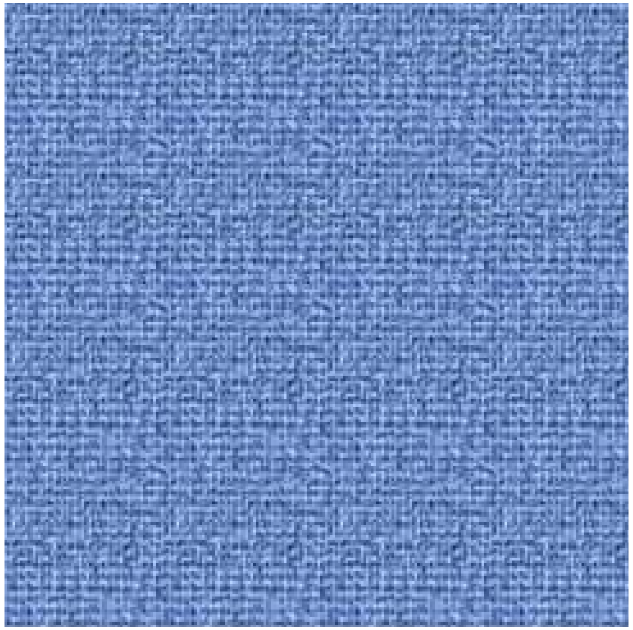
Aula 23A

Aplicação de Texturas



Detalhes de Superfícies

Modelos de iluminação não são apropriados para descrever completamente uma superfície

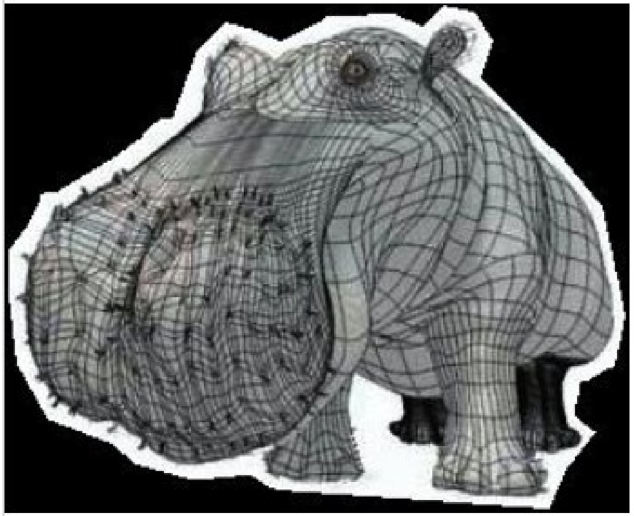
Superfícies pintadas com padrões ou imagens
- A capa ou uma página de um livro

Superfícies com padrões regulares
- Tecidos ou uma parede de tijolos
- Em princípio é possível modelar esses detalhes com geometria e usando materiais de propriedades óticas distintas
- Na prática, esses efeitos são modelados usando uma técnica chamada ***mapeamento de textura***

Detalhes de Superfícies

Texturas

- Superfícies no mundo real são muito complexas
- Não se pode modelar todos os detalhes
- Como adicionar detalhe na superfície?



Modelo Geométrico

Modelo Geométrico
+
Sombreamento

Modelo Geométrico
+
Sombreamento
+
Texturas

Detalhes de Superfícies

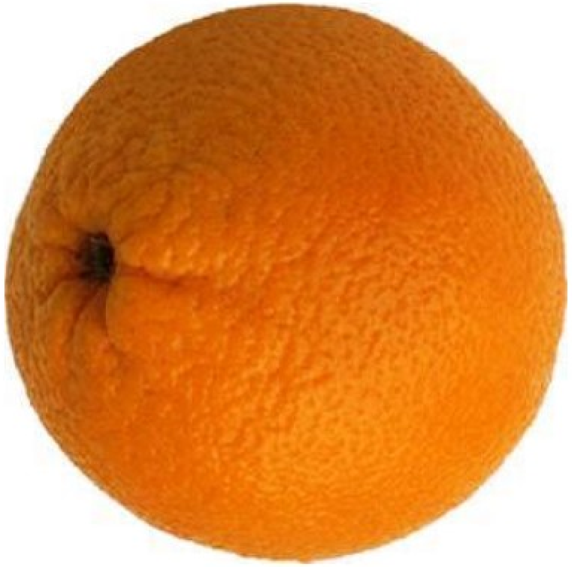
Limites da Modelagem Geométrica

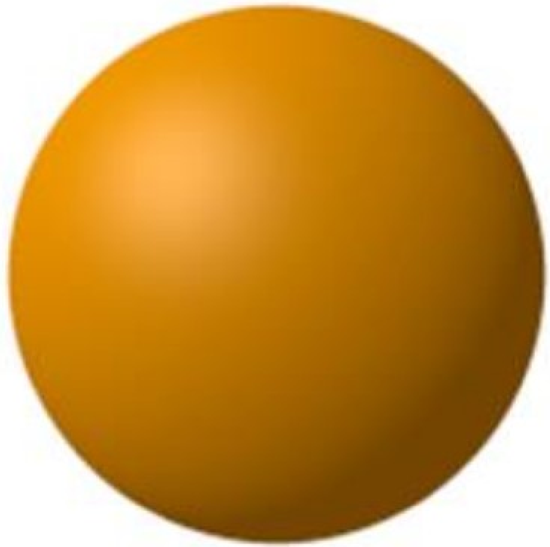
Mesmo as placas gráficas mais recentes, que processam bilhões de polígonos por segundo, com poder de processamento de dezenas de TeraFLOPs, são incapazes de criar realismo em

- Nuvens
- Relva
- Terreno
- Pele
- …

Detalhes de Superfícies

- **Exemplo**
  Modelagem de uma Laranja





Solução:
Tirar foto de uma laranja
Aplicar imagem em uma esfera

Mapeamento de Texturas

Texture Mapping

- Usa imagens para preencher os polígonos

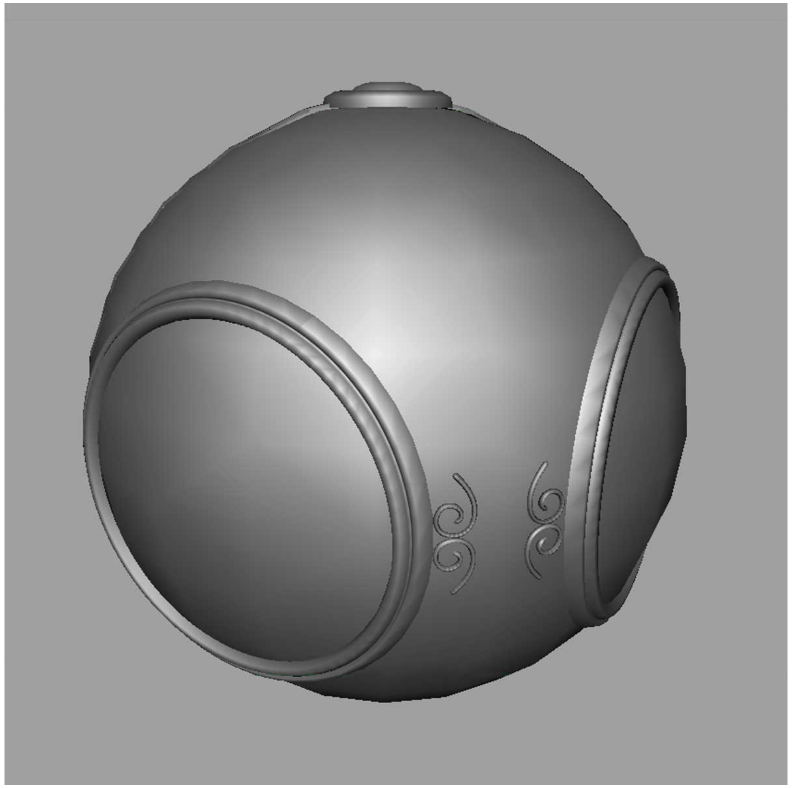
Bump Mapping

- Altera as normais à superfície durante a visualização

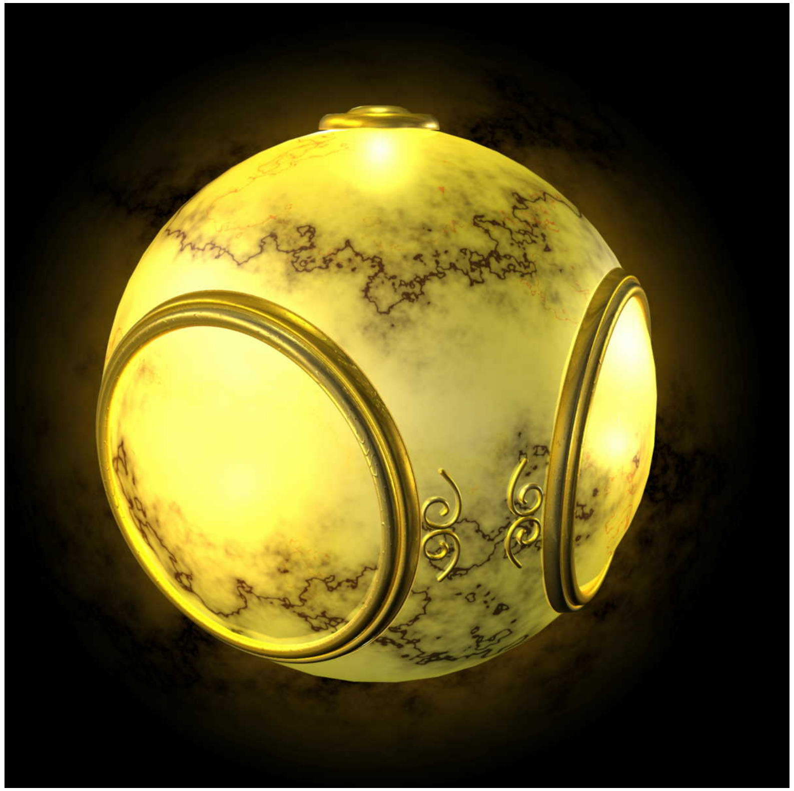
Environment (reflection) Mapping

- Usa um snapshot da cena para fazer texture mapping
- Simular superfícies altamente especulares sem ray-tracing

