Realismo Visual

Cap. 7 (do livro texto)

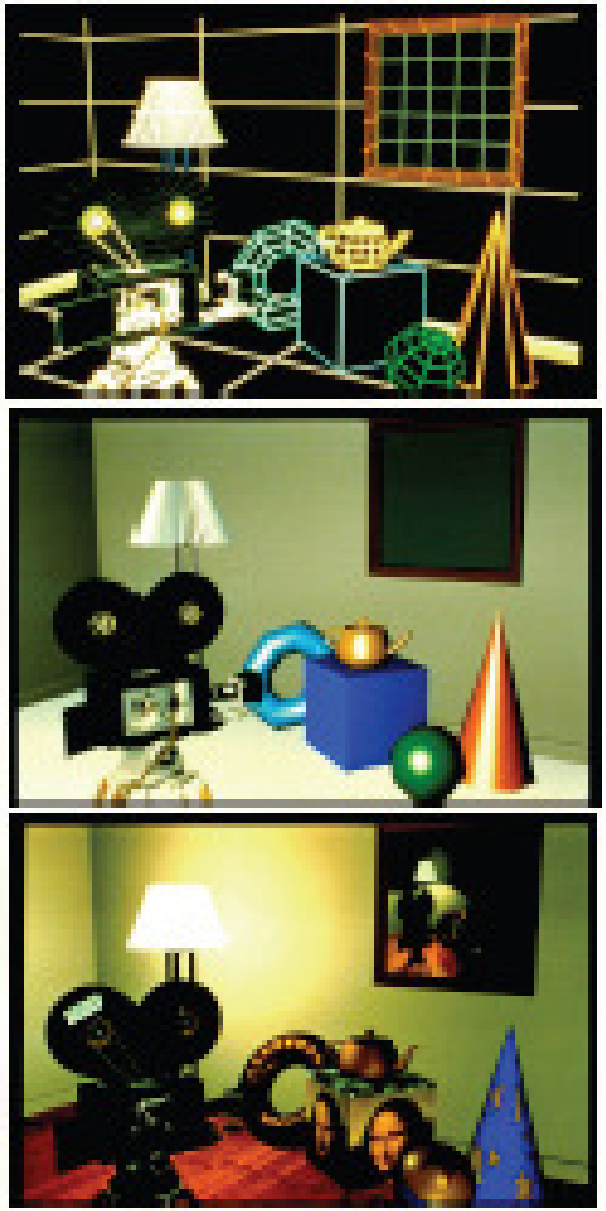
Graduação – UFF - 2014

Objetivos

Melhorar o entendimento das cenas e objetos criados

Possibilidade de representação de dados, objetos e cenas complexas

Realismo até o nível desejado da forma adequada para a aplicação (real time x perfeição física da cena)



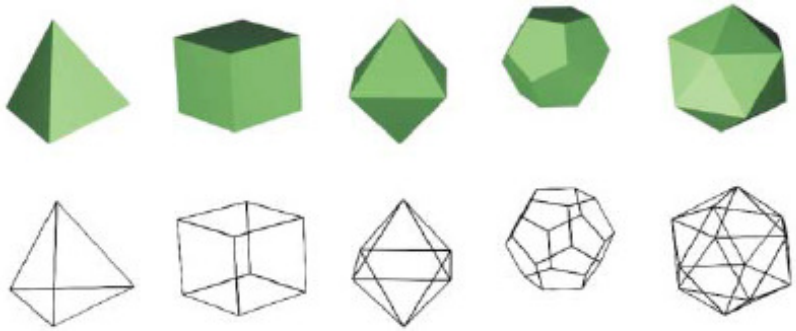
Nível adequado do realismo

Remoção de partes invisíveis do objeto (linhas, superfícies e oclusões por outros objetos)

Sombreamento das diversas superfícies ou *Shading* :

reflexão difusa,

reflexão especular



Demais níveis de detalhes:

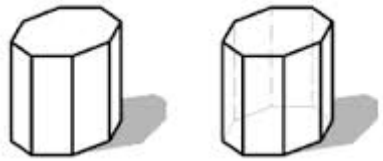
Sombras (*shadows*)

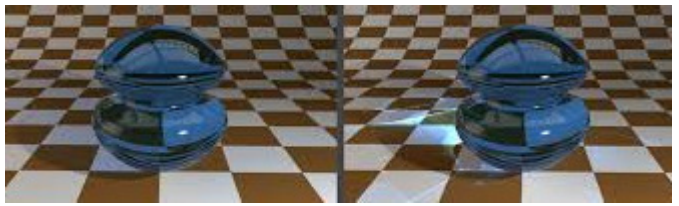
Reflexão,

transparências,

refração,

Texturas

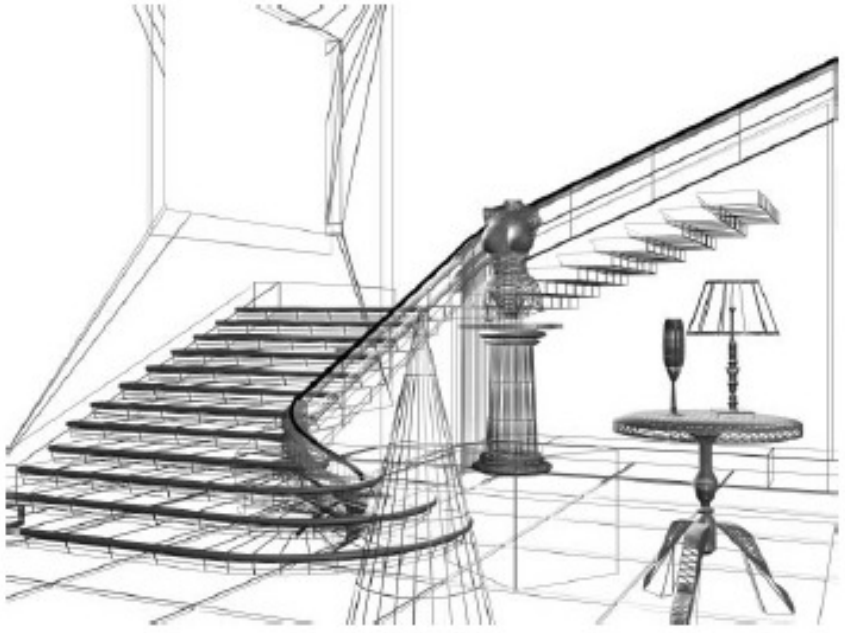




Wire frame : adequado para posicionamentos e desenho, mas não realístico

Todas as linhas são mostradas.

Passo seguinte do realismo eliminar **partes da cena que não são vistas quando objetos opacos são vistos de determinada direção.**



Tratamento de *hiddens* ou *Hidden Line/surface problem*

Eliminação de linhas:
caso particular da

definição de que faces ou superfícies são ocultas por outras do objeto ou cena.

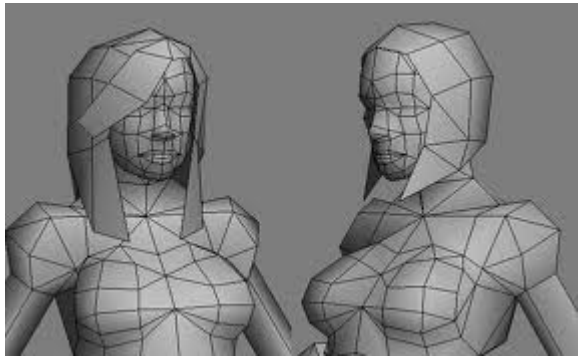
Técnicas de visibilidade

Back face culling

Priority fill ou *painter's algorithm*

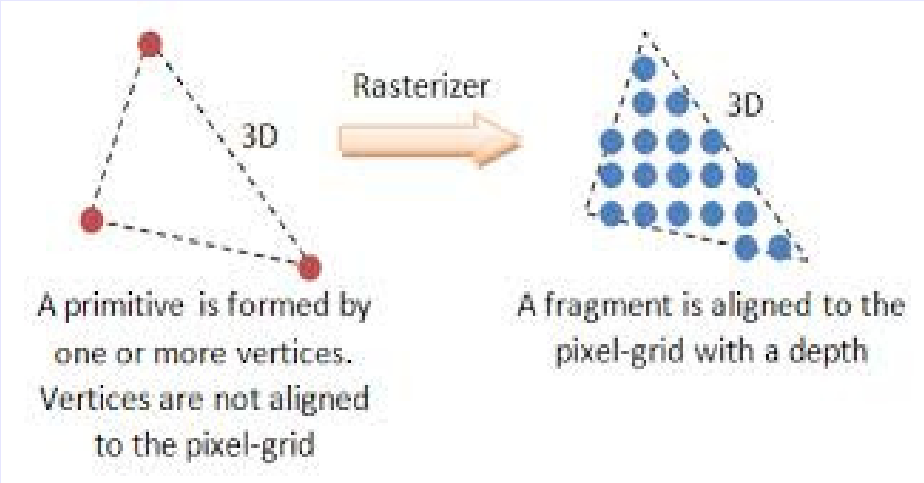
Z- buffer

Ray casting
(*Ray tracing simplificado*
ou aproximado)



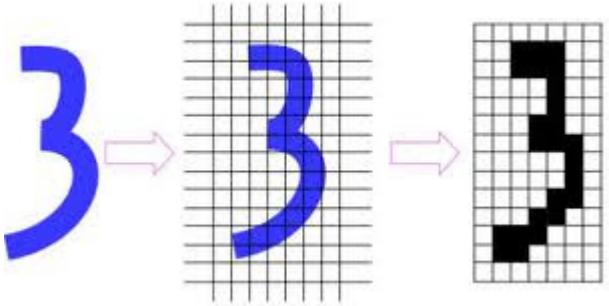
HÁ ALGORITMOS NA FORMA **VETORIAL** E **RASTER**

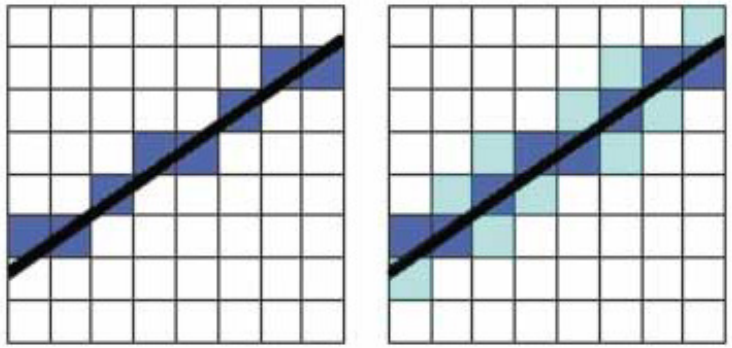
RASTER: o objeto em 3D é tratado na forma final quando já "*discretizado*" em pixels.



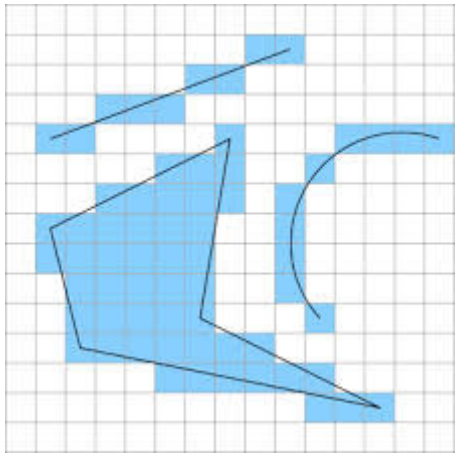
Rasterisation
(ou **rasterization**)
converte uma imagem descrita como vector format para a forma de pixels (dots) para representação em video, printer ou storage in a bitmap file format.

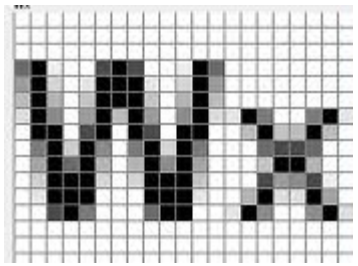
Alising → antialising



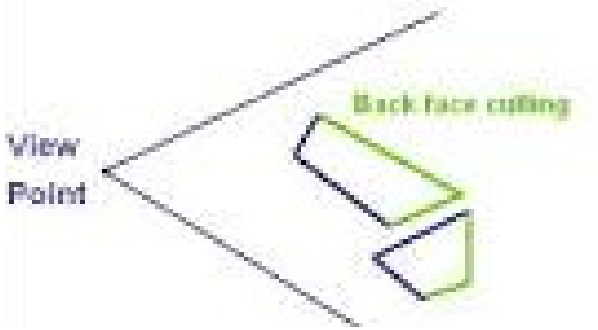


Rasterizar = Usar a malha de pixels para descrever os objetos!





Back face culling



Demo: em javascript:

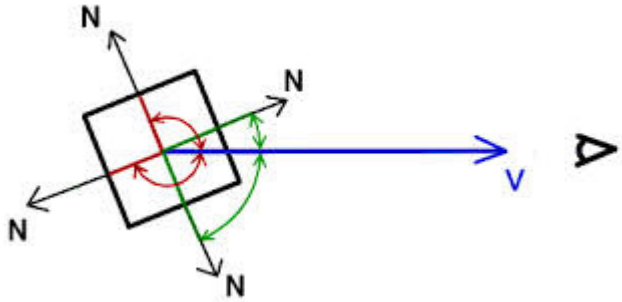
http://echolot-1.github.io/back-face-culling-demo/

echolot-1/**back-face-culling-demo**

*Em CG **back-face culling** determina quando a face de um objeto será **visivel**.*

*Esse processo torna o **rendering** mais **eficiente pois reduz o número de poligonos** a ser desenhado.*

Back face culling



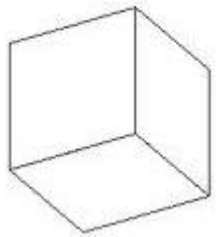
Idéia básica:

Remover faces traseiras dos objetos em relação ao observador

Adequadas para objetos convexos.

OBS :

Ser **não convexo** ≠ ser **côncavo**

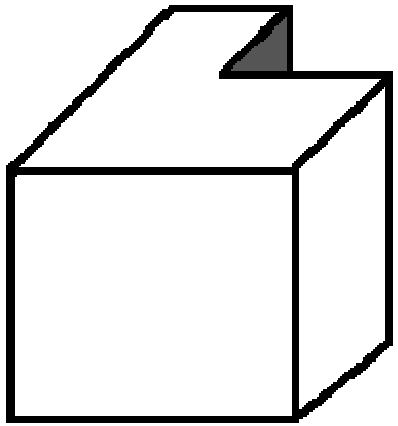


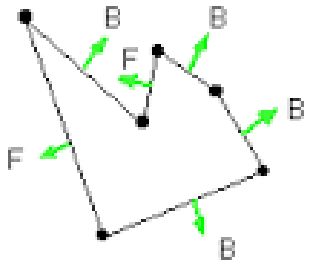
Objetos convexos

Definição:

Formado por faces convexas.

i.e. Formado por polígonos convexos: nos quais a **ligação entre quais quer 2 pontos** internos nunca passa por uma parte externo a face:



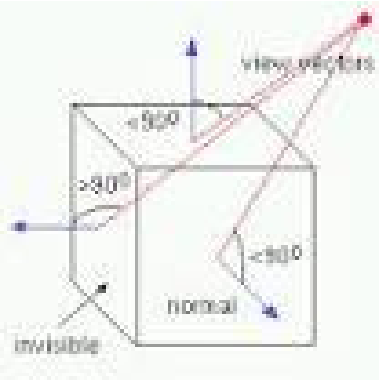


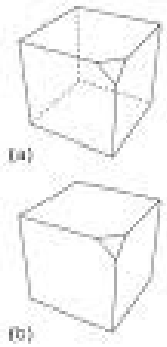
Algoritmo no espaço do objeto

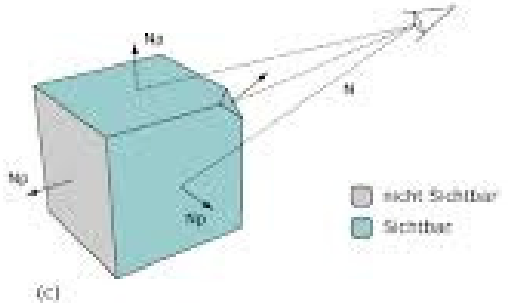
Usa-se a **direção que as normais** às faces fazem com a direção de visualização.

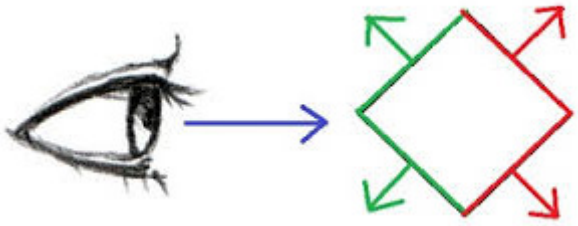
Entre **-90** graus e **90** graus a **face é visível** pelo observador (ou a face é de frente) .

(Back face culling, método de Roberts ou teste da normal)



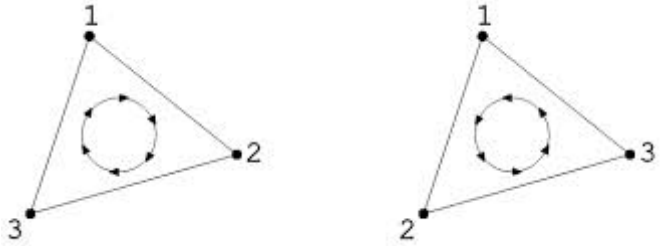


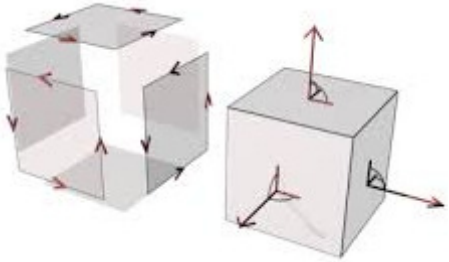


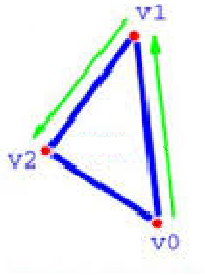


1-Obtêm a normal às faces

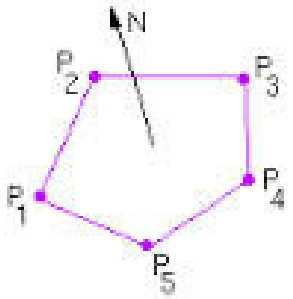
Através do cálculo do **produto vetorial** de dois vetores da face: a ordem dos vértices é importante!







$$N = (V_1 - V_0) \times (V_2 - V_0)$$



$$(V_1 - V_0) \times (V_2 - V_0) = -(V_2 - V_0) \times (V_1 - V_0)$$

2 - Define-se o vetor da direção de visão

3- Verifica-se o ângulo!

Através do **produto interno** entre as normais e a direção de visão, (não é preciso calcular o ângulo) apenas ver se o resultado **é maior que zero** → ângulo entre -90 ° e 90 ° !

