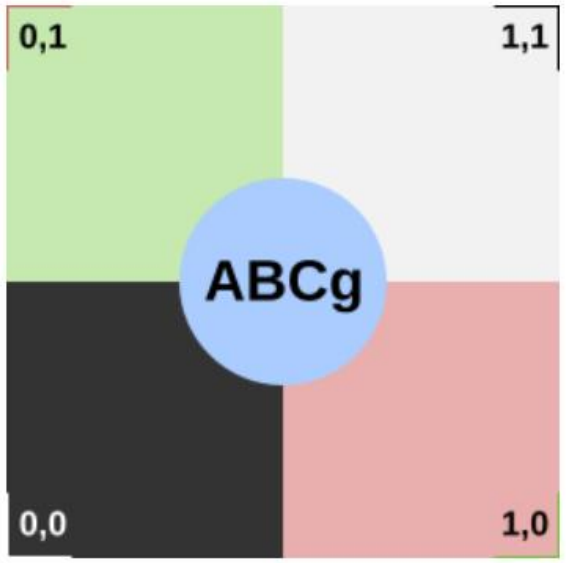
# Mapeamento esférico

- Coordenadas uv são mapeadas segundo coordenadas polares esféricas
- Se esfera na origem:
  - Texels da linha v =0 -> polo sul da esfera (v = 1 -> polo norte)
  - Texels com v = 0,5 -> equador no plano y = 0

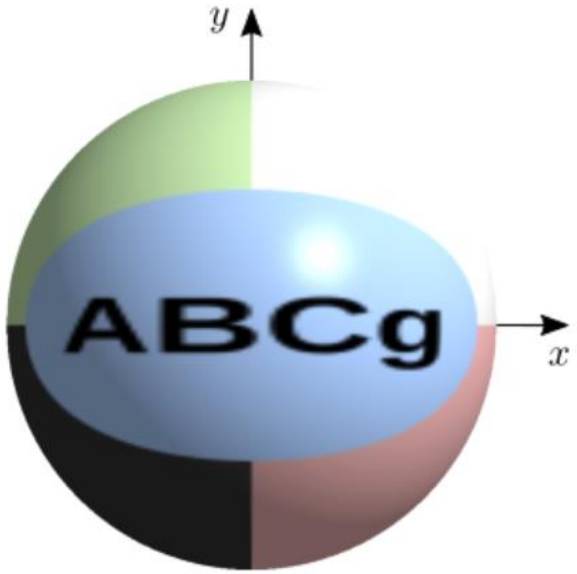



Mapeamento esférico
(coordenadas polares esféricas)
Veja detalhes aqui

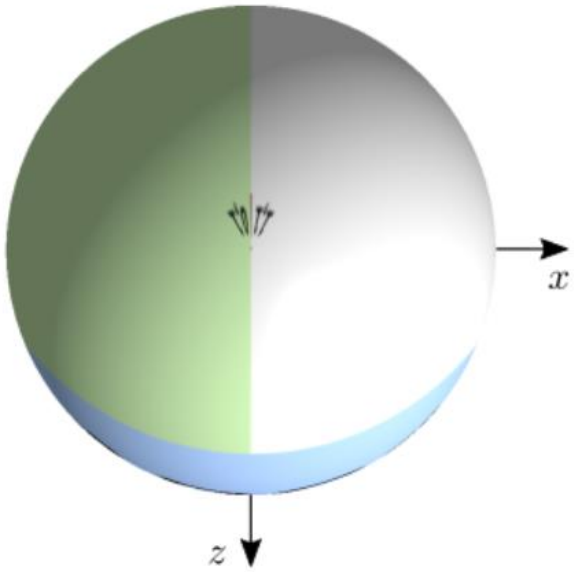
# Mapeamento esférico




Textura

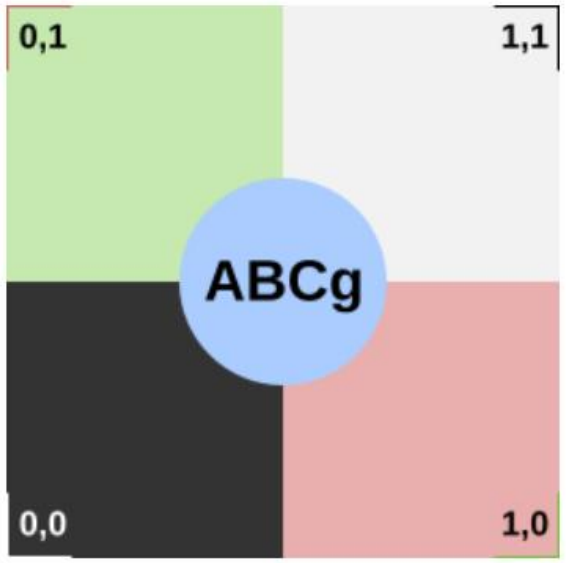



Esfera unitária
vista de frente

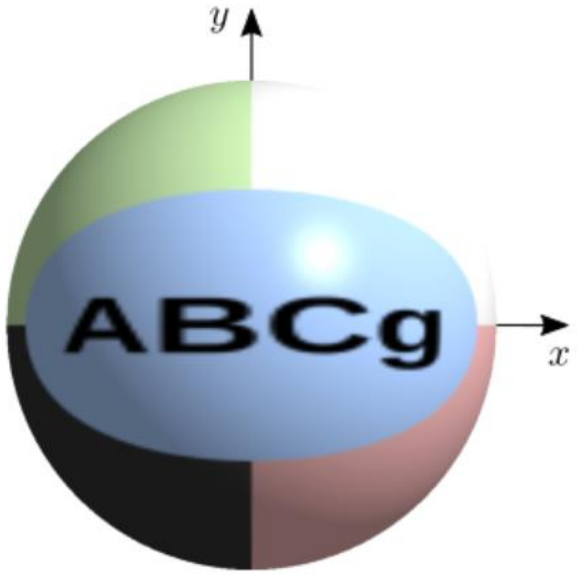



Mesma esfera
vista de cima
(polo norte)