Mapeamento de Textura



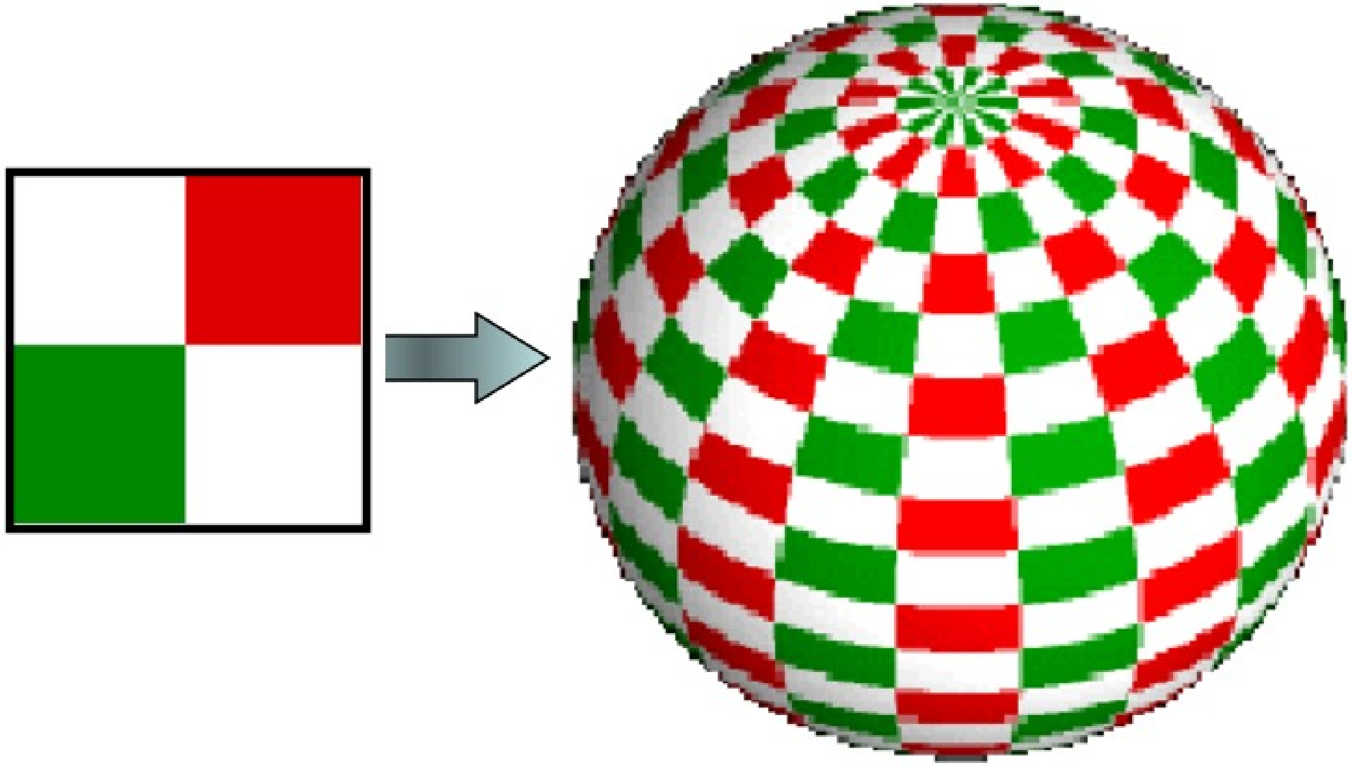
Modelo geométrico
+ Sombreamento



Modelo geométrico +
Mapeameneto de textura

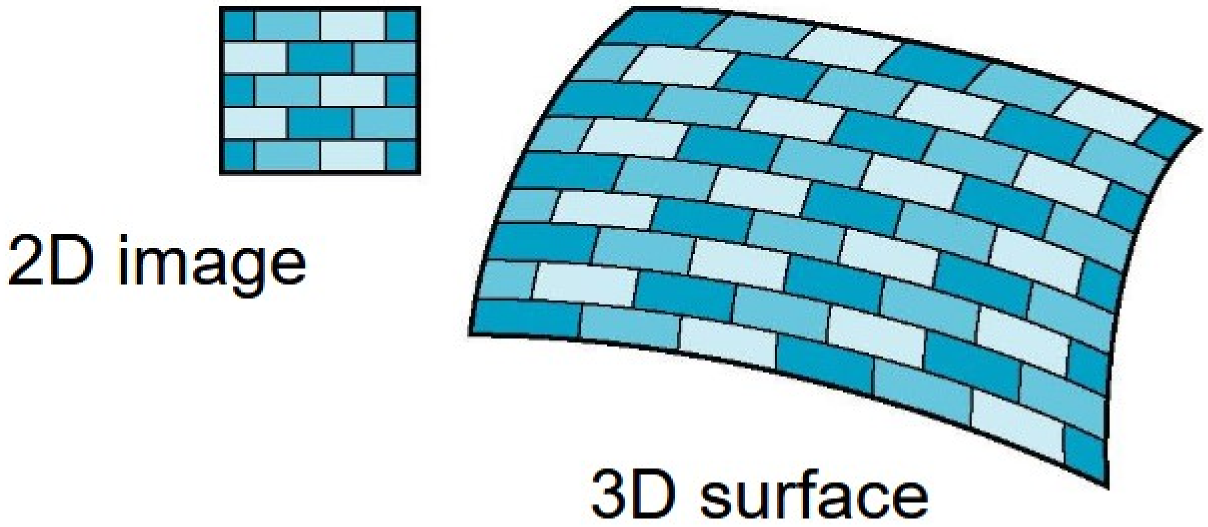
Mapeamento de Textura

A idéia é reproduzir sobre a superfície de algum objeto da cena as propriedades de alguma função – ou mapa - bidimensional (cor, por exemplo)



Mapeamento de Textura

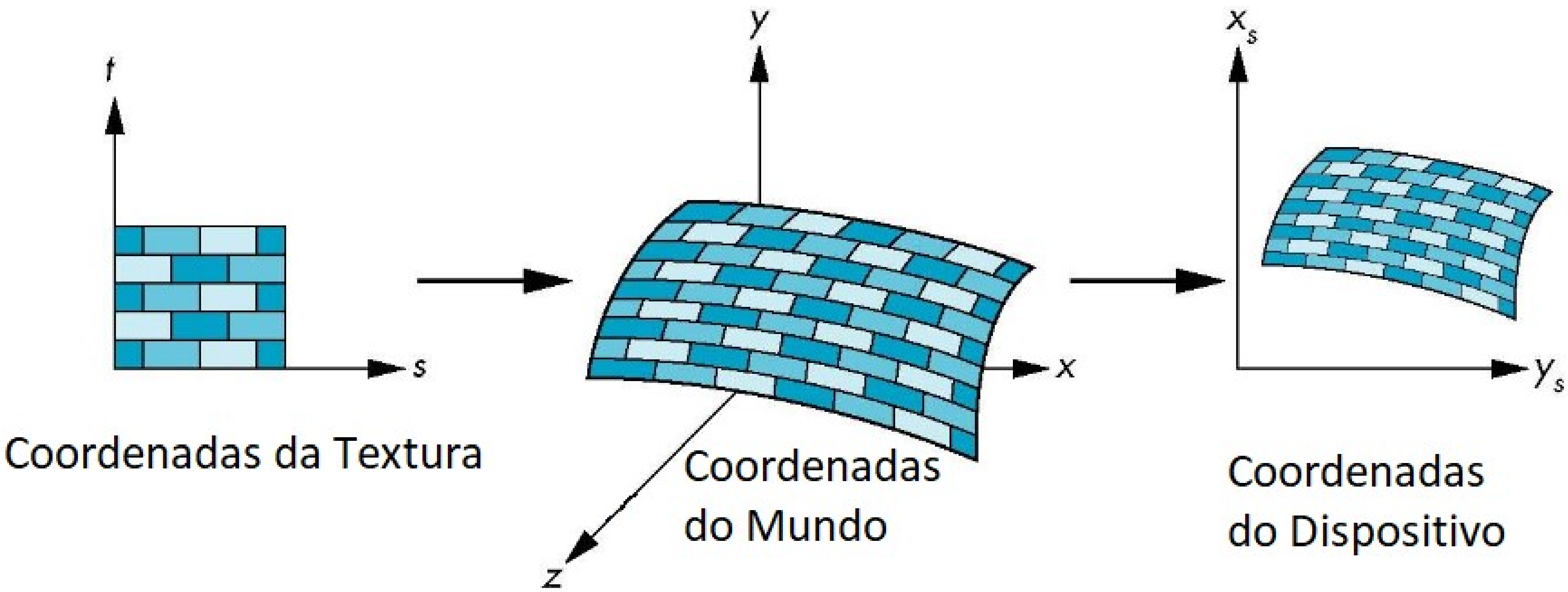
O Mapeamento de texturas envolve o uso de 3 ou 4 sistemas de coordenadas



