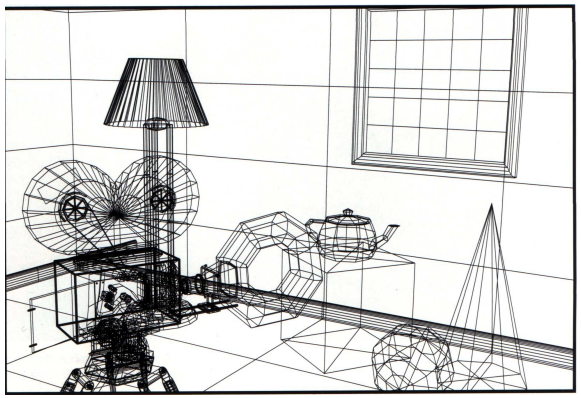
Síntese de Imagem
Cálculo de Visibilidade

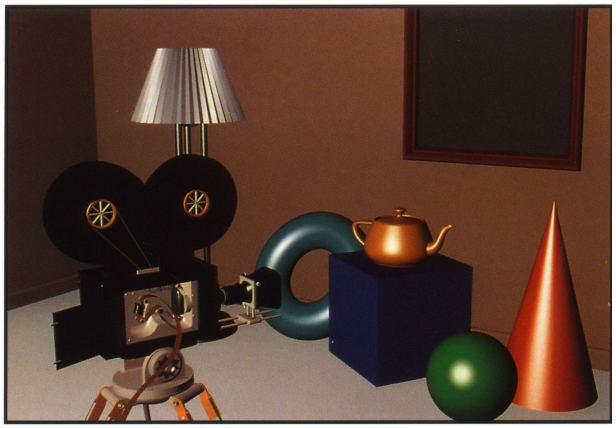
Sistemas Gráficos/

Computação Gráfica e Interfaces

Síntese de Imagem

A **síntese de imagem** (do inglês ***rendering***) consiste na criação de imagens com elevado grau de realismo a partir da descrição dos objectos nela contidos (geometria e interacção com a luz), fontes de luz e posicionamento do observador.





Síntese de Imagem

Índice

Síntese de Imagem
- Cálculo de Visibilidade -

Objectivo: a partir de um conjunto de objectos 3D, determinar quais as linhas ou superfícies dos objectos que são visíveis, quer a partir do centro de projecção (para projecção em perspectiva) quer ao longo da direcção de projecção (para projecção paralela), de modo a mostrar apenas as linhas ou superfícies visíveis.

Duas aproximações possíveis:

1. Para cada pixel da imagem determinar qual o objecto visível (Espaço Imagem)

```
for (cada pixel na imagem)
{   determinar o objecto mais perto do observador, atendendo aos raios
    de projecção;
    desenhar o pixel com a cor apropriada;
}
```

2. Comparar os objectos entre si de modo a seleccionar a parte visível de cada um (Espaço Objecto)

```
for (cada objecto do "mundo")
{   determinar as partes do objecto não obstruídas por ele ou por
    outros objectos;
    desenhar as partes visíveis na cor apropriada;
}
```

Síntese de Imagem
- Cálculo de Visibilidade -

Backface culling

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço objecto

Algoritmos no espaço objecto para determinação de linhas visíveis:

Ao volume: Algoritmo de Roberts

À aresta: Algoritmo de Appel, Loutrel, Galimberti e Montanari

Nestes algoritmos todas as arestas são testadas para produzir uma lista com os segmentos visíveis de todas as arestas.

Ao Volume: supõe-se que uma aresta pode ser oculta pelo volume de um objecto.

À Aresta: o teste é efectuado aresta contra aresta observando que a visibilidade de uma aresta goza de coerência, o que permite determinar a invisibilidade de uma aresta a partir da invisibilidade de outra aresta que possua com ela um vértice comum.

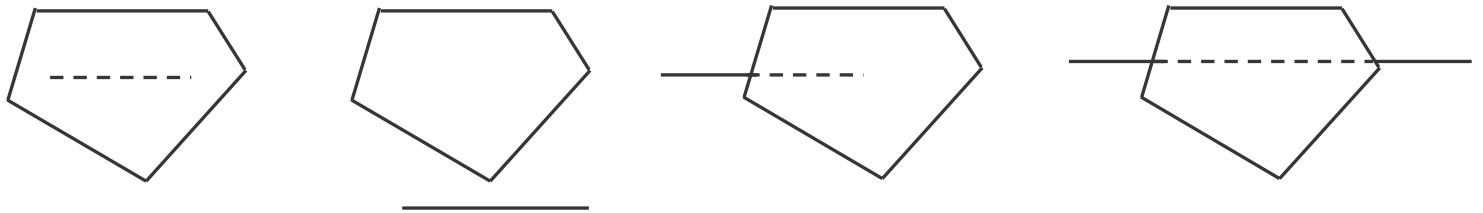
Coerência de aresta: uma aresta só altera a sua visibilidade onde se cruza por trás de uma aresta visível.

Síntese de Imagem
- Cálculo de Visibilidade – Alg. espaço objecto

Ao Volume - Algoritmo de Roberts (1963)

Requisito: cada aresta deve pertencer a uma face de um **poliedro** convexo. Poliedros côncavos devem ser partidos em vários convexos para poder aplicar o algoritmo.

1. Remover todas as faces posteriores dos objectos *(backface culling)* e correspondentes arestas
2. Comparar as arestas restantes contra cada **volume** (poliedro) da cena; deste teste podem ocorrer 4 situações:



- Aresta completamente oculta pelo volume.
- Aresta não oculta
- Uma parte da aresta não é oculta
- Duas partes da aresta não são ocultas

Síntese de Imagem
- Cálculo de Visibilidade – Alg. espaço objecto

À aresta - Algoritmo de Appel, Loutrel, Galimberti e Montanari (1967/9,1970)

