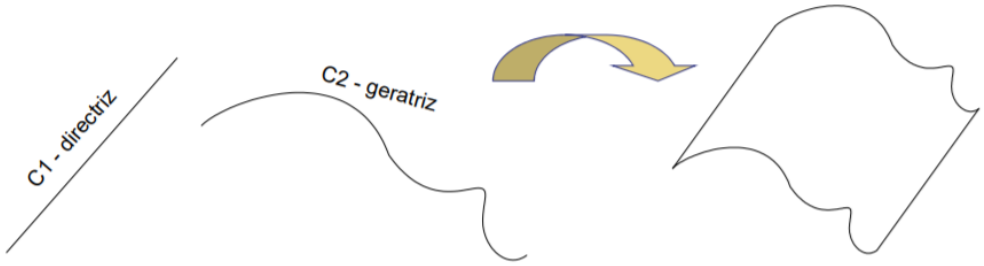
Modelização

Representação computacional de uma entidade real. Trata do problema da definição, criação e manipulação de sólidos/objectos.

A partir de primitivas

- **Primitivas básicas** – uma mesa pode ser obtida por justaposição e transformações geométricas de uma única primitiva básica: paralelepípedo
- **Representação por varrimento (sweep)** – Construção de objectos baseados na seguinte noção: uma curva C2 quando deslocada no espaço ao longo de uma trajetória definida por uma outra curva C1 descreve uma superfície.

  A curva C1 designa-se por **contorno ou diretriz** e C2**, caminho ou geratriz**.



Representação por subdivisão do espaço (Modelos de Subdivisão - ou Decomposição - do Espaço) – representação de um sólido através da "enumeração" do espaço tridimensional por ele ocupado

- **Enumeração exaustiva** – o espaço é decomposto segundo uma grelha tridimensional definida por volumes de forma e dimensão idênticas. A cada um destes cubos dá-se o nome de voxel (análogo a um pixel num espaço bidimensional)
  - Vantagens:
    - **Simplicidade** - operações lógicas (união, intersecção) entre modelos fáceis de efectuar;
    - **Facilidade de deteção de sobreposições entre objetos e a sua adjacência**, através da inspecção das células contíguas às células das fronteiras dos objectos;
    - **Operações** - por exemplo, para determinar se um dado ponto pertence ou não ao sólido basta verificar se o ponto pertence a algum dos voxels
  - Desvantagens:
    - Voxels cúbicos não permitem representações exatas;
    - Dificuldade de representação de objectos com superfícies curvas. É necessária uma grande quantidade de espaço para o armazenamento;
    - **Quanto menores forem os voxels, maior é precisão da representação, mas maior o espaço para armazenar a estrutura**

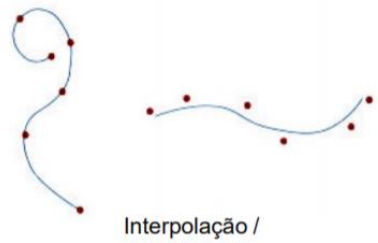
Representação por fronteira – Neste tipo de representação o sólido é definido indiretamente através da superfície que o delimita

Tipos de superfície

- **Superfícies poligonais (planos)**
  - Vantagens:
    - Superfícies planas formadas permitem uma descrição/tratamento através de equações lineares com todas as vantagens daí inerentes

        - Aplicação eficaz de transformações geométricas e algoritmos de recorte, visibilidade e de geometria
        - Ótimo desempenho
    - Desvantagens:
        - Um dos maiores inconvenientes da representação poligonal (superfícies planas) é a impossibilidade de representar de uma forma natural detalhes/superfícies curvas

- **Superfícies não-poligonais (curvas)**
    - Certas curvas podem ser determinadas a partir de um conjunto de pontos conhecidos, denominados pontos de controlo



    - Interpolações polinomiais conduzem, regra geral, a soluções bastantes "oscilatórias" produzindo resultados inaceitáveis
    - Alternativas:
        - **Splines** – interpolações por partes, impondo certas restrições na interseção das várias partes (continuidade, diferenciabilidade)
        - **Curvas e superfícies de Bezier, Hermite e B-splines, Nurbs** (Non-Uniform Rational B-Splines são um tipo especial de B-Splines, onde os pontos de controlo são caracterizados por pesos: quanto mais peso tiver um Control Vertex, mais a curva será "atraída" por este) – permitem o controlo da curva pela especificação de um conjunto de pontos (pontos de controlo)
        - **Quádricas**

Renderização

Consiste na conversão de um objeto tridimensional (3D), descrito através de modelos geométricos, numa figura 2D, a ser depois exibida num dispositivo de visualização.

