Tratamento de linhas e superfícies escondidas

Professor: Luciano Ferreira Silva, Dr.

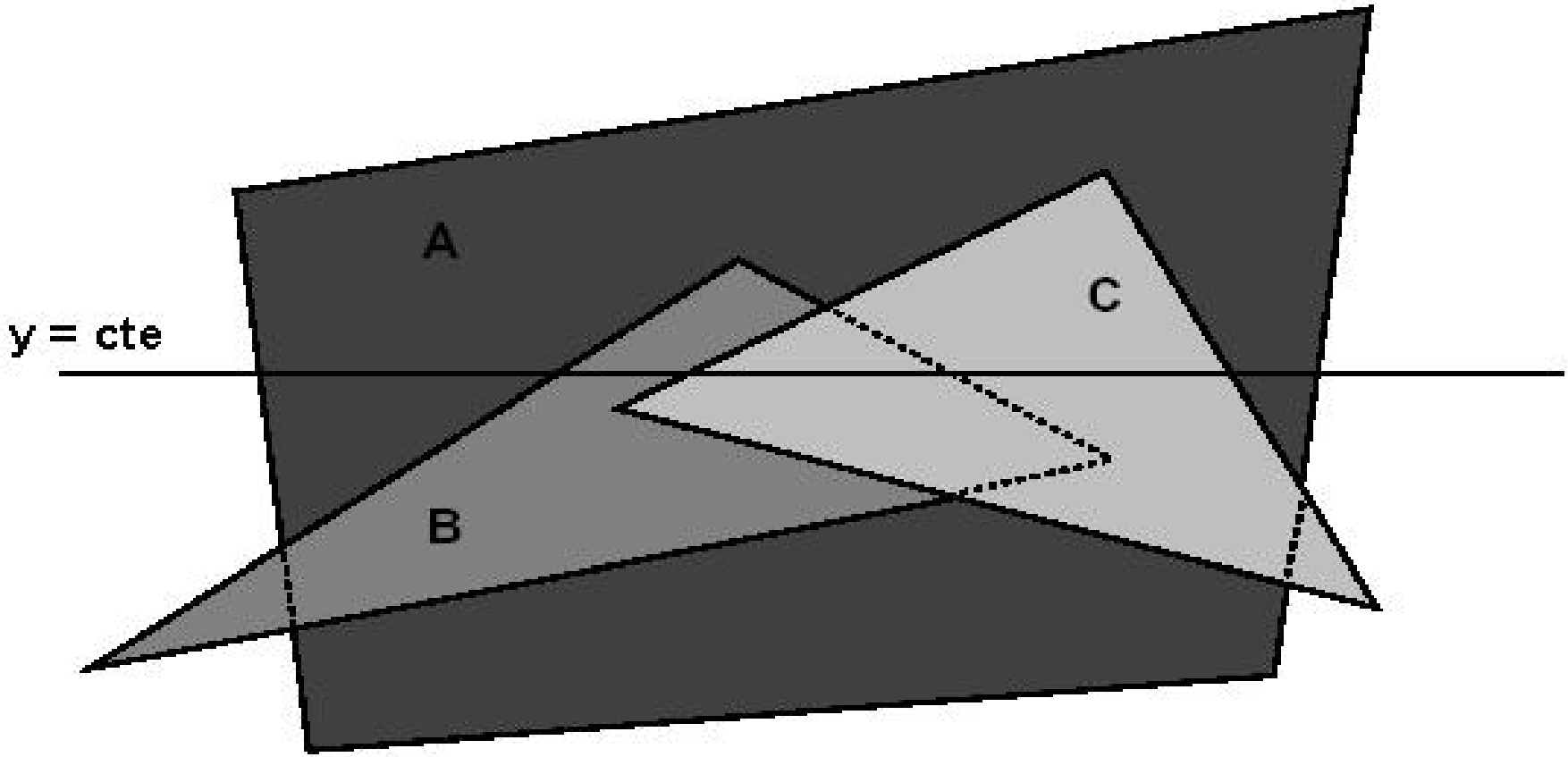
Linhas de Varredura - 'Scan Line'