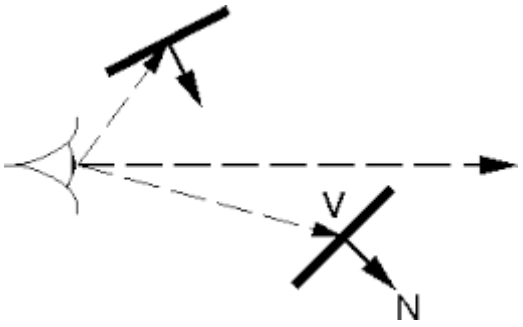
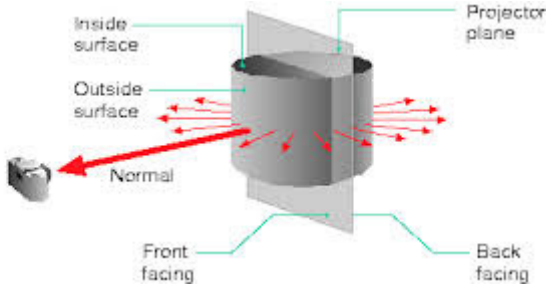


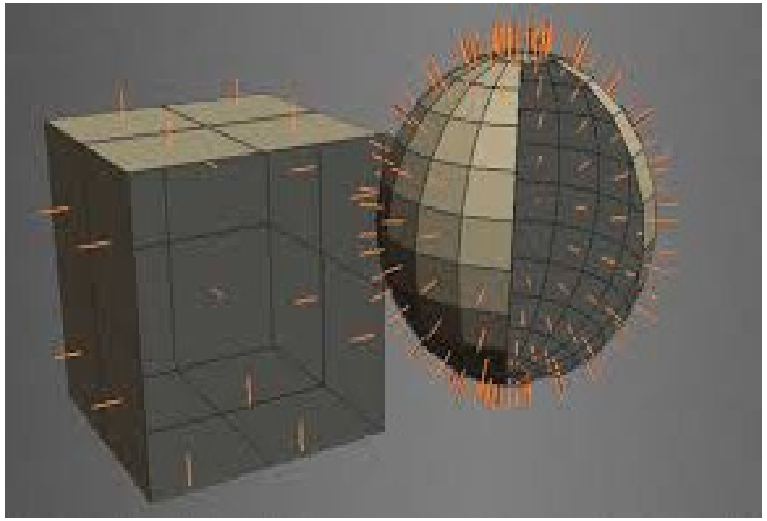
figure 206
Inside and outside surface

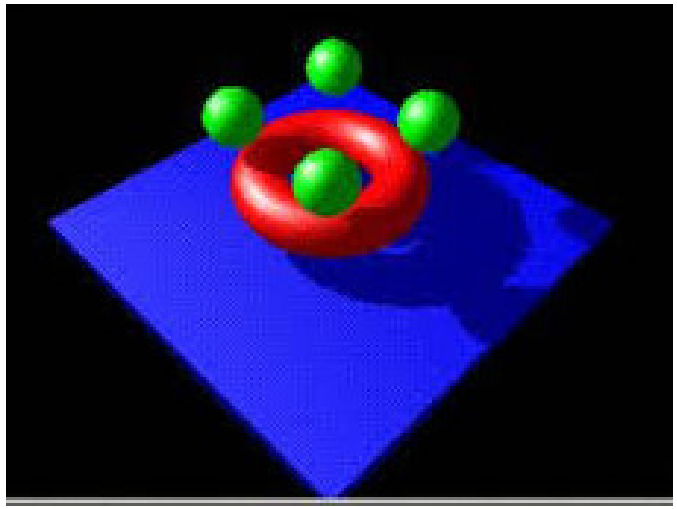


$$(V_0 - P) \cdot N \geq 0$$

Revisitando a fórmula de Euler

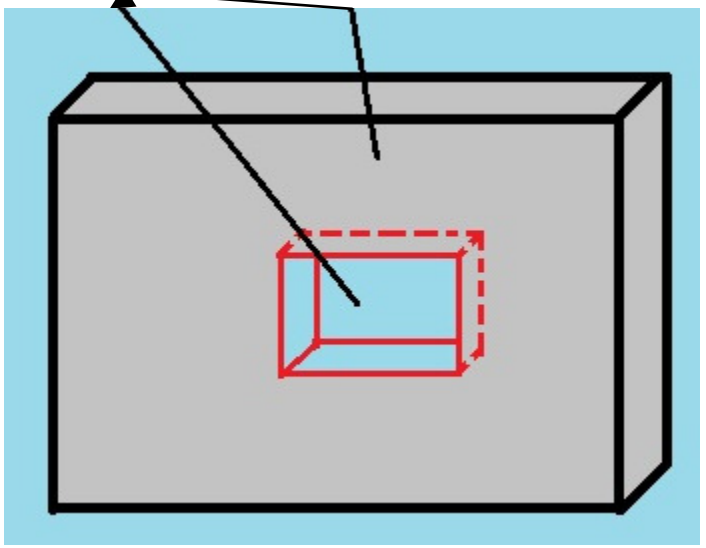
Objetos *topologicamente equivalentes* → se feitos de materiais deformáveis poderiam ser transformados uns nos outros.





Revisitando a fórmula de Euler

Genus G de um objeto : menor **número de furos** que trespassam o objeto.

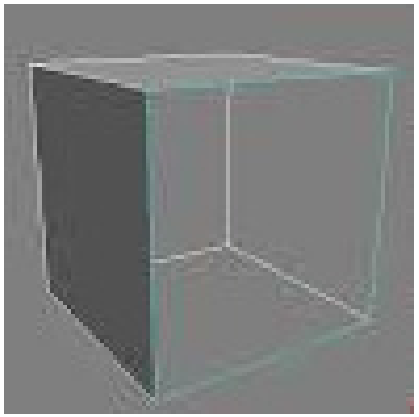


Genus G=**1**

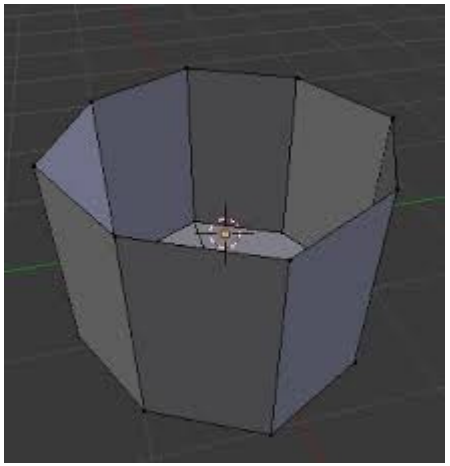
Qual o genus de uma tubulação em Y?
Resposta: Veja o vídeo no Breno onde ele mostra isso por deformação!
Segue o link do vídeo no youtube: http://youtu.be/QkcryL4f6hE

Revisitando a fórmula de Euler

Buracos H : menor **número de furos** que não trespassam ou loops fechados de faces.



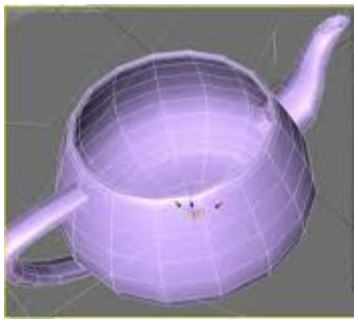
Buracos H=**1**



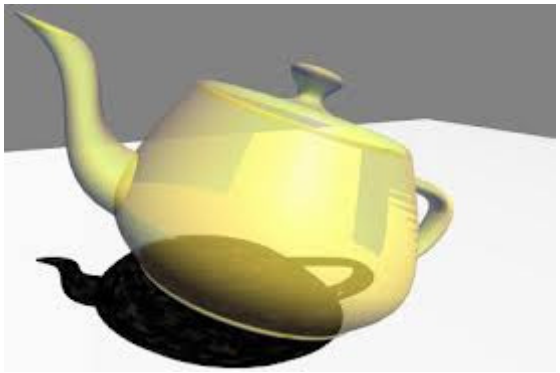
Revisitando a formula de Euler→ **Euler - Poincaré**:

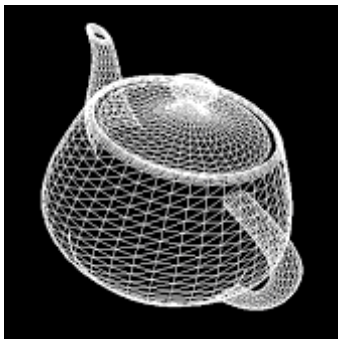
Componentes separáveis ou partes conectadas: C

formula de Euler - Poincaré: V-A+F-H = 2(C-G)



H=1 e G=?





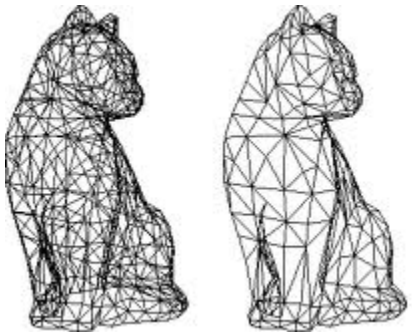
Utah teapot

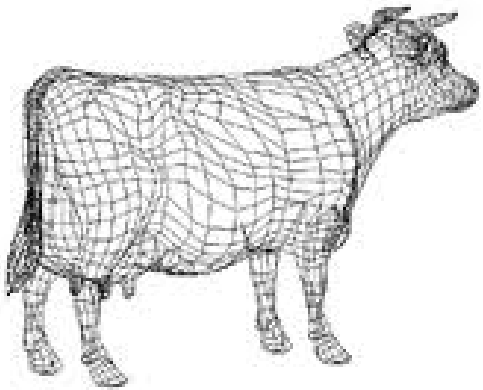
fórmula de Euler : V – A + F = 2

Um **teapot** não é uma **chaleira** ! Nunca é usado para por água no fogo e a ferver!

Importante da modelagem correta para o de uso do objeto adequadamente

Já definir se há **buracos H**, ou furos **trespassantes G** ou **partes conectadas C**, na modelagem inicial do objeto.





Qual o ***Geno*** de um corpo humano para uma modelagem que o tratasse por dentro, como para uma endoscopia?

Painter's algorithm

Painter's algorithm, ou **priority fill**, é uma das soluções mais simples para o problema de Visibility 3m 3D CG.

Na projeção de cena 3D para o plano do video 2D é nescessario **decidir que faces são visiveis ou escondidas (** hidden) .

O nome "painter's algorithm" se refere a tecnica usado por pintores : primeiro pintam detalhes mais longes da cena de depois os cobrem com as partes mais proximas.

O painter's algorithm desenha os polygonos da cena pela sua distancia (depth) os representando nesta orden : dos mais longes para os mais proximos (**farthest to closest)**.

Cobrindo assim as parte invisiveis — ou seja o visibility problem é resolvido com algum custo extra (the cost of having painted invisible areas of distant objects).

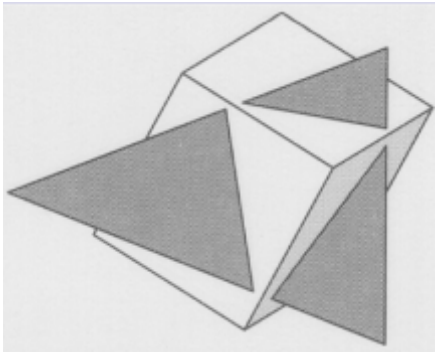
A ordem usada é chamada ***depth order. Esssa ordenação tem uma boa propriedade***: if one object obscures **part** of another **then it is painted after** the object that it obscures.

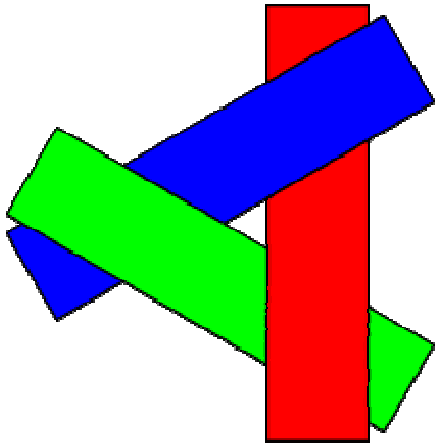
Painter's algorithm

Possibilidade de falha → quando parte de uma face se sobrepoem a outra→ solução divisão da face
(**Newell's Algorithm**).

Essa falha do algoritmo levou ao desenvolvimento do método de

z-buffer ou **depth buffer**





z-buffer algorithm

Idéia básica: testar a distancia (z – depth) de cada superfície para determinar a mais próxima (visible surface).

Considera um array : z buffer(x, y) para cada pixel (x, y) .

Esse array é inicializado com maximum depth. Apos isso o algorithm segue como:

z-buffer algorithm

```
for each polygon P
    for each pixel (x, y) in P
        compute z_depth at x, y
        if z_depth < z_buffer (x, y) then
        set_pixel (x, y, color)= intensidade de P em (x,y)
z_buffer (x, y) = z_depth
```

Vantagem do z-buffer:

sempre funciona e é de simples implementação!

z-buffer *algorithm*

Considerando o quando um ponto é opaco ou transparente.

Conceito de canal alfa ou composição de transparência:

Alpha compositing: processo de combinar a imagem com o fundo crianado a aparencia de **partial** or **full transparency**.

Idéia de translúcidos – modelo RGBα

Considere 2 polígonos, um **vermelho=R (red 1** , 0 , 0, 0.5 **),** e o outro **azul=B(blue** 0 , 0 , 1, 0.5 **)** *rendered* em um fundo **verde=G(green background** (0 , 1 , 0 , 0**).**

Ambos **50% transparentes**. Se o **V(red)** estiver na frente de todos, depois o **azul (blue)** e o **verde** for o fundo (**green background**).

No final deve-se ter 50% R, 25% G e 25% B (Rederizando de traz para a frente):

Green background. (0 , 1 , 0)

Poligono blue : (0 , 0.5 , 0.5) – conta 50% da cor sobre o fundo!

Poligono red: (0.5 , 0.25 , 0.25) – conta 50% da cor sobre o fundo!

z-buffer *algorithm com* canal alfa!
ou
Alpha-blending + the Z-buffer

Given: A list of polygons {P1,P2,.....Pn} and a background

Output: A COLOR which displays the intensity of the polygon surfaces.

Initialize: z-depth and z-buffer(x,y) , -buffer(x,y)=max depth; and
COLOR(x,y)=background at (x,y)

Begin:

z-buffer *algorithm com* canal alfa!

```
for(each polygon P in the polygon list)
do{
    for(each pixel(x,y) that intersects P)
    do{
       Calculate z-depth of P at (x,y)
       If (z-depth < z-buffer[x,y])
       then{
              z-buffer[x,y]=z-depth;
     COLOR(x,y)=Intensity of P at(x,y);
            }
       #considerando α:
      Else if (COLOR(x,y).opacity < 100%)
      then{ COLOR(x,y)=Superimpose COLOR(x,y) in front of Intensity of P at(x,y); }
      #End consideração do α:
        }
    }
display COLOR array.
```

Ray tracing *simplificado ou aproximado ou*

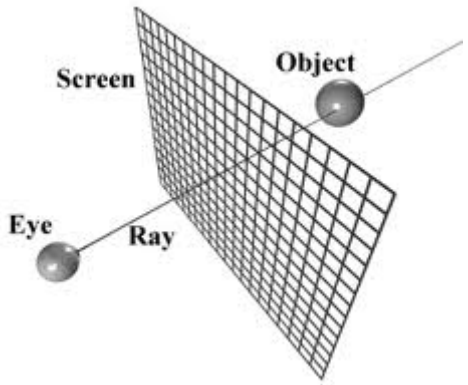
Ray casting lança raios a partir do observador de forma a perceber a distância dos objetos que compõem a cena.

Os raios são emitidos **a partir do observador**, (no sentido inverso do que acontece na natureza), para reduzir recursos computacionais (pois a maior parte dos raios de luz que partem da fonte não chegam ao observador).

Ray casting

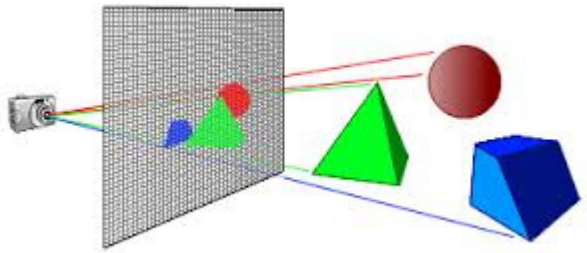
Supõe-se um raio do olho do observador passando por **cada ponto da tela** a ser desenhada. O ponto da tela receberá a cor do objeto que for atingido na cena pelo raio.

O calculo das interseções é o ponto chave do algoritmo.



Ray casting

permite remover as superfícies escondidas utilizando as informações obtidas a partir das primeiras intersecções encontradas pelos raios lançados a partir do observador.



Ray tracing (rastreamento)

Método recursivo, onde recorre ao lançamento de raios secundários a partir das interseções dos raios primários com os objetos.

Ray casting é apropriado para a renderização de jogos 3D em tempo-real.

Durante a viagem do raio pode acontecer: absorção, reflexão ou refração. A superfície pode refletir toda ou apenas uma parte do raio numa ou mais direção. A soma das componentes absorvidas, refletidas e refratadas tem que ser igual ao inicial.

Um *modelo de iluminação*

é um modelo utilizado para calcular a intensidade de luz observada em um ponto na superfície de um objeto.

Modelos :

- Empíricos x Físicos
- Locais x Globais

Sombreamento das diversas superfícies

Shading :

Shading se refere ao processo de alterar ou não a color do objeto/superficie/poligono numa cena 3D, basedo em um modelo de iluminação para o criar um efeito realistico.

Modelos mais comuns:

Flat ou constantes, intensidade variável, normais variáveis, como funções de reflexão bidirecionais (BRDF), radiosidade.

Modelo de iluminação: empírico e local.

Quando se *renderiza* um objeto onde o **tom de um ponto** é determinado por:

- A descrição das fontes de luz disponíveis
- As superfícies dos objetos da cena
- A posição relativa entre as fontes de luz e as superfícies dos objetos

Descrição das fontes de luz disponíveis

Deve incluir detalhes como:

- Onde estão localizadas nas coordenadas da cena
- Intensidade, cor, número
- Tipo:

Ambiente – uniformemente distribuída em todas as direções da cena

Direcional, Pontuais ou

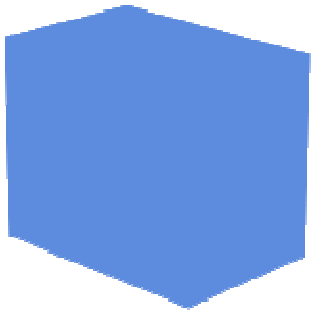
Com áreas de dimensões definidas

Shading com luz ambiente

Intensidade constante, cor constante → afeta igualmente todas as faces e objetos da cena:

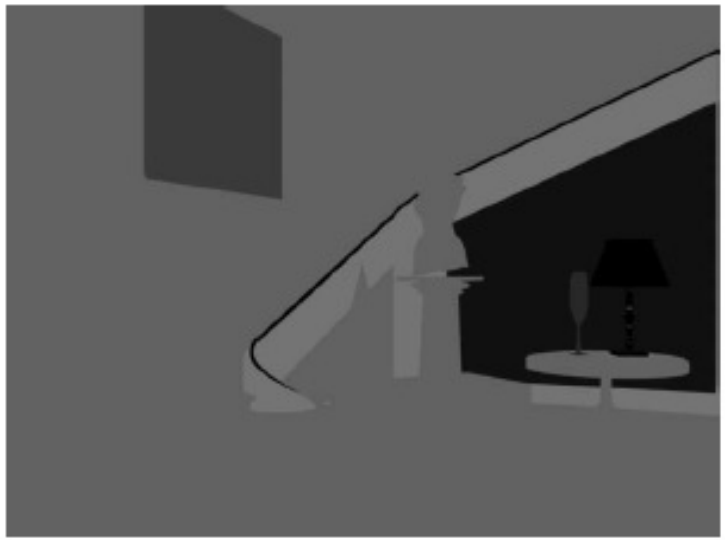
Difusa e non-directional lighting

Se I = intensidade da luz no ponto em estudo, Ia = intensidade da luz ambiente no ponto em estudo, ra= coeficiente de reflexão entre 0 e 1



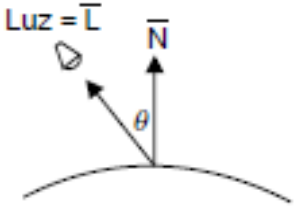
$$I = I_a \, r_a$$

Paralelepípedo e cena sob luz ambiente



-

Luz pontual

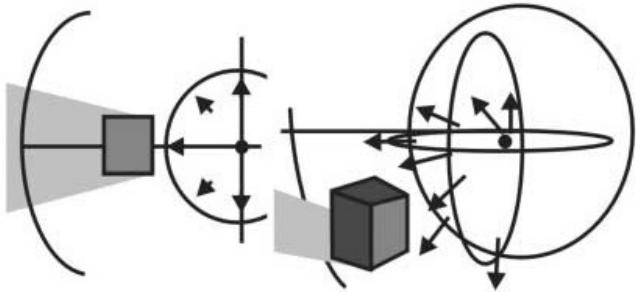


A luz é distribuída a partir de um ponto igualmente em todas as direções.

$$I = I_d\, r_d \cos\theta$$

Pode deixar as superfícies e as sombras com os limites muito intensos

$$I = I_a\, r_a + I_d\, r_d \cos\theta$$



Boa aproximação quando:

1) A fonte está suficientemente distante da cena.