Ray-tracing – gera modelos globais, isto é, modelos em que são consideradas interações entre os vários objetos existentes (sendo possível incluir sombras entre outros fenómenos óticos)

Para cada pixel da imagem

- É traçado um raio e determinado (interseção) qual o objeto que se encontra mais próximo do observador
- Uma vez detetada a superfície do objeto mais próximo determinam-se as características desse ponto de interseção (cor, sombras...) com base nas características da superfície do material e da(s) fonte(s) de luz existentes

Renderização poligonal – gera modelos locais, isto é, modelos em que não se consideram as interações entre os objetos

- Cálculo das propriedades dos vértices (cor)
- Constituição da imagem 2D por um processo de projeção dos vértices 3D que definem os objetos

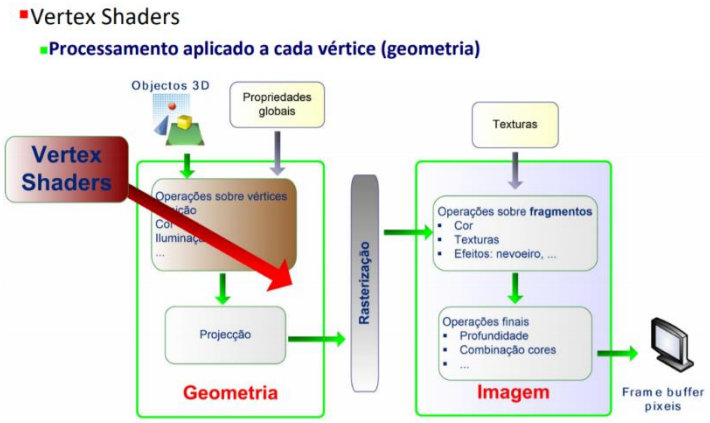
- Interpolação dos pontos interiores

Pipeline Fixo

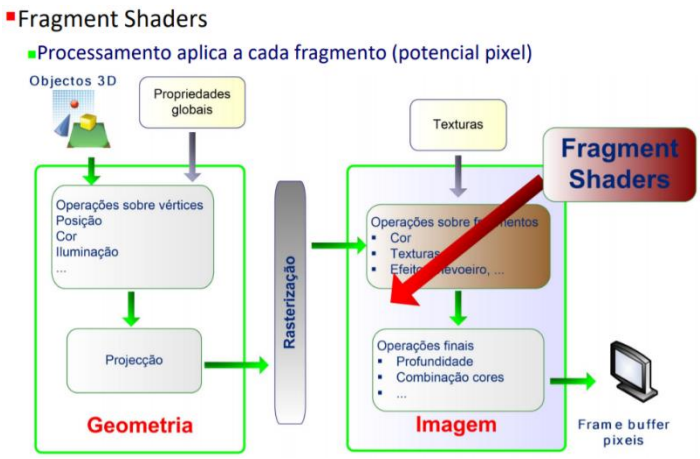
- Funcionamento “declarativo”
- Impõe uma criatividade e liberdades limitadas
- Qualquer operação realiza todas as operações do pipeline

Pipeline Programável

- Shaders são “pequenos” programas executados na GPU
- Utilizados para personalizar o processamento dos vértices e pixels da cena
- Permitem criar cenas realísticas, com efeitos impossíveis (ou muito dificilmente alcançáveis) com pipeline fixo



Permite alterar diretamente: Coordenadas, Cor, Iluminação, Normal
Não pode alterar: Volume de visualização, Perspetiva



Permite alterar diretamente: Texturas, Cores de cada fragmento, Efeitos (por exemplo nevoeiro)
Não pode alterar: Testes aos fragmentos: profundidade, Combinação de cores