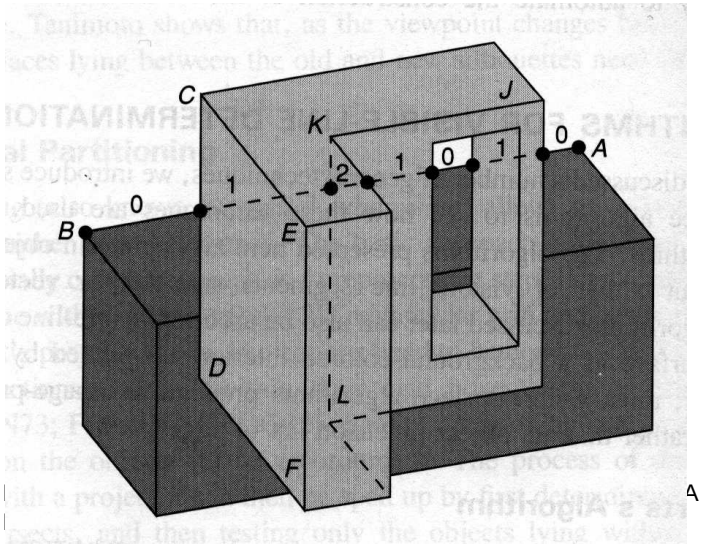
Ao contrario do alg. de Roberts trata ao nível do polígono.

1. Determinar as faces orientadas para o observador *(backface culling).*
2. Calcular a "*Quantitative Invisibility*" de um vértice para cada objecto.

   "*Quantitative Invisibility*" **QI** de um ponto: é o número de polígonos orientados para o observador que ocultam esse ponto.

   Quando uma aresta passa por detrás de um polígono, a sua **QI** é incrementada de 1, e quando deixa de ser ocultada é decrementada de 1.

- Quando se chega ao vértice final de uma aresta, o valor **QI** desse vértice é o valor inicial para as arestas que emanem desse vértice.



<u>*Contour line*</u>: é definida como uma aresta partilhada por um polígono *back-facing* com outro front-facing, ou um polígono *front-facing* isolado.

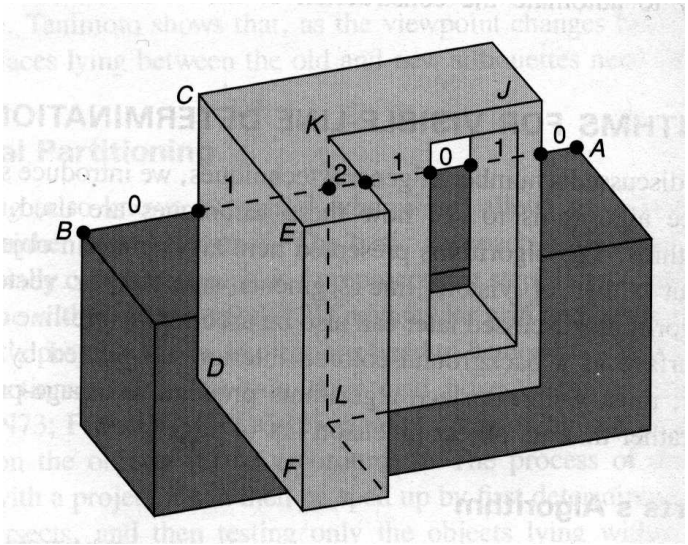
<u>*Contour lines*</u>: AB, CD, DF, KL

CE, EF, JK não são (porque são partilhadas por polígonos *front-facing*)

Síntese de Imagem
- Cálculo de Visibilidade – Alg. espaço objecto

O **QI** é alterado quando a aresta passa por trás de uma *contour line*.

Na figura os pontos indicam as intersecções da projecção da aresta AB com as projecções das *contour lines*.



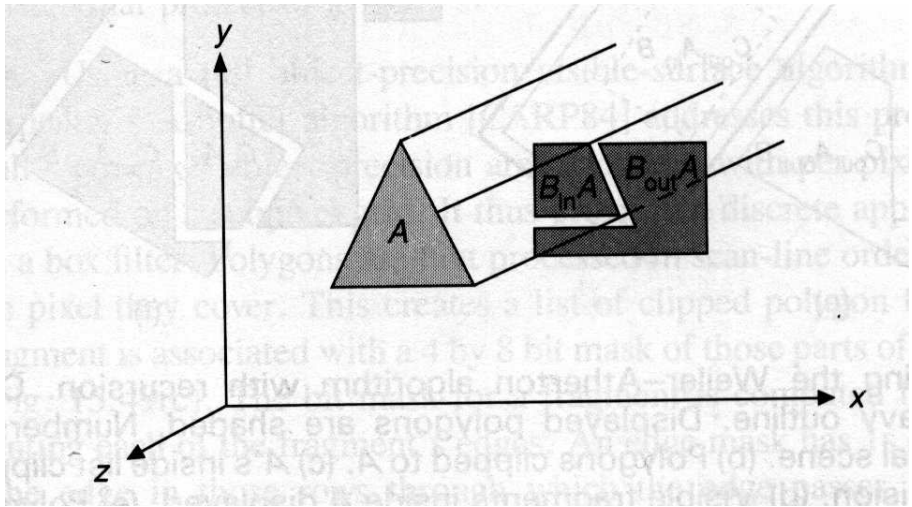
No final apenas os segmentos com valor **QI** igual a **zero** são visíveis.

Síntese de Imagem
- Cálculo de Visibilidade – Alg. espaço objecto

Determinação de faces visíveis: **Atherton & Weiller (1977)**

Algoritmo orientado à área como o de Warnock, mas subdivide a área de ecrã pela fronteira dos polígonos em vez de áreas rectangulares.

Requisito: exige a aplicação de um algoritmo sofisticado de *clipping* de polígonos, capaz de efectuar *clipping* de um polígono concavo com buracos contra um outro qualquer.