2) A fonte tem dimensões pequenas comparadas aos demais objetos.

Caso o ângulo varia ponto a ponto:

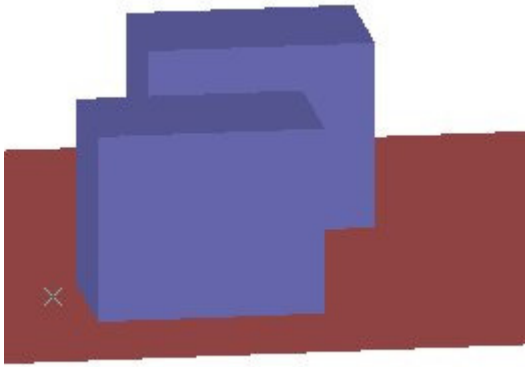
$$I = I_d r_d (\overline{u_e}.\overline{u_n})$$

Luz direcional

A direção da iluminação é considerada, mas áreas mais distantes e mais próximas com mesmo ângulo em relação a luz são iluminadas igualmente.

Isso não é muito realístico pois se espera que áreas mais distantes da luz fiquem mais escuras!

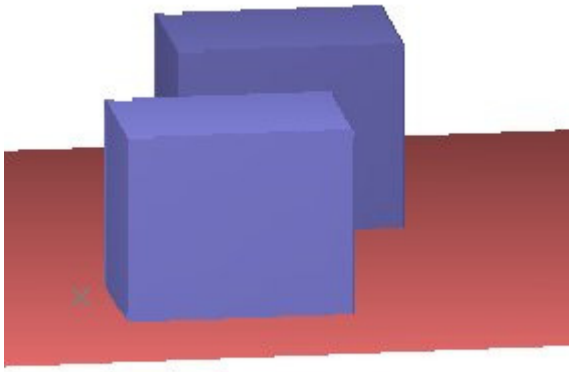
A atenuação com a distância pode ser de diversas maneiras: linear, quadrática, com fator de atenuação, associada ou não a constantes, etc.



$$I = I_a . r_a + I_d r_d \cos\theta / (d+k)$$

$$I = I_a . r_a + I_d r_d \cos\theta / (d+k)^2$$

$$I = I_a . r_a + f_{at} I_d r_d \cos\theta$$



Superfícies dos objetos da cena

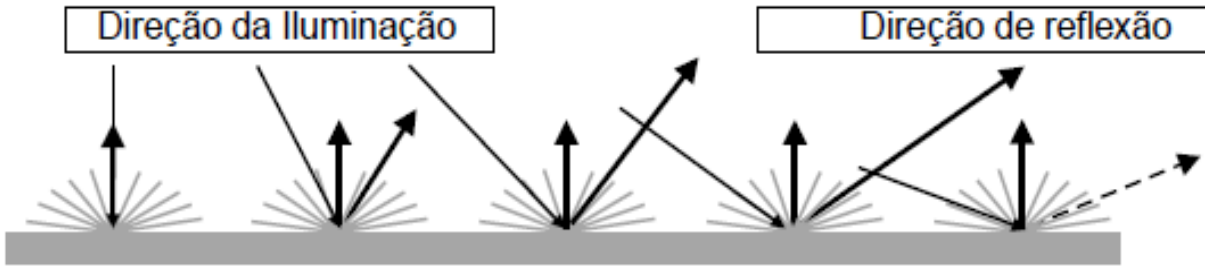
Forma como a luz é refletida pela superfícies:

- Reflexão difusa (superfícies foscas, sem lustro ou brilho) – aparece com mesma intensidade em todas as direções (dull, matte)
- Reflexão especular – tem um efeito de ter algum brilho , como se fosse de metal, ou encerada.

As superfícies dos objetos da cena

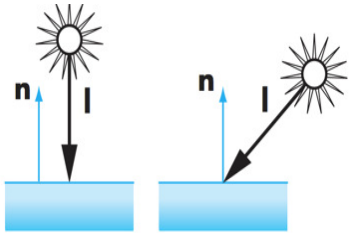
Modelo **Lambertiano** ou de objeto foscos

A intensidade da luz é independente do posição do visualizador da cena. Iluminação isotrópica. A intensidade luminosa obedece a **lei dos cossenos de Lambert**



$$I_d = k_d(\mathbf{l} \cdot \mathbf{n})L_d.$$

$$\cos\theta = \mathbf{l} \cdot \mathbf{n}.$$



Johann Heinrich Lambert (1728-1777)

Físico, matemático, astrônomo e filosofo suíço Foi um dos criadores da fotometria e autor de trabalhos inovadores sobre as geometrias não euclidianas. Provou que o *Pi*=π é irracional.

Em 1760, ele publicou o livro : Photometria. Considerando que a **luz viaja em linha reta**, mostrou que a iluminação é proporcional à intensidade da fonte, **inversamente proporcional ao quadrado da distância** da superfície iluminada e ao **ângulo de inclinação da direção da luz** com a superfície. Estes resultados foram apoiados por experiências.

Em Photometria Lambert também formulou **a lei da absorção da luz** e introduziu o termo albedo. A **unidade** fotométrica **Lambert** é em reconhecimento ao seu trabalho.

Lambert também foi pioneiro no desenvolvimento de **modelos de cores tridimensionais** combinando pigmentos vermelhos, amarelos e azuis, e com branco.

Função das cores dos objetos

Dependendo da forma de representação se usam as expressões anteriores separadamente para cada canal RGB, ou HSV, ou seja considerado I=cada canal da imagem:

$$I_I = I_{aI} \cdot r_{aI} + f_{at}\, I_{dI}\, (r_{dI} \cos\theta) + f_{at}\, I_d\, r_s \cos^n \alpha$$

Reflexão especular

Em algumas superfícies funcionam como "espelhos" onde os fótons não interagem com os pigmentos, refletindo toda a cor original que nelas chega.

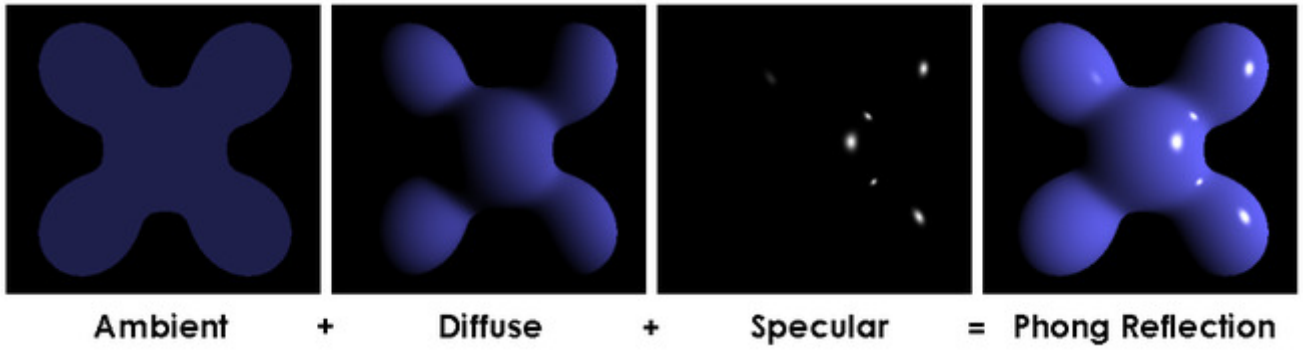
Modelos:

- Cook-Torrance
- Phong

Phong reflection model

Empírico e local.

Combina luz difusa (superfícies foscas) e especular (superfícies com brilho)



Luz branca e objeto azul

Bui Tuong Phong (1942- 1975)

Vietnamita, nascido em Hanoi, (Bui é o sobrenome e Phong seu nome, no Vietnam como o Brasil se considera muito o primiero nome). Se formou como engenheiro em Toulouse, e entrou para o IRIA (*Institut de Recherche en Informatique et en Automatique*) em 1968. Ph.D. na University of Utah em 1973. Professor da Universidade de Stanford até morrer de leucemia.

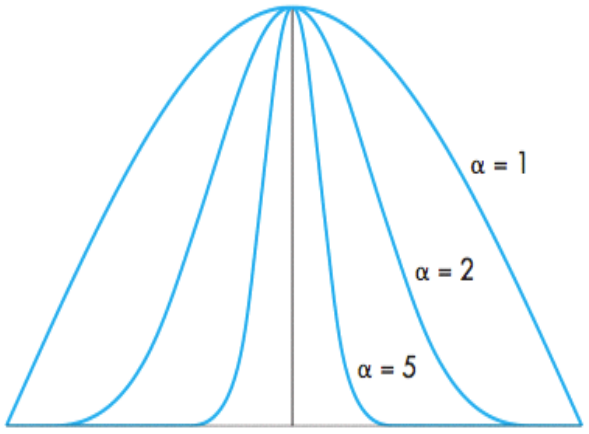
No modelo de Phong a intensidade luminosa é proporcional ao ângulo entre o observador e a direção de reflexão

Usando parâmetros R entre zero e um:

$$I = I_a + I_d + I_s = L_a R_a + L_d R_d + L_s R_s$$

$$I_s = k_s L_s \cos^{\alpha} \phi .$$

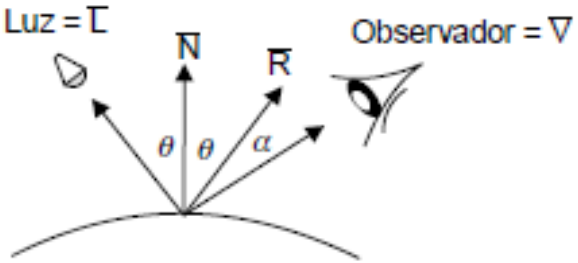
$$\cos \phi = r \cdot v$$



Phong reflection model

Aparece mais um ângulo na expressão!

$$I = I_a . r_a + f_{at} I_d (r_d \cos\theta + r_s \cos^n \alpha)$$



$$I = I_a . r_a + f_{at} I_d (r_d (u_e . u_n) + r_s (u_r . u_v)^n)$$

Múltiplas fonte:

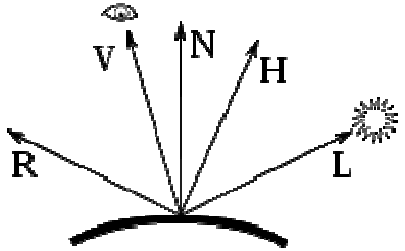
$$I = I_a . r_a + \sum_{j=1}^{j} I_{dj} (r_d (u_{ej} . u_n) + r_s (u_{rj} . u_v)^n) / (d_j + k_j)$$

ou

$$I = I_a . r_a + \sum_{j=1}^{j} f_{atj} I_{dj} (r_a (u_{ej} . u_n) + r_s (u_{rj} . u_v)^n)$$

Outros:

ângulo entre o observador e a direção de reflexão é substituído pelo metade do ângulo que a luz refletida faz com a normal



Uma alternativa ao modelo de luz especular de Phong é o uso do vetor de *intensidade de luz especular máxima* ou *vetor de caminho médio*, H, esse vetor é definido usando a direção da fonte de luz, L, e de visualização, V, como:

$$\overline{H} = \frac{\overline{L} + \overline{V}}{\left|\overline{L} + \overline{V}\right|} \text{ ou } \overline{H} = (\overline{L} + \overline{V})/2 \text{ se } |\overline{L}| = |\overline{H}| = 1$$

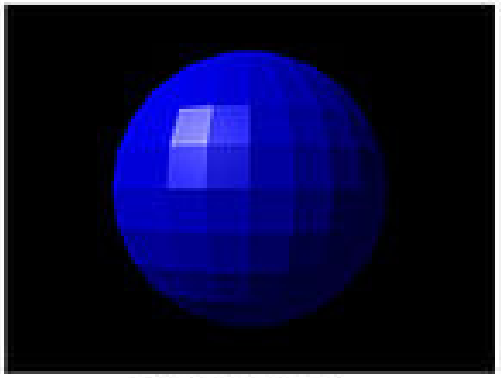
Flat shading

Produz bons resultados apenas se o objeto for mesmo de faces planas.

Cada polígono que compõem o objeto tem seu tom baseado no ângulo de sua normal com a direção da luz, sua cor e a cor da fonte de luz.

De modo que toda a face tem um tom constante.

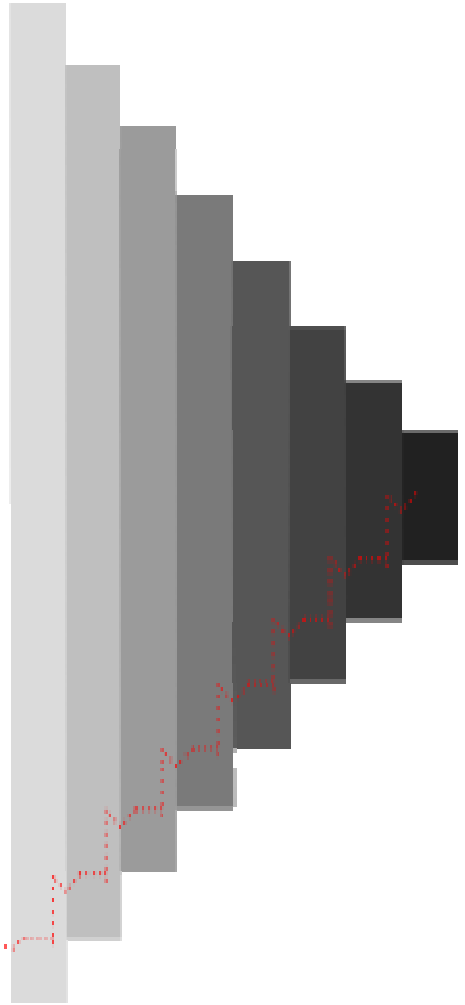
Efeito de bandas de Mach



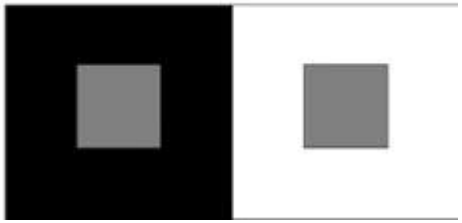
FLAT SHADING

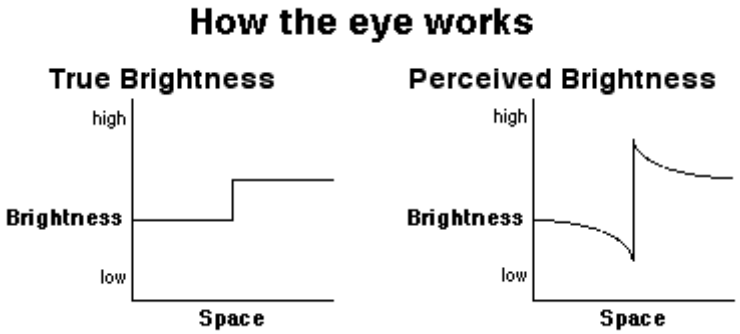
As **bandas de Mach**

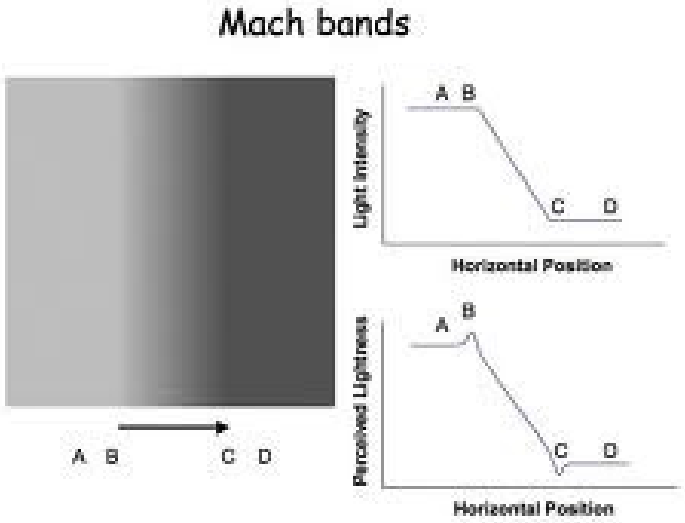
são efeitos de intensificação de contraste observados pelo olho humano: diferente gradiente de luminosidade tem sua fronteira com contraste amplificado. O nome desta ilusão é uma referência a Ernst Mach.

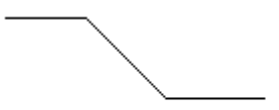


Bandas de Mach, intensidades constantes ou variações de intensidade constantes





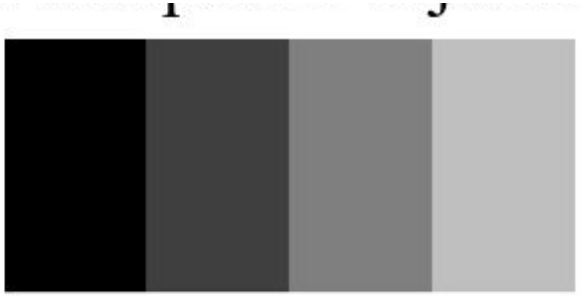




Actual Luminance Profile

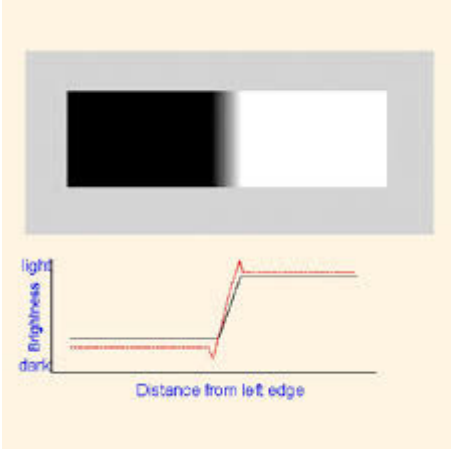
Perceived Luminance Profile

Mach Bands:
Perceived changes in luminance near the edges of a luminance gradient.