Geometria

Produto Interno – operação que transforma dois vectores num escalar (se produto interno = 0, os vetores são perpendiculares)

$$\vec{v} \cdot \vec{u} = \sum_{i=1}^{n} v_i u_i = v^T u$$

Norma

$$\|\vec{v}\| = \sqrt{\vec{v} \cdot \vec{v}} = \sqrt{\sum_{i=1}^{n} v_i v_i}$$

Versor – norma unitária, representa a direção do vetor

$$\vec{n}_v = \frac{\vec{v}}{\|\vec{v}\|}$$

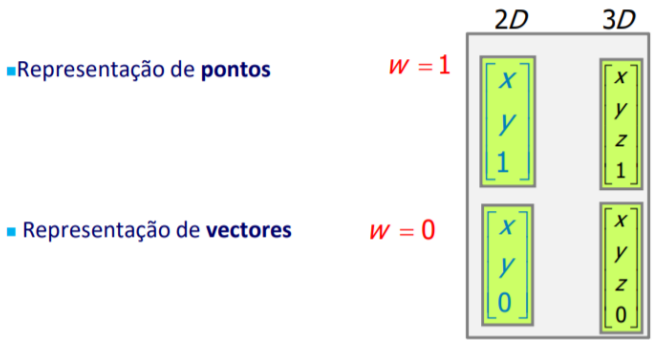
Produto Vetorial – operação entre dois vetores cujo resultado é um vetor perpendicular a ambos

$$\vec{w} = \vec{u} \times \vec{v} = \begin{vmatrix} \vec{n}_x & \vec{n}_y & \vec{n}_z \\ u_x & u_y & u_z \\ v_x & v_y & v_z \end{vmatrix} \quad \vec{w} = \vec{u} \times \vec{v} = \begin{vmatrix} \vec{n}_x & \vec{n}_y & \vec{n}_z \\ 5 & 3 & 0 \\ 2 & 4 & 0 \end{vmatrix} = (3x0 - 4x0)\vec{n}_x - (5x0 - 2x0)\vec{n}_y + (5x4 - 3x2)\vec{n}_z$$

Tipos de transformações

- **Transformações rígidas** (ou Euclidianas) – preservam ângulos e distâncias
  - Translação
  - Rotação
  - Reflexão
- **Transformações ortogonais** – preservam ângulos mas não necesseriamente comprimentos
  - Translação
  - Rotação
  - Escala uniforme
- **Transformação linear** – a origem é mapeada na origem
- **Transformação afim** – transformações lineares em que são permitidas translações

Coordenadas homogéneas – representação unificadora que simplifica a composição e a inversão de transformações



Matrizes de transformações

Translação

$$\begin{bmatrix}\tilde{x}\\ \tilde{y}\\ \tilde{z}\\ 1\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 & tx\\ 0 & 1 & 0 & ty\\ 0 & 0 & 1 & tz\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}; tx, ty, tz \in \mathbb{R}$$

Escala

$$\begin{bmatrix}\tilde{x}\\ \tilde{y}\\ \tilde{z}\\ 1\end{bmatrix} = \begin{bmatrix}Sx & 0 & 0 & 0\\ 0 & Sy & 0 & 0\\ 0 & 0 & Sz & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}; Sx, Sy, Sz \in \mathbb{R}$$

Rotação

$$\begin{bmatrix}x'\\ y'\\ 1\end{bmatrix} = \begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}; \theta \in \mathbb{R}$$

$$\begin{bmatrix}\tilde{x}\\ \tilde{y}\\ \tilde{z}\\ 1\end{bmatrix} = \begin{bmatrix}\cos\varpi & 0 & -\sin\varpi & 0\\ 0 & 1 & 0 & 0\\ \sin\varpi & 0 & \cos\varpi & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}$$

Rotação em torno do eixo dos yy

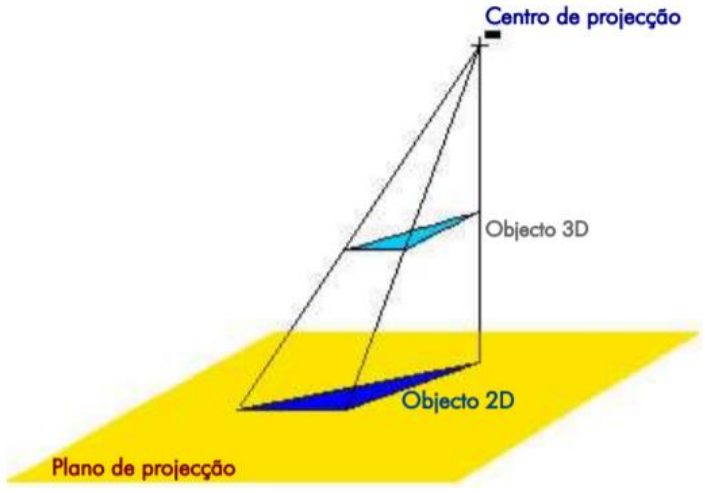
Distorção

$$\begin{bmatrix}1 & a & c & 0\\ b & 1 & d & 0\\ e & f & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