# Mapeamento esférico

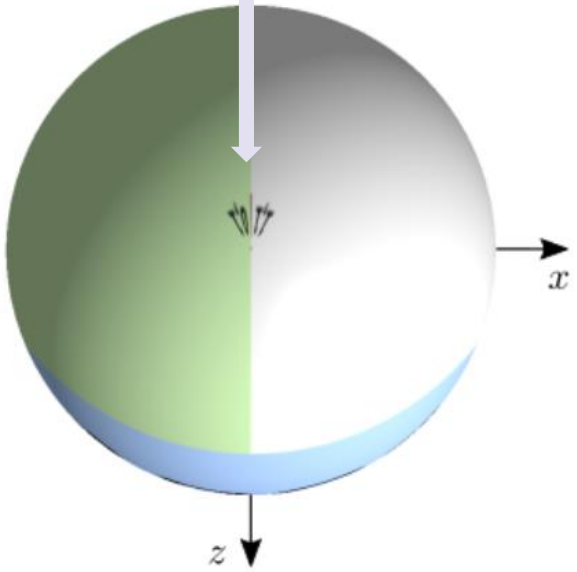



Textura




Esfera unitária
vista de frente

Toda a linha da textura mapeada aqui

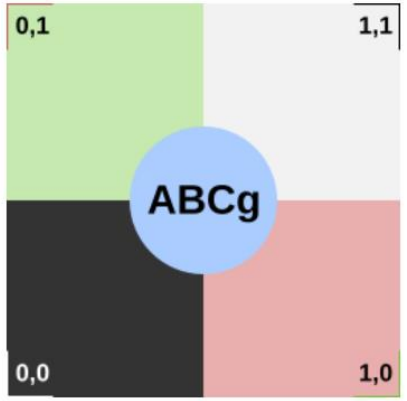


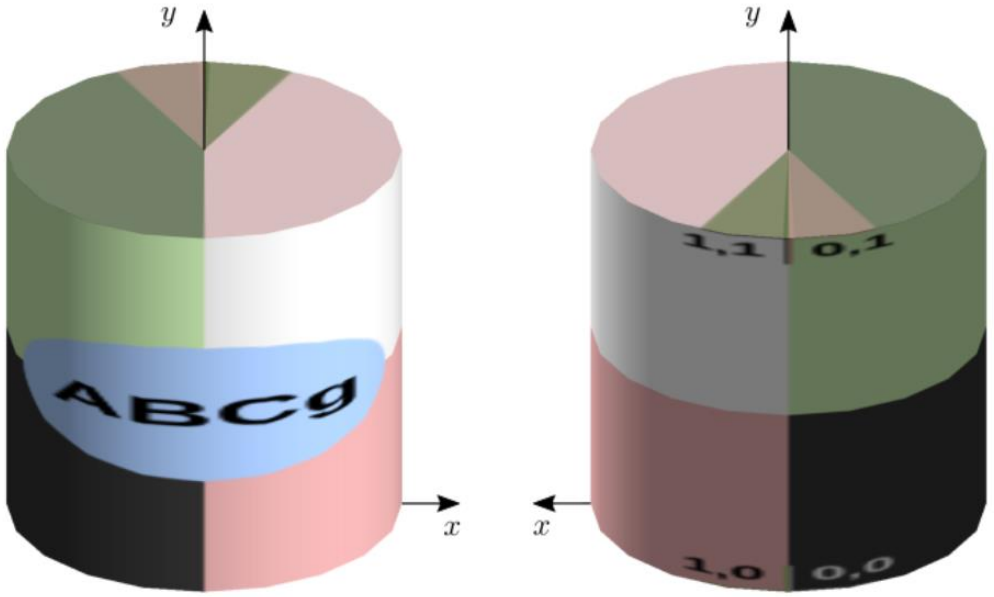

Mesma esfera
vista de cima
(polo norte)

# Mapeamento cilíndrico

- Mapeamento de tal forma que se o objeto é um cilindro unitário alinhado com o eixo y e com base em y = 0...
  - ... O resultado será equivalente a envolver a área lateral do cilindro com a textura.

Veja como os lados esquerdo (u = 0) e direito (u = 1) da textura se unem na parte de trás do cilindro



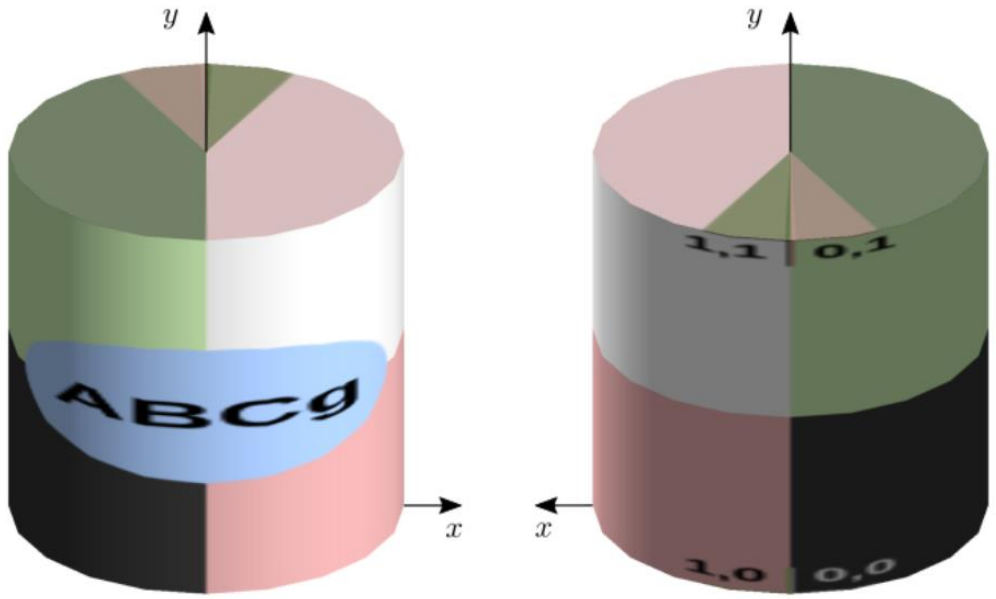
# Mapeamento cilíndrico

- Mapeamento de tal forma que se o objeto é um cilindro unitário alinhado com o eixo y e com base em y = 0...
  - ... O resultado será equivalente a envolver a área lateral do cilindro com a textura.

(u,v) definidos usando coordenadas polares cilíndricas. Veja mais detalhes aqui

$$u = \frac{\arctan2\left(p_x, p_z\right)}{2\pi} + 0.5,$$

$$v = \frac{\arcsin\left(\frac{p_y}{|\mathbf{p}|}\right)}{\pi} + 0.5,$$



# Mapeamento em duas fases

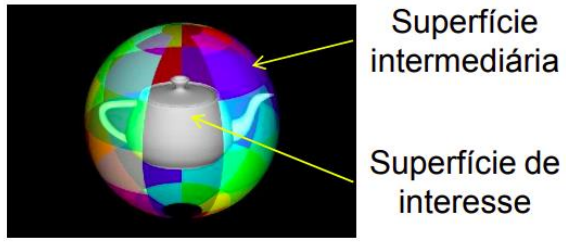
- Modelos mais complexos podem ser englobados por uma superfície simples de um modelo intermediário (cubo, esfera, cilindro):

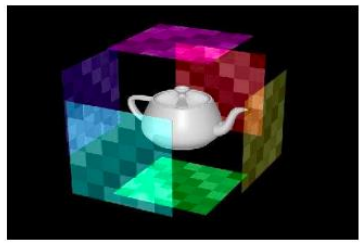
1. A textura é aplicada na superfície simples do modelo intermediário.