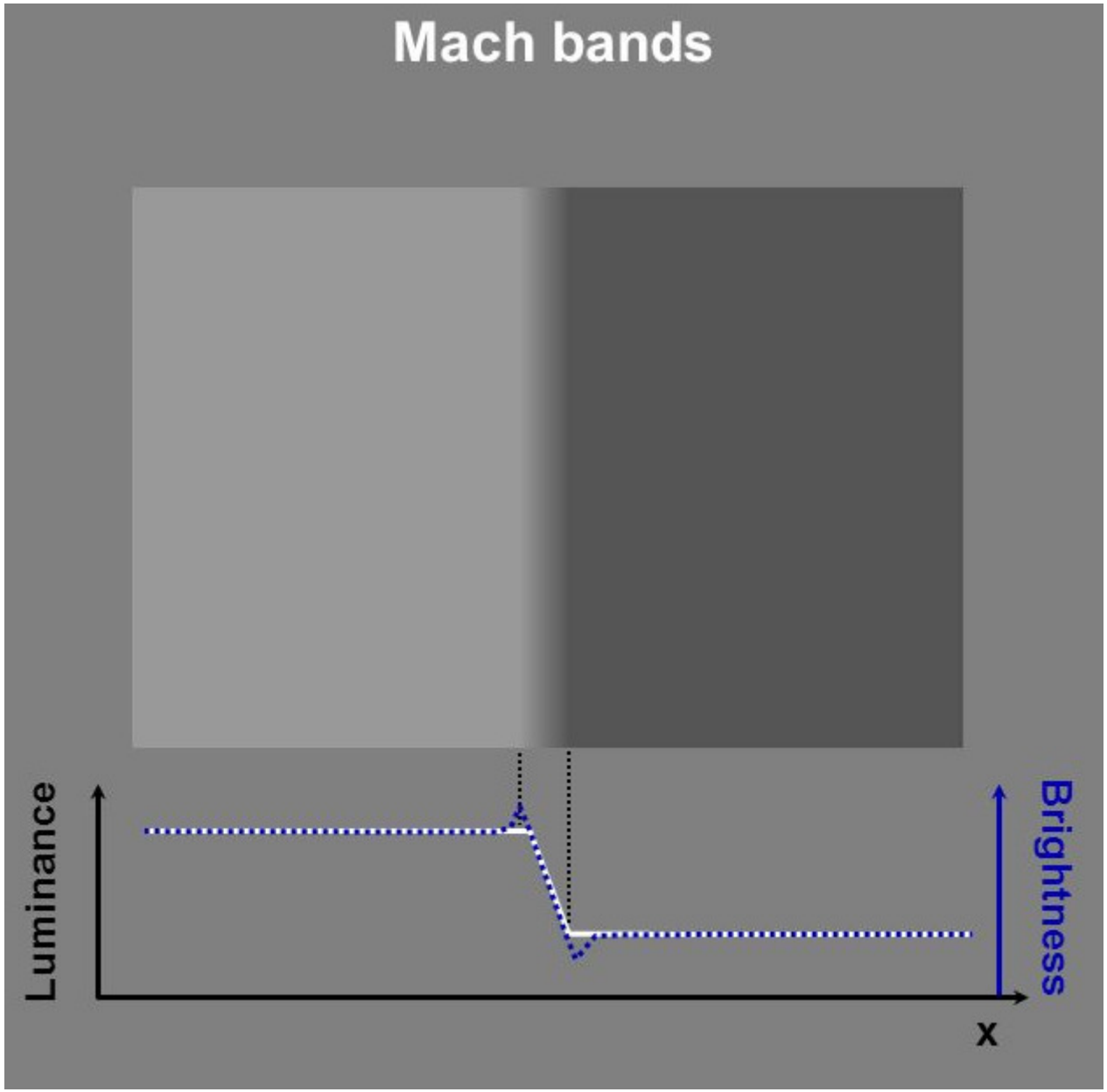
signal original

signal avec Mach bands



Bandas de Mach, intensidades constantes ou variações de intensidade constantes

Bandas de Mach, intensidades constantes ou variações de intensidade constantes



Ernst Waldfried Josef Wenzel Mach

(Brno, 1838 — Vaterstetten, 1916) foi um físico e filósofo austríaco. austríaco.2

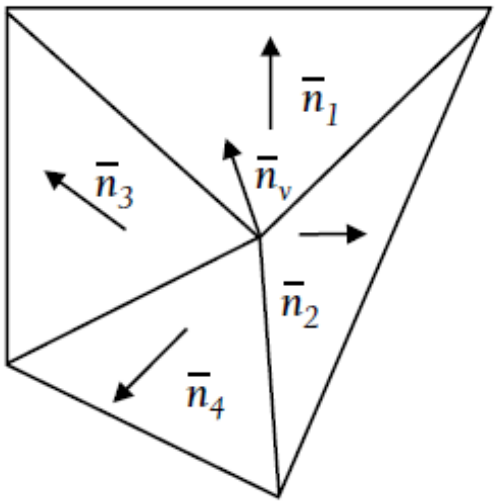
Foi professor de matemática em Graz.

Depois de física em Praga, quando opôs-se à introdução da língua tcheca como idioma oficial na mesma universidade, alinhando-se entre os partidários da dominação alemã na região.

Smooth shading

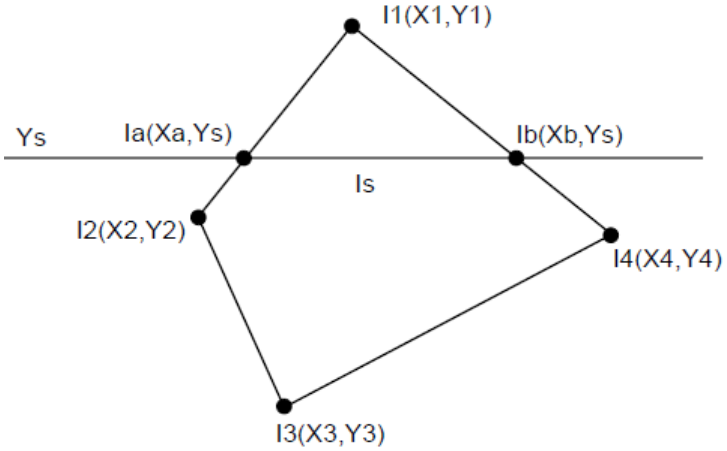
O sombreamento varia de pixel para pixel:

Gouraud shading – calcula a iluminação dos vértices e faz interpolação linear no interior. Supõem a normal nos vértices como média das normais das faces que chegam ao vértice.



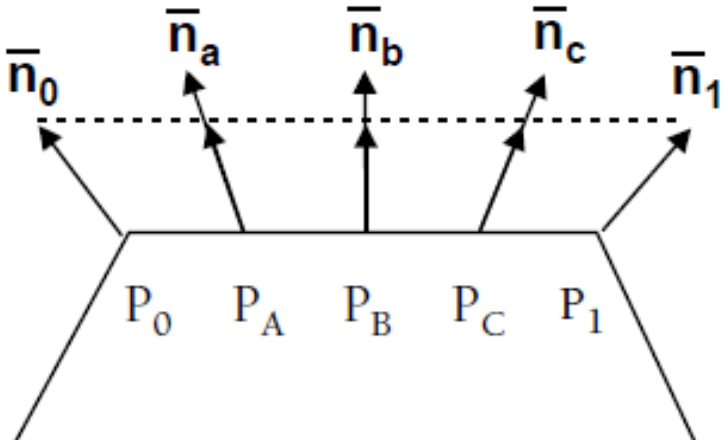
$$\overline{n_i} = \frac{n_i}{|n_x| + |n_y| + |n_z|}$$

$$n_v = \sum_{i=1}^{k} \frac{\overline{n_i}}{k}$$



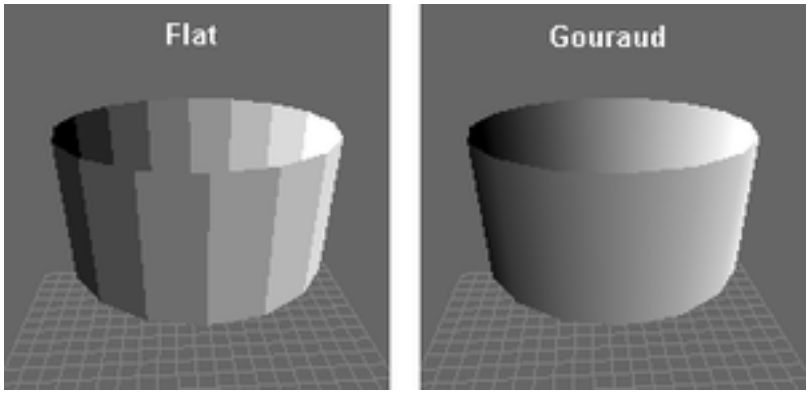
Smooth shading

Phong shading – faz a interpolação das normais dos polígonos já rasterizados para daí calcular o tom do ponto da superfície



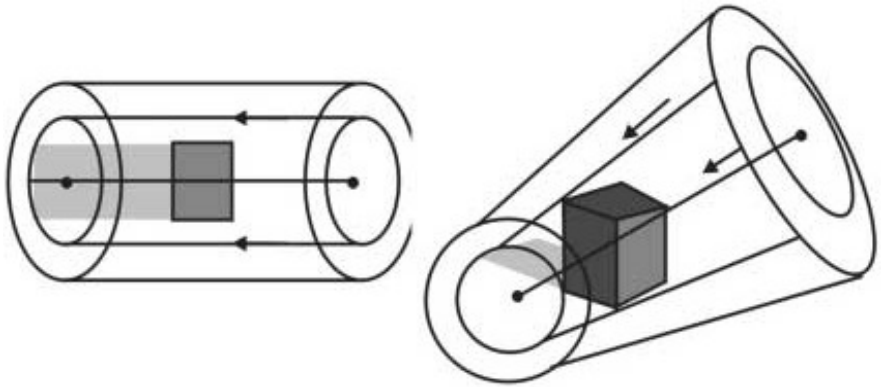
Henri Gouraud (1944- ...)

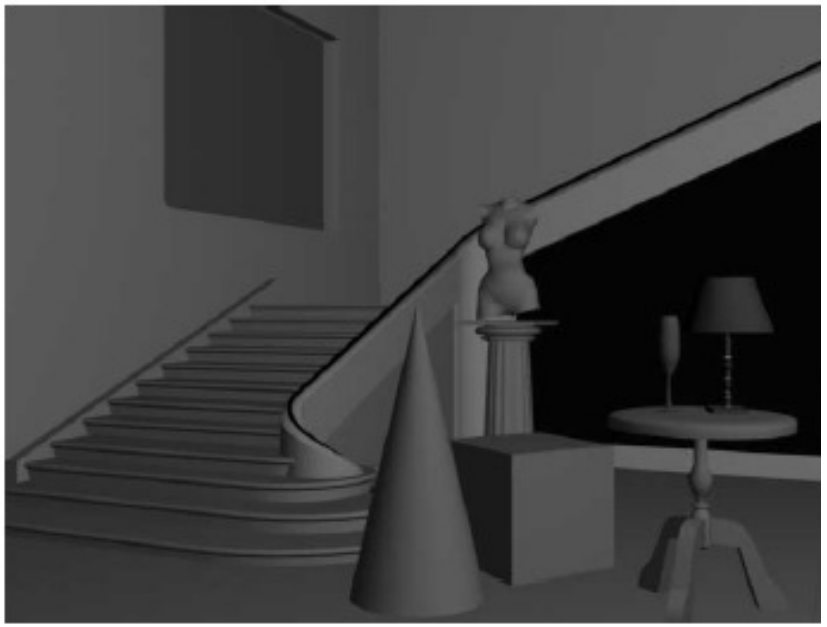
Frances, estudo de 1964–1967 na Escole Central de Paris, em 1971 recebeu seu Ph.D. pela Universidade de Utah pelo trabalho de titulo: **Computer display of curved surfaces**



Luz direcional

A direção da iluminação é considerada, mas áreas mais distantes e mais próximas com mesmo ângulo em relação a luz são iluminadas igualmente



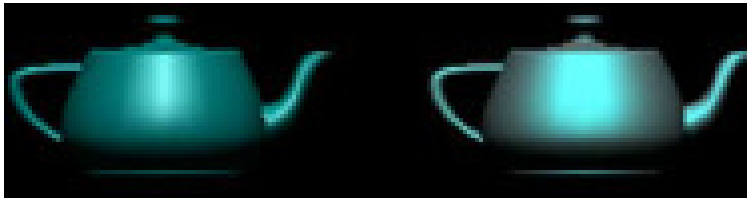


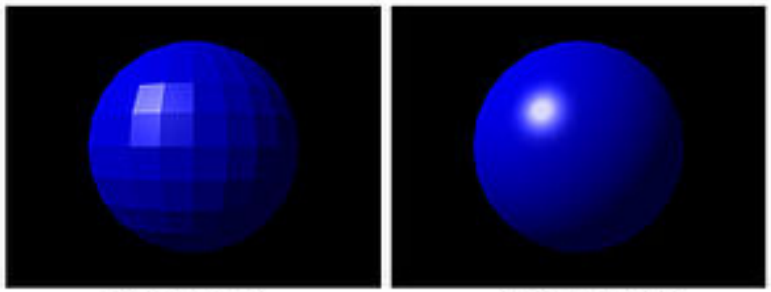
Tratamento de Iluminação especular

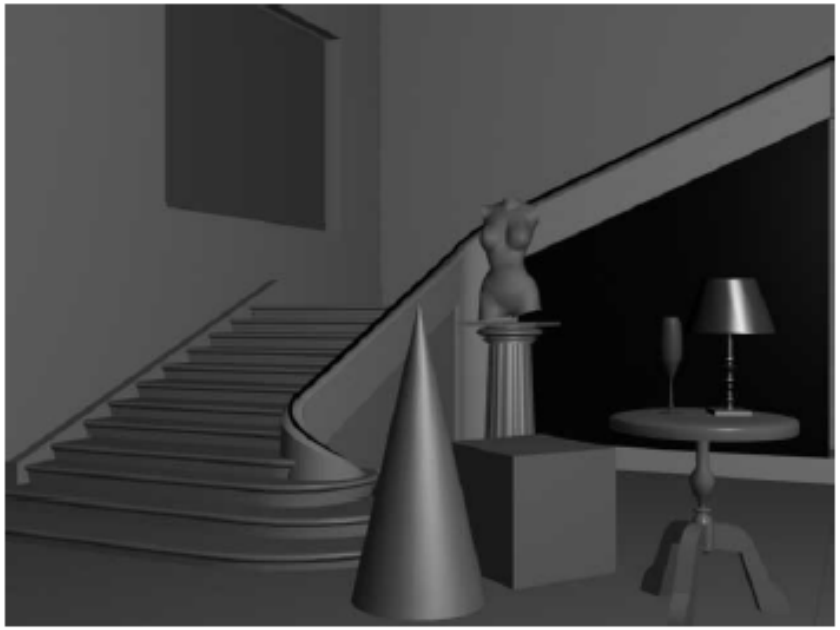
Shininess Coefficient

Ou coeficiente de brilho da luz especular:
Metais entre 100 e 200
Plástico entre 5 e 10

Única diferença é o coeficiente
de brilho da luz especular







Texturas: **Texture mapping**

Texturas: Texture mapping – permite dar a uma face plana um aparência bem complexa!

Edwin Catmull em 1974, em sua tese de doutorado, foi o primeiro a adicionar detalhes de textura na superfície de modelos 3D por mapeamentos

Fazer um **texture map** em uma superfície é como aplicar uma **folha de papel auto adesivo nela "contact" para lhe dar um aspecto** semelhante **a textura do** desenho deste papel !

Edwin Earl "Ed" Catmull (1945, ...)

Americano formado em **ciência da computação e atualmente presidente da Pixar e Disney Animation.**

Tem feito diversas contribuições a CG.

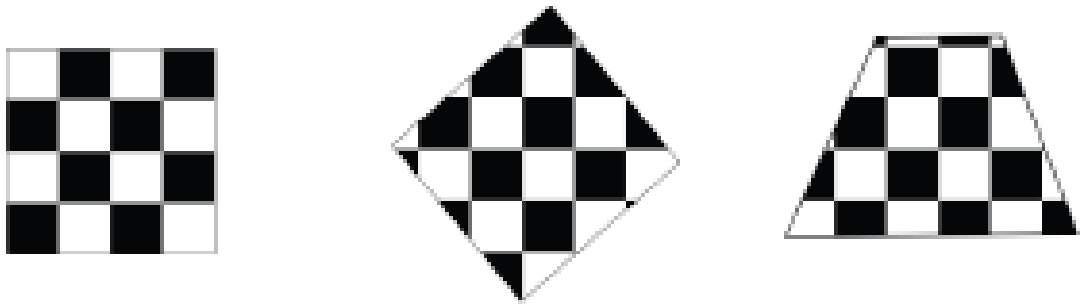
Em 2001, recebeu um **Oscar** "for significant advancements to the field of motion picture rendering".

Em 2006, foi premiado com a IEEE von Neumann medal pelas suas contribuições na ***modelagem em CG, animação e rendering***.

Texture mapping pode ser:

Paramétrico ou não-Paramétrico:

Pode ser fixo (em tamanho ou orientação) ou se deformar com o objeto

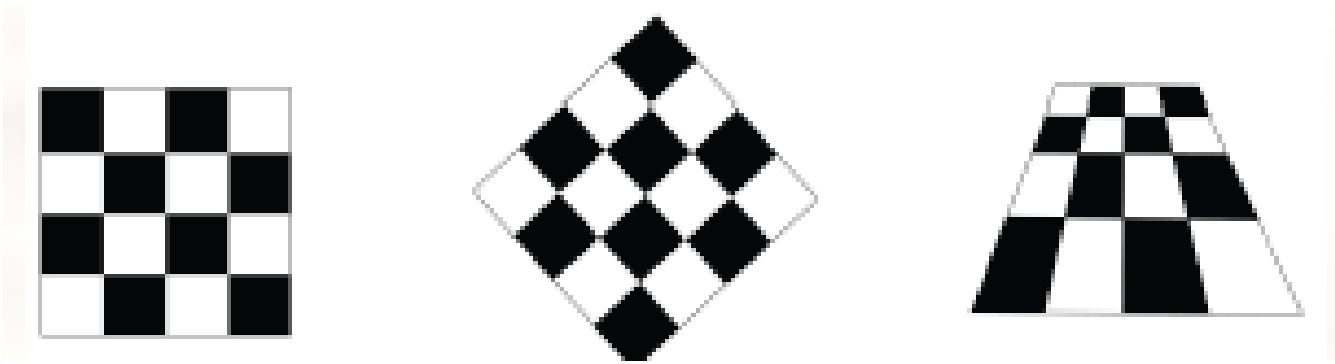


Não-Paramétrico: parece que se usou FORMA COMO CORTADOR SOBRE UM TECIDO OU MATERIAL

Paramétrico

se deforma com o objeto

Parece que A TEXTURA É do material DO OBJETO!
TEM A MESMA deformação QUE ELE NA CENA OU NA ANIMAÇÃO!

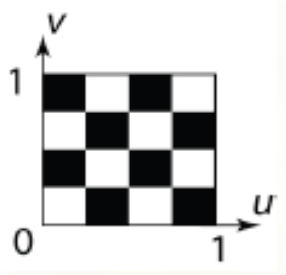


Aplicar um **texture map** é

Associar às coordenadas da face , as coordenadas do mapa de textura 2D: chamadas coordenadas UV.