Mapeamento de Textura

Sistemas de Coordenadas

- Coordenadas da Textura (s, t)
- Coordenadas do Mundo (x, y, z)
- Coordenadas do Dispositivo



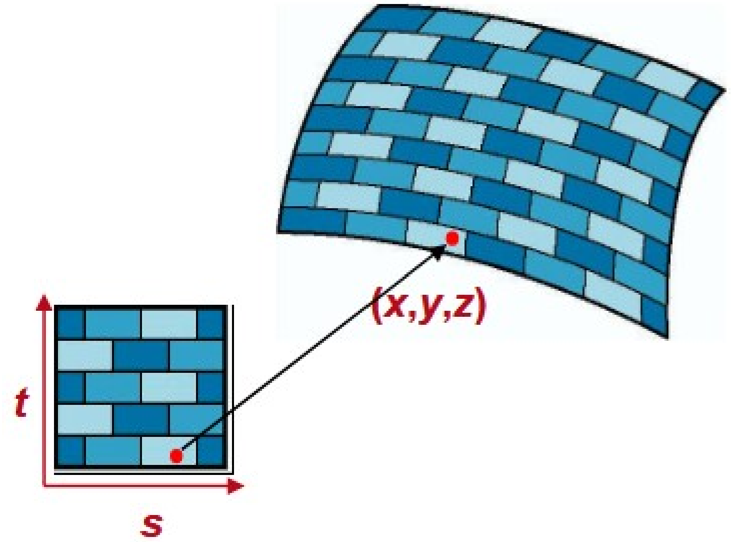
Mapeamento de Textura

Mapeamento de coordenadas (Textura → Superfície)

Para mapear as coordenadas da textura em um ponto na superfície, toma-se uma posição (s,t) na textura, e a mapeia na posição (x,y,z) da superfície

ou seja:

$$x = X(s, t)$$
$$y = Y(s, t)$$
$$z = Z(s, t)$$

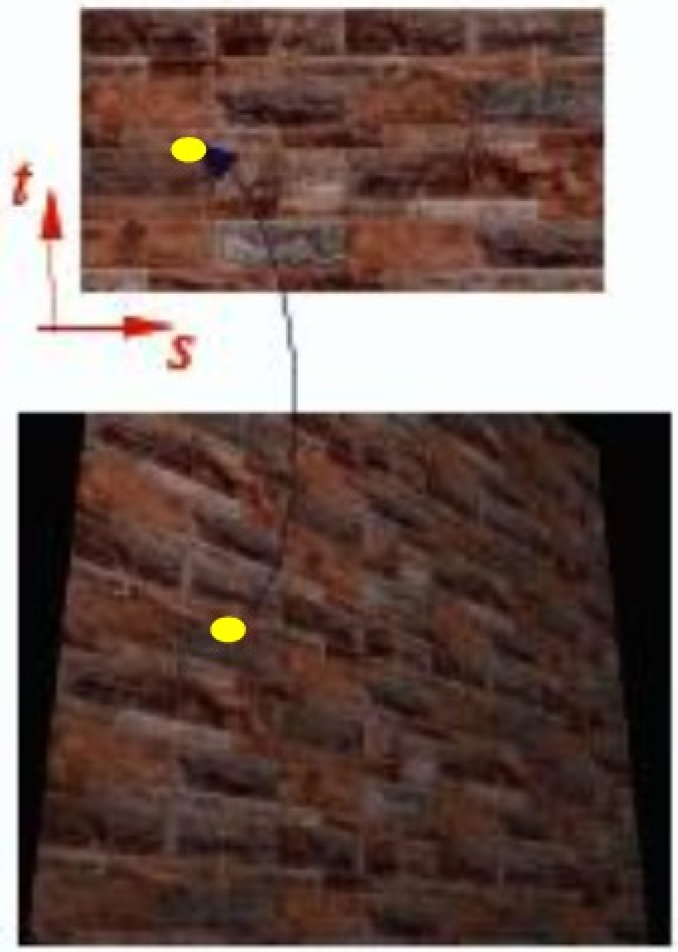


Mas na realidade deve-se fazer exatamente o inverso, ou seja, ir do dispositivo para a textura

Mapeamento de Textura

Backward Mapping

Dado um ponto no objeto, deseja-se saber a que ponto na textura ele corresponde



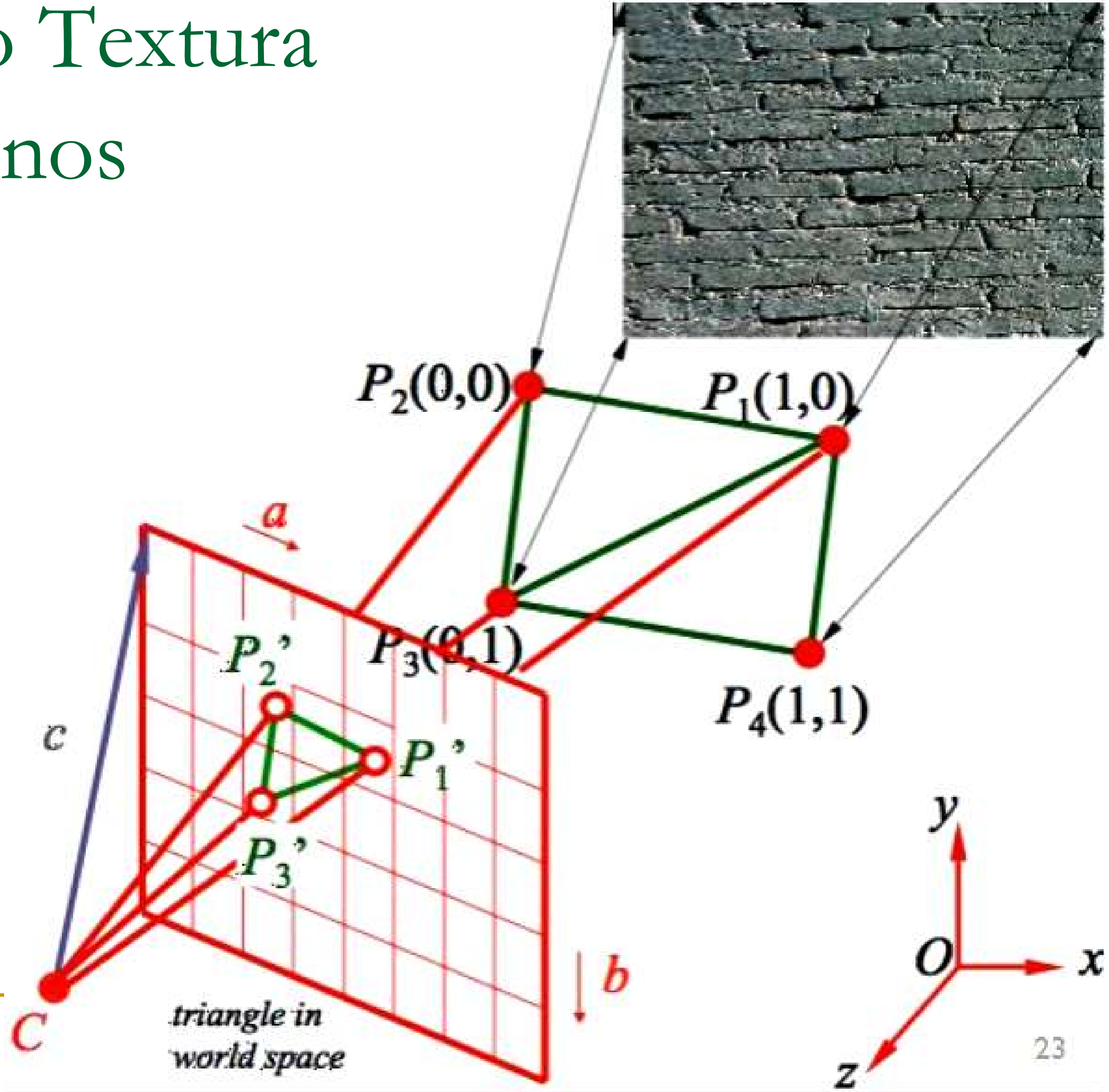
O mapeamento necessário é:

$$s = s(x,y,z)$$
$$t = t(x,y,z)$$

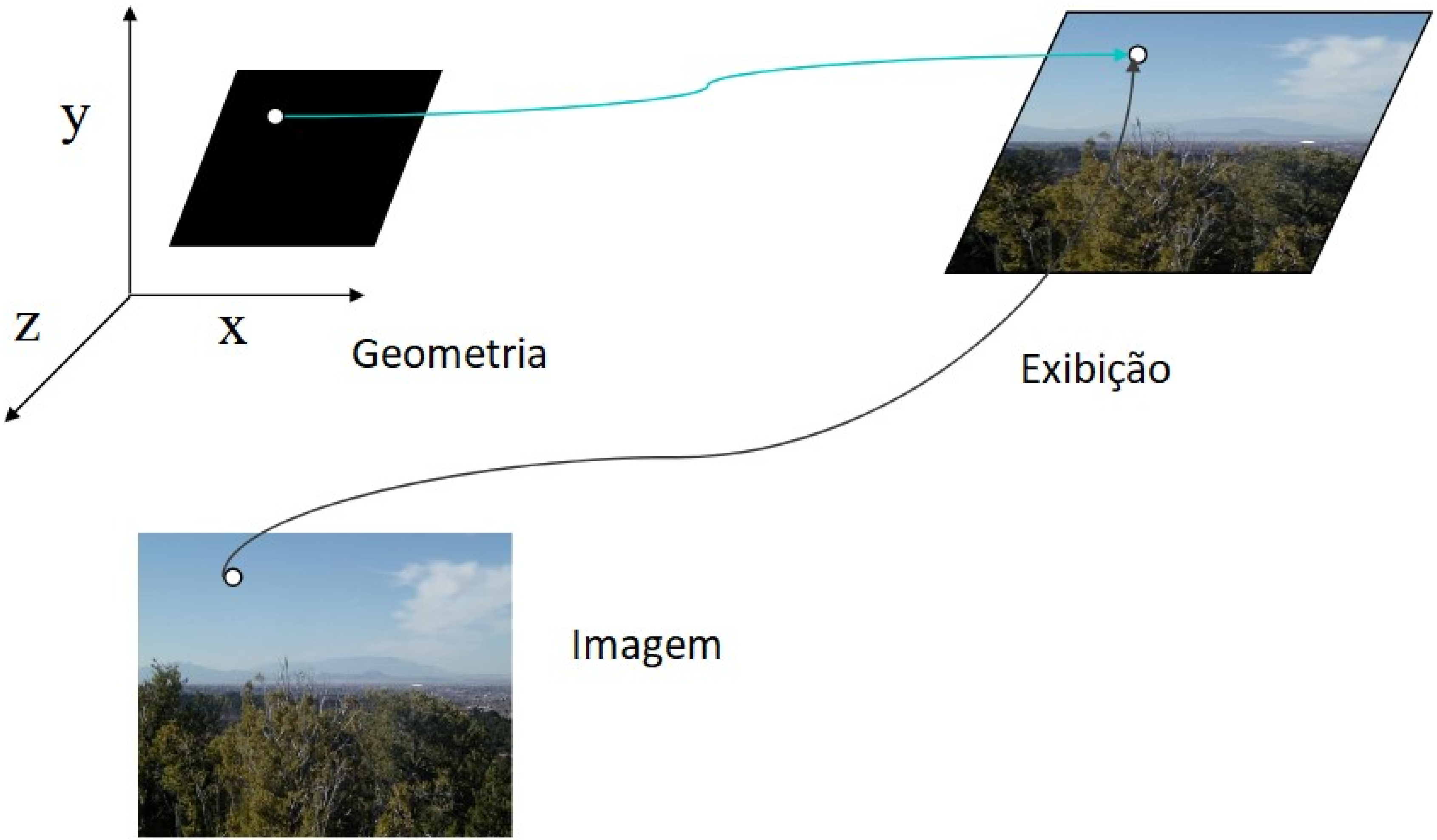
ou

$$s = s(u,v)$$
$$t = t(u,v)$$

Mapeando Textura em Polígonos



Mapeamento de Texturas



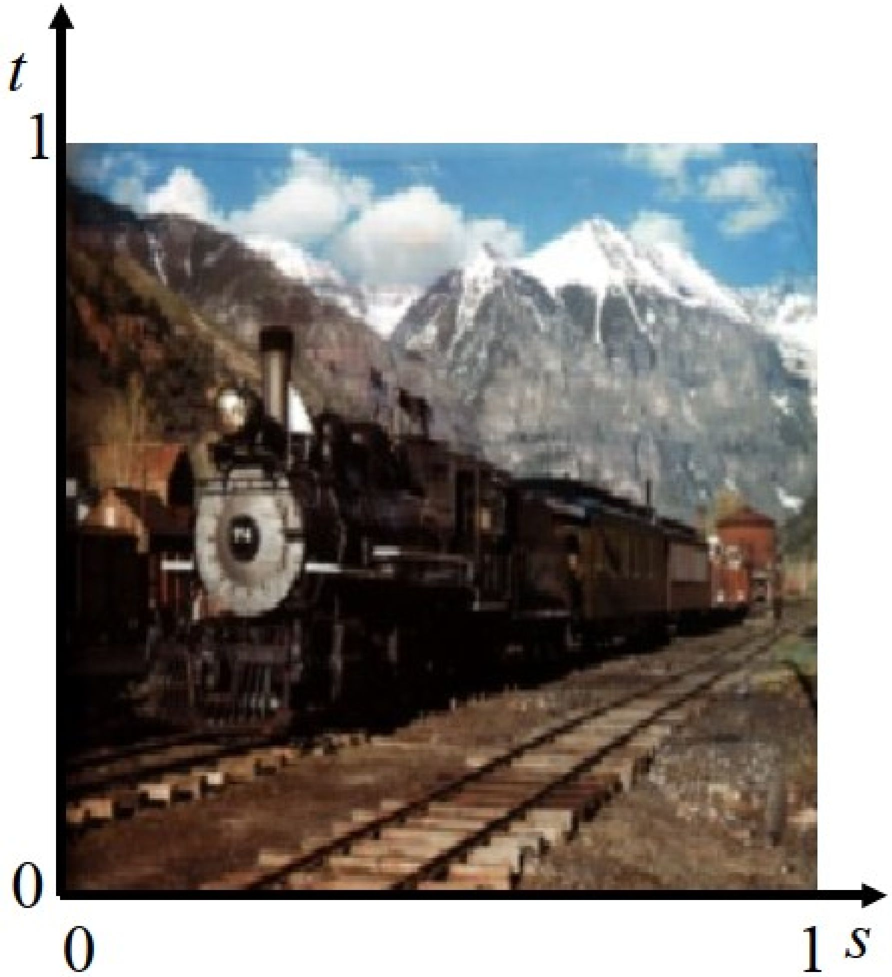
Espaço de Textura

Texturas 2D são funções $T(s, t)$ cujo domínio é um espaço bidimensional e o contradomínio pode ser cor, opacidade, etc

- É comum ajustar a escala da imagem de tal forma que a imagem toda se enquadre no intervalo $0 \le s, t \le 1$
- Normalmente a função em si é derivada de alguma imagem capturada

Se a imagem está armazenada numa matriz $Im\,[0..N{-}1\ ,\ 0..M{-}1]$

Então $T\,(s, t) = Im\,[\ (1 - s)\ N\ ,\ t\ M$



Aplicação do Modelo de Iluminação

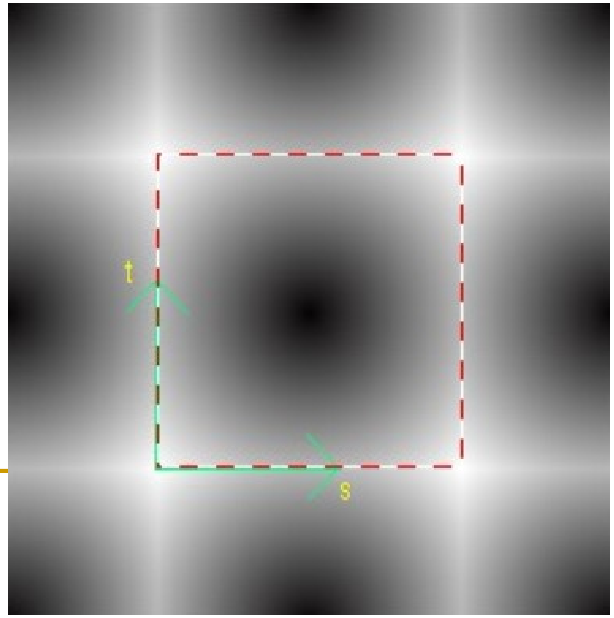
• Pode ser vantajoso assumir que o padrão da imagem se repete fora desse intervalo

$T\,(s,\,t) = Im\,[\,(1 - s)\,N \bmod N,$

$[t\,M] \bmod M\,]$

• A função de textura pode ser também definida algebricamente:

$T\,(s,t) = \text{Raiz}\,[\,(s - 0.5)^2 + (t - 0.5)^2$



Função de Mapeamento

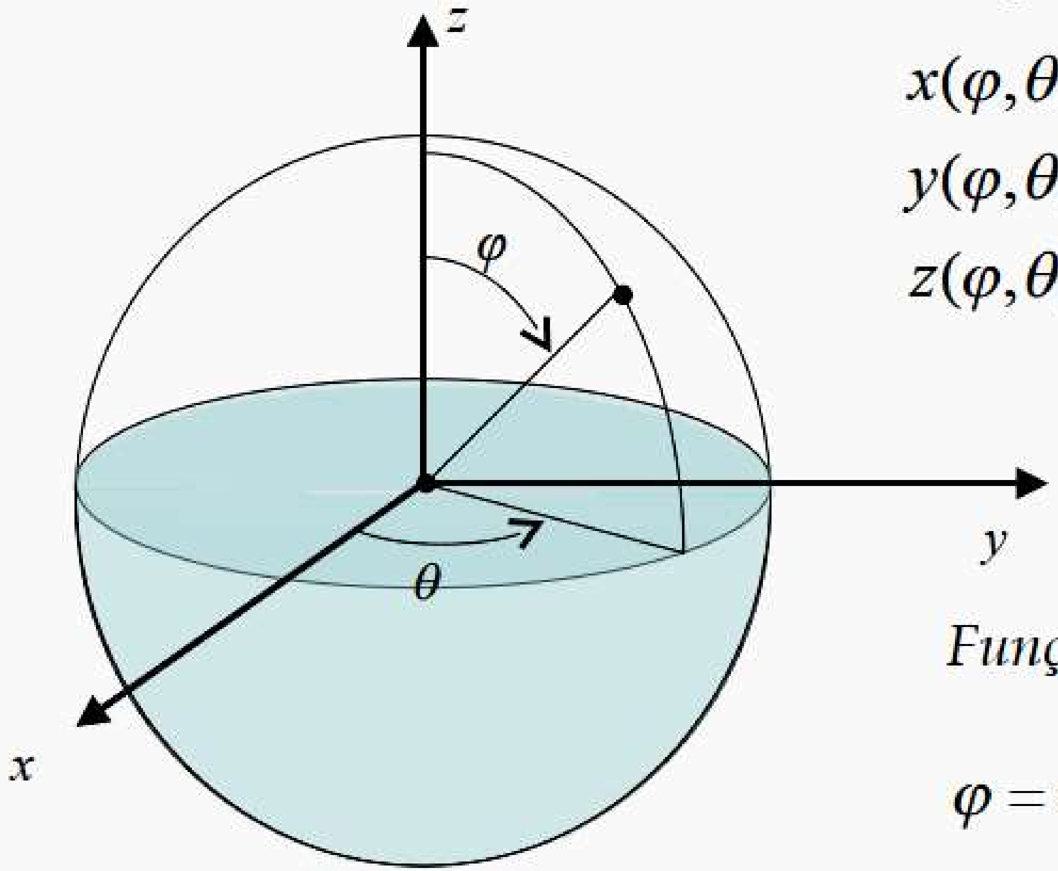
Retorna o ponto do objeto correspondente a cadaponto do espaço de textura $(x, y, z) = F(s, t)$