2. A textura é mapeada do objeto intermediário para o objeto alvo.

# Mapeamento em duas fases

- Modelos mais complexos podem ser englobados por uma superfície simples de um modelo intermediário (cubo, esfera, cilindro):



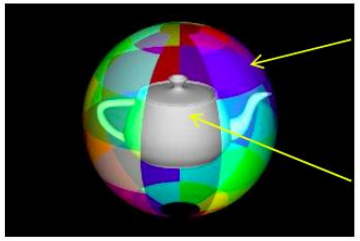



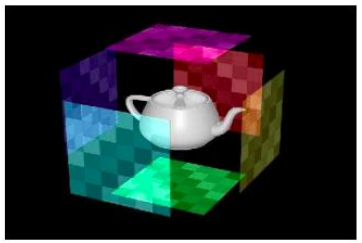
Fase 1: Escolher modelo/superfície intermediário

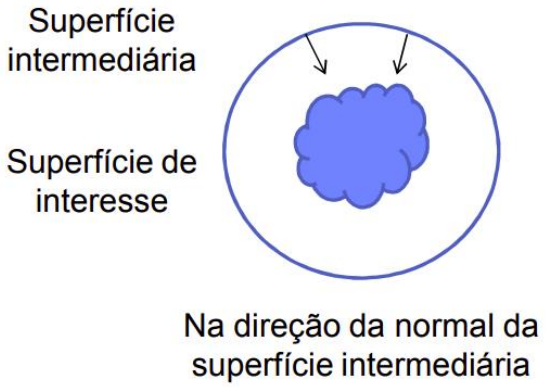
# Mapeamento em duas fases

- Modelos mais complexos podem ser englobados por uma superfície simples de um modelo intermediário (cubo, esfera, cilindro):

Fase 1: Escolher modelo/superfície intermediário





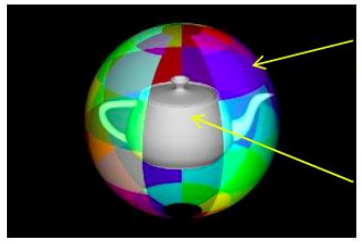


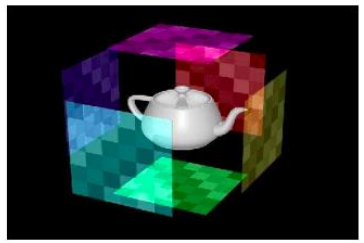

Fase 2: Escolher estratégia de mapeamento

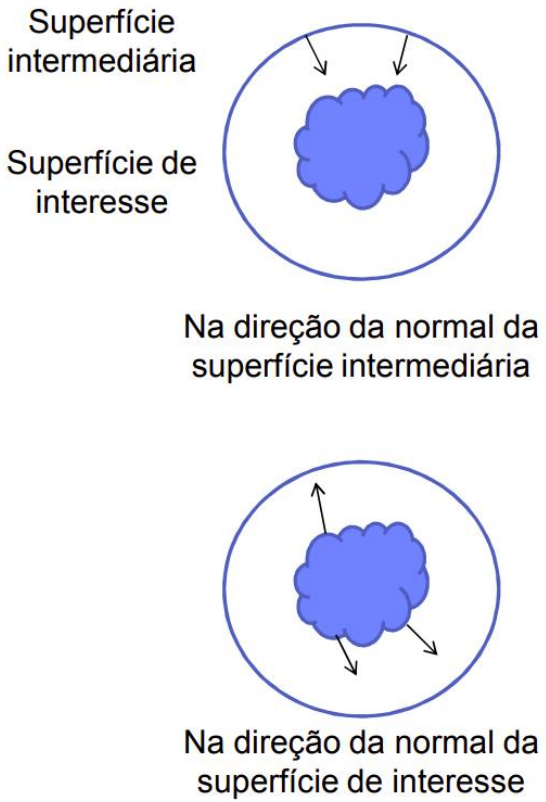
# Mapeamento em duas fases

- Modelos mais complexos podem ser englobados por uma superfície simples de um modelo intermediário (cubo, esfera, cilindro):

Fase 1: Escolher modelo/superfície intermediário





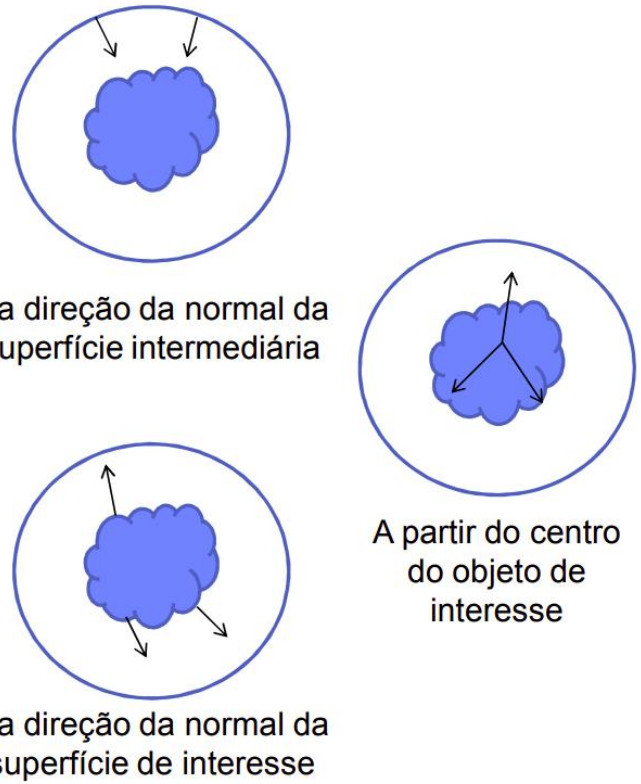



Fase 2: Escolher estratégia de mapeamento

# Mapeamento em duas fases

- Modelos mais complexos podem ser englobados por uma superfície simples de um modelo intermediário (cubo, esfera, cilindro):

Fase 1: Escolher modelo/superfície intermediário



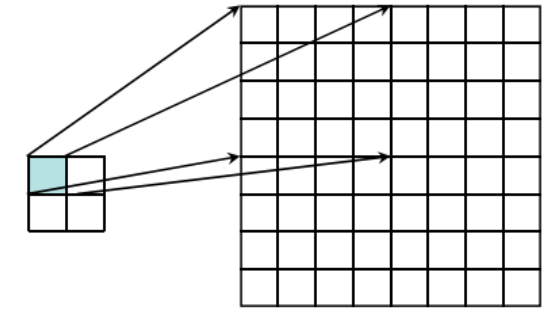

Fase 2: Escolher estratégia de mapeamento

# Filtros

- **Pixels não são texels** (raramente temos 1:1)