- Utiliza um plano de varredura (y = cte) por vez;
- Examina os polígonos cortados por este plano de varredura;
- Requer duas arrays:
  - ✓ Intensidade;
  - ✓ Profundidade;

Linhas de Varredura - 'Scan Line'

- Para todo pixel da linha de varredura, faça depth[x,y]= Max e intensidade[x,y]= valor de fundo da tela - background;
- Para cada polígono na cena, encontre todos os pixels no atual plano de varredura "y" que corta o polígono
  - Calcule a profundidade z do polígono no (x,y)
  - se z < profundidade (x), faça depth[x] = z e intensidade [x] = a intensidade correspondente ao polígono.
- Após todos polígonos terem sido considerados, os valores contidos na array "intensidade" representam a solução.

Linhas de Varredura - ‘Scan Line’



Linhas de Varredura - 'Scan Line'

Considerações:

- A tabela de arestas é desenvolvida e armazenada de forma a conduzir o processo de análise das faces: contém as arestas de todas as projeções dos polígonos;
- Existe uma tabela de polígonos:
  - Coeficientes da equação do plano do polígono;
  - Informação da cor do polígono;
  - Flag de controle - inicial: falso - para indicar se estamos dentro ou fora do polígono;

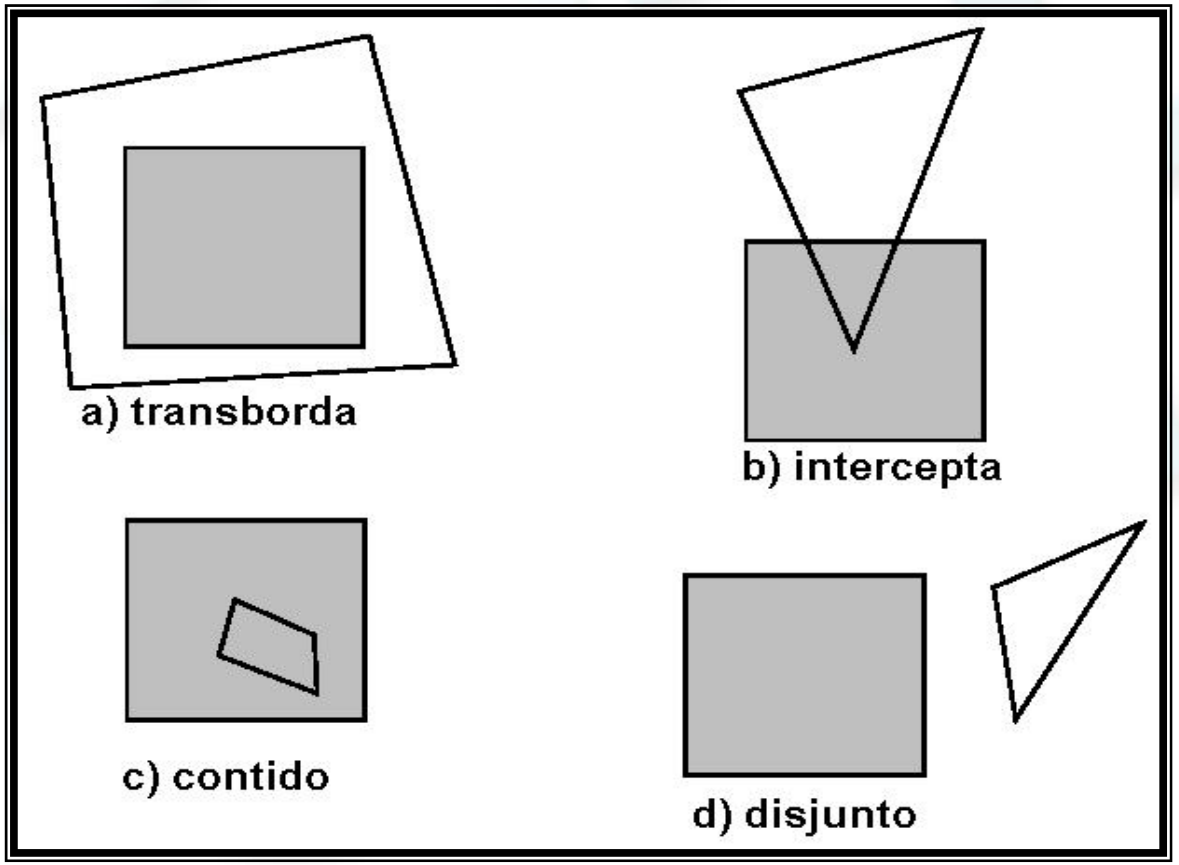
Linhas de Varredura - ‘Scan Line’