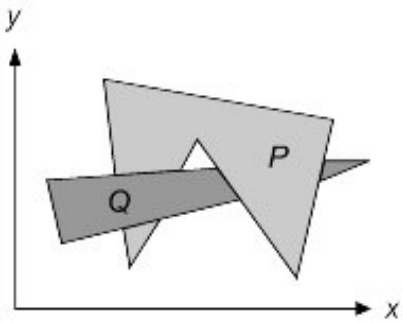
Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço objecto - **Atherton & Weiller**

Procedimento:

Ordenar os polígonos pela coordenada **Z**

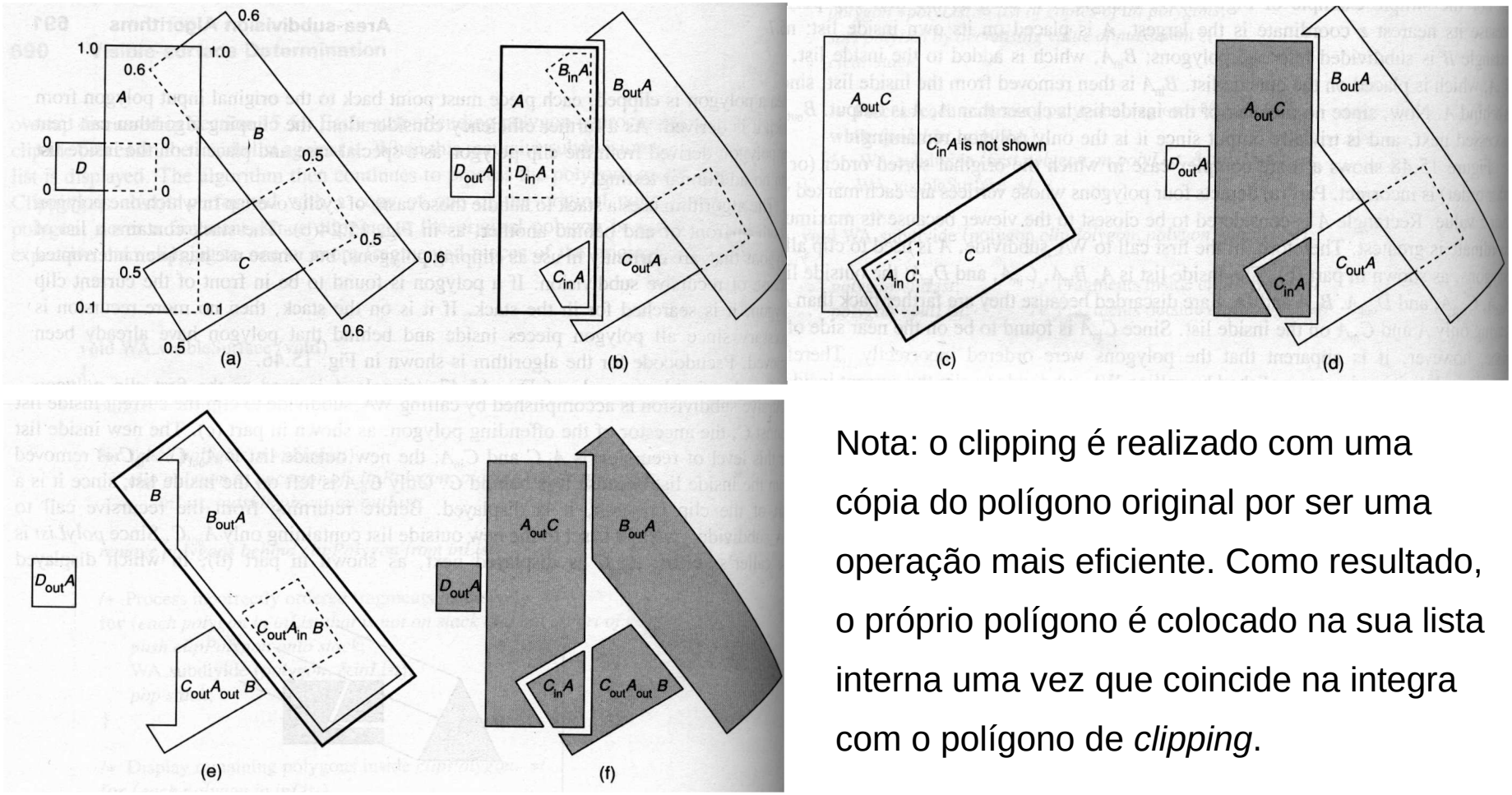
1. O polígono mais próximo do observador é usado para efectuar *clipping* dos restantes, resultando duas listas contendo os polígonos (ou parte deles) que estão dentro e fora da região de *clipping*.
2. Os polígonos da lista interior mais distantes que o actual são marcados como invisíveis.
3. Nos casos de mútua oclusão, como na figura abaixo, a lista interna vai conter polígonos que estão a ocultar o actual. Estes são usados para efectuar clipping sobre o polígono inicial (chamada recursiva)

   Nestes casos é usada uma stack para que o programa não chame novamente o algoritmo com o mesmo polígono que originou inicialmente a chamada recursiva.
4. O polígono é desenhado antes de retornar. A lista exterior desta fase que contém apenas a parte exterior do polígono inicial, retorna como lista interna. Assim ao retornar a parte visível do polígono inicial é desenhada.
5. Volta a 2, para processar a lista de polígonos exteriores.



Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço objecto - **Atherton & Weiller**

Na figura os valores indicam a coordenada Z de cada vértice.



Nota: o clipping é realizado com uma cópia do polígono original por ser uma operação mais eficiente. Como resultado, o próprio polígono é colocado na sua lista interna uma vez que coincide na integra com o polígono de *clipping*.

Síntese de Imagem
- Cálculo de Visibilidade – Alg. espaço imagem

1. Orientada à área: **Algoritmo de Warnock** (1969)
2. Orientado à linha: **Linha de Varrimento**
3. Orientado ao pixel: **Z-buffer, Ray Casting**