# Filtros

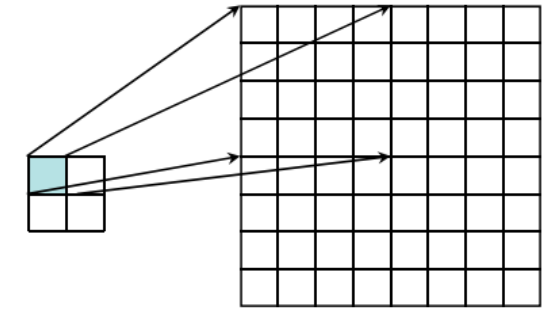
- **Pixels não são texels** (raramente temos 1:1)
  - **Magnificação:** quando o polígono é maior que a textura
    - Vários pixels serão mapeados para o mesmo texel



Magnificação

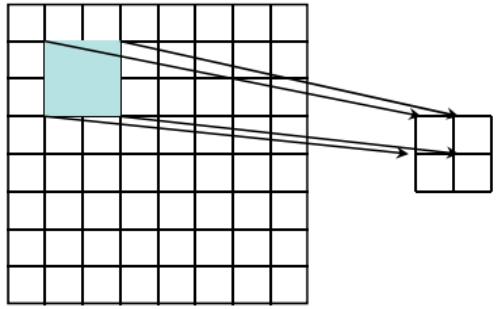
# Filtros

- **Pixels não são texels** (raramente temos 1:1)
  - **Magnificação:** quando o polígono é maior que a textura
    - Vários pixels serão mapeados para o mesmo texel



Magnificação

  - **Minificação:** quando a textura é maior que o polígono
    - Vários texels serão mapeados para o mesmo pixel



Minificação

# Filtros

- **Pixels não são texels** (raramente temos 1:1)
  - **Magnificação:** Vários pixels serão mapeados para o mesmo texel
  - **Minificação:** Vários texels serão mapeados para o mesmo pixel

  - O que fazer nesses casos? Aplicar **filtros** [de interpolação para escolher a cor correspondente a uma posição (u,v)].

# Filtros

- **Pixels não são texels** (raramente temos 1:1)
  - **Magnificação:** Vários pixels serão mapeados para o mesmo texel
  - **Minificação:** Vários texels serão mapeados para o mesmo pixel

  - O que fazer nesses casos? Aplicar **filtros** [de interpolação para escolher a cor correspondente a uma posição (u,v)].


    - `GL_NEAREST`: escolhe o valor do texel mais próximo da coordenada da textura

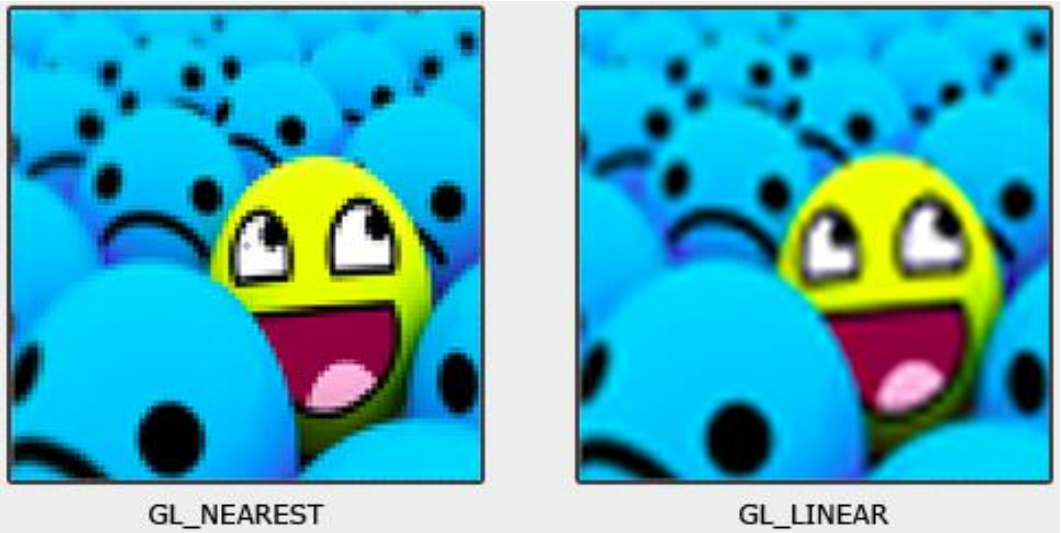
# Filtros

- **Pixels não são texels** (raramente temos 1:1)
  - **Magnificação:** Vários pixels serão mapeados para o mesmo texel
  - **Minificação:** Vários texels serão mapeados para o mesmo pixel

  - O que fazer nesses casos? Aplicar **filtros** [de interpolação para escolher a cor correspondente a uma posição (u,v)].

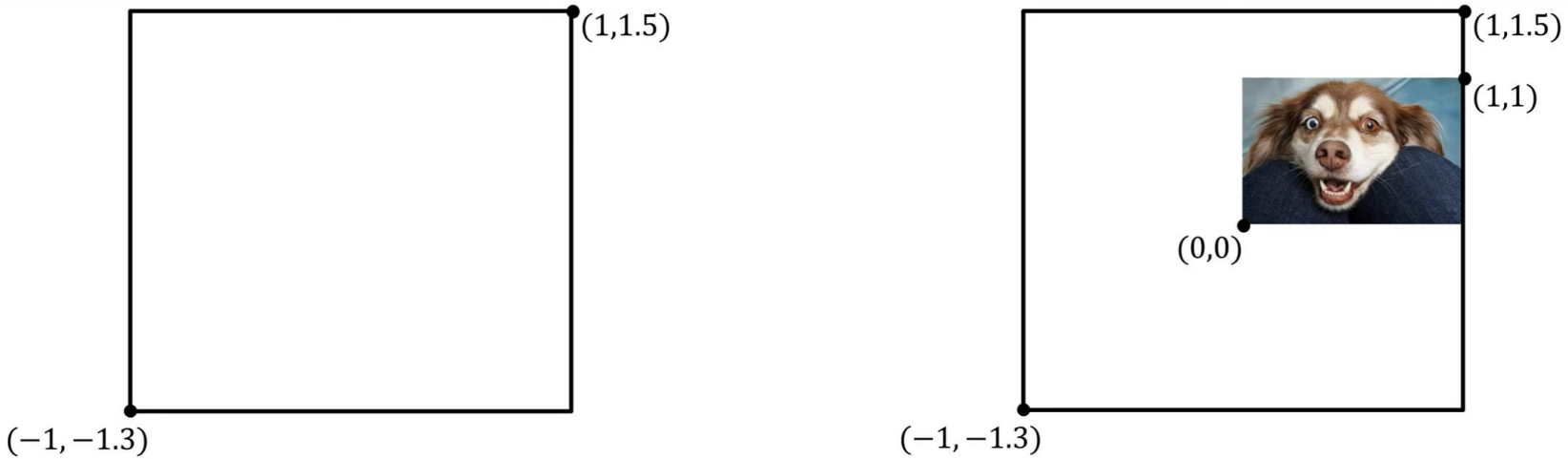

    - `GL_NEAREST`: escolhe o valor do texel mais próximo da coordenada da textura
    - `GL_LINEAR`: escolhe o valor por interpolação entre os quatros texels mais próximos