Projeção

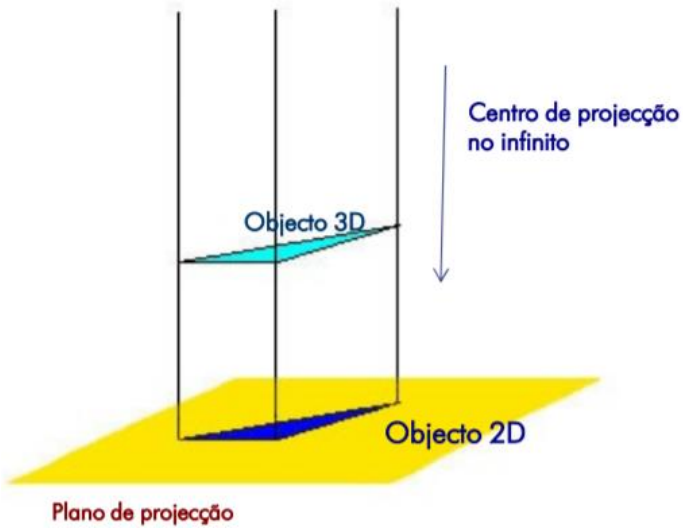
Processo que mapeia uma entidade tridimensional (3D), numa superfície planar (2D)

- **Perspetiva**
  - Apresenta um resultado mais familiar ao observador humano
  - O centro de projeção encontra-se a uma distância finita do plano de projeção



$$\begin{bmatrix}x'\\ y'\\ z'\\ w\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & \frac{1}{d} & 0\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}$$

- **Paralela**
  - Tipo mais simples de projeção
  - A imagem (projeção) dos pontos são obtidos segundo a mesma direcção de projeção
  - Pode ser entendida como uma projeção perspectiva em que o centro de projeção está no infinito



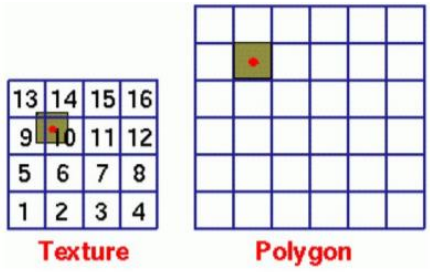
$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Textura

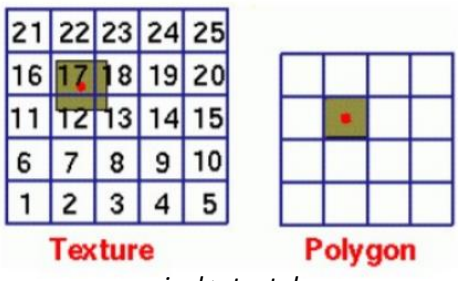
Mapeamento

- **Direto** – caso seja possível estabelecer uma relação direta entre os pontos da textura e o objeto (caso dos polígonos)
- **Uso de superfície intermédia envolvente** – mapeamento em 2 etapas:
  - Considerar uma superfície simples (esfera, cubo, cilindro) com uma determinada textura que envolva o objeto
  - Mapear a textura dessa superfíce simples sobre a superfície do objeto

<u>Muito dificilmente o mapeamento dos textels coincidirá diretamente com os vértices do polígono</u>



pixel < textel



pixel > textel

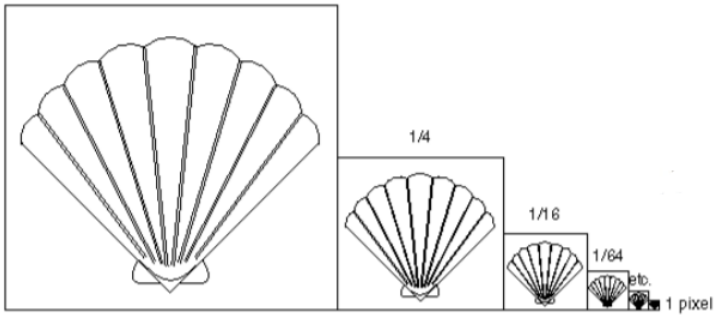
<u>Solução:</u> filtros de interpolação a aplicar nas operações de aumento/diminuição

- **Nearest-Neighbor** – usar o textel mais próximo
- **Linear Interpolation** – usar os vizinhos para o cálculo

