Computação Gráfica

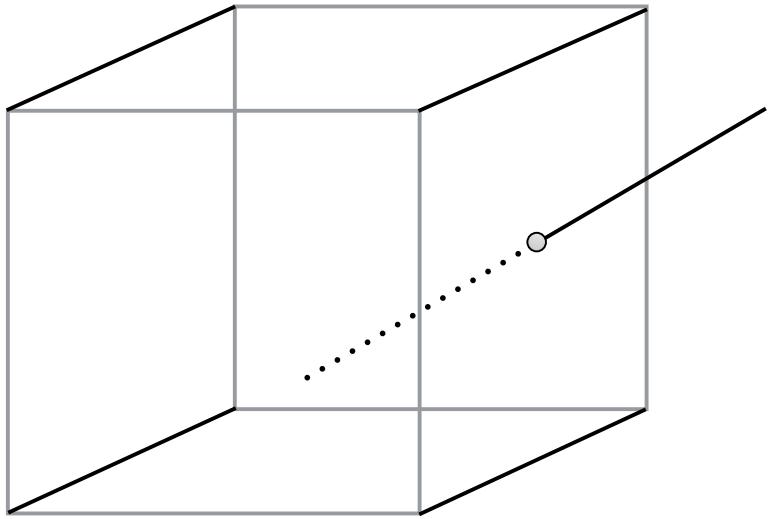
TCC-00291

Assunto: Clipping

Introdução

Próximas etapas

O que acontece com primitivas geométrica que estão fora do volume de visão?

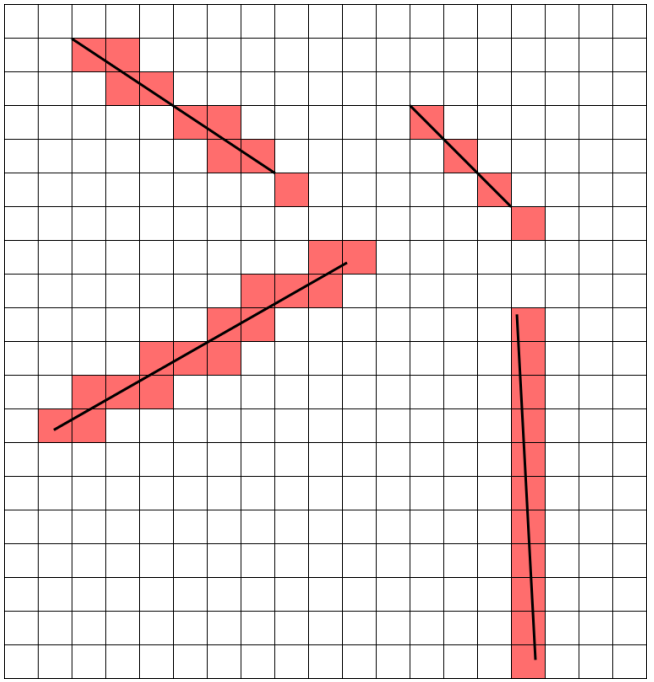


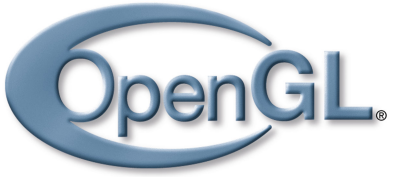
Introdução

Próximas etapas

O que acontece com primitivas geométrica que estão fora do volume de visão?

Como linhas são desenhadas?





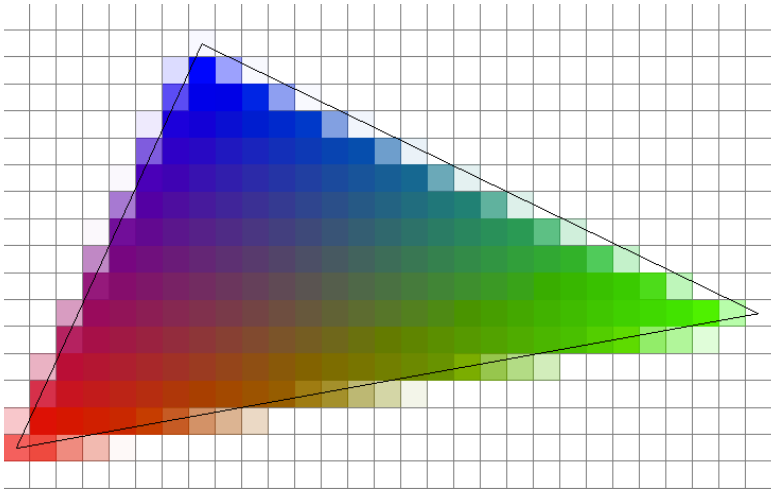
Introdução

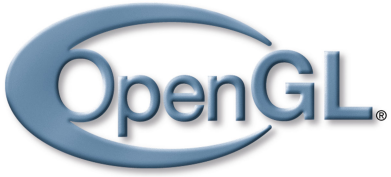
Próximas etapas

O que acontece com primitivas geométrica que estão fora do volume de visão?