# Filtros




# Repetição de Texturas

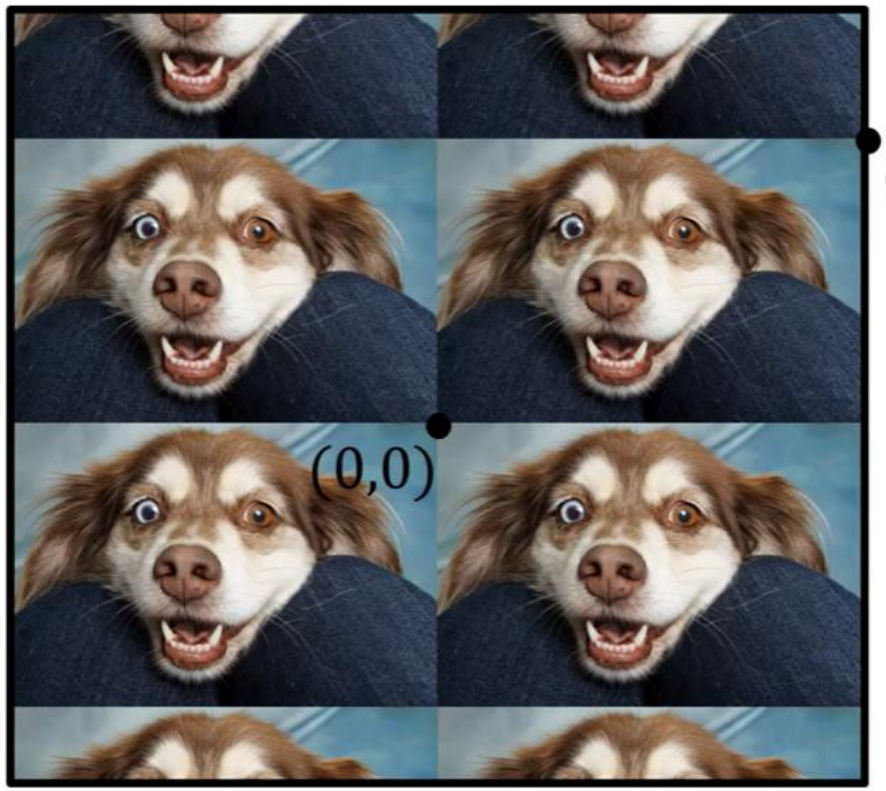
- Qualquer função $f : \mathbb{R}^3 \mapsto \mathbb{R}^2$ que mapeie pontos (x,y,z) do espaço 3D para pontos (u,v) do espaço 2D é uma função de mapeamento de textura.
  - E se (u,v) estiverem fora do intervalo [0,1]? Veremos agora.




Coordenadas de textura

# Repetição de Texturas

- GL_REPEAT: repete a textura fora do intervalo (*default* no OpenGL)

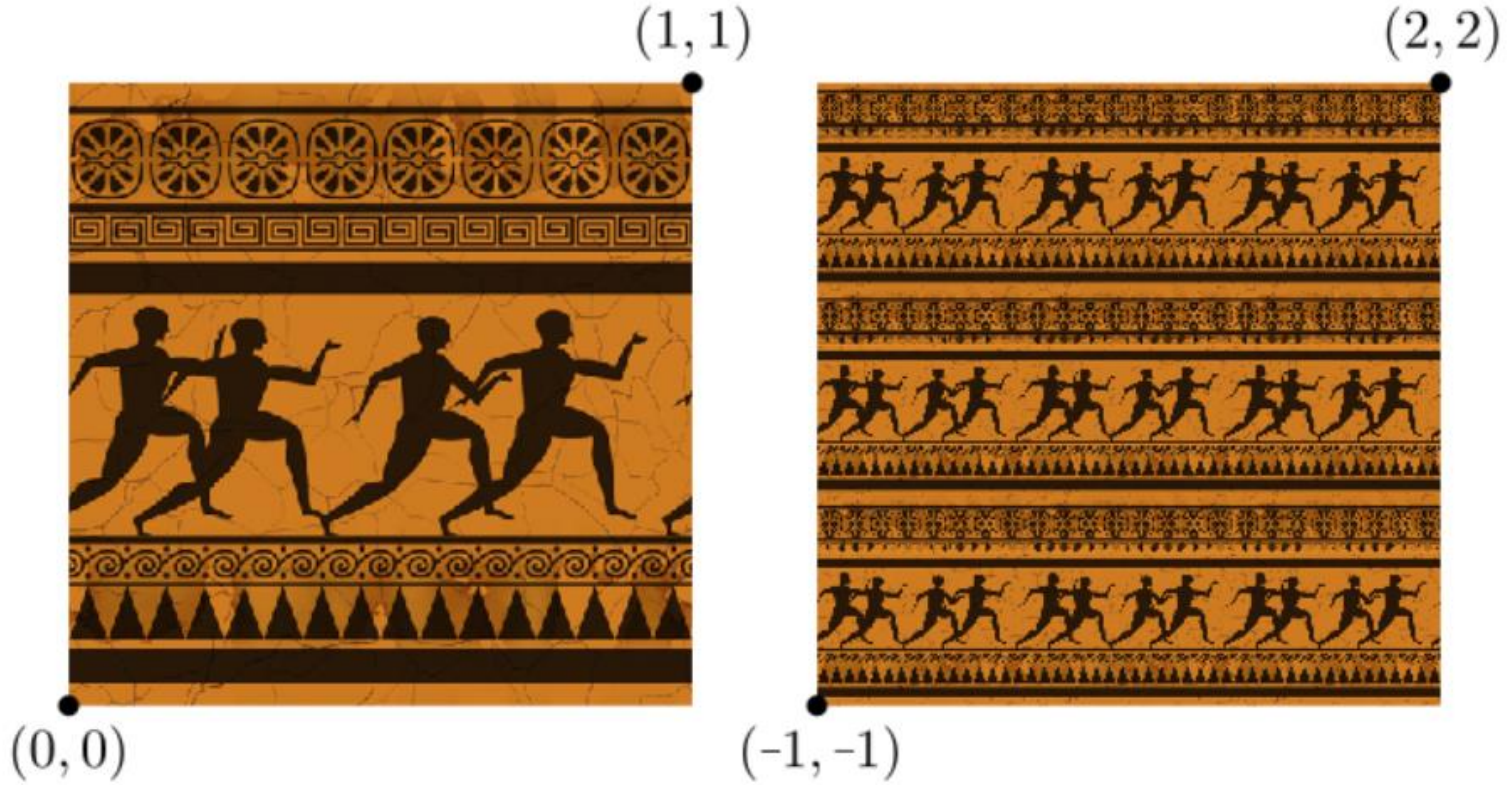



GL_REPEAT

Implementação:

$$u' = u - \text{floor}(u)$$
$$v' = v - \text{floor}(v)$$

# Repetição de Texturas



# Repetição de Texturas

- `GL_MIRRORED_REPEAT`: repete a textura, espelhando-a em u e/ou v


GL_MIRRORED_REPEAT

Implementação:

$$u' = u - \text{floor}(u)$$
$$\textbf{if } \text{impar}(\text{floor}(u))$$
$$u' = 1 - u'$$

$$v' = v - \text{floor}(v)$$
$$\textbf{if } \text{impar}(\text{floor}(v))$$
$$u' = 1 - v'$$

# Repetição de Texturas

- `GL_CLAMP_TO_EDGE`: fixa as coordenadas no intervalo [0,1]. O resultado é a repetição dos valores das primeiras e últimas linhas/colunas da textura

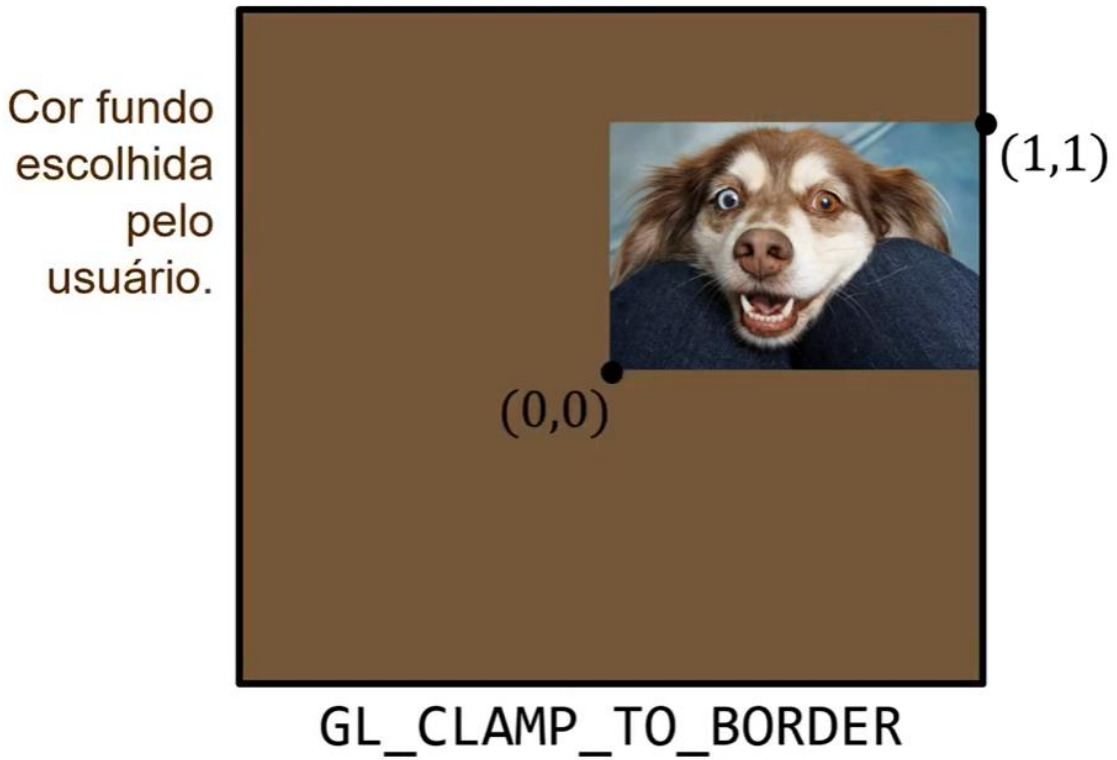

GL_CLAMP_TO_EDGE

Implementação:

$$u' = \max(\min(u, 1), 0)$$
$$v' = \max(\min(v, 1), 0)$$

# Repetição de Texturas

- `GL_CLAMP_TO_BORDER`: o que está fora do intervalo [0,1] assume uma cor dada pelo usuário



# Mapeamento de Texturas na prática

- Onde encontramos as informações de textura para nossos modelos?

- Até hoje, sempre definimos os vértices do modelo manualmente.

- Na prática:
  - Temos centenas/milhares de vértices (modelos complexos)
  - Cada vértice tem atributos, por exemplo:
    - Posição, Coordenadas de textura (u,v), normal, etc...

# Mapeamento de Texturas na prática

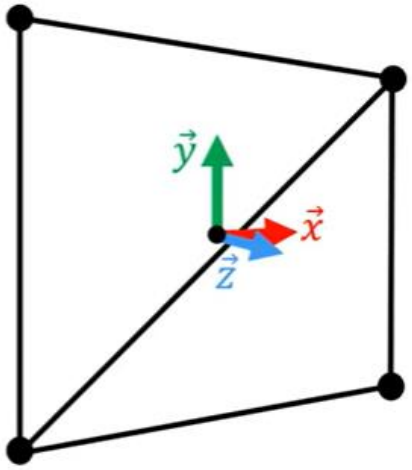
Objetos/Modelos no formato *WaveFront.*

- Arquivo .obj

```
# PLANO.obj
# Lista de vértices
v -0.5  -0.5  0
v  0.5  -0.5  0
v -0.5   0.5  0
v  0.5   0.5  0

# Coordenadas de textura
vt 0.0 0.0
vt 1.0 0.0
vt 0.0 1.0
vt 1.0 1.0

# Lista de faces
f 1/1 2/2 4/4
f 1/1 4/4 3/3
```




# Mapeamento de Texturas na prática

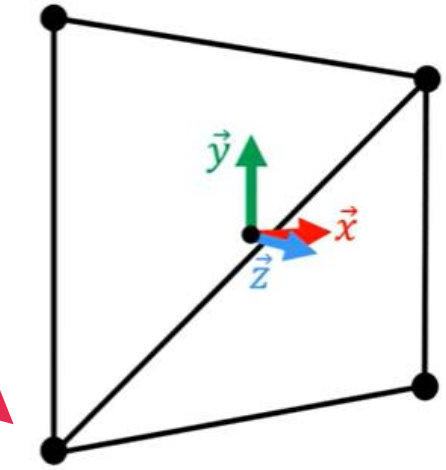
Objetos/Modelos no formato *WaveFront*.