Algoritmo de Warnock

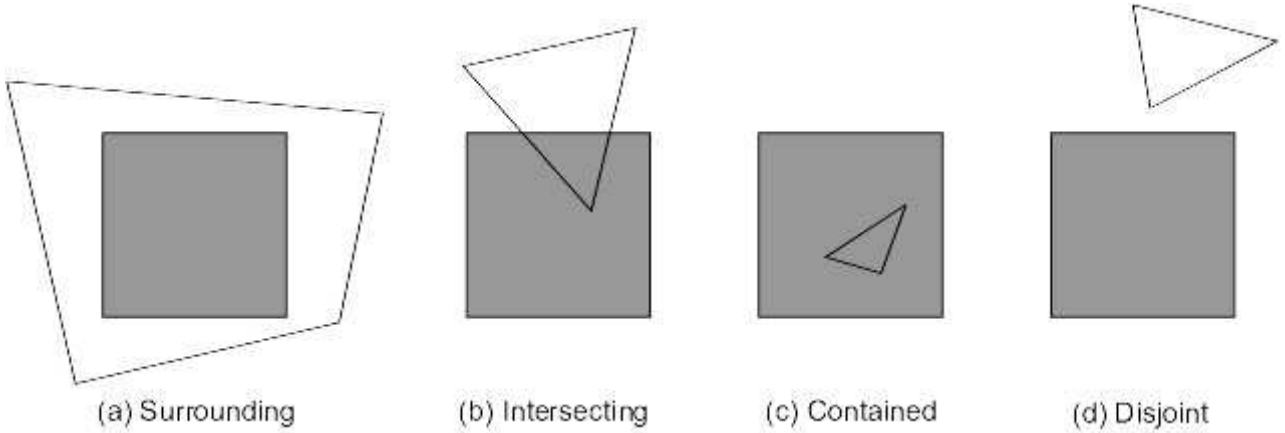
- O algoritmo divide sucessivamente a imagem projectada em áreas rectangulares.
- Se for fácil decidir quais os polígonos visíveis na área, então estes são mostrados; senão, a área é dividida em áreas mais pequenas à qual a avaliação é aplicada recursivamente.
- Quanto menores forem as áreas menor número de polígonos estarão sobrepostos nessas áreas e mais facilmente se poderá decidir qual o polígono a desenhar.

O algoritmo utiliza o conceito de coerência de área: um grupo de pixels adjacentes é habitualmente coberto pela mesma face visível.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: alg. de Warnock

Procedimento alg. Warnock:

- Divisão da área em 4 blocos iguais. Em cada fase da subdivisão, a projecção de cada polígono terá uma das 4 situações em relação a cada área:

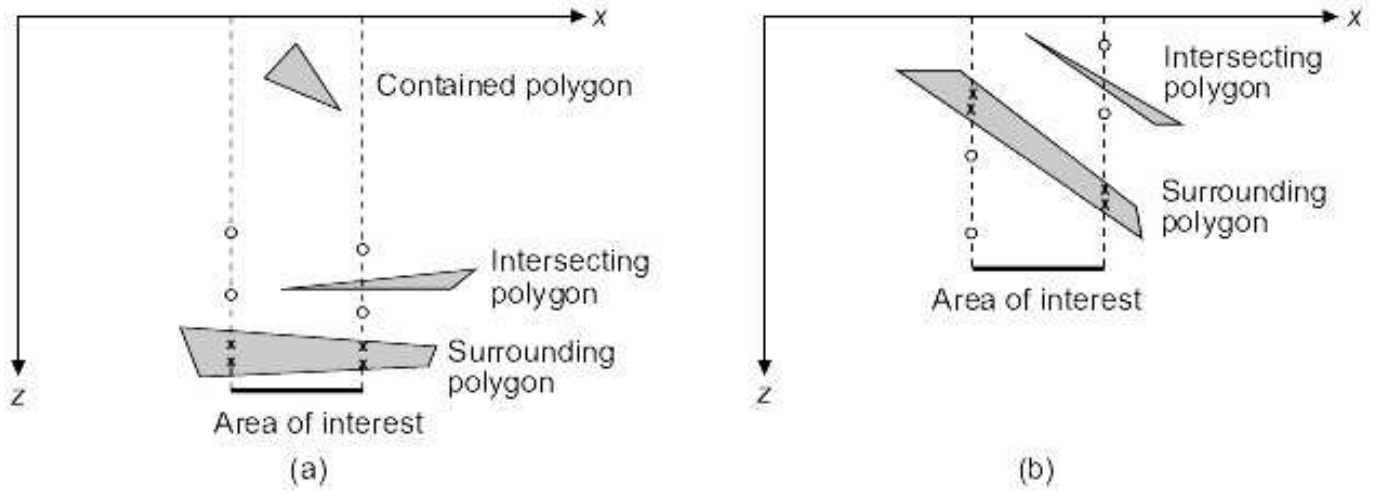


As quatro situações em que a decisão é possível, não havendo mais subdivisão:

1. Todos os polígonos estão fora da área → Pintar a área à cor de fundo.
2. Apenas um polígono que intersecta ou que está totalmente dentro da área (caso b. e c.). Preencher a área com a cor de fundo e depois pintar a parte do polígono que se sobrepõe nessa área.
3. Caso a), i.e. apenas um polígono que ocupa toda a área, não havendo mais nenhum projectado nessa área. Pintar a área à cor do polígono.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: alg. de Warnock

4. Mais do que um polígono intersecta, está contido ou rodeia a área, mas um rodeia a área e está à frente de todos os outros. Pintar a área com a cor deste último. Caso a) na figura abaixo.



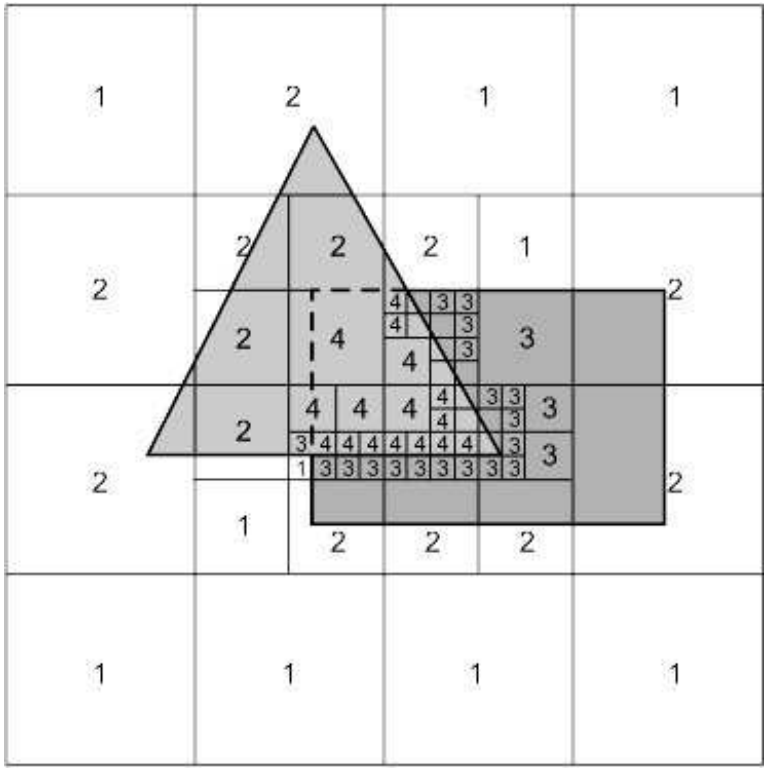
No caso b) a área é subdividida, resultando em cada caso uma das 4 situações anteriores.