Quanto maior é a alteração de escala maior podem ser os problemas

Solução: **MipMap** – várias imagens com nível de resolução decrescente

- Pré-definir/calcular imagens a mais baixas resoluções
- Cada imagem tem metade da resolução da antecessora (1/4) (2^n)
- Escolher a mais adequada à escala atual (feito automaticamente)



Bump Mapping

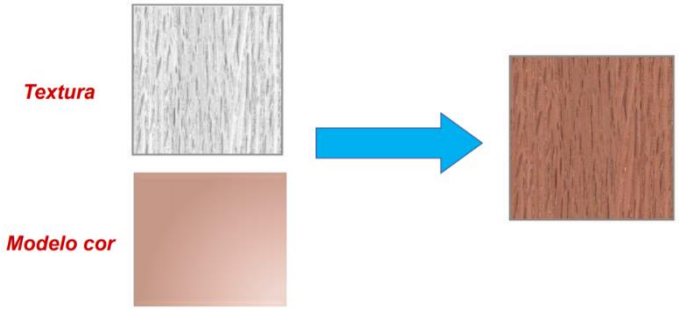
- Simula depressões numa superfície através da perturbação angular variável da normal à superfície em pontos distintos
- A normal modificada é usada no modelo de iluminação (reflexão) e produz diferenças de intensidade que simulam as depressões e ondulações na superfície

Texturas Procedimentais

Texturas definidas explicitamente, geradas a partir de um programa ou modelo sem recorrer a uma imagem digital. São definidas por um algoritmo matemático capaz de as "criar". Geralmente usadas para geração de fenómenos da natureza (água, montanhas, nuvens).

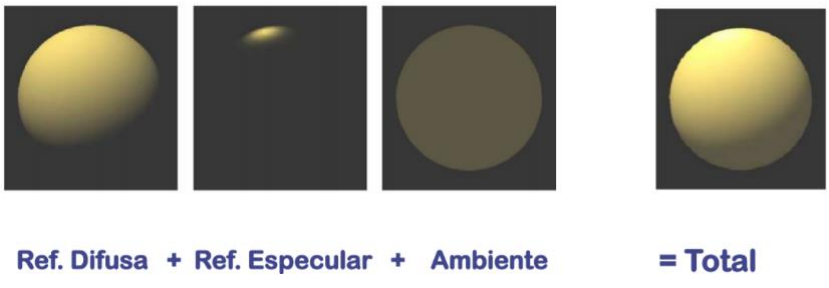
Exemplo: Perlin Noise – "Aleatório orientado": Considerar vários sinais "suaves" e efetuar a sua adição

Blending



Iluminação

Modelo de Phong



- Iluminação ambiente
  - Representa a reflexão de toda a luz indireta (assume-se que esta é originada a partir de todas as direções)
  - Valor constante
  - Não depende da posição do observador

$$I_A = K_A I_{amb}$$

Coeficiente de reflexão ambiente

- Reflexão difusa
  - Luz reemitida em todas as direções
  - Intensidade dependente do ângulo entre a fonte de luz e a normal ao corpo
  - Não depende da posição do observador

$$I_B = K_B I (I_L \cdot n) = K_B I \cos\theta$$

Coeficiente de reflexão difusa

- Reflexão especular
  - Uma percentagem da luz incidente é diretamente refletida à superfície
  - A direção de reflexão é a direção especular
  - A reflexão especular é máxima na direção especular atenuando-se acentuadamente (dependente do material) à medida que o ponto de observação se afasta desta direção
  - Coeficiente de especularidade (ns) - indica o grau de polimento da superfície (por exemplo, um espelho apresenta um grau de especularidade elevado)