- Corresponde à forma com que a textura é usadapara “embrulhar” (*wrap*) o objeto

Na verdade, na maioria dos casos, é preciso usar uma função que permita “desembrulhar” (*unwrap*) a textura do objeto, isto é, a inversa da função de mapeamento

- Se a superfície do objeto pode ser descrita em forma paramétrica esta pode servir como base para a função de mapeamento

Parametrização da Esfera



Função de mapeamento

$$x(\varphi,\theta) = \sin\varphi\cos\theta \qquad \varphi = \pi \cdot t$$

$$y(\varphi,\theta) = \sin\varphi\sin\theta \qquad \theta = 2\pi \cdot s$$

$$z(\varphi,\theta) = \cos\varphi$$

Função de mapeamento inversa

$$\varphi = \arccos z \qquad t = \frac{\arccos z}{\pi}$$

$$\theta = \arctan\frac{y}{x} \qquad s = \frac{\arctan\frac{y}{x}}{2\pi}$$

Obtenção da função de mapeamento inversa

$x(\phi,\theta)=\text{sen}\phi.\cos\theta$ ------ (1)

$y(\phi,\theta)=\text{sen}\phi.\text{sen}\theta$ ------- (2)

$z(\phi,\theta)=\cos\phi$ ------------- (3)

$\phi = \pi.t$ ----- (4)

$\theta = 2\pi.s$ --- (5)

usando (3) temos que $\phi = \arccos(z)$ ----- (6)

aplicando (6) na (4) temos $\arccos(z) = \pi.t$ ----- (7)

Assim, $\mathbf{t = arccos(z) / \pi}$ ----- (8)

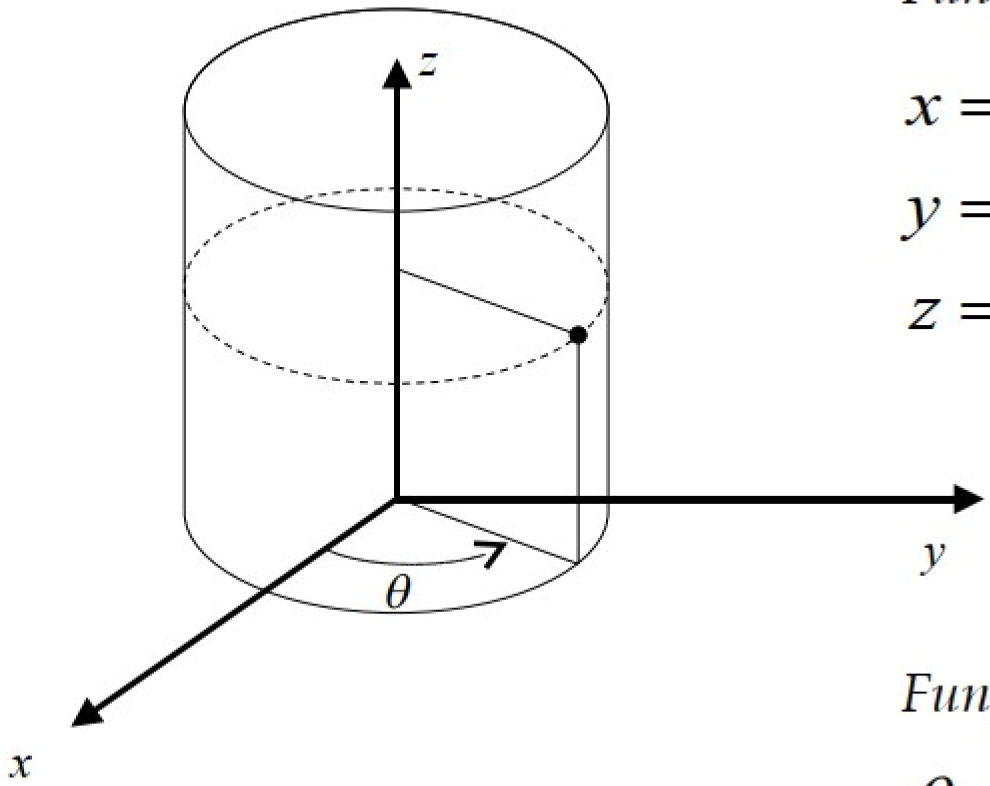
Dividindo (2) por (1) temos que $y/x = \cancel{\text{sen}\phi}.\text{sen}\theta / \cancel{\text{sen}\phi}.\cos\theta$

então, $y/x = \tan\theta$, logo, $\theta = \arctan(y/x)$ ----- (9)

por fim, aplicando (9) em (5) fica $\arctan(y/x) = 2\pi.s$

portanto, $\mathbf{s = arctan(y/x) / 2\pi}$

Parametrização do Cilindro



Função de mapeamento

$$x = \cos\theta \qquad \theta = 2\pi \cdot s$$

$$y = \sin\theta \qquad z = t$$

$$z = z$$

Função de mapeamento inversa

$$\theta = \arctan\frac{y}{x} \qquad s = \frac{\theta}{2\pi}$$

$$z = z \qquad t = z$$

Obtenção da função de mapeamento inversa

$x = \cos\theta$ ------ (1)

$y = \text{sen}\theta$ ------- (2)

$z = z$ ------------ (3)

$\theta = 2\pi.s$ --- (4)

$z = t$ ------- (5)

Dividindo (2) por (1) tem-se que $y/x = \tan\theta$

assim, $\theta = \arctan(y/x)$ ----- (6)

aplicando (6) em (4) tem-se que $\arctan(y/x) = 2\pi.s$

então, $\mathbf{s = \arctan(y/x) / 2\pi}$

por fim, usando (5), $\mathbf{t = z}$

Mapeamento no Cilindro

Equações paramétricas do cilindro

$$
\left.\begin{aligned} x &= r.\cos(2\pi.u) \\ y &= r.\sin(2\pi.u) \end{aligned}\right\} \rightarrow \left\{\begin{array}{l}\text{quando } \mathbf{u} \text{ vai de 0 até 1} \\ \text{cos vai de 0 até } 2\pi\end{array}\right.
$$

$$
z = v.h \rightarrow \left\{\begin{array}{l}\text{quando } \mathbf{v} \text{ vai de 0 até 1} \\ \text{z vai de 0 até h}\end{array}\right.
$$

mapeia um retângulo no espaço *u*,*v* para o cilindro de raio *r* e altura *h* em coordenadas do mundo

Parametrizando Objetos Genéricos

• Quando o objeto não comporta uma parametrização natural?

• Uma sugestão é usar um mapeamento em dois estágios:

- Mapear textura sobre uma superfície simples como cilindro, esfera, etc. aproximadamente englobando o objeto

- Mapear superfície simples sobre a superfície do objeto.

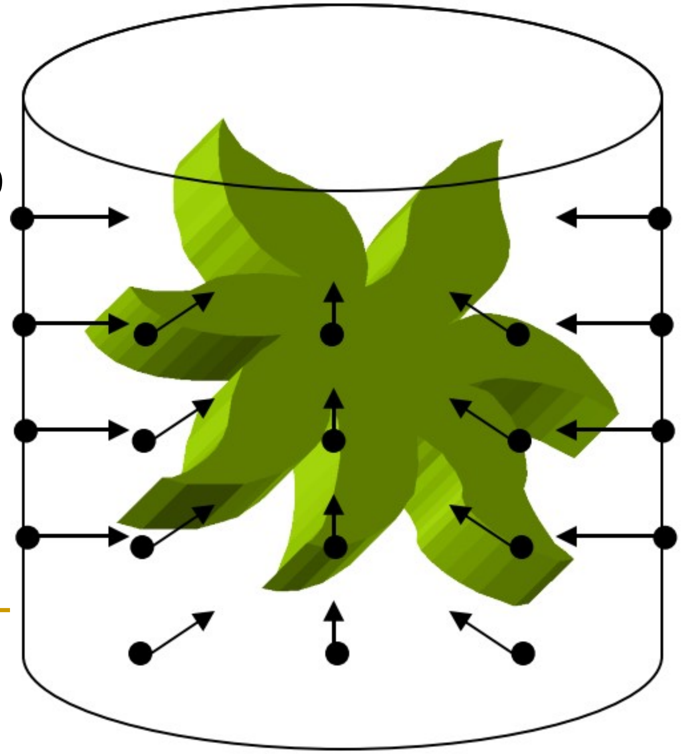
<u>Pode ser feito de diversas maneiras:</u>