E se não for verificada qualquer uma das 4 situações ?

A divisão é feita recursivamente, não descendo abaixo da dimensão do pixel. Neste caso limite, verifica-se qual o polígono mais próximo e pinta-se com a cor correspondente.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: alg. de Warnock

Exemplo da aplicação do algoritmo. Os valores indicam qual a situação verificada para cada área. Área sem número indica que não foi verificada qualquer uma das condições.



Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: linha de varrimento

Algoritmo de Linha de Varrimento

A imagem é criada linha a linha, à semelhança do algoritmo de preenchimento de regiões 2D, designado por **algoritmo da lista das arestas activas**.

Utiliza o conceito de **coerência de linha de varrimento** ou **coerência vertical**: o conjunto de objectos visíveis determinados para uma linha de varrimento numa imagem, difere pouco do conjunto da linha anterior.

E de coerência de aresta: uma aresta só altera a sua visibilidade onde se cruza por trás de uma aresta visível ou quando penetra uma face.

Estruturas de dados utilizadas: Tabela de (novas) Arestas **(ET)**, Tabela de Polígonos **(PT)** e Lista de Arestas Activas **(AEL)**.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: linha de varrimento

(nota: descritores com mais informação do que consta no livro, a vermelho)

Tabela de Arestas (ET): guarda informação de todas as arestas cuja projecção no plano de visualização não é horizontal. As entradas da tabela estão ordenadas de forma crescente pelo menor valor de Y, e contêm inicialmente:

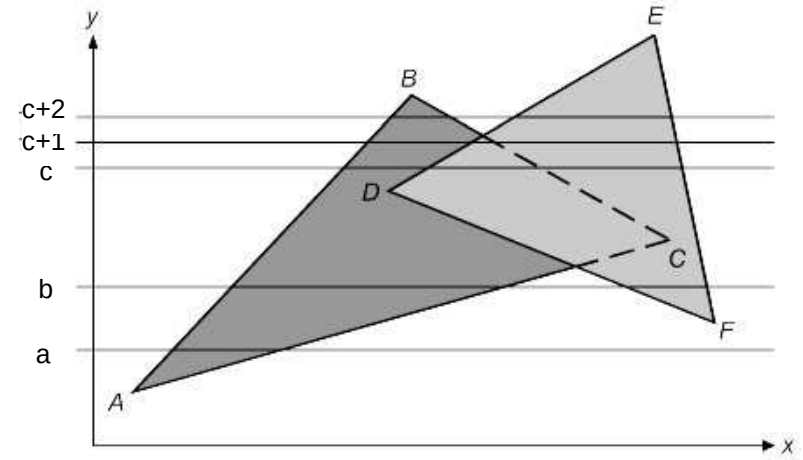
1. Coordenadas (X, Z) do vértice com menor Y
2. Coordenada Y do outro extremo da aresta (ou a altura da aresta Y1-Y0)
3. Incrementos $\Delta X/\Delta Y = \partial X/\partial Y$, $\Delta Z/\Delta Y = \partial Z/\partial Y$, usados na actualização de X e de Z, na passagem para a linha de varrimento seguinte
4. Identificação do(s) polígono(s) a que pertence a aresta (apontador)

Tabela de Polígonos (PT): informação de todos os polígonos, contendo para cada um:

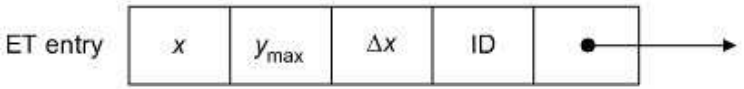
1. Coeficientes da equação do plano (no mínimo, $\Delta Z/\Delta X = \partial Z/\partial X$)
2. Informação da cor
3. Coordenada Z, a recalcular a cada pixel
4. Flag de *in-out*, iniciada a *False*, é usada para controlar se o processamento está dentro ou fora do polígono

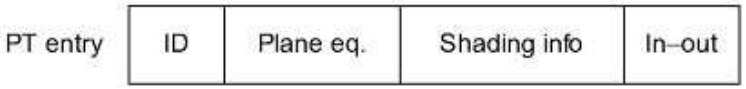
Lista de Arestas Activas (AEL): controla quais as arestas activas na linha de varrimento actual. Reduz o tempo de pesquisa para encontrar as arestas a processar na linha de varrimento actual.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: linha de varrimento



Quando se verifica a sobreposição de polígonos, como na linha ***c***, mais do que um polígono tem a flag ***in-out*** a ***true***. Utilizando a equação do plano de cada polígono, determina-se a coordenada **Z** de cada um para saber qual deles está mais próximo do observador. O problema fica assim reduzido a duas dimensões: X e Z





AET contents

Scan line	Entries			
a	*AB*	*AC*		
b	*AB*	*AC*	*FD*	*FE*
c, c+1	*AB*	*DE*	*CB*	*FE*
c+2	*AB*	*CB*	*DE*	*FE*

Para acelerar o processamento é conveniente manter uma lista com os polígonos que estão com a flag ***in-out*** igual a ***TRUE***.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: linha de varrimento

- Actualização incremental de uma aresta à mudança de uma linha de varrimento:

$$\left.\begin{aligned} x &= x + \Delta y.\left(\frac{\partial x}{\partial y}\right) = x + \left(\frac{\partial x}{\partial y}\right) \\ z &= z + \Delta y.\left(\frac{\partial z}{\partial y}\right) = z + \left(\frac{\partial z}{\partial y}\right) \end{aligned}\right\}, \; visto\ \Delta y = 1$$

- Inicialização da profundidade de um polígono quando *in-out* muda para TRUE:

$$z_{poly} = z_{edge}$$

- Actualização de $\boldsymbol{Z_{poly}}$ ao fim de Δx pixels:

$$z = z + \Delta x.\left(\frac{\partial z}{\partial x}\right)$$

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: Z-Buffer

Algoritmo Z-Buffer

Um dos algoritmos mais simples de implementar quer em hardware quer em software. Não exige qualquer pré-processamento de ordenação nem efectua comparação objecto-objecto.