$$I_S = K_S I\ (r \cdot v)^{ns} = K_S I \cos^{ns}\gamma$$

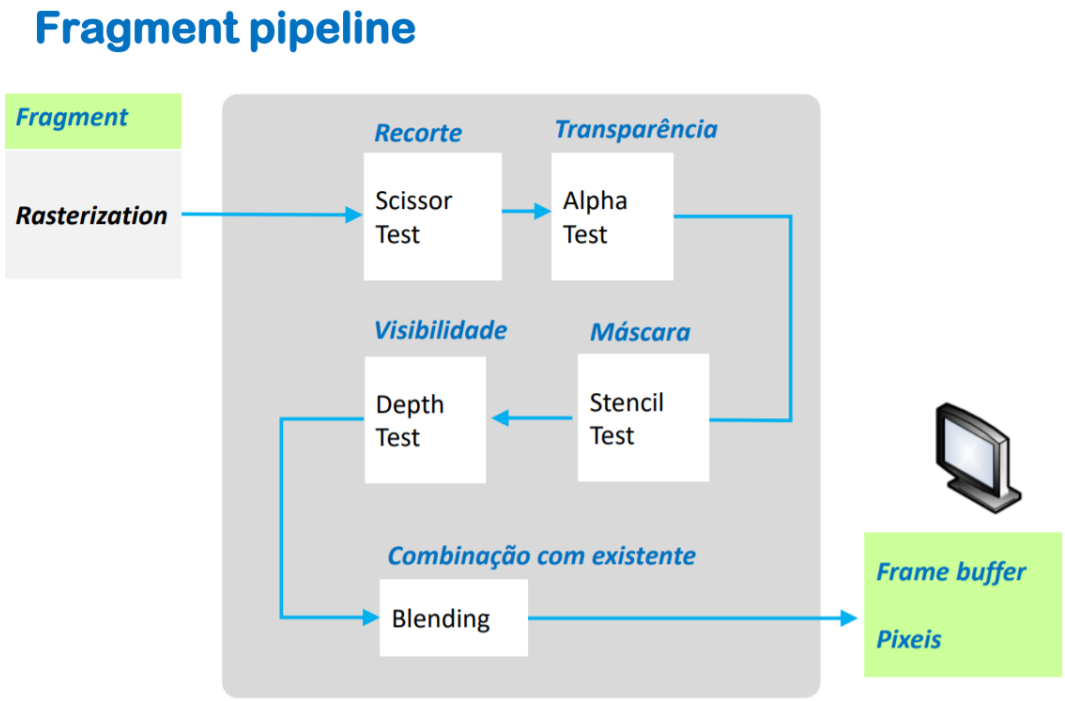
Coeficiente de reflexão especular

Interpolação/coloração de polígonos

(A interpolação é aplicada a triângulos)

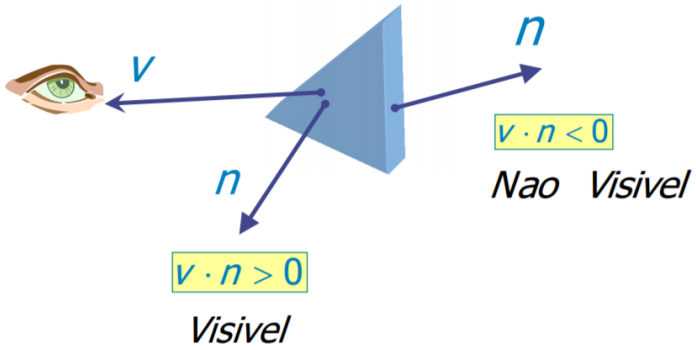
- Flat Shading – a cor é constante para todo o polígono
- Gouraud Shading – interpolação com base na cor. Operação é realizada durante a rasterização, sendo que os pixels são preenchidos da esquerda para a direita, de cima para baixo
- Phong Shading – interpolação com base na normal. Para cada ponto do polígono determina-se a sua normal interpolando bi-linearmente, com base nos vetores normais dos seus vértices

Operações sobre Fragmentos



Visibilidade (operam no espaço do objeto – 3D) – Desenha apenas os objetos que sejam visíveis (os que estão "por trás" não é necessário desenhar)

- Back-face detection – renderizar apenas as superfícies do lado do observador (verifica o sinal da componente n da normal à superfície)



- Z-buffer – 2 buffers:
  - Frame buffer, bidimensional, com a intensidade (cor) do último objeto desenhado
  - Depth buffer (Z-buffer), bidimensional, com a componente z do último objeto desenhado

  Durante a rasterização de cada polígono, cada pixel passa por um teste de profundidade: se a profundidade do pixel for menor que a registrada no z-buffer pinta o pixel (atualiza o buffer de cor) e atualiza o buffer de profundidade, caso contrário, ignora.
  - Vantagens
    - Normalmente implementado por H/W

        - Eficiente para polígonos
        - Pode “facilmente” ser estendido para outras superfícies
    - Desvantagens
        - Rasterização independente da visibilidade (todos os polígonos são rasterizados)
        - Erros na quantetização (valores finitos de profundidade podem originar distorções significativas)

Pinta de trás para frente - Desenha primeiro os opacos e só no fim os transparentes

Recorte (operam no espaço da imagem – 2D) – Permite eliminar todos os polígonos que estejam fora do volume de visualização

- Algoritmo de Cohen-Sutherland (segmentos de reta)