Considerações de ordem prática:

- A tabela de arestas (TA) contém todas as arestas cortadas pela LV, ordenadas de acordo com a interseção ‘x’ com a LV;
- Quando a primeira linha da TA é considerada, o flag do seu polígono é tornado “verdadeiro”;
- Quando há dois flag ‘verdadeiro’, então uma comparação dos valores de ´z´ define qual dos dois está mais próximo, este polígono torna-se o polígono corrente e define a cor;
- Se, ao cruzar uma aresta, o flag torna-se falso, significa que a LV saiu do polígono:
  - deve ser procurado o polígono que está mais próximo ao observador, se a aresta for do polígono corrente
  - se não for uma aresta do polígono corrente, nada deve ser feito.

Subdivisão por Áreas - Warnock

- Estratégia: dividir para conquistar!!
- Há quatro possibilidades: área x polígono

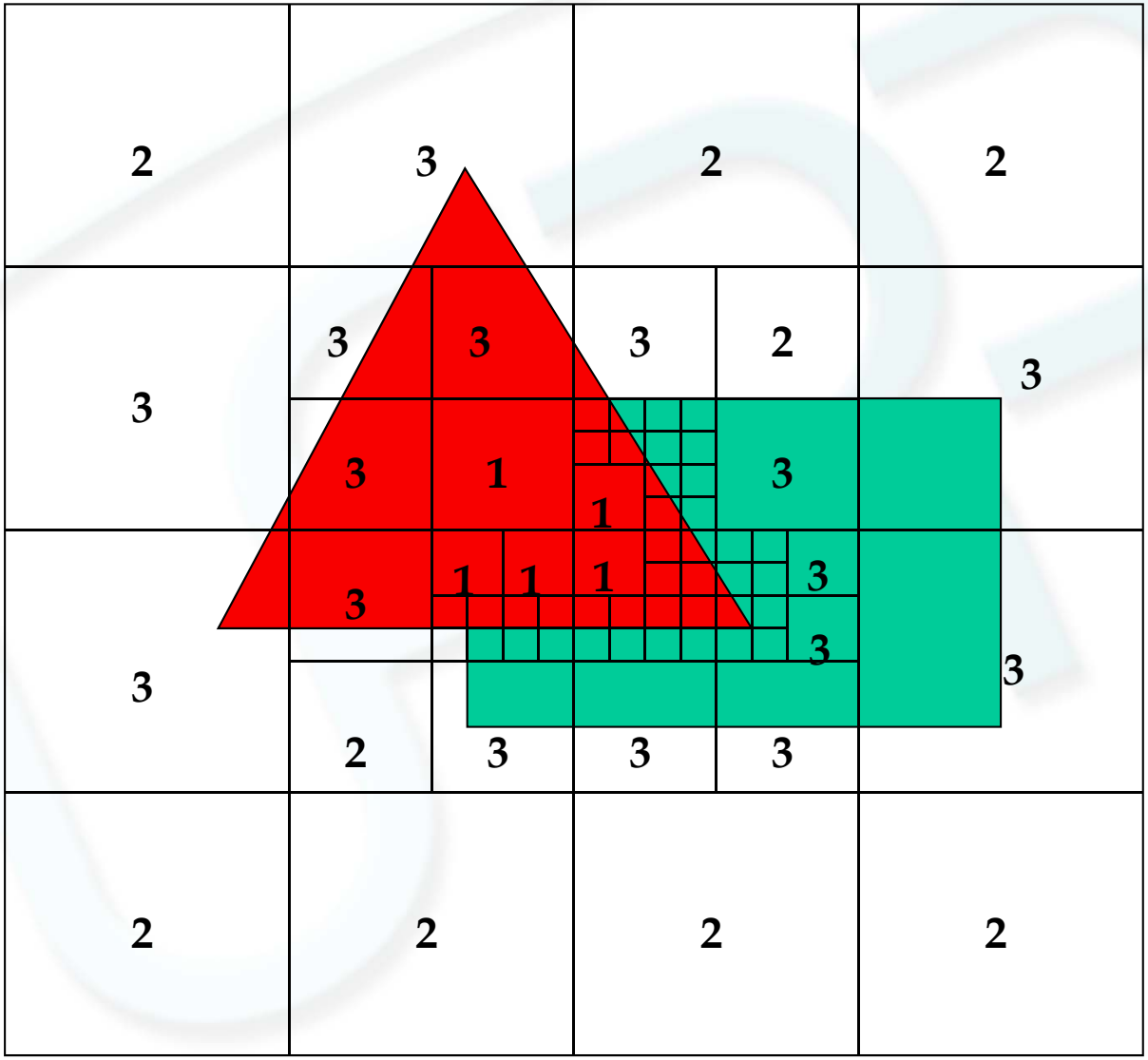


Subdivisão por Áreas - Warnock

- **Há 4 casos possíveis:**

1. Há somente 1 polígono que transborda a área => toda a área é colocada na cor do polígono;
2. Todos polígonos disjuntos à área => valor do pixel = fundo de tela;
3. Somente 1 polígono contido / interceptado => solução é completar o restante da área para a cor do fundo;
4. Há mais de um polígono contido, interceptado ou que transborde e pelo menos um deles transborda. É promovido um teste para detectar se o que transborda esta a frente dos demais, analisando a coordenada z de cada um dos planos.