Requisitos: Um *frame buffer* para a imagem final e um segundo *buffer* para guardar o valor Z correspondente a cada pixel, designado de Z-Buffer.

Procedimento:

1. Preencher com zeros o *Z-Buffer* e o *frame buffer* com a cor de fundo (background). O maior valor de Z em *Z-Buffer* será o correspondente ao plano frontal de *clipping*.
2. Percorrer cada polígono (Scan-convert), por qualquer ordem.
3. Se o ponto (x,y) actual do polígono em processamento, estiver mais próximo do observador do que o ponto actual do Z-Buffer, então este ponto substitui o anterior.

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: Z-Buffer

Algoritmo Z-Buffer

```
/* inicializa buffers*/
for(y=0; y<YMAX; y++)
     for(x=0; x<XMAX; x++)
     {    bufferIm(x,y) = BACKGROUND_COLOR;
          bufferZ(x,y) = 0;
     }
for (cada polígono)
{    for (cada pixel na projecção do polígono)
     {    /* calcula Z para (x,y) no poligono*/
          pz = poligonoZ(x,y);
          if (pz > bufferZ(x,y))
          {    bufferZ(x,y) = pz;
               bufferIm(x,y) = cor_poligono();
          }
     }
  }
```

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: Z-Buffer

Optimização no processamento:

O valor *z* do ponto *(x+ 1, y)* no polígono pode ser obtido a partir do valor *z* em *(x,y)* se atendermos que o polígono é plano.

Para obter *z* temos de resolver a equação *Ax+By+Cz+D=0* em ordem a *z*:

$$z = \frac{-D - Ax - By}{C}$$

Se em *(x,y)* a equação tem o valor z_1 então em *(x+ 1, y)* o valor de *z* pode ser calculado:

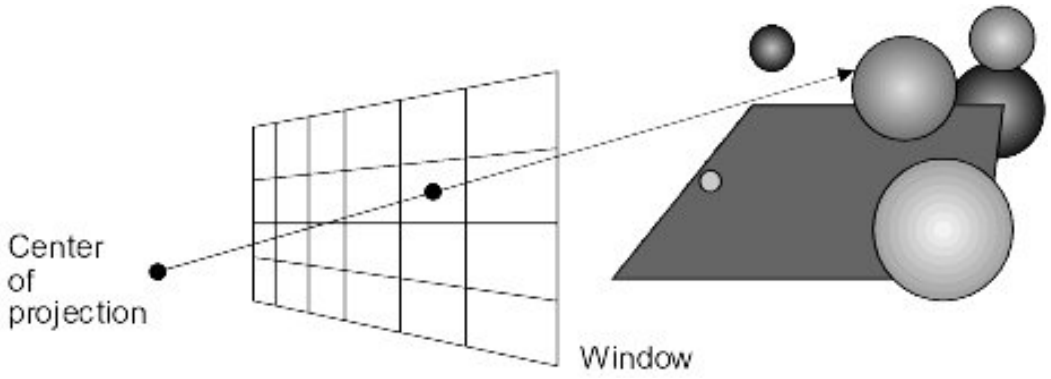
$$z = z_1 + \Delta x . \frac{\partial z}{\partial x} = z_1 + 1 . \frac{\partial z}{\partial x}$$

$$z = z_1 - \frac{A}{C}$$

Síntese de Imagem - Cálculo de Visibilidade
Alg. espaço imagem: Ray-casting

Algoritmo Ray-casting

A superfície visível em cada pixel da imagem é determinada traçando um raio de luz imaginário a partir do centro de projecção (observador), passando pelo centro do pixel para a cena 3D. A cor em cada pixel é definido pela intersecção com o objecto mais próximo.




```
Definir Centro de Projecção e window no plano de visualização
for(cada linha da imagem)
       for(cada pixel da linha)
       {      Definir o raio que a partir do centro de projecção passa no pixel
              for (cada objecto na cena)
              {      if ((objecto intersectado) e (mais próximo do que registo anterior))
                         registar intercepção e referência do objecto
              }
              atribuir ao pixel a cor da intercepção mais próxima
       }
```

Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades

Algoritmos tipo Lista de Prioridades

- Alg. Newel, Newel & Sancha
- Binary Space-Partitioning Trees

Objectivo: determinar a ordem de visibilidade para os objectos (polígonos), assegurando assim que a imagem será correctamente criada se os objectos forem desenhados por certa ordem:

1. pintar as faces mais afastadas do observador em primeiro lugar
2. à medida que outras, mais próximas, vão sendo pintadas, ocultam as anteriores.

(Algoritmo do “Pintor”)

Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades - **Alg. Newel, Newel & Sancha**

Alg. Newel, Newel & Sancha (Depth-sort algorithm)

Procedimento: pintar os polígonos por ordem decrescente da distância ao observador. Para isso são realizados 3 passos:

1. Ordenar os polígonos por ordem crescente de ***z*** (dos mais afastados para os mais próximos)

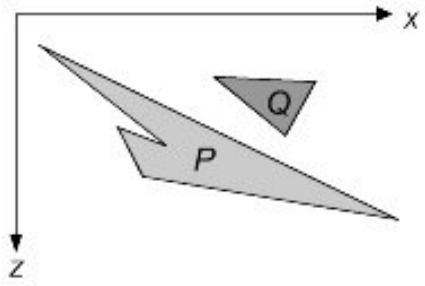
2. Resolver qualquer ambiguidade na ordenação, nomeadamente se houver sobreposição de polígonos na coordenada ***z***. Poderá ser necessário dividir polígonos.

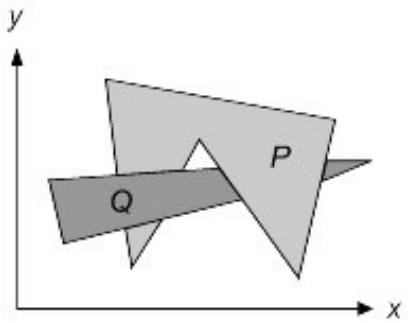
3. Pintar os polígonos por ordem do mais afastado para o mais próximo.