Determinar o código para P1 e para P2

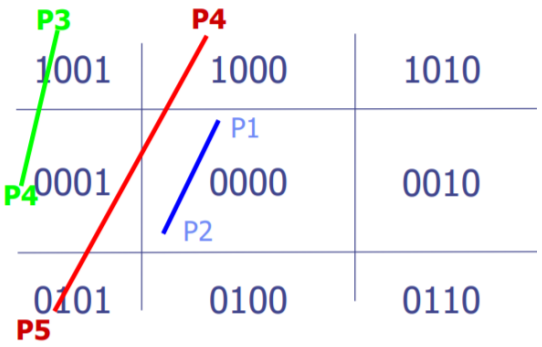
Repete

- 1 - Aceitação trivial – Stop
- 2 - Rejeição trivial – Stop
- 3 - Determina bit mais à esquerda à do *ponto P*
    - Indica “qual o lado” do ponto fora
    - *Ordem pré-definida (cima, baixo, direita, esquerda)*
- 4 - Determinar ponto de intersecção P1
    - Posição do bit indica qual a aresta que intersecta
- 5 - Substituir ponto P pelo P1

Até aceitação ou rejeição trivial

- Se segmento de recta está dentro da Janela de Selecção,
    - Teste **OR** - todos os bits devem possuir o valor zero
- Segmento de recta fora da Janela de Selecção
    - Teste **AND** - se o resultado for diferente de zero a linha está toda fora (**mesmo lado**)

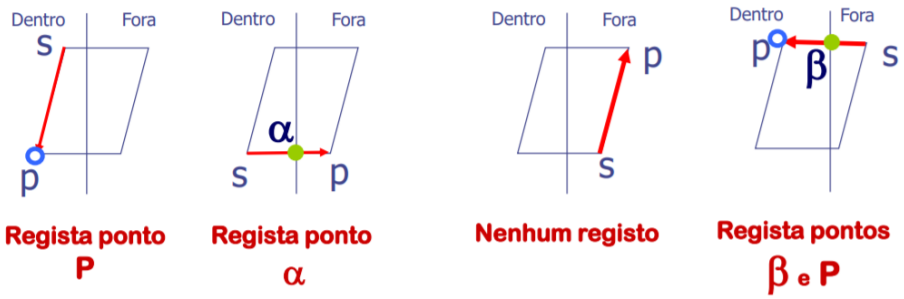
- **Aceitação trivial** (OR=0)
    - P1|P2=0000
        - 0000|0000 = 0000
- **Rejeição trivial** (AND ≠0)
    - P3&P4
        - 1001&0001=0001≠0000
- **Não trivial** (AND=0)
    - P4&P5=0000
        - 1000&0101=0000



- Algoritmo de Sutherland-Hodgman (Polígonos relativamente a polígonos)

▸***Pré-processamento***:

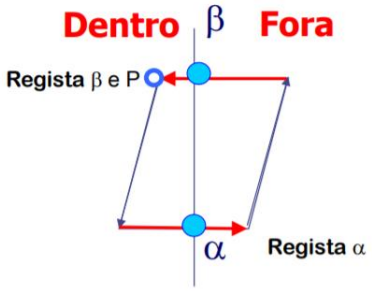
▸Definir uma lista para os vértices p=v1,…vn de acordo com as regras:



Regista ponto P | **Regista ponto α** | **Nenhum registo** | **Regista pontos β e P**

▸Distinguir pontos de intersecção gerados

- De fora para dentro – **tipo β**
- De dentro para fora – **tipo α**



▸***Pós-processamento***

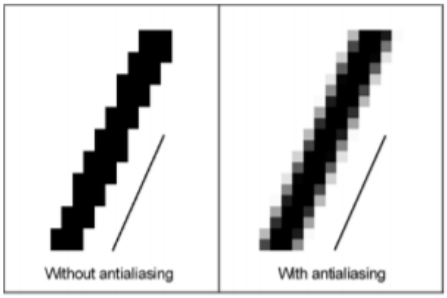
- Definição/eliminação de listas

▸**Nova lista:**

- ao encontrar um ponto do tipo α, ligar com o último ponto β encontrado

Rasterização

Processo de amostragem. Transformação do domínio contínuo para o discreto. Técnicas de antialiasing podem/devem ser empregues.
A Rasterização é uma aproximação de variáveis contínuas para um espaço discreto. Por exemplo: uma reta descrita matematicamente é infinitesimalmente contínua, não importa o quão pequeno um segmento da reta é observado, é impossível determinar qual é o próximo ponto depois de um determinado ponto; não



existem quebras. Porém num espaço discreto existem quebras, e é possível visualizar cada ponto individualmente.
Assim, podemos definir a rasterização de retas como a discretização de um modelo matemático para um espaço de uma matriz quadrada: o ecrã.