Subdivisão por Áreas - Warnock

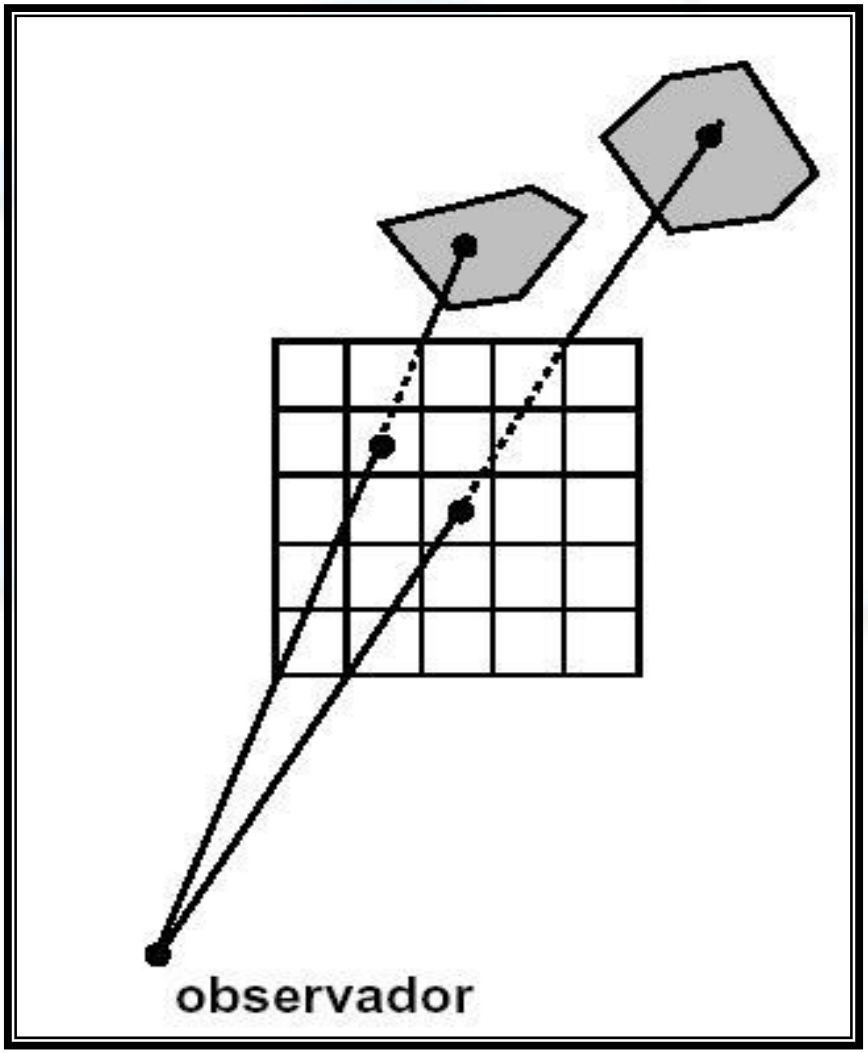


Subdivisão por Áreas - Warnock

- **Observação:**
  - ✓ *É obvio que os 3 primeiros casos acima são de solução muito simples.*
  - ✓ *Nos casos da hipótese 4, onde mais de um polígono pertencem à mesma área, necessitamos olhar a coordenada do plano na qual está contido e ver qual delas está mais próxima do observador.*
  - ✓ *Se, no entanto, os planos dos 2 polígonos se interceptam, será necessário uma subdivisão para análise posterior.*

Ray Casting - Traçado de raios

- Determinar a visibilidade de superfícies, a partir de raios imaginários, que saem dos pontos de vista até o ponto de um objeto de cena;



Ray Casting - Traçado de raios

Considerações práticas:

- ✓ Testes são feitos, de forma a identificar os polígonos que o raio intercepta;
- ✓ A superfície visível é aquela que está mais próxima do observador;
- ✓ Definição do raio:
  - Origem Ro e vetor direção Rd
  - R (t) = Ro + t * Rd
- ✓ Podem ser adaptados para tratar a cor da cena - ray tracing.

Equação do Plano

- **Este algoritmo trabalha com as faces do objeto e utiliza a equação do plano para implementar as faces do objeto à ser exibido;**

- **Equação do plano: Ax + By + Cz + D = 0, onde:**

  - ✓ (x,y,z) é qualquer ponto no plano;

  - ✓ os coeficientes A, B, C e D são constantes que descrevem as propriedades espacial do plano.

Equação do Plano

- **Dados $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ e $P_3(x_3, y_3, z_3)$ pertencentes ao plano. Os coeficientes do plano podem ser obtidos pelos cálculos:**
  - ✓ $A = y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2);$
  - ✓ $B = z_1(x2 - x_3) + z_2(x_3 - x1) + z_3(x_1 - x_2);$
  - ✓ $C = x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2);$

- **A orientação de uma superfície plana no espaço pode ser descrita pelo vetor normal do plano.**
  - ✓ Este vetor normal do plano possui as coordenadas cartesianas (A,B,C).

Equação do Plano

- **Para verificar se um plano é visto pelo observador, aplicamos a equação do plano para coordenadas do observador (x_0, y_0, z_0). Daí, temos:**

$Ax_0 + By_0 + Cz_0$

$< 0 \rightarrow$ o observador está fora, então o plano deve ser exibido;

$= 0 \rightarrow$ o observador está na fronteira.