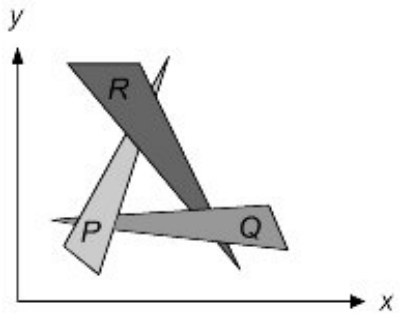
Síntese de Imagem - Cálculo de Visibilidade

Alg. tipo lista de prioridades - - **Alg. Newel, Newel & Sancha**

Tipo de ambiguidades que podem surgir na ordenação dos polígonos:

a) 

b) 

c) 

Pré-processamento:

Ordenar os polígonos pela coordenada ***Z do vértice mais afastado***

Processamento:

Para o último polígono **P** da lista, verificar se existe algum polígono **Q** cujo maior ***Z*** seja mais afastado do que o menor ***Z*** de **P**, e que esteja a ser obstruído por **P**. Se não estiver, então **P** pode ser desenhado.

São efectuados até **5 testes** para cada polígono **Q** nas condições anteriores. Se um deles for verificado então **P** pode ser desenhado antes de **Q**.

Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades - Binary Space-Partitioning Trees

Binary Space-Partitioning Trees

Eficaz para cálculo de visibilidade em cenas estáticas onde há **variação do ponto de visão**.

Principio de funcionamento: se podermos definir um plano que separe dois conjuntos de polígonos (**cluster1** e **cluster2)**, então podemos dizer que o **cluster1,** que está do mesmo lado do observador, pode obstruir, mas não ser obstruído, pelo **cluster2** do outro lado do plano. Se este último for desenhado primeiro, obtemos uma representação correcta da cena.

Cada **cluster** pode ser também dividido recursivamente, até chegarmos ao nível do polígono.

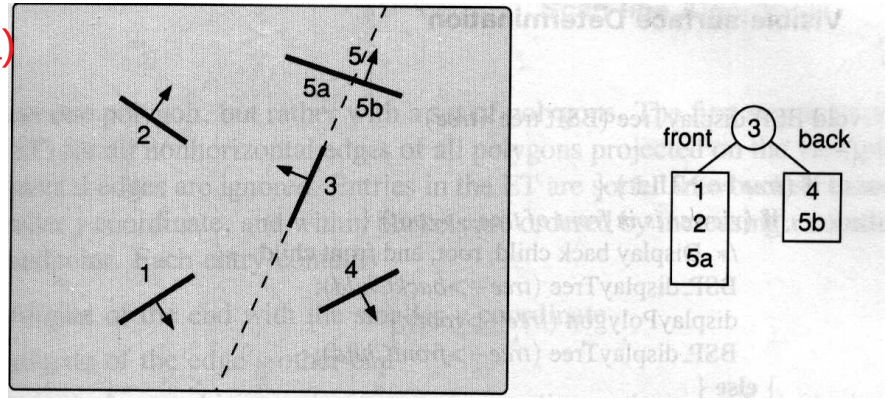
Algoritmo para construir a árvore binária:

1. Escolher um dos polígonos para raiz da árvore binária.
2. A raiz é usada para dividir o espaço em 2: um dos espaços contém os polígonos que estão à frente dele em relação à sua normal; a outra parte contém os que estão atrás.
3. Recursivamente, dividir cada um dos subespaços obtidos na iteração anterior.

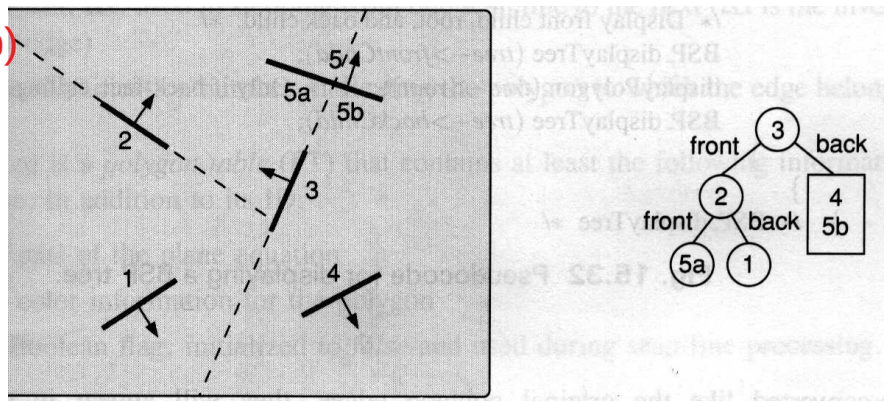
Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades - **Binary Space-Partitioning Trees**

Exemplo de criação de árvore para um conjunto de polígonos.

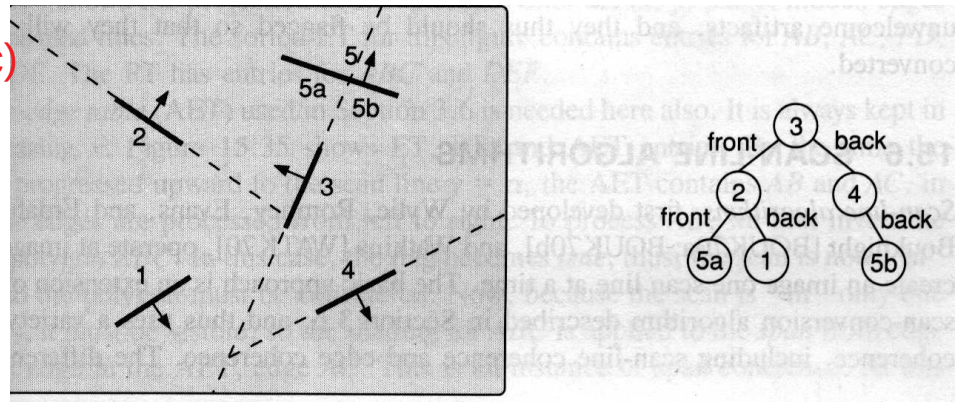
a)