• Raios passando pelo centróide do objeto

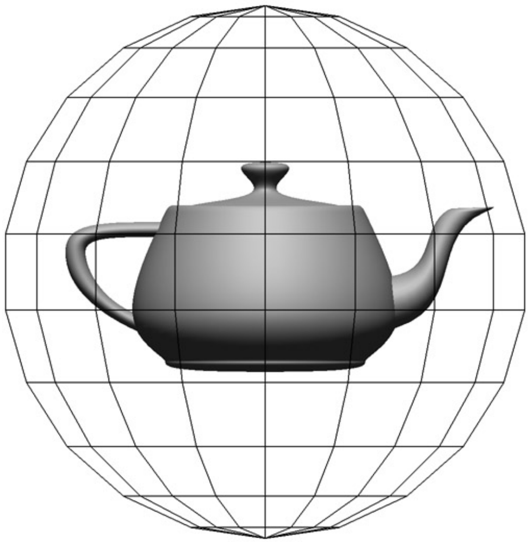
• Raios normais à superfície do objeto

• Raios normais à superfície simples

• Raios refletidos (*environment mapping*)

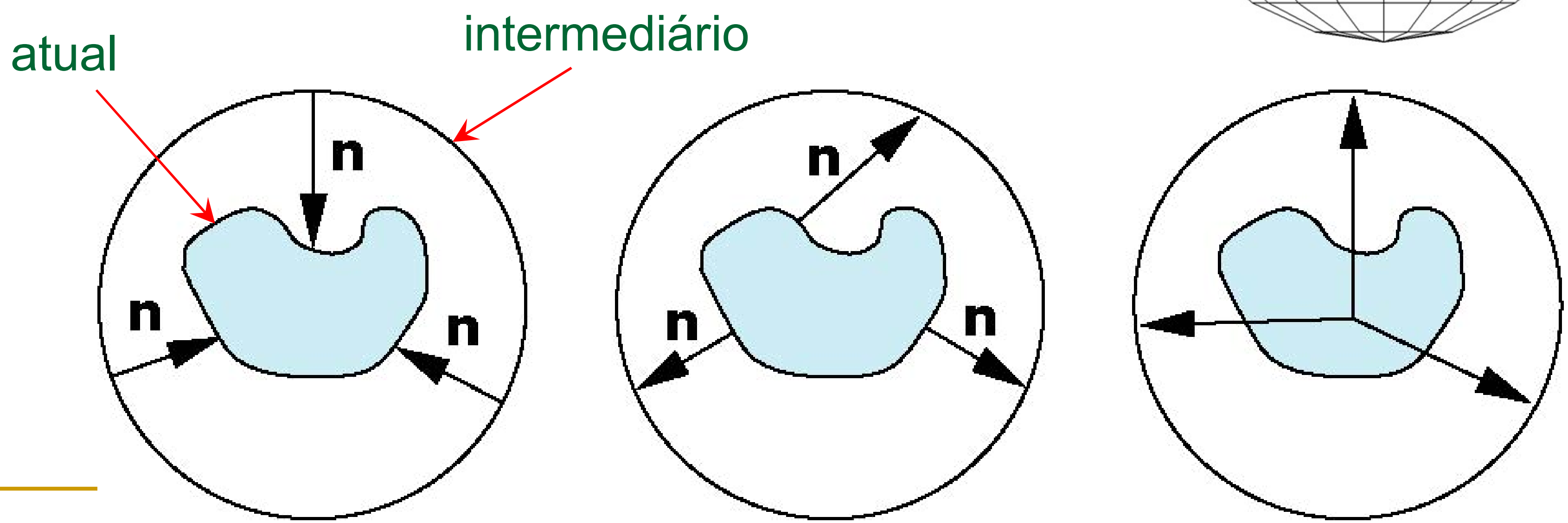


Parametrizando Objetos Genéricos
Second mapping



Mapeia de um objeto intermediário para o objeto atual

- Normais do objeto intermediário para o atual
- Normais do objeto atual para o intermediário
- Vetores do centro para o intermediário



Exemplos

Parametrização cúbica

Projetada em uma esfera

Projetada em um cilindro

Exemplos

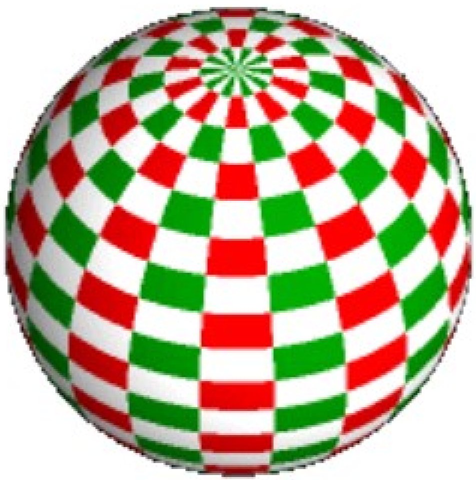
Parametrização cilíndrica

Projetada em uma esfera

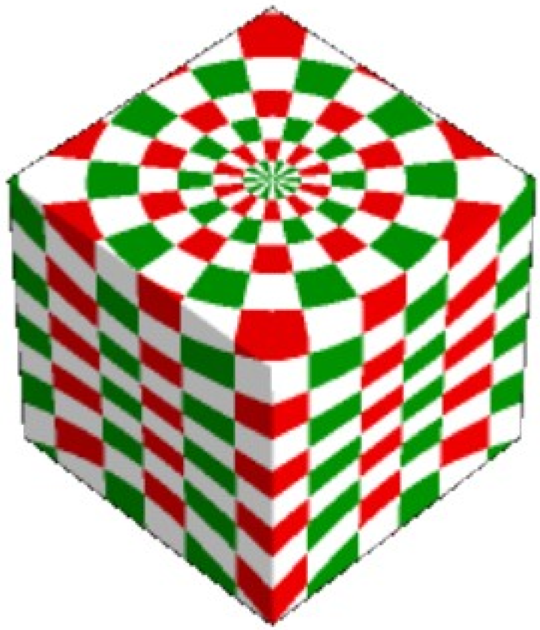
Projetada em um cubo

Exemplos

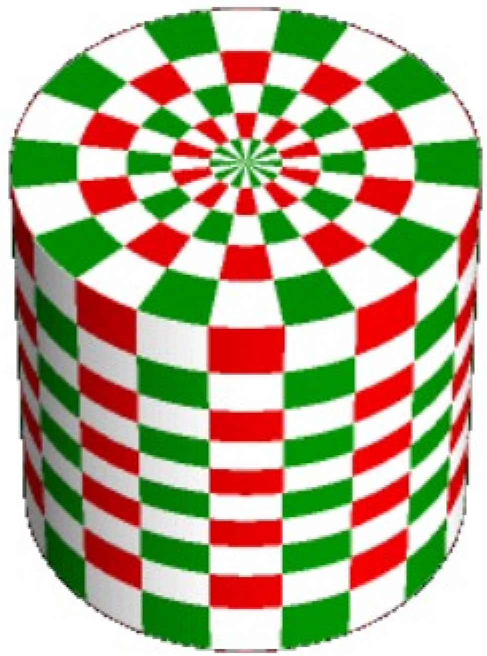
Parametrização esférica



Projetada em um cubo

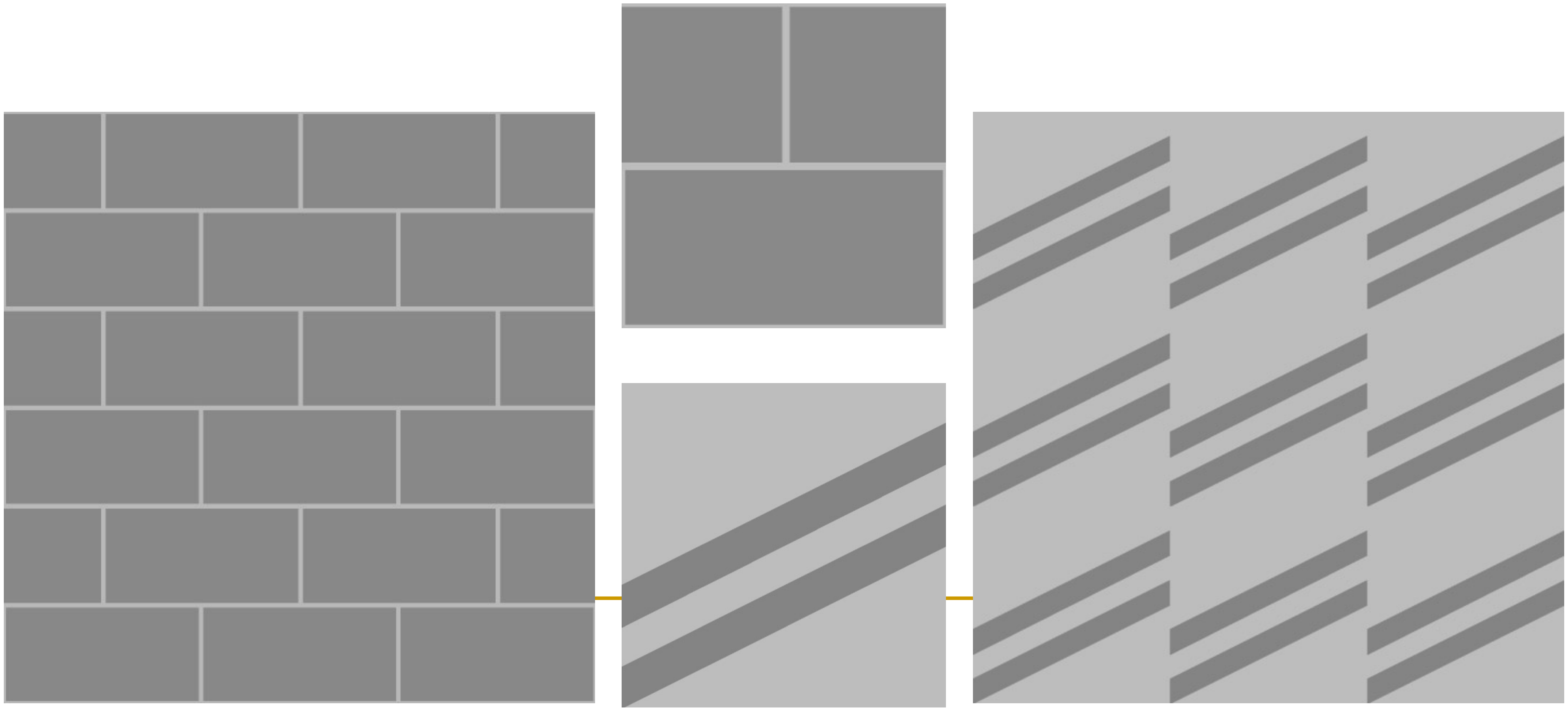


Projetada em um cilindro



Texturas Repetidas

As texturas repetidas precisam ser continuas ao longo das bordas para prevenir desencontros