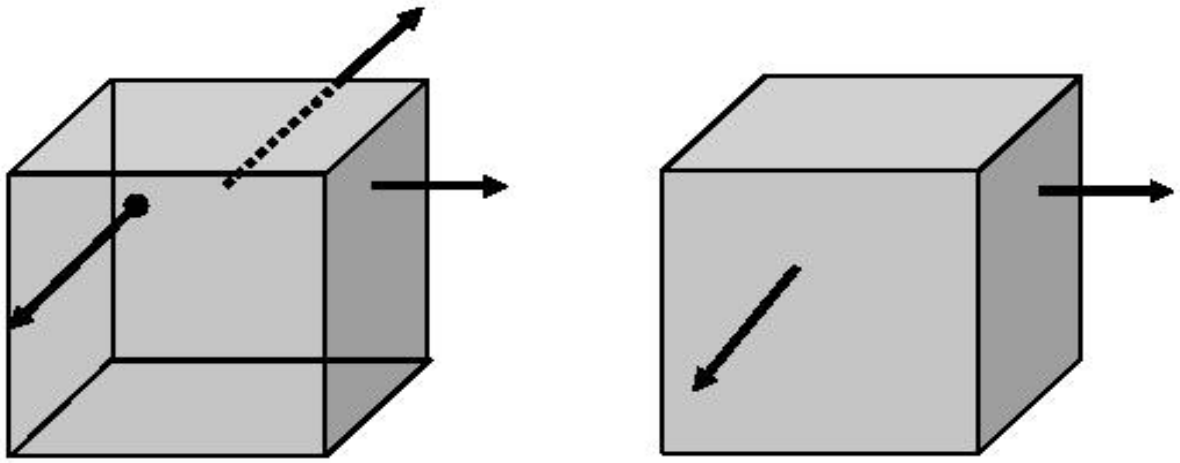
$> 0 \rightarrow$ O observador está dentro, então o plano deve esta escondido.

Back Face Removal

- Determinar as faces que estão de “costas” para o observador;
- Eliminar estas faces do desenho;
- Idéia simples: custo computacional pode ser elevado;
  - ✓Memória;
  - ✓Velocidade de processamento;
- Trabalho base: Sutherland – 1974;

Back Face Removal

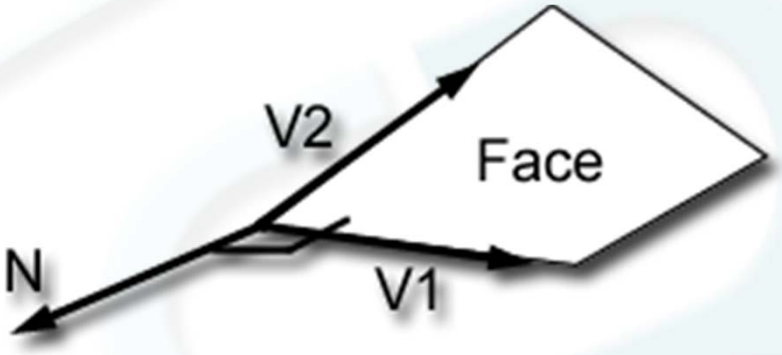
- **Estratégia:**
  - ✓ Descrever as arestas de cada face no sentido anti-horário para quem está fora do objeto;
  - ✓ O vetor normal de cada face indica o exterior do objeto;