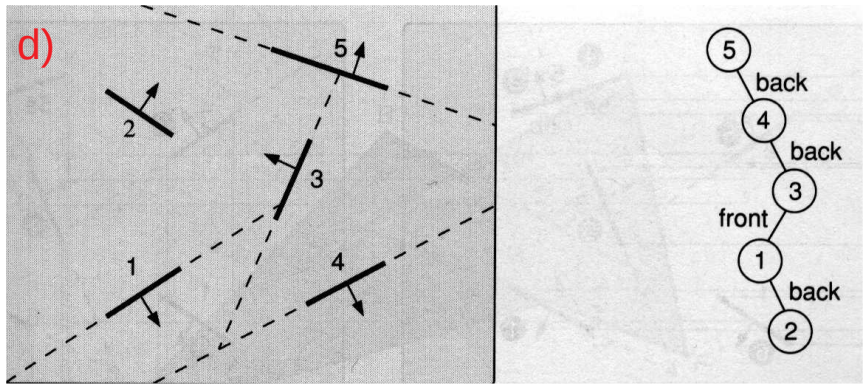
b)



c)



d)



d) Árvore obtida se o primeiro polígono for o 5

Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades - **Binary Space-Partitioning Trees**

Pseudo-código para obter a lista de polígonos correcta, dado o ponto de observação:

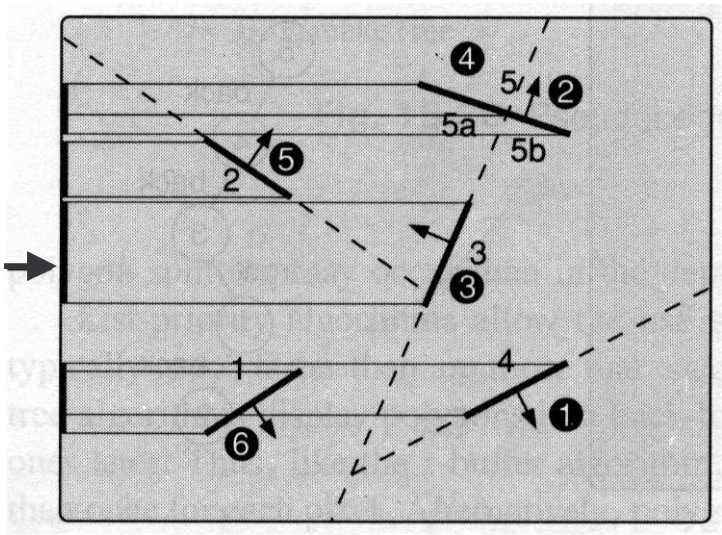
```
void BSP_displayTree(BSP_tree *tree)
     if (tree != null) {
         if (viewer is in front of tree->root){
            /* display back child, root, and front child */
            BSP_displayTree(tree->backChild);
            displayTree(tree->root);
            BSP_displayTree(tree->frontChild);
         } else {
            /* display front child, root, and back child */
            BSP_displayTree(tree->frontChild);
            displayTree(tree->root);   /* se back-face culling off */
            BSP_displayTree(tree->backChild);
         }
     }
}
```

Nota: Se o *backface culling* estiver activo, os polígonos que não estão voltados para o observador não são desenhados.

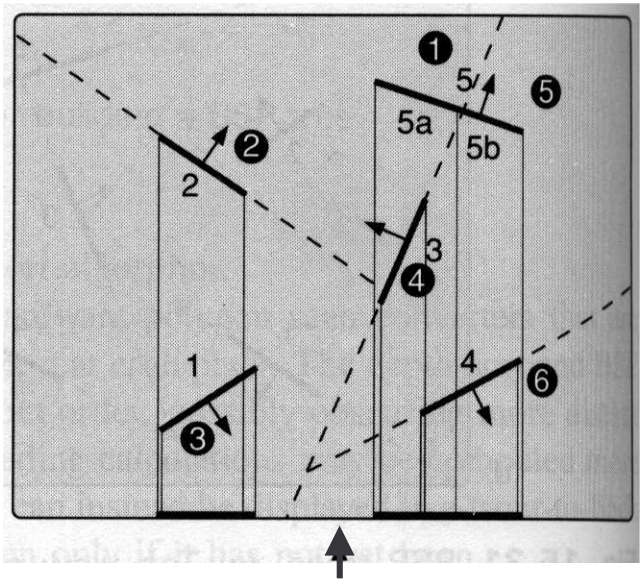
Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades - **Binary Space-Partitioning Trees**

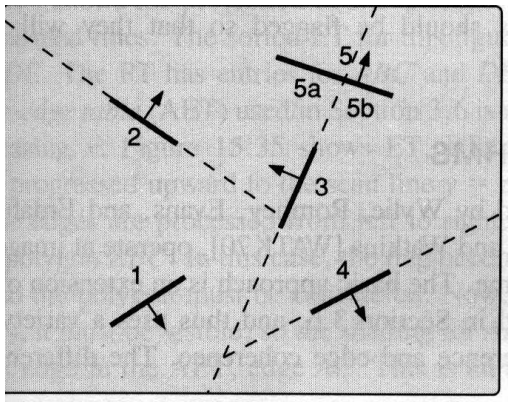
Resultado da consulta à árvore para dois pontos de observação distintos:

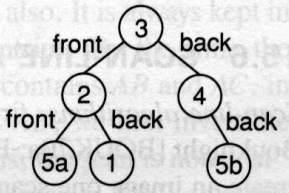
a)



b)







003

Síntese de Imagem - Cálculo de Visibilidade
Alg. tipo lista de prioridades - **Binary Space-Partitioning Trees**

Observação: este algoritmo pode auxiliar na operação de clipping. Qualquer polígono cujo plano não intersecte o volume de visualização (VV), tem uma subárvore que está totalmente fora do VV, não sendo por isso pesquisada.

Escolha do nó *root* de cada subárvore: o algoritmo funciona qualquer que seja o polígono, no entanto o melhor será aquele que origine um menor número de divisões dos restantes polígonos.

Heurística: testar aleatoriamente alguns polígonos (ex: 6) e escolher para *root* o que originar menor número de divisões.