- Arquivo .obj

```
# PLANO.obj
# Lista de vértices
v -0.5  -0.5  0
v  0.5  -0.5  0
v -0.5   0.5  0
v  0.5   0.5  0

# Coordenadas de textura
vt 0.0 0.0
vt 1.0 0.0
vt 0.0 1.0
vt 1.0 1.0

# Lista de faces
f 1/1 2/2 4/4
f 1/1 4/4 3/3
```




# Mapeamento de Texturas na prática

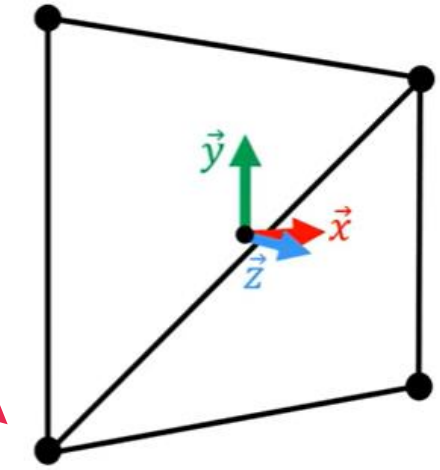
Objetos/Modelos no formato *WaveFront*.

- Arquivo .obj

```
# PLANO.obj
# Lista de vértices
v -0.5  -0.5  0
v  0.5  -0.5  0
v -0.5   0.5  0
v  0.5   0.5  0

# Coordenadas de textura
vt 0.0 0.0
vt 1.0 0.0
vt 0.0 1.0
vt 1.0 1.0

# Lista de faces
f 1/1 2/2 4/4
f 1/1 4/4 3/3
```




# Mapeamento de Texturas na prática

Objetos/Modelos no formato *WaveFront.*

- Arquivo .obj

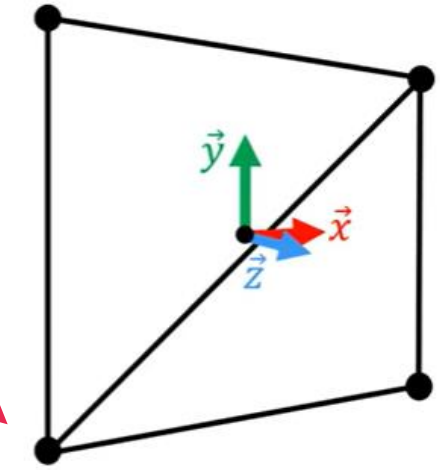
Como sei o valor para um ponto interno?

Interpolação dos valores dos vértices

```
# PLANO.obj
# Lista de vértices
v -0.5  -0.5  0
v  0.5  -0.5  0
v -0.5   0.5  0
v  0.5   0.5  0

# Coordenadas de textura
vt 0.0 0.0
vt 1.0 0.0
vt 0.0 1.0
vt 1.0 1.0

# Lista de faces
f 1/1 2/2 4/4
f 1/1 4/4 3/3
```

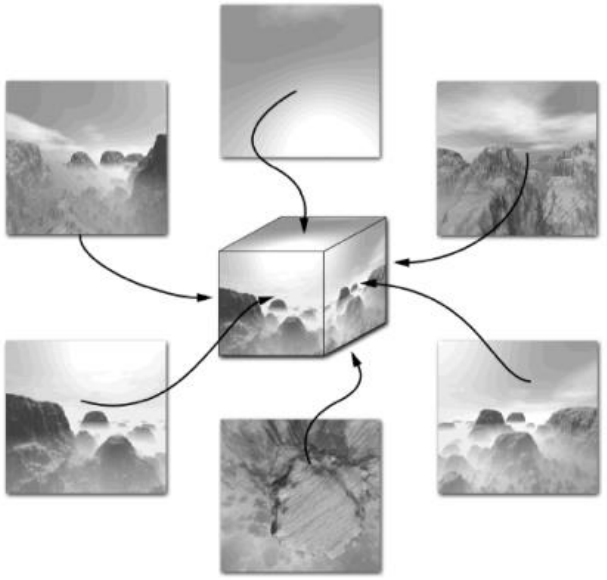


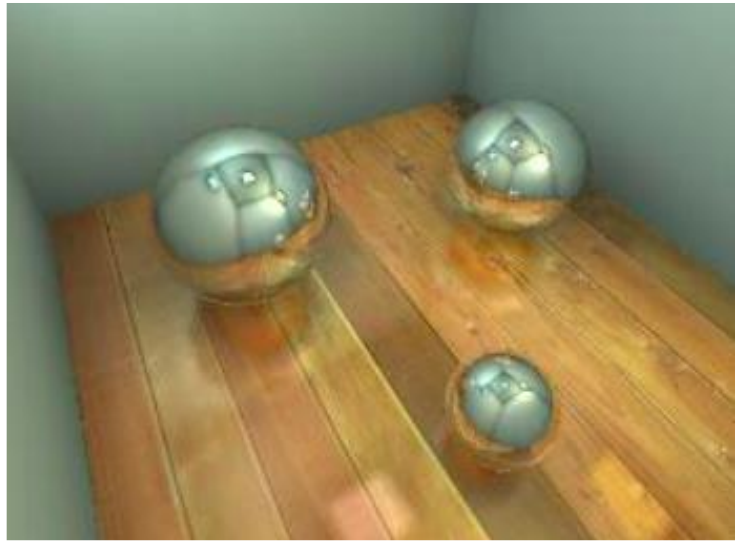

# Mapeamento de Texturas na prática

Jupyter Notebook
(Aula 06.Ex01)

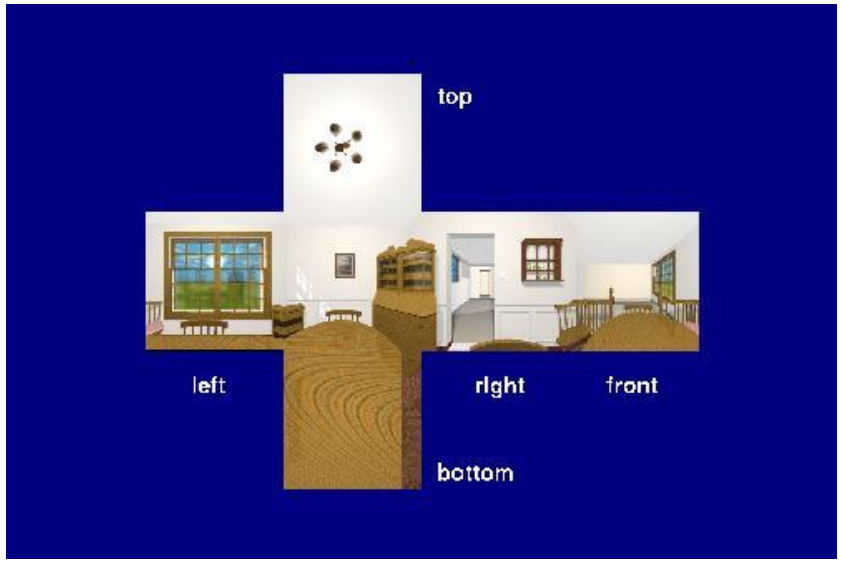
# Outros tipos de mapeamento de texturas