ESPAÇO DE TEXTURA (u,v)

As texturas são definidas em um sistema cartesiano normalizado [0,1]x[0,1]



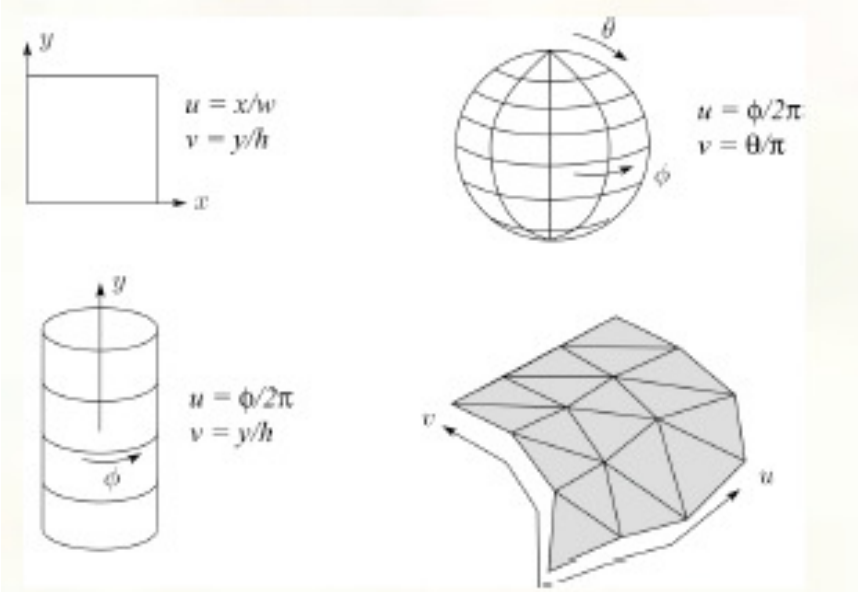
E depois usadas para

"encapar" nossos objetos,

De modo que quando

A superfície de mover

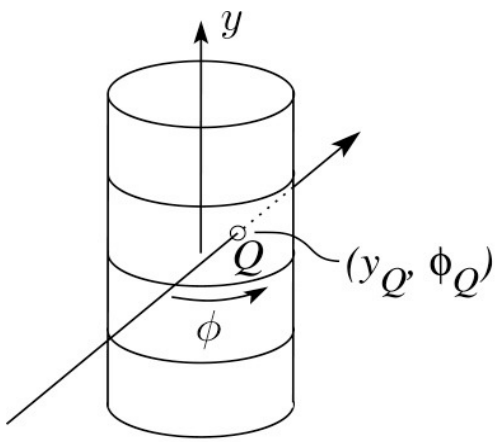
ela vá junto!



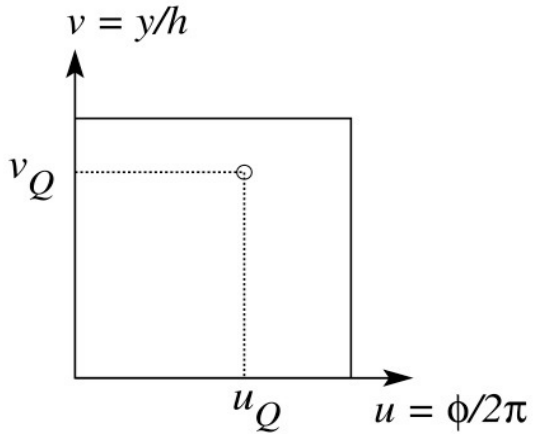
O mapa de textura é uma imagem

Essa imagem deve ser convertida para [0,1]x[0,1] e depois para as coordenadas onde será mapeada

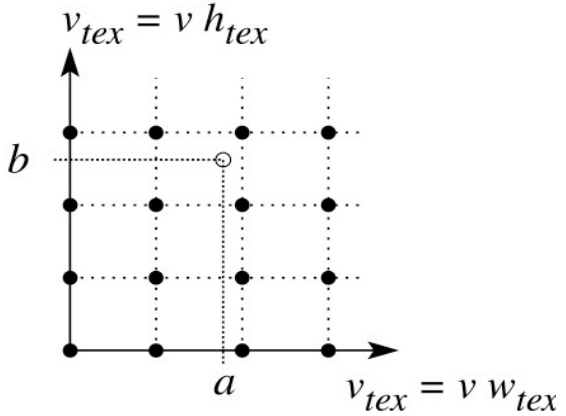
(u_{tex},v_{tex}) entre os valores: ([0.. w_{tex}], [0.. ht_{ex}])



Ray intersection

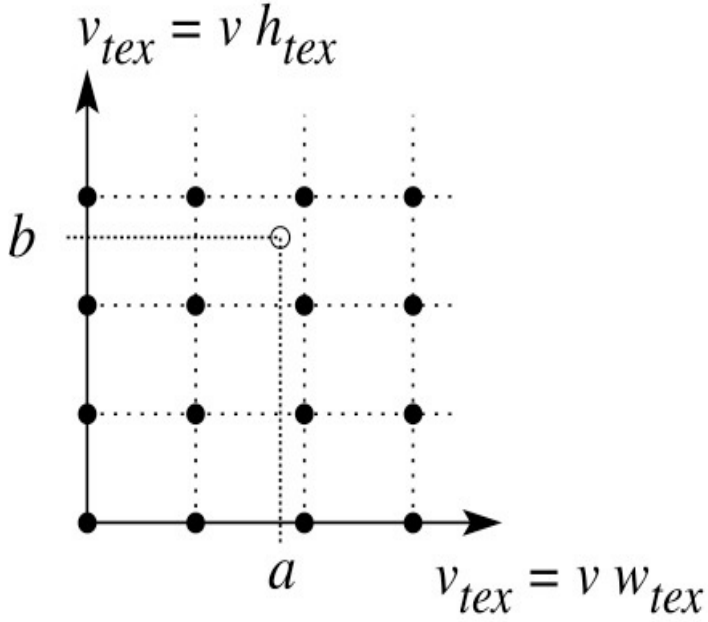


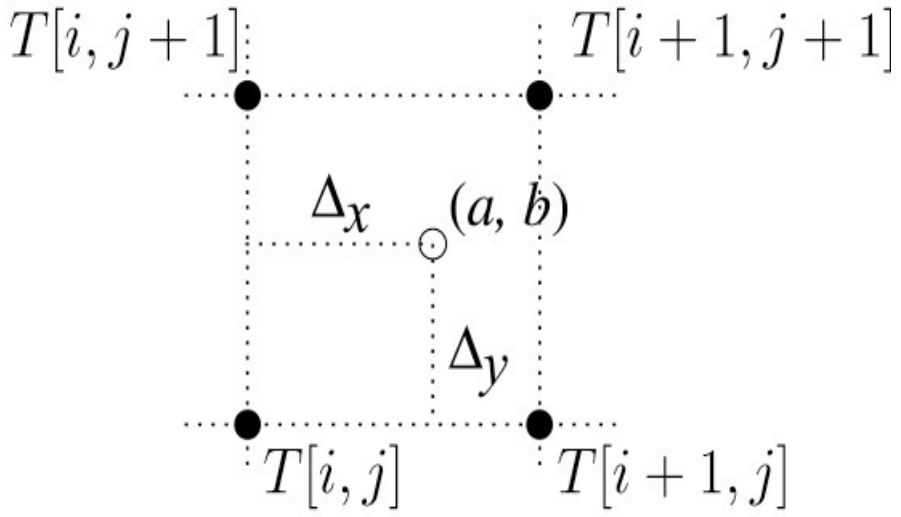
Mapping to
abstract texture coords



Mapping to
texture pixel coords

Reamostrando pelo uso de interpolação bilinear quando necessário





$T(a,b) = T[i + \Delta x, j + \Delta y]$
$= (1- \Delta x)(1- \Delta y)\ T[i, j]+ \Delta x\ (1- \Delta y)\ T[i +1, j]$
$+(1- \Delta x)\ \Delta y\ T[i, j +1]+ \Delta x\ \Delta y\ T[i +1, j +1]$

Os mapas podem fazer mais que apenas mudar os tons

Podem mudar a **geometria da superfície** em que serão mapeados, por exemplo:

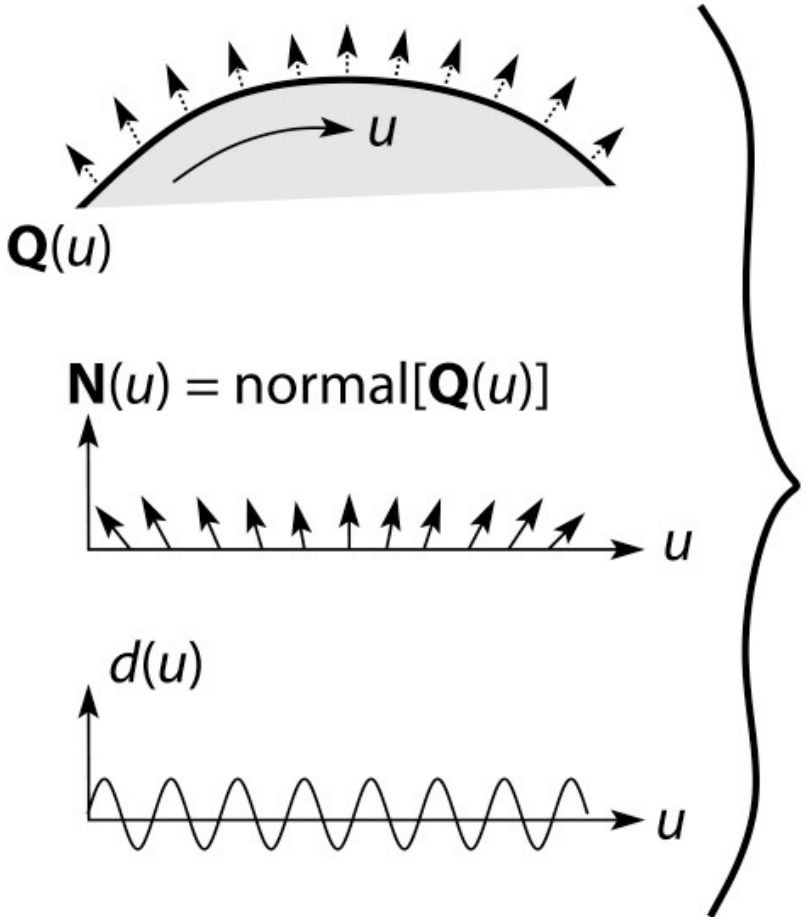
Se a superfície poder ser descrita como função de um para metro u: Q(u)

Sua normal, será geralmente também uma função: N(u)

Assim:

Displacement mapping (emboss)

Teremos assim uma nova superfície:

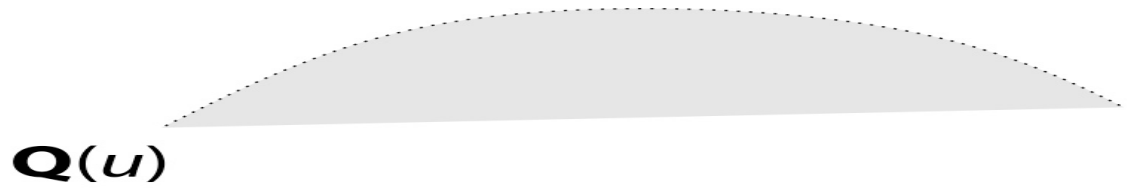


$$\tilde{\mathbf{Q}}(u) = \mathbf{Q}(u) + d(u)\mathbf{N}(u)$$

Bump map

Continuando com essa ideiam pode-se pensar em modificar a normal da superfície (depois de fazer os tratamentos de hiddens , apenas na hora de produzir seu shading).

$$\tilde{\mathbf{N}} = \text{normal}[\tilde{\mathbf{Q}}(u)]$$



Multitexturing ocorre quando mais de um mapeamento é aplicado na face ao mesmo tempo.

Alterações na imagem final ao ser adicionada

Da fase de rendending, como:

Textura para cor difusa

textura do bump map

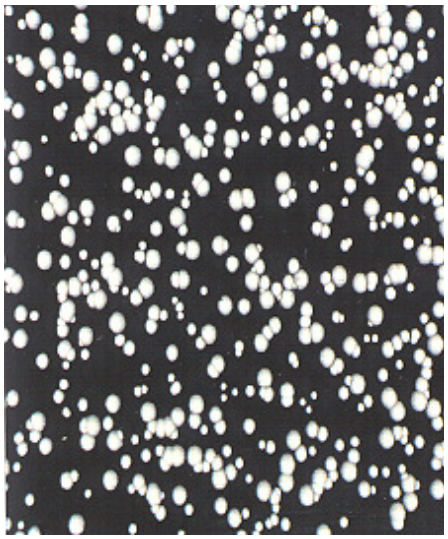
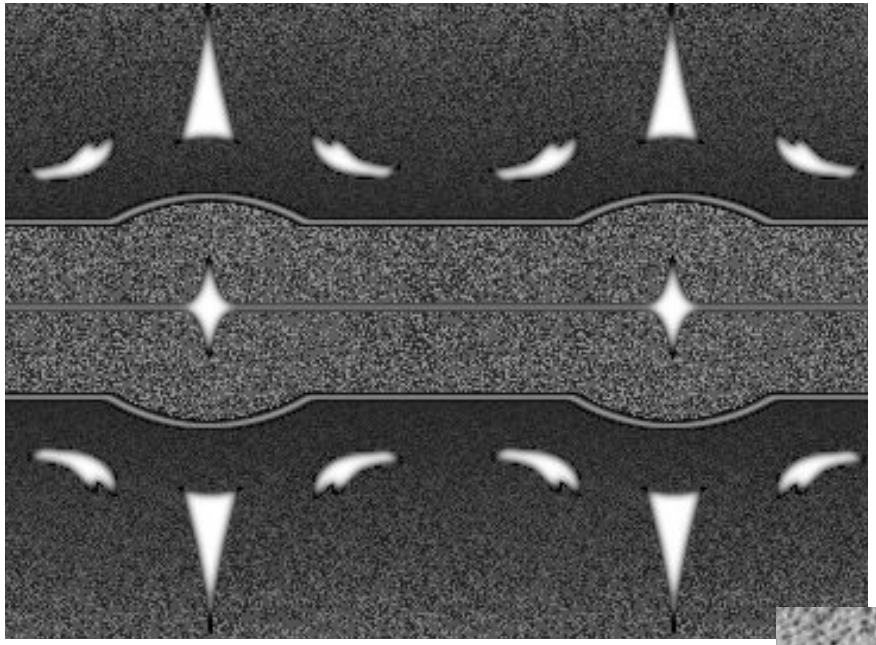


imagem final



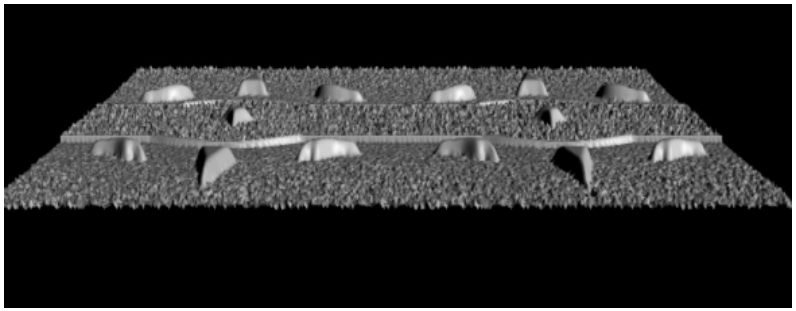
Mesma textura em fase plana

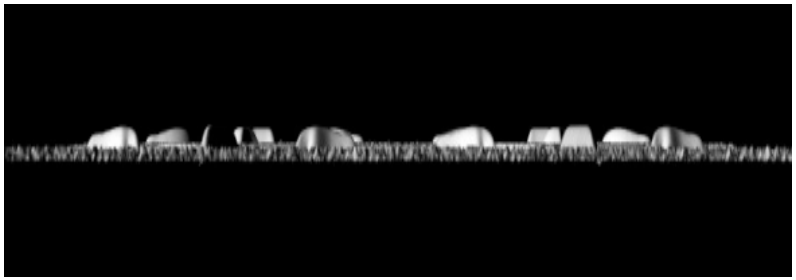
Como:

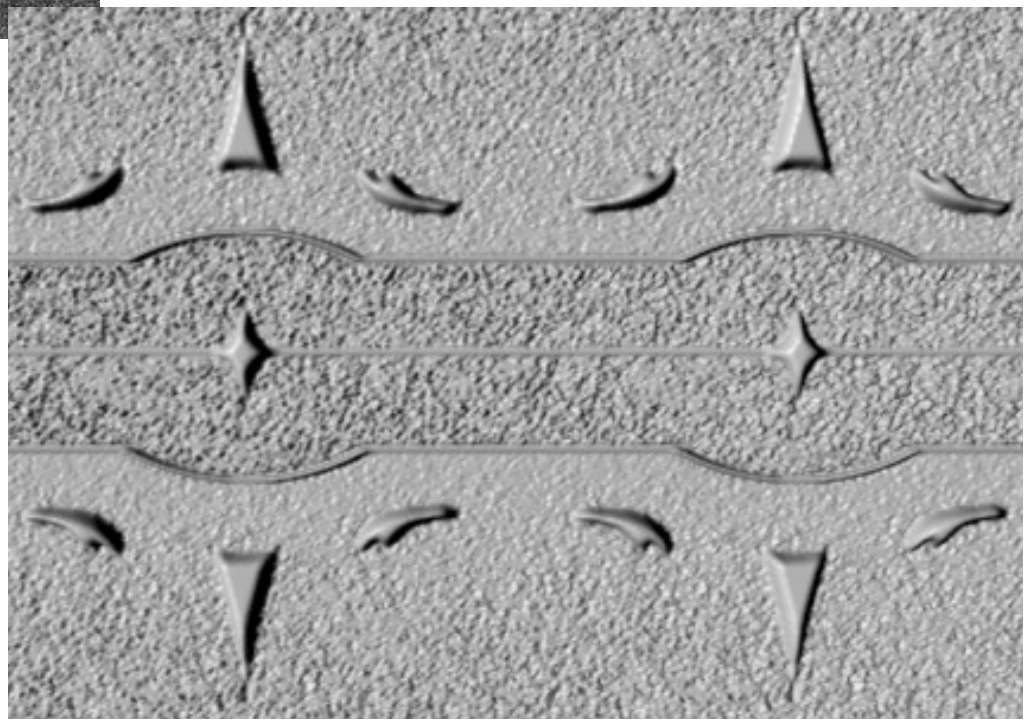
displacement map

+

bump map





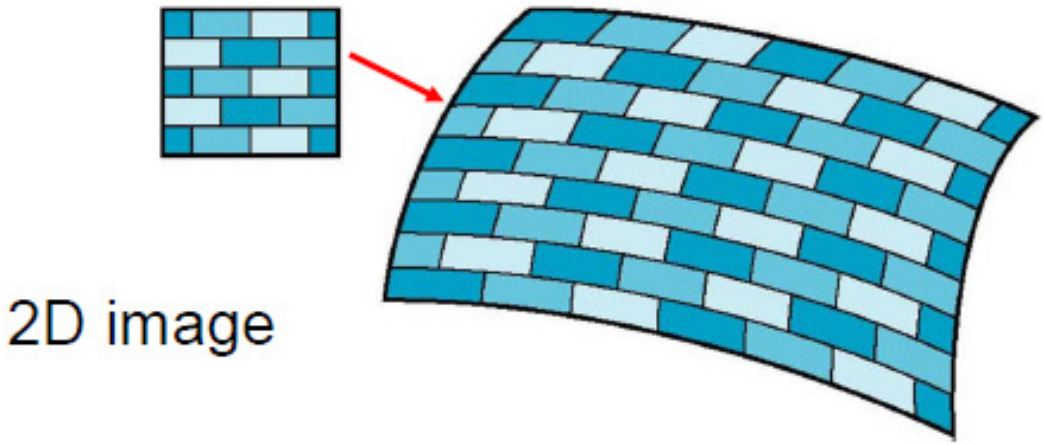


Sobre uma fase cilíndrica

Displacement map

Displacement map + bumpmap

Sistemas de coordenadas envolvidos:



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

Forward mapping

Forwards x backwards mapping

Mapeamento direto e inverso

Dado um ponto da tela (pixel) queremos saber a que ponto do objeto ele corresponde (inverso),

e

dado um ponto do objeto, queremos saber a que ponto da textura ele corresponde (direto)