vetores normais às faces

Back Face Removal

- **1º passo: Cálculo do vetor normal;**



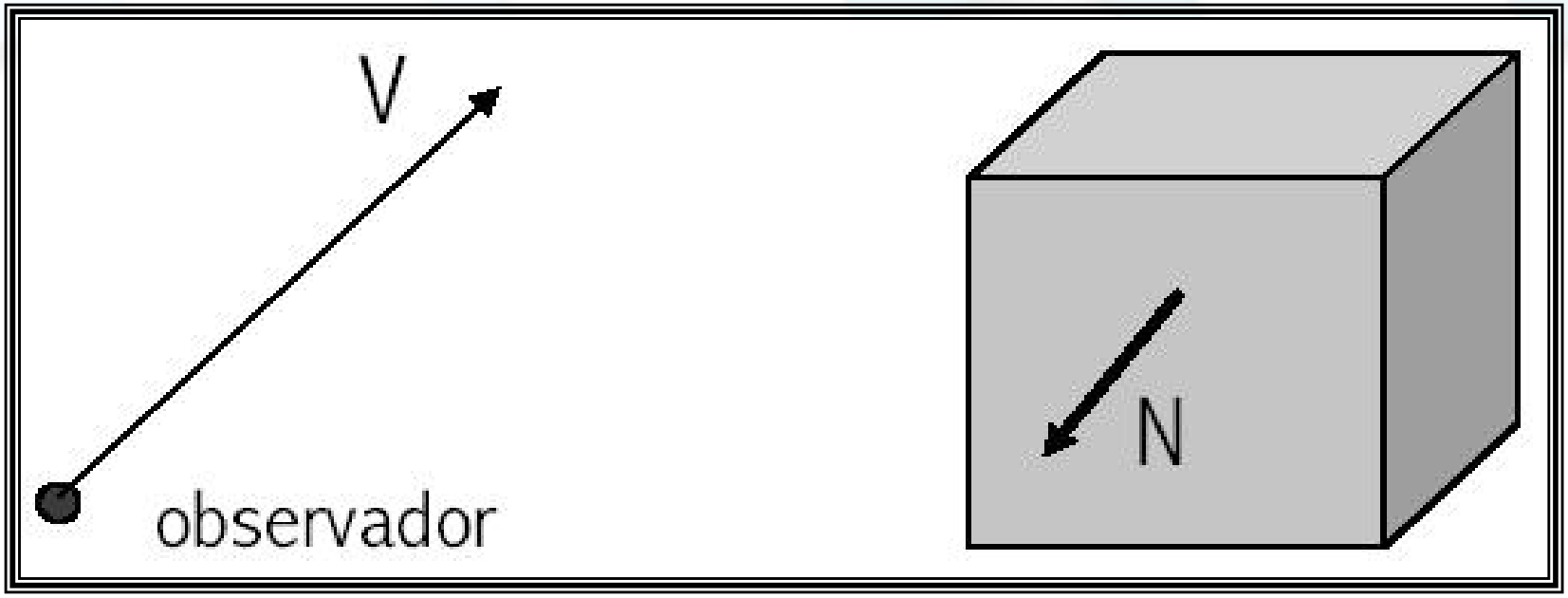
produto vetorial:

$$N = v1 \times v2$$

```
N.x = v1.y * v2.z - v1.z * v2.y;
N.y = v1.z * v2.x - v1.x * v2.z;
N.z = v1.x * v2.y - v1.y * v2.x;
```

Back Face Removal

- **2º passo: determinação de faces frontais;**
  - ✓ Análise do ângulo formado entre a direção de observação e o vetor normal de cada face;
    - Uso do produto escalar entre os dois vetores;



Back Face Removal

- Produto escalar
  - expressão cartesiana:

$$v.n = |v|.|n|.\cos\theta$$

  - expressão analítica:

$$v.n = v.x * n.x + v.y * n.y + v.z * n.z$$

Back Face Removal

- **3º passo: validação das face;**
  - ✓ v.n = 0 → temos faces perpendiculares → esconde;
  - ✓ v.n > 0 → vetores com ângulo maior que 90º → esconde;
  - ✓ v.n < 0 → vetores com ângulo menor que 90º → exibe;