- **Mapeamento de Reflexão:** reflete na superfície dos objetos os elementos que compõem a cena



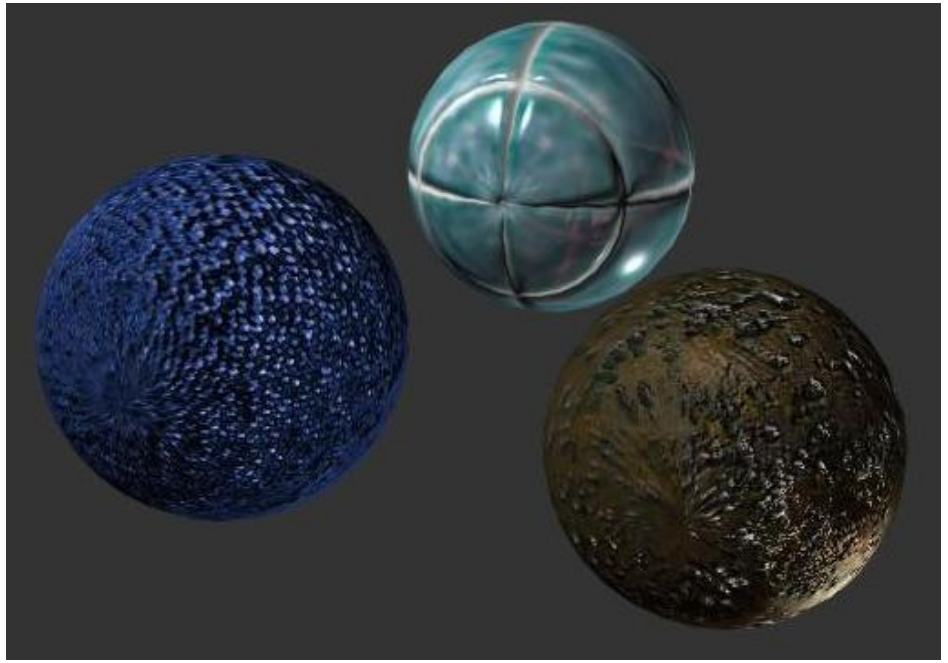


# Outros tipos de mapeamento de texturas

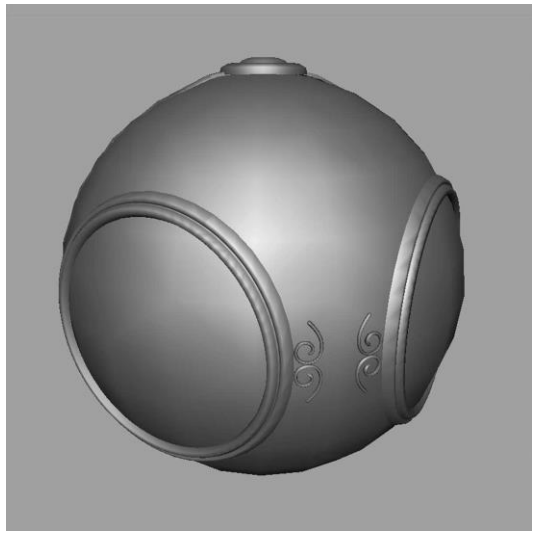



# Outros tipos de mapeamento de texturas

- **Mapeamento Bump:** técnica de perturbação para dar efeito de superfície áspera

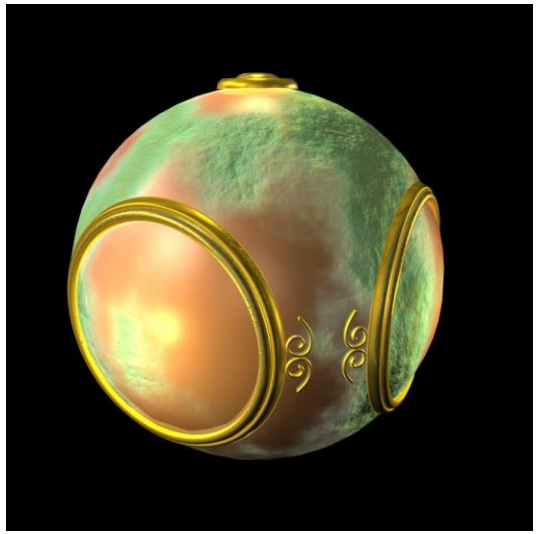
# Mapeamentos – Resumo



Sem textura

Com textura

Mapeamento de reflexão



Bump

# Bibliografia

- Essa aula foi baseada no seguinte material:


- Video-aula Prof. Eduardo S L Gastal, UFRGS. https://www.youtube.com/watch?v=89TyBovmIZU *(Acessado em 06/08/2024)*

- www.brunodorta.com.br/cg/texmapping.html *(Acessado em 06/08/2024)*

- Computação Gráfica, Slides de Ricardo Marcacini, Maria Cristina, Alaor Cervati. ICMC/USP.

- Hughes, J. F., Van Dam, A., Foley, J. D., McGuire, M., Feiner, S. K., & Sklar, D. F. (2014). Computer graphics: principles and practice. Terceira Edição. Pearson Education.

# arctan2

O ângulo é calculado corretamente para $p_z > 0$. Entretanto, a imagem da função arco tangente está restrita ao intervalo $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$. Para que $\theta$ seja um ângulo em um intervalo de 360 graus, precisamos ajustar o intervalo da função arco tangente de acordo com o sinal de $p_x$ e $p_z$. Isso pode ser feito através da definição de uma função $\mathbf{arctan2}(p_x, p_z)$ , que retorna um ângulo no intervalo $(-\pi, \pi]$:

$$\operatorname{atan2}(p_x, p_z) = \begin{cases} \arctan\left(\dfrac{p_x}{p_z}\right) & \text{se } p_z > 0, \\ \arctan\left(\dfrac{p_x}{p_z}\right) + \pi & \text{se } p_z < 0 \text{ e } p_x \geq 0, \\ \arctan\left(\dfrac{p_x}{p_z}\right) - \pi & \text{se } p_z < 0 \text{ e } p_x < 0, \\ +\dfrac{\pi}{2} & \text{se } p_z = 0 \text{ e } p_x > 0, \\ -\dfrac{\pi}{2} & \text{se } p_z = 0 \text{ e } p_x < 0, \\ \text{indefinido} & \text{se } p_z = 0 \text{ e } p_x = 0. \end{cases}$$