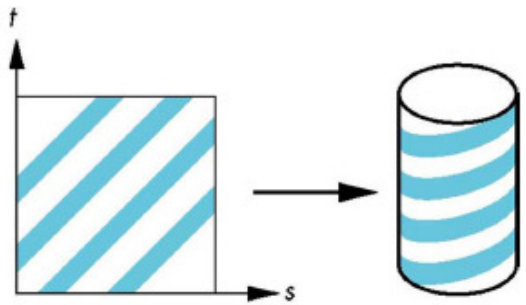
Mapeamento em 2 partes

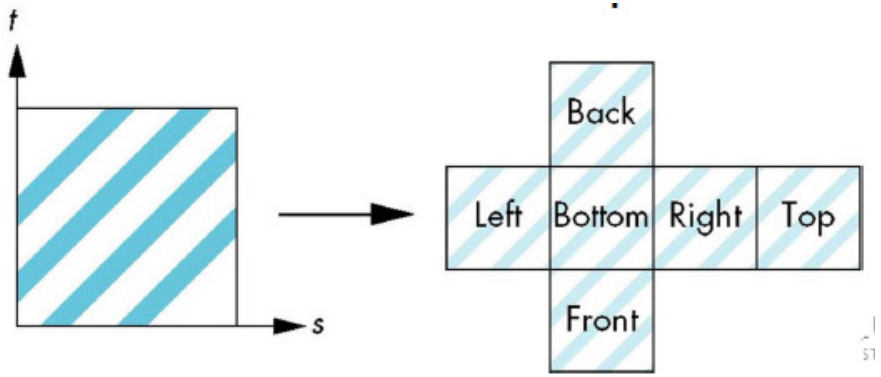
1- textura em uma forma intermediaria mais simples (cubo – projeções ortográficas, cilindro de altura h e raio r, esfera de raio r)

$$x = r \cos 2\pi u$$
$$y = r \sin 2\pi u$$
$$z = v/h$$

$$s = u$$
$$t = v$$

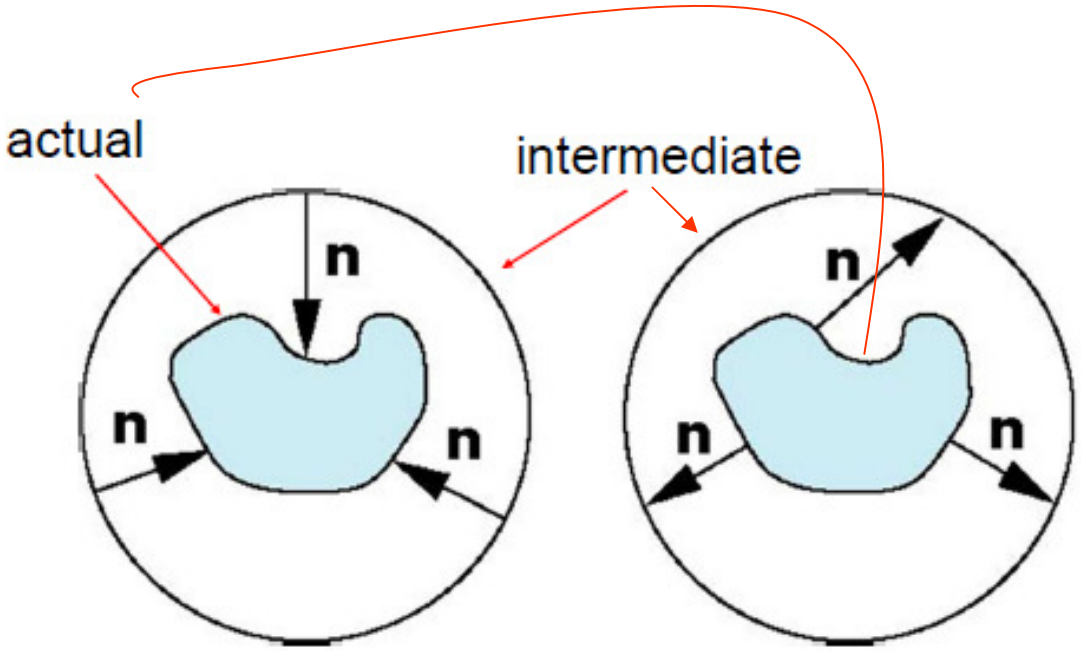
$$x = r \cos 2\pi u$$
$$y = r \sin 2\pi u \cos 2\pi v$$
$$z = r \sin 2\pi u \sin 2\pi v$$



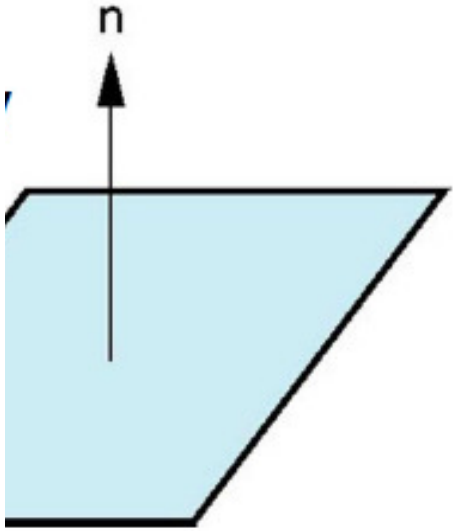


Mapeamento em 2 partes

2- da superfície mais simples para o objeto real, usando as normais (da intermediária para o objeto e do objeto para a intermediária)



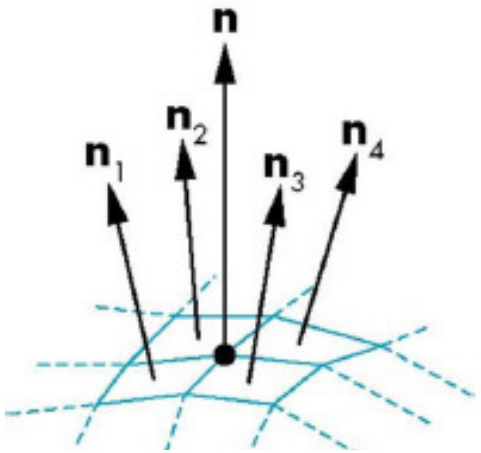
Normais



Superfícies planas:

Equação do plano:

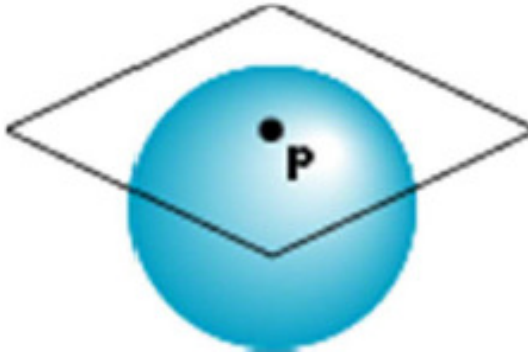
$ax+by+cz+d = 0$



Normal = (a,b,c)

Os por 3 pontos do plano: p0, p1, p2

$$\mathbf{n} = (p_2\text{-}p_0) \times (p_1\text{-}p_0)$$



Círculos: raio do ponto ao centro

Environment ou reflection map

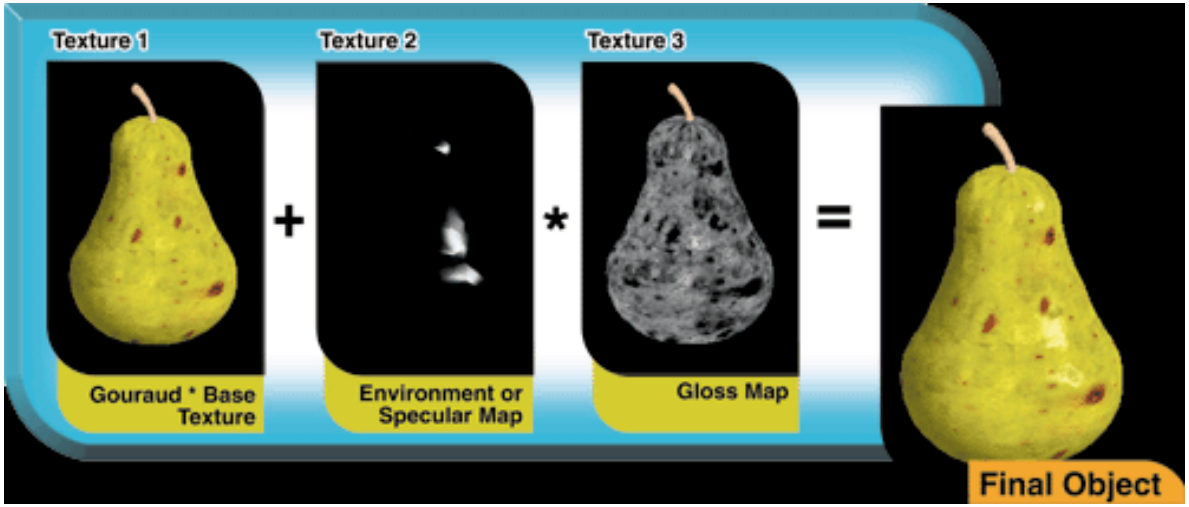
Usa para modelar o ambiente em uma superfície, como espelho.

Funciona bem com apenas um objeto na cena,

Dando uma ótima idéia de reflexão sem usar nenhum raio ou pode ser unido ao raytracing

Mapas podem ser combinados em diversos níveis

Para produzir um grau de realismo na cena de maneira simplificada



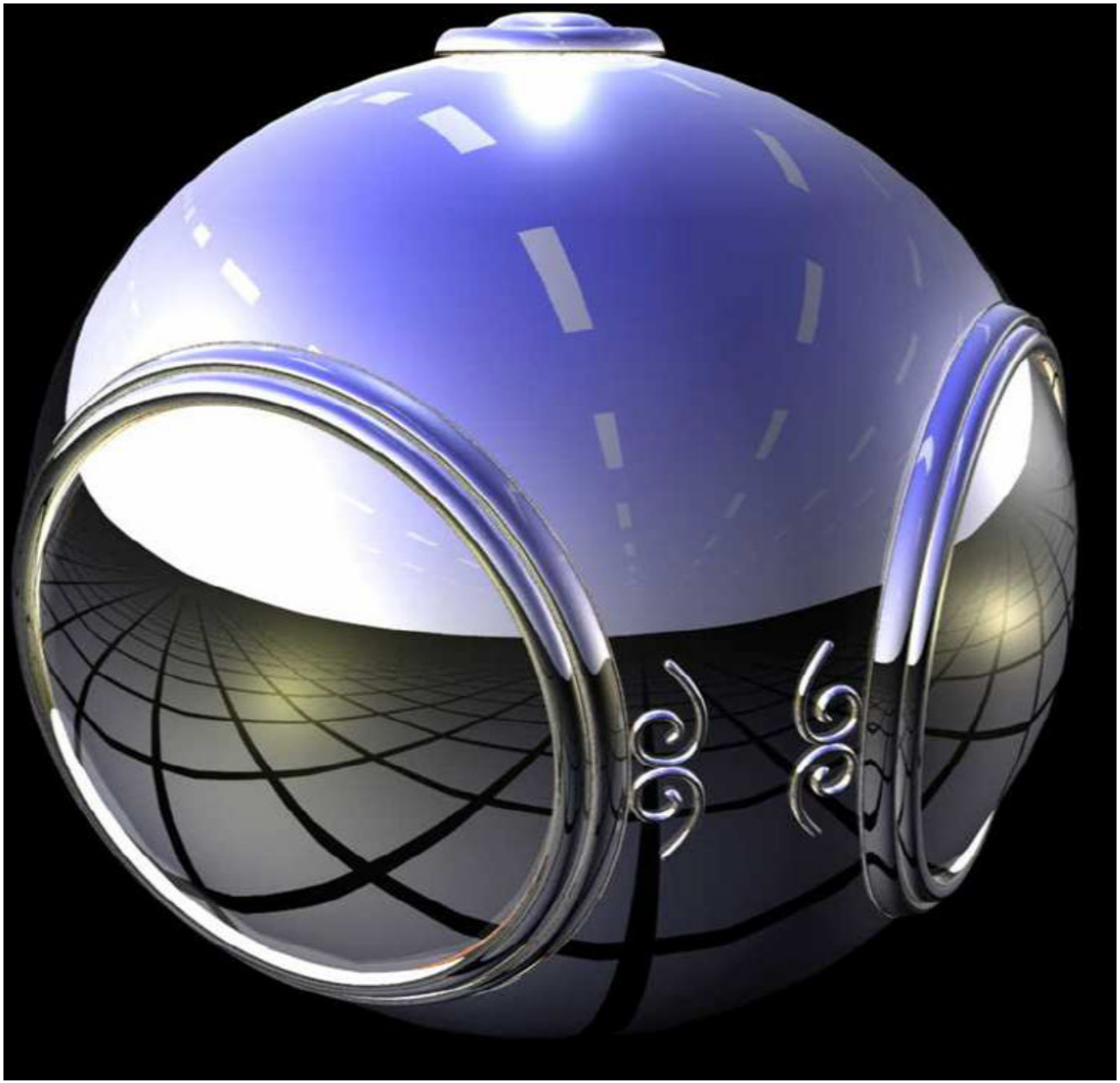
https://noppa.oulu.fi/noppa/kurssi/521493s/luennot/521493S_3-d_graphics_vi.pdf

Shading

Modelo Phong

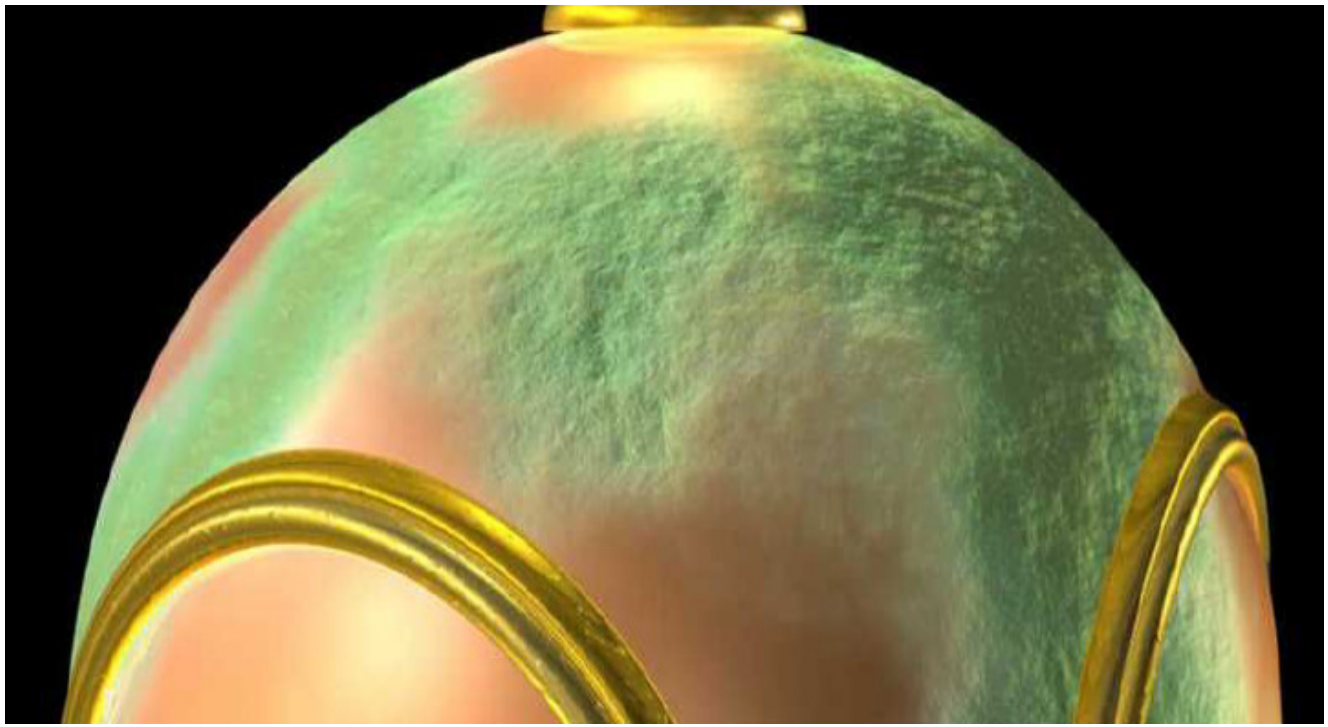
Texture map

Environment Mapping



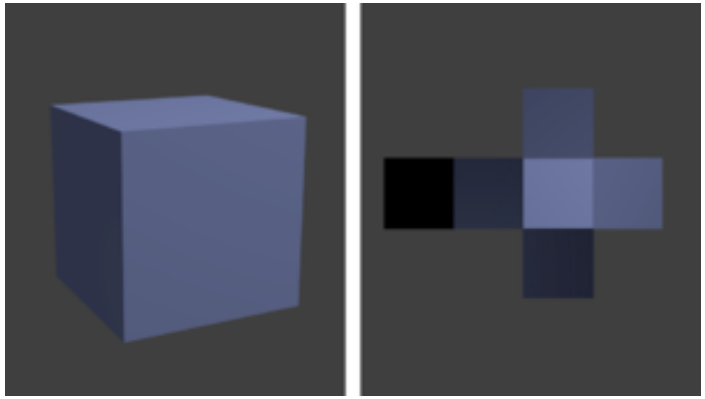
detalhes

Bump Mapping



Lightmap

Mapeamento que contém a intensidade luminosa das faces. Útil em objetos que permanecem estáticos em games. Geralmente flat , sem incluir a idéia da direção da iluminação . Presentes na maioria dos plug-ins 3D



http://www.computerhistory.org/

The Utah Teapot

Criado por Martin Newell na University of Utah em 1975.

Tem sido usado como modelo 3D por 40 anos para verificar modelos de iluminação , cor, realismo, etc.

Qual o nome da estrutura de dados logo do Blender?

(Ton Roosendaal 2002) Suzanne?

Procedural texture generation method

As texturas podem ser geradas por programação (procedures) e não apenas por captura de texturas já existentes

Geradores de padrões fractais são muitos úteis para isso!