Repeating textures

Uma textura pode ser repetida através do objeto com as equações:

$$u = fract\left(\frac{repeat_x}{(x_{\max} - x_{\min})} * (x - x_{\min})\right) * T_u$$

$$v = fract\left(\frac{repeat_y}{(y_{\max} - y_{\min})} * (y - y_{\min})\right) * T_v$$

Mapeamento de Texturas em OpenGL

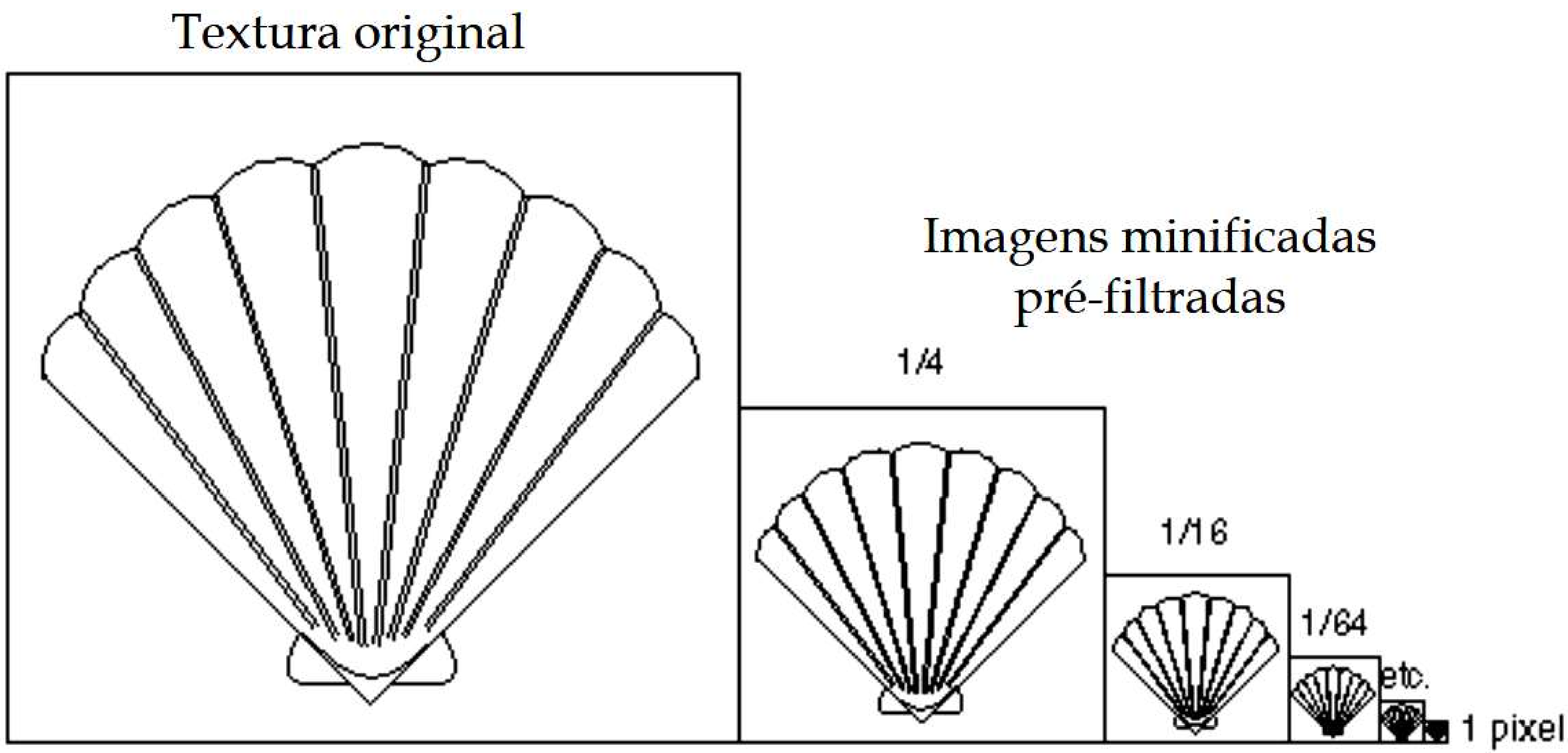
1. Configurar diversos parâmetros
   - **Modos de filtragem**
     - Magnificação ou minificação
     - Filtros mipmap de minificação
   - **Modos de repetição de padrões**
     - Cortar ou repetir
   - **Funções de aplicação de textura**
     - Como misturar a cor do objeto com a da textura
       (Misturar, modular ou substituir texels)

2. Especificar coordenadas de textura
   - Por vértice → glTexCoord*
   - Coordenadas computadas automaticamente
     → glTexGen*

Texturas Mipmap

Como é preciso ajustar o tamanho da imagem contendo a textura aos polígonos, é comum usar diferentes tamanhos de imagens com as texturas:

- nos polígonos pequenos, usar as menores
- nos polígonos grandes, usar as maiores



Texturas Mipmap

- Permite que texturas de diferentes níveis de resolução sejam aplicadas de forma adaptativa
- Reduz aliasing devido a problemas de interpolação
- O nível da textura na hierarquia mipmap é especificada durante a definição da textura

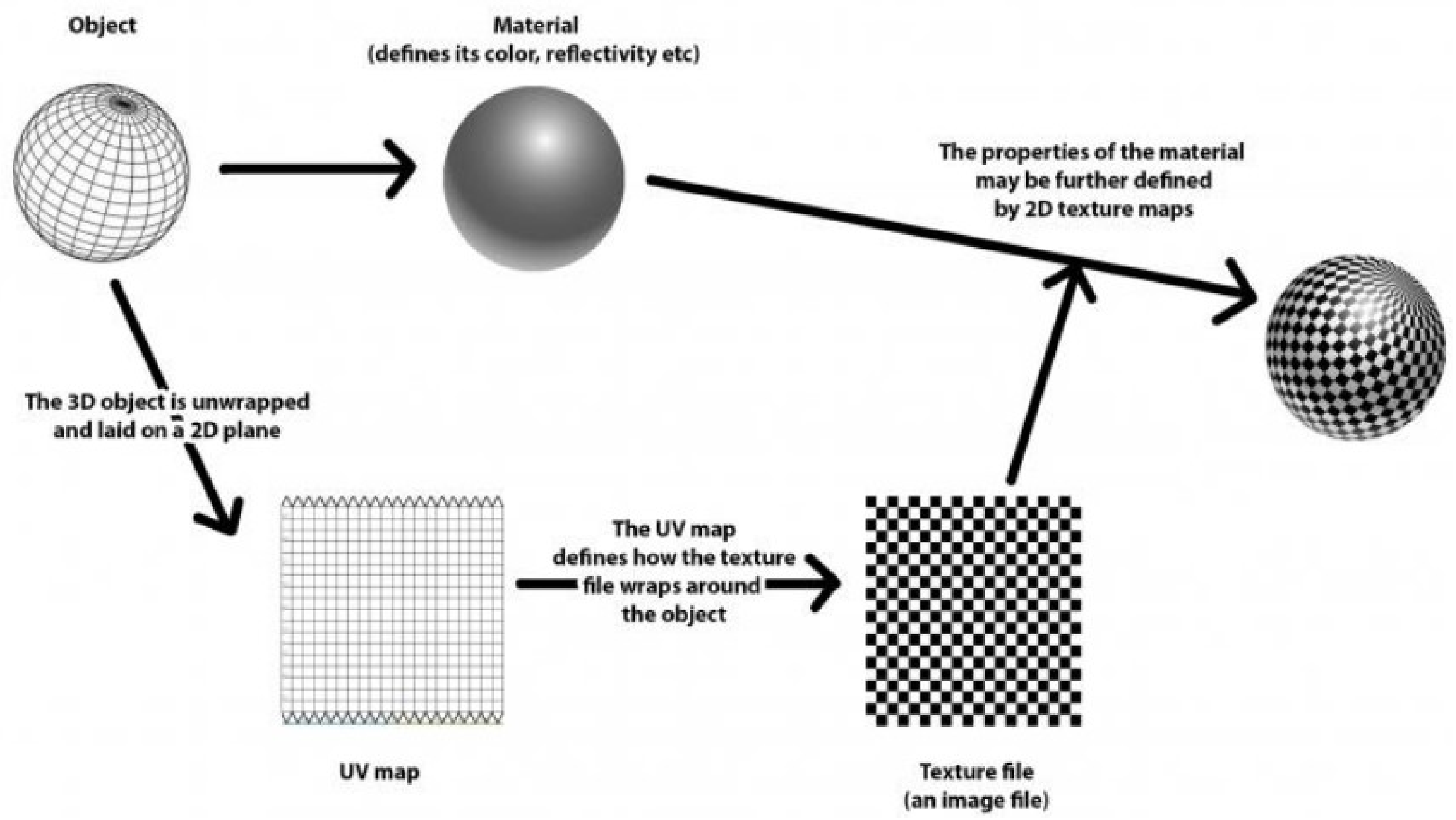
glTexImage*D(***GL_TEXTURE_*D, level, …*** **)**