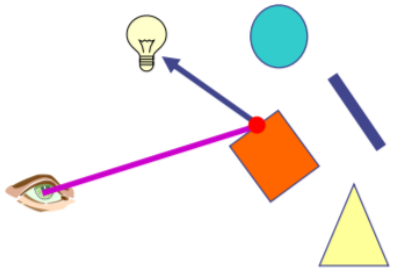
Ray-tracing e Ray-casting

Algoritmo:

1. Construir um raio desde o Centro de projecção que passe pelo plano de imagem
2. Determinar a ponto/superfície onde o raio intersecta
3. Determinar a cor nesse ponto

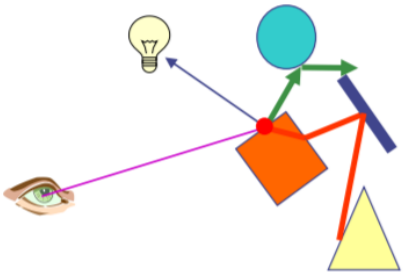
1. Ray – casting : *Iluminação local*

- A ideia é aplicada uma única vez (para o ponto detectado).
- ***Limitações***
  - Sombras?
  - Transparências?
  - Inter-reflexões?



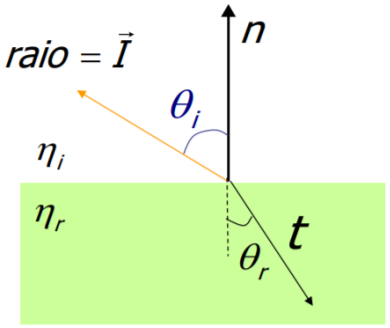
2. Ray – tracing : *Iluminação global*

- A ideia é aplicada iterativamente



2. Refractiva: Direcção de refracção:

Lei de snell

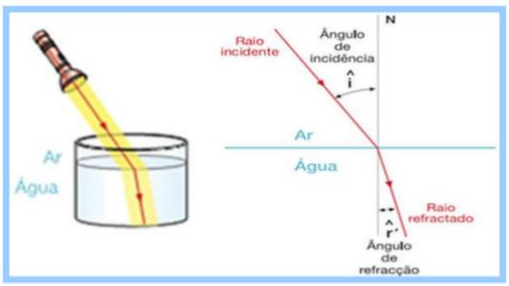


Índice de Refracção

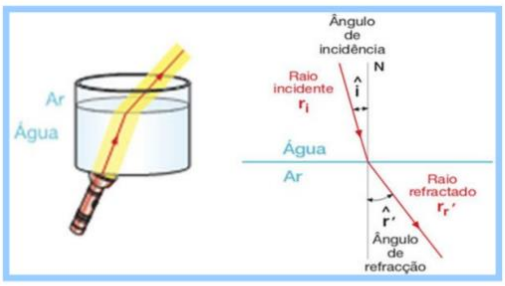
$$\eta = \frac{\eta_j}{\eta_r} = \frac{\text{velocidade da luz no vacuo}}{\text{velocidade da luz no material}}$$

Lei de Snell **para objectos sólidos**

$$\eta_i \sin\theta_i = \eta_r \sin\theta_r$$

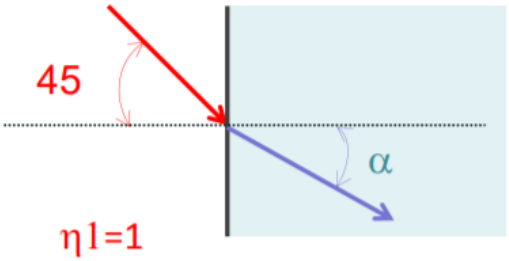


$$\eta_1 < \eta_2$$



$$\eta_1 > \eta_2$$

Exemplo



η1=1
Ar

η2=sqrt(2)
Agua

$$\sin 45\ \eta 1 = \sin\alpha\ \eta 2$$

$$\frac{\sqrt{2}}{2} 1 = \sin\alpha\ \sqrt{2}$$

$$\sin\alpha = \frac{1}{2}$$

$$\alpha = 30$$