Level of details (mip maps)

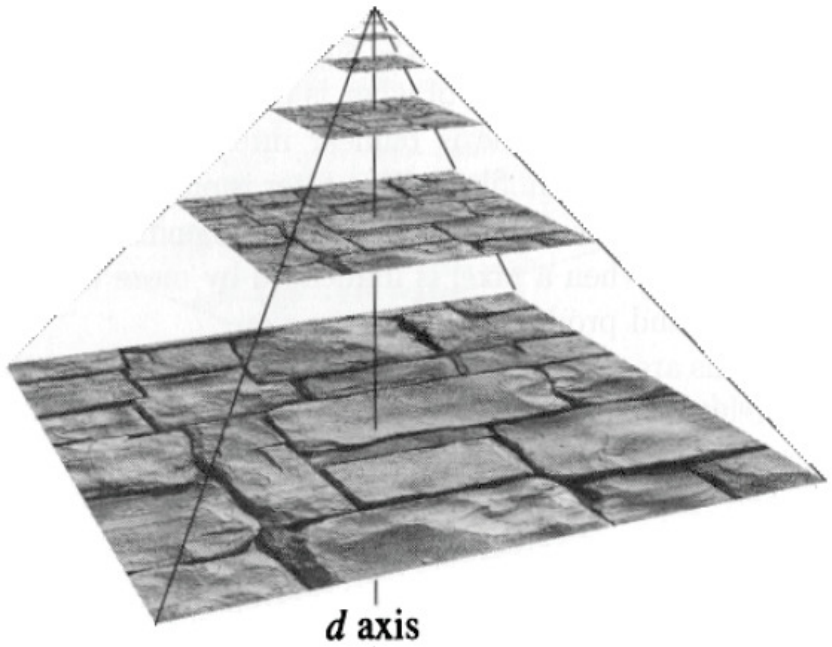
Alterado detalhes da textura com a distância ao observador

Também pode ser simulado com filtros

Que diminuem a resolução

"MIP" acronym of the phrase *multum in parvo* =

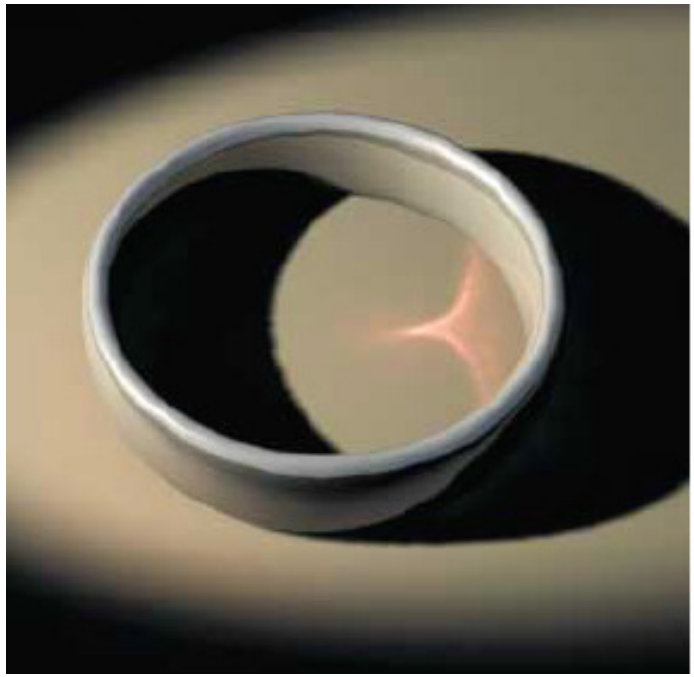
"much in little"

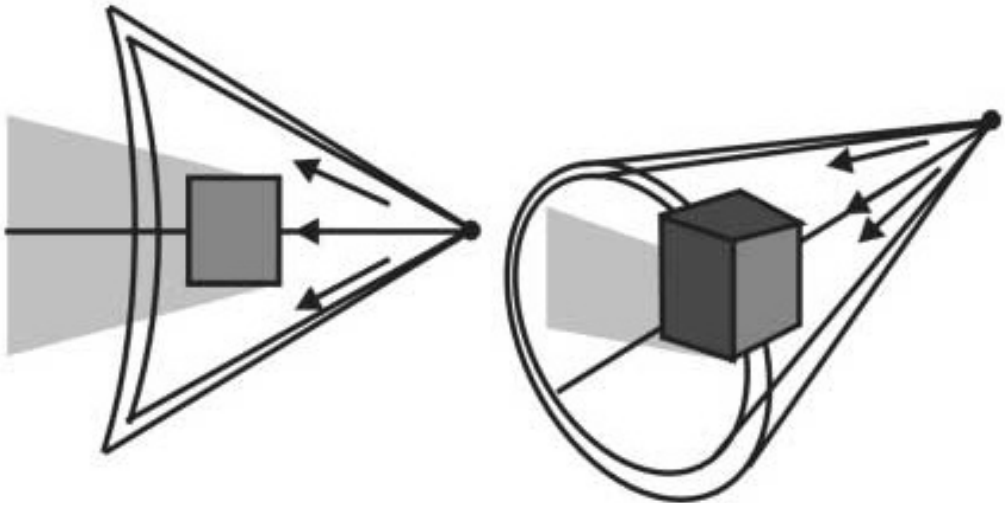


Sombras, refração, reflexão e efeitos específicos

Sombras,

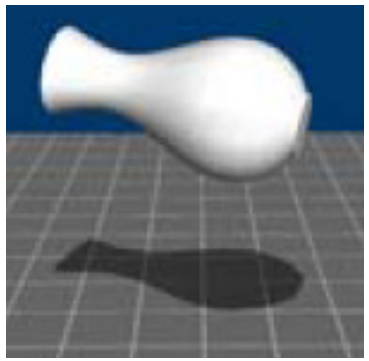
podem ser consideradas por diversos métodos, de simples projeções , passando por texturas até os métodos globais (seção 7.3.6 do livro texto tem boa revisão do assunto) !





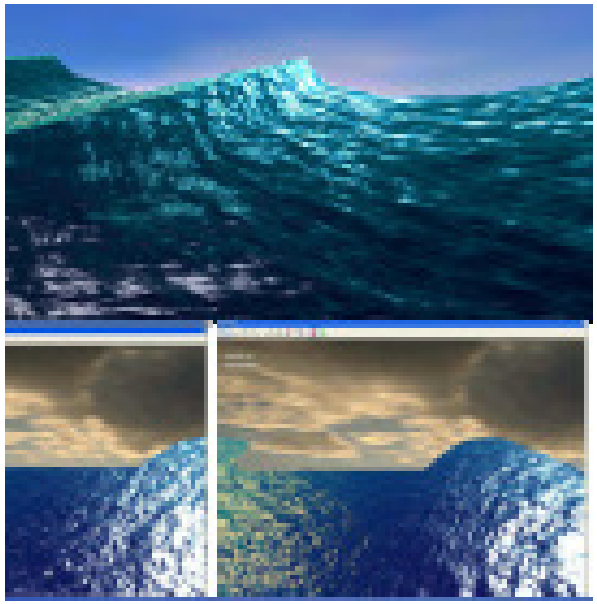
Sobras planas e projetadas:

Sombra=umbra e penumbra



Cautics

São padrões de luz (refletidas e refratadas) que parecem concentrar a luz em alguns pontos. Ocorrem em vidros, água, modelagens de ondas, piscinas e outras situações que concentramos raios luminosos



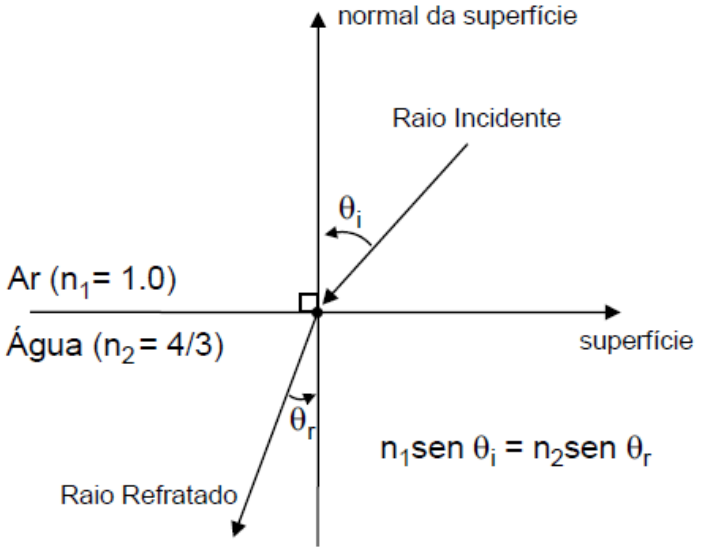
Refração

Quando o feixe de luz penetra em alguns materiais sua trajetória muda de ângulo de acordo com a diferença de densidade dos meios.

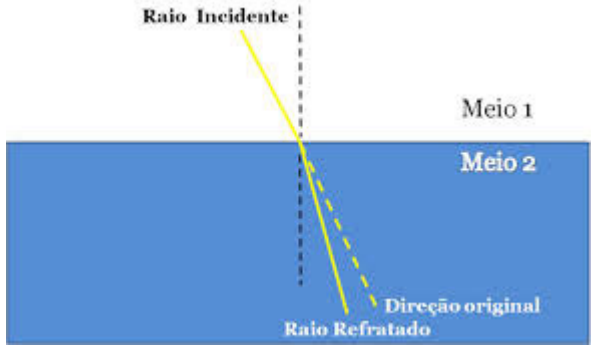
Lei da refração ou de Snell:

$$\frac{\text{sen}\,\theta_i}{\text{sen}\,\theta_r} = n_{21}$$

n_{21} é uma constante, chamada índice de refração ou IR



Exemplo de alguns IR:



Material	IR
Ar (em temperatura e pressão padrão ou STP)	1,0003
Água	1,33
Álcool etílico	1,36
Vidro	1,66
Plástico	1,51
Vidro Denso	1,52
Sal	1,53
Quartzo	1,46
Cristal	1,58
Diamante	2,42

Transparência

$$I = t\, I_1 + (1-t)\, I_2,\ 0 \leq t \leq 1$$

onde, I_1 é a superfície visível, I_2 é a superfície imediatamente atrás da superfície visível, e t é o fator de transparência para I_1. Se I_2 também é transparente, o algoritmo é aplicado recursivamente até encontrar uma superfície opaca ou o fundo da cena.

Modelos de iluminação globais

Ao contrario dos modelos locais que consideram a superfície a luz e o observador, os globais consideram todos os objetos da cena, precisam ter toda a base de dados dos objetos

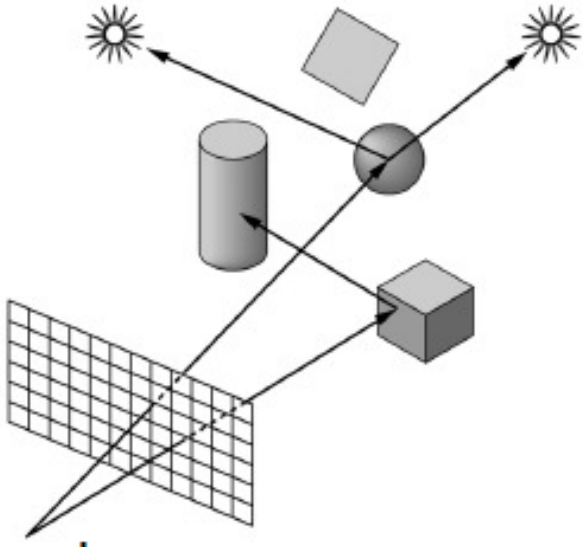
Principais: Raytracing e radiosidade

Não produzem os mesmo efeitos nem são adequados pra as mesmas coisas!

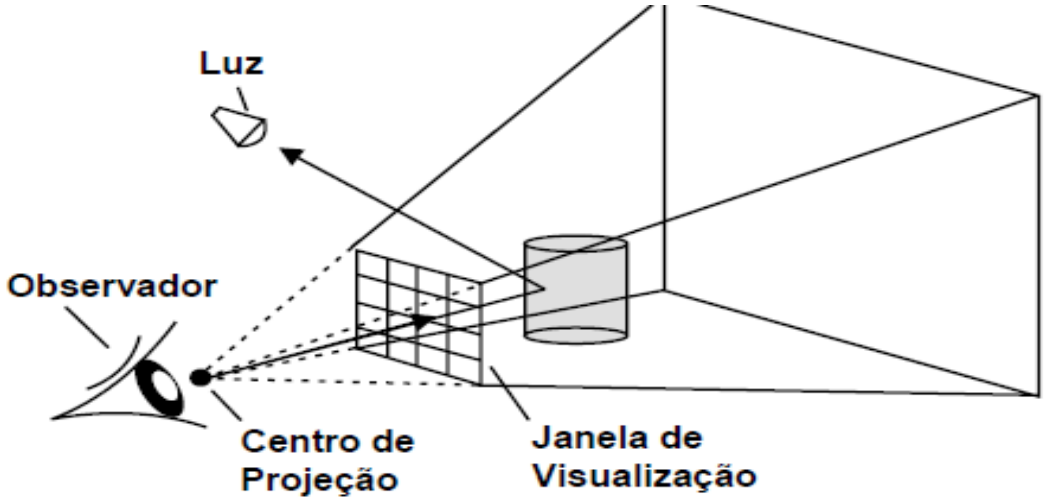
Lentos para real time!

Raytracing

Bom para:
reflexões,
transparências,
objetos fáceis de
calcular interseções
(superfícies,
planas, esférica,
E cilíndricas)



Ray tracing



é uma técnica para gerar uma imagem, seguindo o caminho da luz através de pixels em um plano de imagem e simulando os efeitos de seus encontros com objetos.

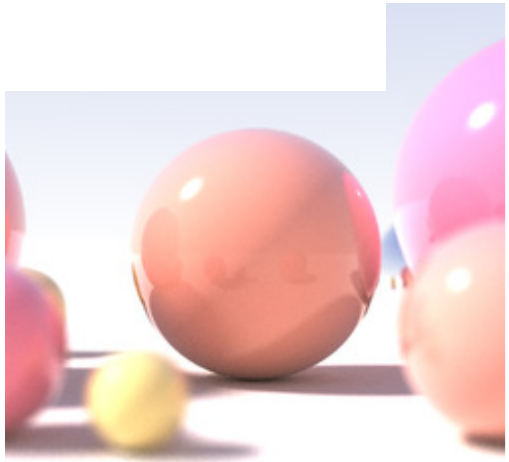
é capaz de produzir um elevado realismo visual, geralmente maior do que o dos métodos de processamento locais típicos, mas em um maior custo computacional.

Isso faz com ray tracing mais adequado para aplicações em que a imagem pode ser renderizada lentamente, como em imagens de cinema e televisão, efeitos visuais, e pouco adequada para aplicações em tempo real, como jogos, onde a velocidade é fundamental.

é capaz de simular uma variedade de efeitos ópticos, como os fenômenos de dispersão, reflexão e refracção.

O algoritmo de Ray tracing considera os seguintes pontos:

- Os raios são disparados de forma sistemática, de modo que cada um deles corresponda a um pixel na tela.
- Após o disparo, o raio percorre o espaço podendo atingir um objeto ou sair da cena.
- Se atingir algum objeto, o ponto de intersecção é calculado. As contribuições das fontes de luz para cada ponto, levando em conta a sombra de outros objetos, também são calculadas.
- Se o objeto for opaco, a soma dessas contribuições será a intensidade luminosa total naquele ponto.
- Caso contrário, as contribuições devidas aos reflexos e refrações, serão também computadas. O pixel correspondente pode, então, ser exibido.
- Se não houver interseção, o pixel terá a cor de fundo.



Algoritmo clássico

Para cada pixel da tela:

1. Trace um "raio" a partir do observador até a cena a ser representada através de um pixel da tela;
2. Determine qual o primeiro objeto a interceptar esse raio;
3. Calcule a cor ambiente da superfície do objeto no ponto de interseção baseado nas características do objeto e na luz ambiente;
4. Se a superfície do objeto for reflexiva, calcule um novo raio a partir do ponto de interseção e na "direção de reflexão";
5. Se a superfície do objeto for transparente, calcule um novo raio a partir do ponto de interseção.
6. Considere a cor de todos os objetos interceptados pelo raio até sair da cena ou atingir uma fonte de luz, e use esse valor para determinar a cor do pixel e se há sombras.

Espelhos:

O ray tracing deve considerar os raios refletidos toda vez que o coeficiente de reflexão de uma superfície for diferente de zero. O coeficiente de reflexão varia entre 0 e 1, determinando que quantidade de energia do raio de luz deve ser considerada como absorvida pelo objeto em questão, compondo uma soma ponderada das componentes de cor para o pixel na tela. Um espelho possui um coeficiente de reflexão próximo de 1, ou seja, nessa superfície todos os raios incidentes devem ser refletidos com o mesmo ângulo de incidência em relação à direção da reta normal à superfície. Além do coeficiente de reflexão, as superfícies também apresentam um coeficiente de refração que expressa a maneira pela qual a luz passa através de um meio para outro.

Cálculo de interseções:

A tarefa principal do ray tracing consiste no cálculo da interseção de um raio com o objeto. Para essa tarefa, utiliza-se normalmente a representação paramétrica de um vetor ou reta. Cada ponto (x, y, z) ao longo de um raio com origem no ponto (x_0, y_0, z_0) e direção do ponto (x_0, y_0, z_0) para o ponto (x_1, y_1, z_1) é definido em função do parâmetro t, (com valores no intervalo [0,1] pelas equações paramétricas da reta):

$x = x_0 + t(x_1 - x_0);$ $\quad x = x_0 + t\Delta x;\ \Delta x = x_1 - x_0$

$y = y_0 + t(y_1 - y_0);$ $\quad y = y_0 + t\Delta y;\ \Delta y = y_1 - y_0$

$z = z_0 + t(z_1 - z_0);$ $\quad z = z_0 + t\Delta z;\ \Delta z = z_1 - z_0$

(x_0, y_0, z_0) for considerado o centro de projeção, ou o olho do observador

(x_1, y_1, z_1) for o centro de um pixel na "janela"

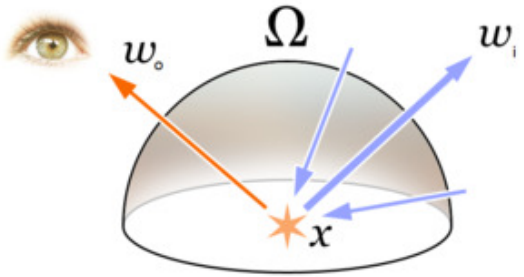
t varia de 0 a 1 entre esses pontos.
valores de t maiores que 1 correspondem a pontos depois da janela

Radiosidade

considera a solução da integral de rendering (equilíbrio da radiância em um ponto ou a conservação da energia) para modelar a iluminação.

O nível de realismo da modelagem é muito maior.

Considera a função bidirecional de distribuição da reflectancia- **bidirectional reflectance distribution function (BRDF)**.

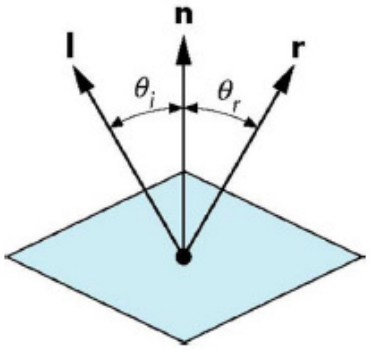


Os anteriores todos consideram

Que os 3 vetores estão no mesmo plano

(reflexão ideal)

$$\mathbf{r} = 2\,(\mathbf{l} \cdot \mathbf{n})\,\mathbf{n} - \mathbf{l}$$



Radiosidade é:

- uma aplicação do método de elementos finitos para resolver a equação de renderização para cenas com superfícies que refletem a luz de forma difusa.

-um algoritmo de iluminação global: a iluminação não vem apenas a partir das fontes de luz, mas todas as superfícies de cena interagindo uns com os outros.

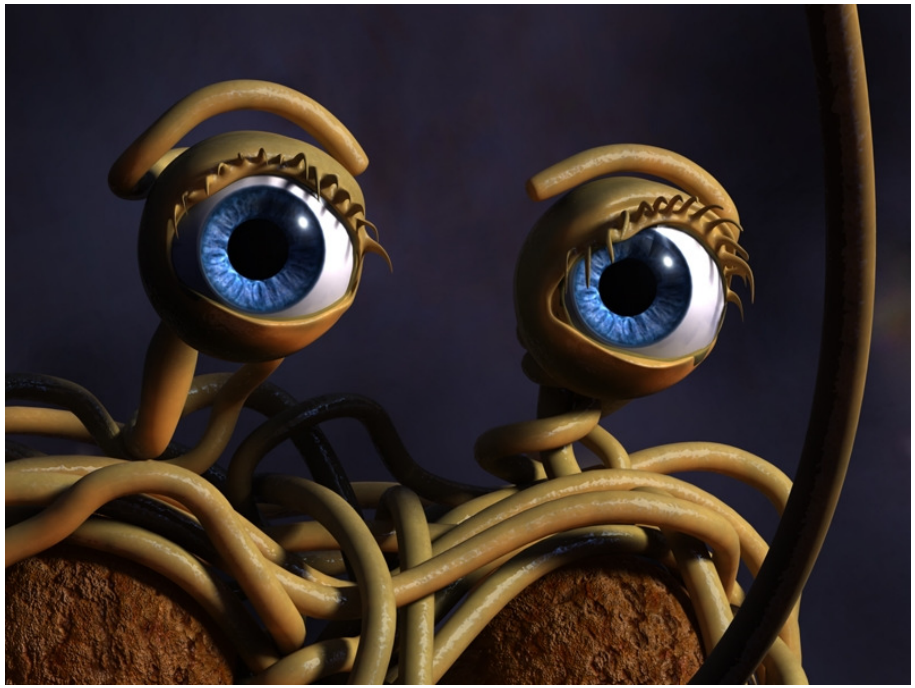
-**independente do ponto de vista**, o que aumenta o volume dos cálculos envolvidos, mas torna-os **úteis para todos os pontos de vista.**

- inicialmente uma aplicação desenvolvidos na área de transferência de calor, posteriormente adaptada para a aplicação de computação gráfica (1984 na Universidade de Cornell).