Processo de Mapeamento de Texturas

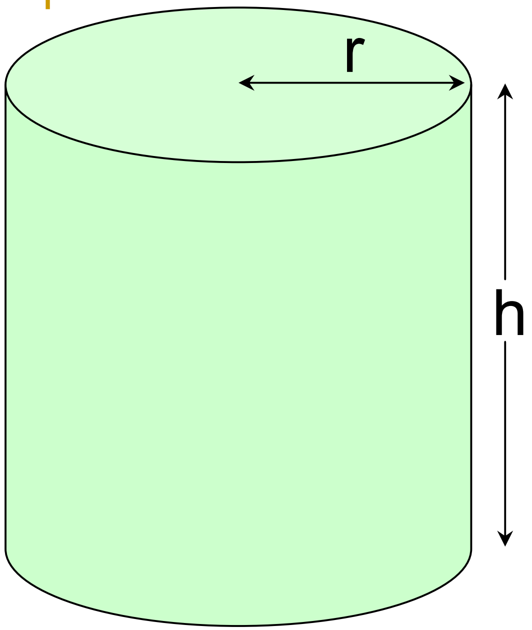
- **Projeção do pixel sobre a superfície**
  - Pontos da superfície correspondentes aos vértices do pixel
- **Parametrização**
  - Coordenadas paramétricas dos vértices do pixel projetados
- **Mapeamento inverso**
  - Coordenadas dos vértices no espaço de textura
- **Média**
  - Cor média dos "Texels" proporcional à área coberta pelo quadrilátero

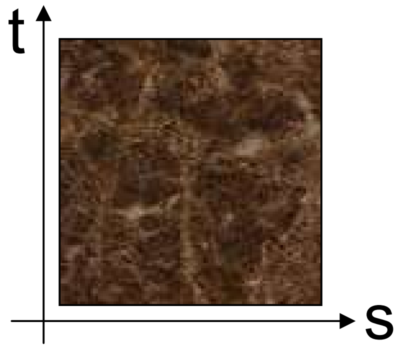
Textura + Brilho

Calcula o brilho e, aplica a textura usando o valor de brilho



Exemplo de aplicação de textura





$x = r.\cos\theta$ ---- (1)

$y = r.\text{sen}\theta$ ---- (2)

$z = h.t$ ------ (3)

de (1) e (2) $\theta = 2\pi.s$ ---- (4), pois se s vai de 0 até 1, então, θ vai de 0 até 2π

dividindo (2) por (1) tem-se $\theta = \arctan(y/x)$ ---- (5)

aplicando (5) em (4) fica $\arctan(y/x) = 2\pi.s$ ---- (6)

logo $\boxed{s = \arctan(y/x) / 2\pi}$

em (3), quando t vai de 0 até 1, então, z vai de 0 até h

tem-se que $z = h.t$, logo, $\boxed{t = z/h}$

Obs. A função atn(y/x) resulta valores entre $-\pi$ e $+\pi$ assim, para ficar na faixa 0 à 2π, faça **$s = \text{atn}(y/x)+\pi)/(2\pi)$**

Código C++

obs.: A image2 que contém a textura é 320 x 240

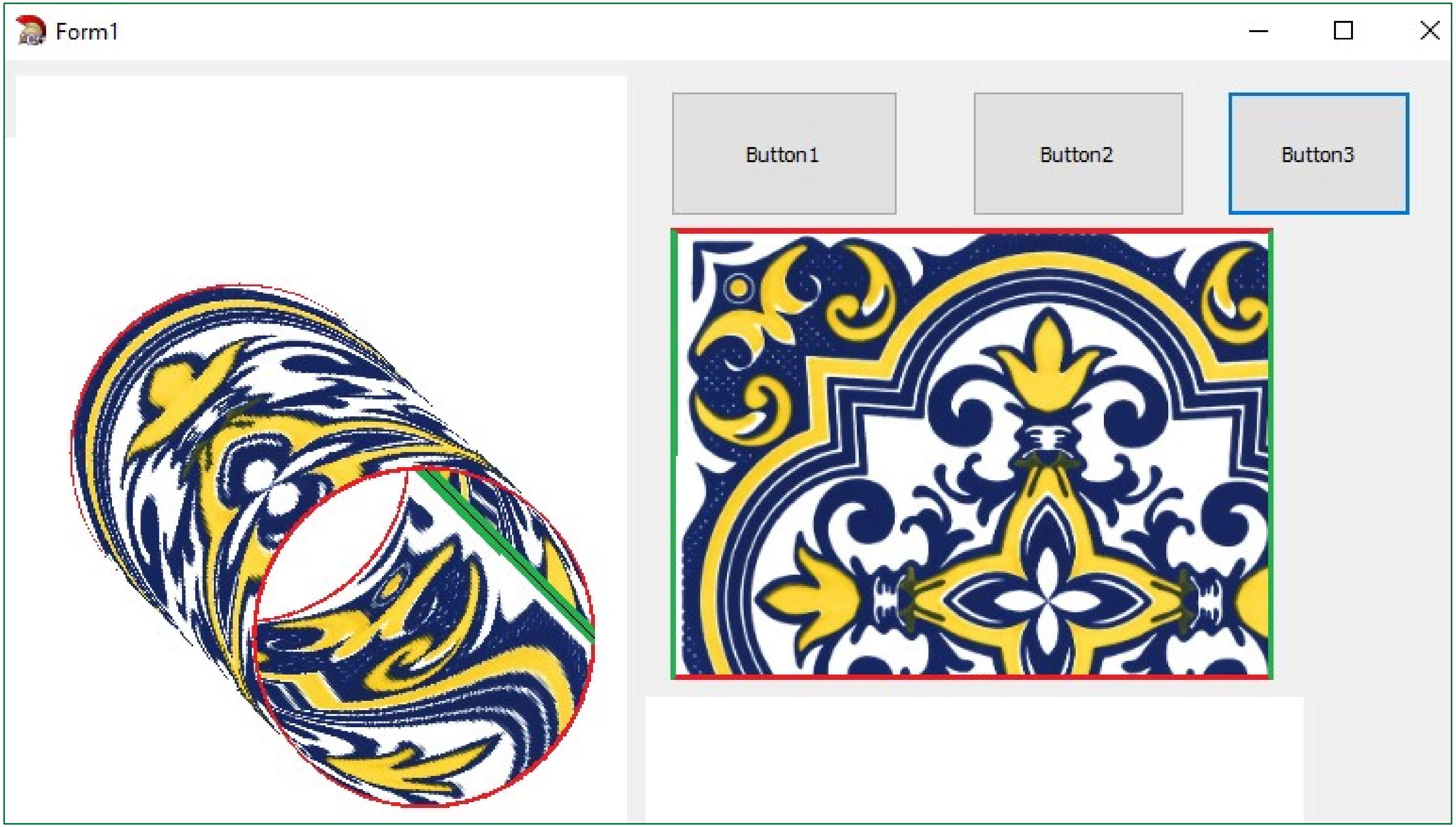
```
float x, y, z, r, h, t, s, pi;
int xt, yt, st, tt;

h = 90.0;
r = 70.0;          pi = 3.14159265;

t = 0.0;
while (t < 1.0)
 {
    s = 0.0;
    while (s < 1.0)
     {
      x = r * cos(2 * pi * s);    //calcula as coordenadas do cilindro
      y = r * sin(2 * pi * s);    //calcula as coordenadas do cilindro
      z = h * t;                  //calcula as coordenadas do cilindro
      s = s + 0.002;
      xt = x + z * 0.70710678118;   // cavaleira
      yt = y + z * 0.70710678118;
      st = s * 320;   // como estamos desenhando de 0 até 1 para s e t,
      tt = t * 240;   // usamos eles para percorrer os pixels da textura
      Image1->Canvas->Pixels[200+xt][200+yt] = Image2->Canvas->Pixels[st][tt];
     }
     t = t + 0.002;
 }
```

Neste caso, não há necessidade de usar a função de mapeamento inversa

Código C++ - Resultado



Prática para entregar agora

Dado um mapeamento de uma textura em um cilindro,

$x = r.\cos\theta$ $r = 30$

$y = r.\text{sen}\theta$ $h = 55$

$z = h.t$

Supondo:

$x = -9{,}270509831$

$y = 28.531695488$

$z = 27.5$

Resposta com 4 casas decimais

Aplique a função de mapeamento inversa para obter os valores de **s** e **t**

respostas: s = 0.3 e t = 0,5

Prática para entregar agora

Dado um objeto definido por:

$z = x^2+y$ com $x \in [10,30]$ e $y \in [20,40]$

Encontre a equação para z, quando $x \in [0,1]$ e $y \in [0,1]$

Resposta: $z = (x.20+10)^2+(y.20+20)$