Color bleeding

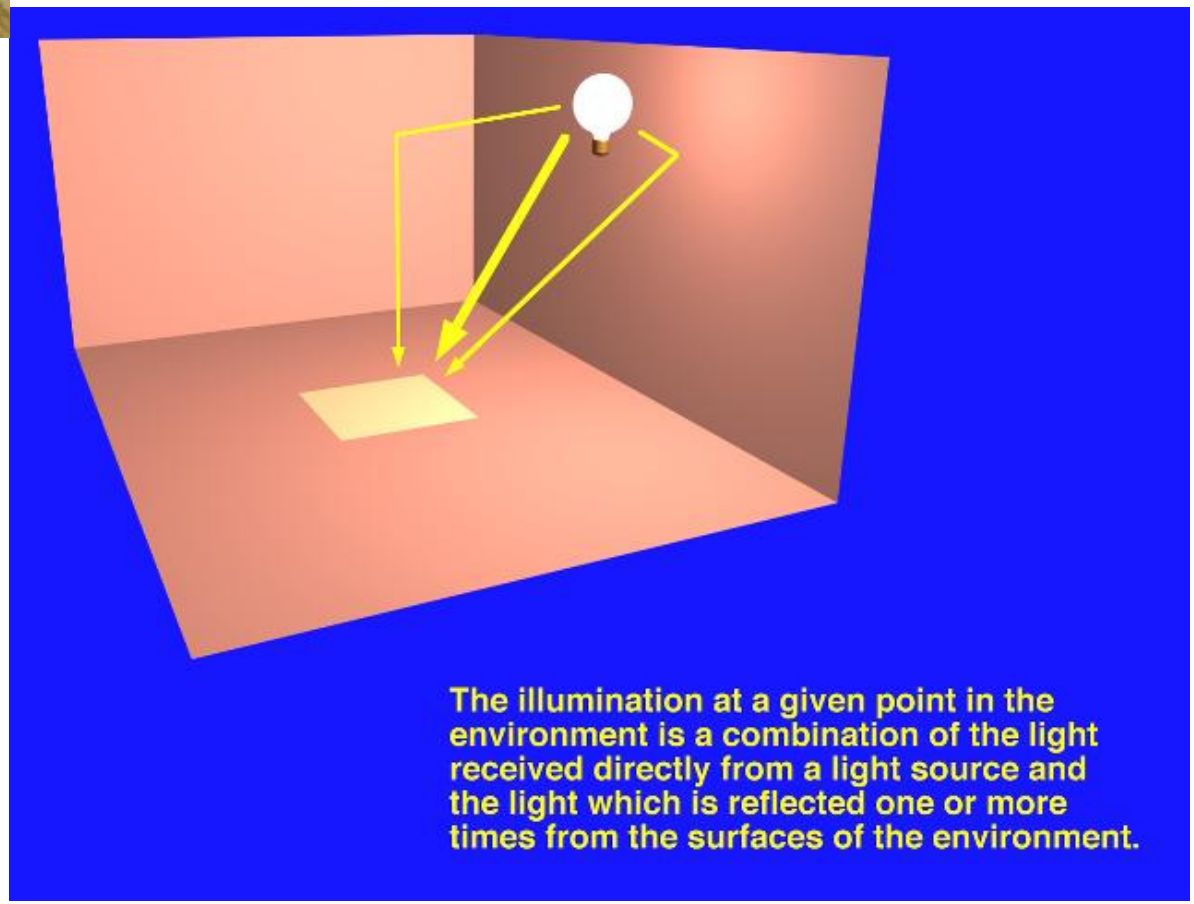
Em rendering , **color bleeding** é a ocorrência de colorização de um objeto ou superfície pela cor refletida de superfícies próximas.

Ocorre principalmente quando se usa Radiosity para a cena 3D.

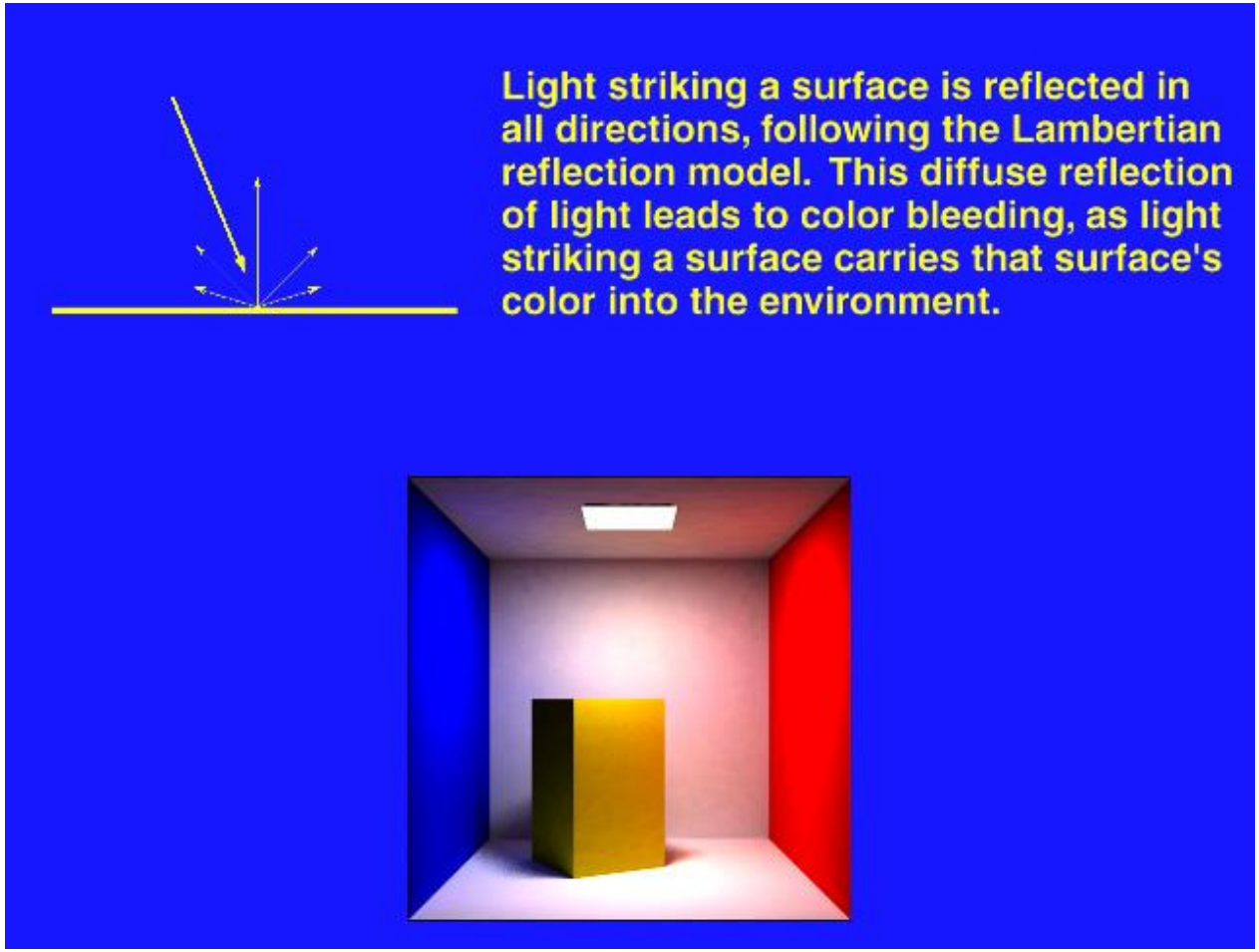


Radiosidade: discretiza o ambiente em um malha

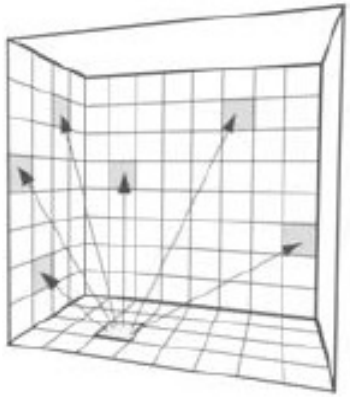
Os limites da malha devem coincidir com os limites das zonas de diferença de iluminação



Balanço ou equilíbrio de energia radiante



Radiosidade:



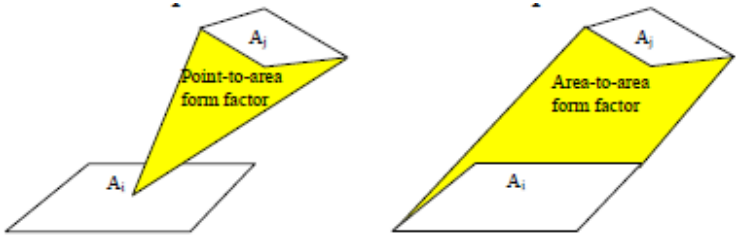
O método da radiosidade é baseado em um modelo simples de balanço de energia. Na sua origem, o cálculo da radiosidade empregado em Transmissão de Calor não é mais do que a aplicação da lei da conservação da energia a cada uma das superfícies de um recinto ou cena, e pressupõe a existência de equilíbrio térmico. Em cada superfície de um modelo, a quantidade de energia emitida é a soma entre a energia que a superfície emite internamente mais a quantidade de energia refletida. A quantidade de energia refletida pode ser caracterizada pelo produto entre a quantidade de energia incidente na superfície e a constante de reflexão da superfície.

$$B_j = \rho_j H_j + E_j$$

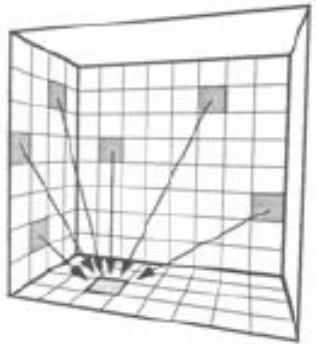
onde B_j é a radiosidade da superfície j, ρ_j sua reflectividade, H_j a energia incidente nesta superfície e E_j a energia emitida pela superfície j

Radiosidade

A radiosidade de uma superfície é a energia dissipada. Isso é usado para determinar a intensidade luminosa da superfície. A quantidade de energia emitida por uma superfície deve ser especificada como um parâmetro do modelo, como nos métodos tradicionais onde a localização e a intensidade das fontes de luz devem ser especificadas. A reflectividade da superfície também deve ser especificada no modelo, como nos métodos de iluminação tradicional. A única incógnita da equação é a quantidade de luz incidente na superfície. Esta pode ser encontrado somando-se todas as outras superfícies à quantidade de energia refletida que contribui com a iluminação dessa superfície:



$$H_j = \sum_{i=1}^{n} B_j F_{ij}$$



onde H_j é a energia incidente na superfície j, B_j a radiosidade de cada superfície i da cena e F_{ij} uma constante i j.

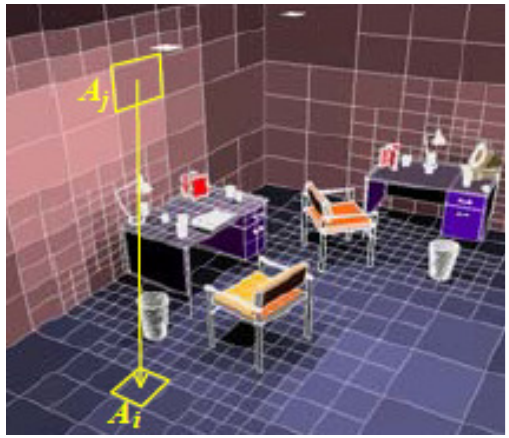
A constante dessa equação é definida como a fração de energia que sai da superfície i e chega na superfície j, e é, portanto, um número entre 0 e 1. Essa constante pode ser calculada por métodos analíticos, ou através de semelhança geométrica.

A equação da radiosidade fica assim:

$$B_j = E_j + \rho_j \sum_{i=1}^{n} B_j F_{ij}$$

A consideração de todas as superfícies da cena forma uma seqüência de N equações lineares com N incógnitas, o que leva a uma solução matricial:

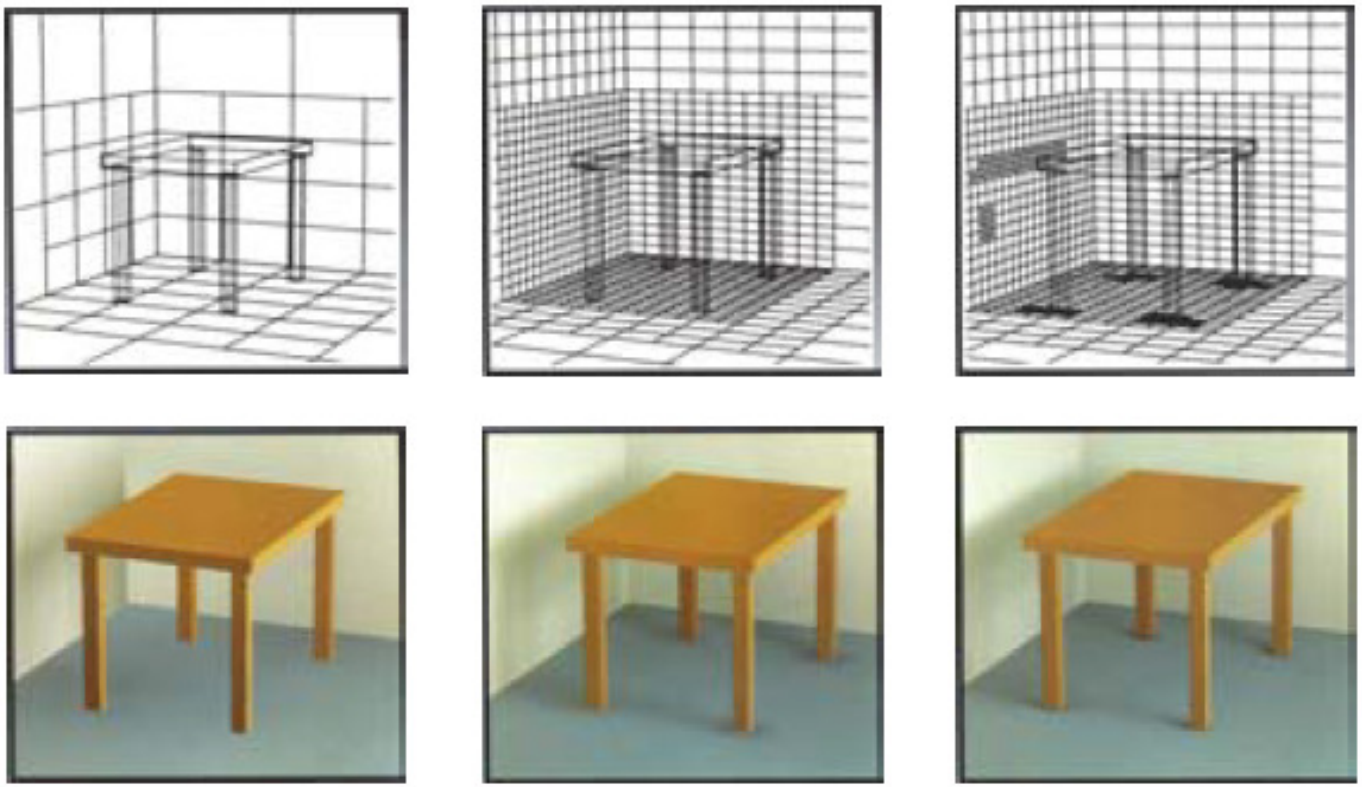
$$\begin{bmatrix} 1-\rho_1 F_{11} & 1-\rho_1 F_{12} & \Lambda & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_1 F_{22} & \Lambda & -\rho_2 F_{2n} \\ M & M & O & M \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \Lambda & -\rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ M \\ B_3 \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ M \\ E_3 \end{bmatrix}$$



Essa matriz tem duas propriedades interessantes: é diagonalmente dominante e, portanto, converge quando usada no método iterativo de Gauss Seidel, [GOR, 84]. Métodos alternativos para o cálculo dessa matriz já foram propostos por Cohen et al. [COH, 88]. Alguns permitem uma convergência para a solução correta mais rápida que o algoritmo iterativo de Gauss Seidel.

Refinamentos progressivos

Alterando o numero de elementos da malha:



Coarse patch solution
(145 patches)

Improved solution
(1021 subpatches)

Adaptive subdivision
(1306 subpatches)

Sombreamento anisotrópico

Isotrópico x ortotrópico

Photon mapping

Algoritmo de **iluminação global** em 2 passadas (two-pass) que considera modelos de radiância para maior realismo na simulação da refração e reflexão da luz em superfícies transparentes

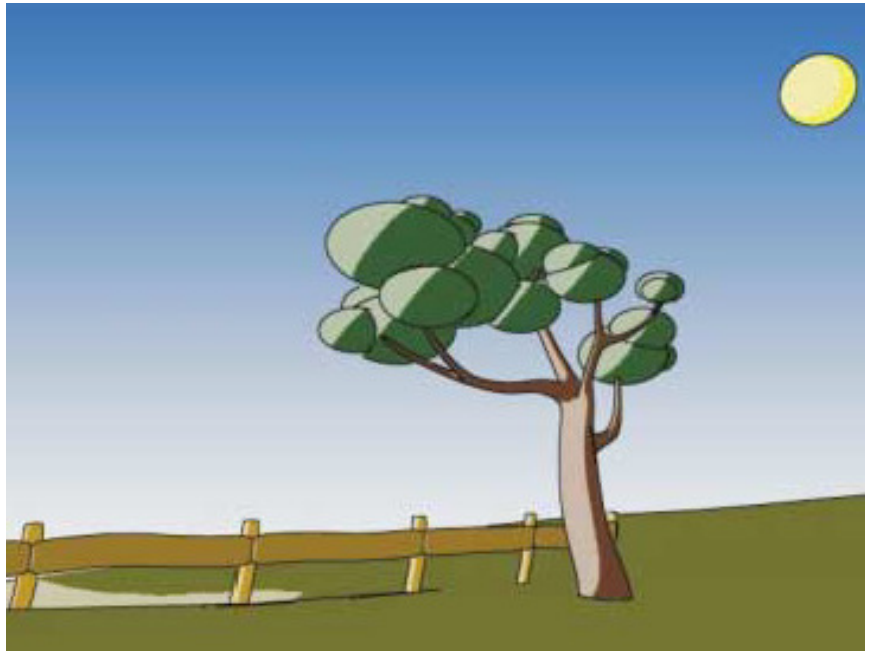
É capaz de simular a refração da luz em meios transparentes tal como o vidro ou a água, inter reflexões difusas entre objetos iluminados, a dispersão da luz sob a superfícies de materiais translúcidos, e efeitos causados por partículas, tal como o **fumaça ou a água de vapor.**

Há muito mais do que isso!

Vimos aqui apenas sobre um realismo fotográfico das imagens, mas há diversas outras formas e esse assunto esta sempre em constante evolução. Assim depois desta leve introdução continue na área! Você já tem a bagagem teórica que precisa para agora descobrir o resto sozinho!

Toon Shading

Stylistic rendering



Bibliografia:

E. Azevedo, A. Conci, Computação Gráfica: teoria e prática, Campus ; - Rio de Janeiro, 2003

J.D.Foley,A.van Dam,S.K.Feiner,J.F.Hughes. Computer Graphics- Principles and Practice, Addison-Wesley, Reading, 1990.

H. Watt, F. Policarpo - The Computer , Addison-Wesley Pub Co (Net); 1998

http://en.wikipedia.org/wiki/Shadow_mapping

https://noppa.oulu.fi/noppa/kurssi/521493s/luennot/521493S_3-d_graphics_vi.pdf

http://graphics.stanford.edu/papers/rad/