Como linhas são desenhadas?

Como polígonos são preenchidos?



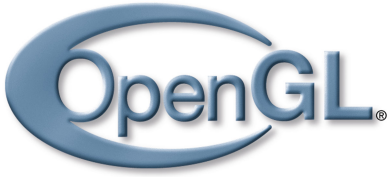


Introdução

Próximas etapas

O que acontece com primitivas geométrica que estão fora do volume de visão?

Estudaremos hoje os algoritmos de Clipping.



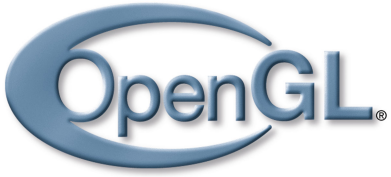
Introdução

Próximas etapas

O que acontece com primitivas geométrica que estão fora do volume de visão?

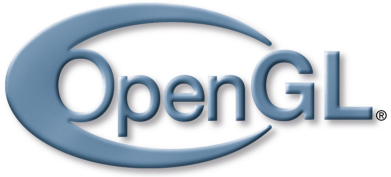
Estudaremos hoje os algoritmos de Clipping.

Obs: Concentraremos o nosso estudo em algoritmos de clipping de linhas e polígonos em 2D.



Clipping
Fundamentos

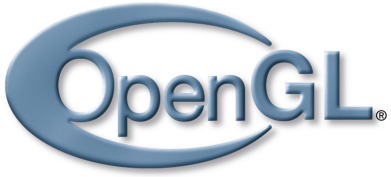
Quando especificamos um volume de visão queremos exibir somente os objetos que estejam no seu interior.



Clipping
Fundamentos

Quando especificamos um volume de visão queremos exibir somente os objetos que estejam no seu interior.

Naturalmente alguns objetos estarão parcialmente visíveis.

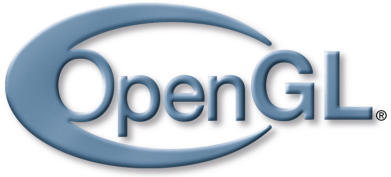


Clipping
Fundamentos

Quando especificamos um volume de visão queremos exibir somente os objetos que estejam no seu interior.

Naturalmente alguns objetos estarão parcialmente visíveis.

A operação que determina quais partes dos objetos estão visíveis é denominada clipping ou recorte.



Clipping

Segmento de reta x retângulo

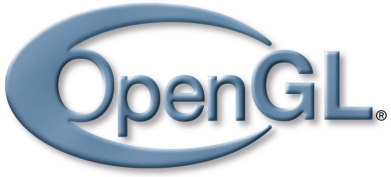
Entrada

Segmento de reta P1-P2

Retângulo de visão (xmin, ymin), (xmax, ymax)

Saída

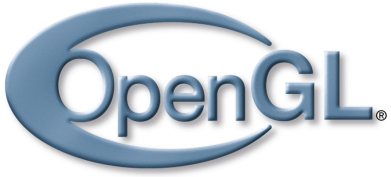
Segmento recortado (possivelmente nulo)



Clipping

Segmento de reta x retângulo

Estudaremos dois algoritmos clássicos:
Cohen-Sutherland e Liang-Barsky

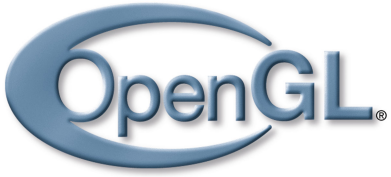


Clipping

Segmento de reta x retângulo

Vértices do segmento são classificados com relação a cada semi-espaço plano que delimita o retângulo.

$$x \geq xmin \quad e \quad x \leq xmax \; e \quad y \geq ymin \; e \; y \leq ymax$$



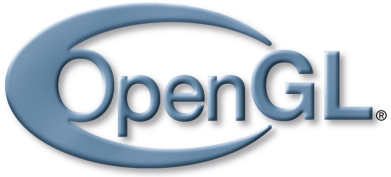
Clipping

Segmento de reta x retângulo

Vértices do segmento são classificados com relação a cada semi-espaço plano que delimita o retângulo.

$$x \geq xmin \;\; e \;\; x \leq xmax \; e \;\; y \geq ymin \; e \; y \leq ymax$$

Obs: Se ambos os vértices são classificados como "fora" em relação a um semi-espaço, descartar o segmento.

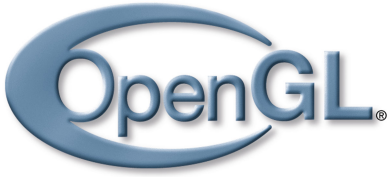


Clipping

Segmento de reta x retângulo

Se ambos são classificados como “dentro”:

Testar o próximo semi-espaço.



Clipping

Segmento de reta x retângulo

Se ambos são classificados como “dentro”:

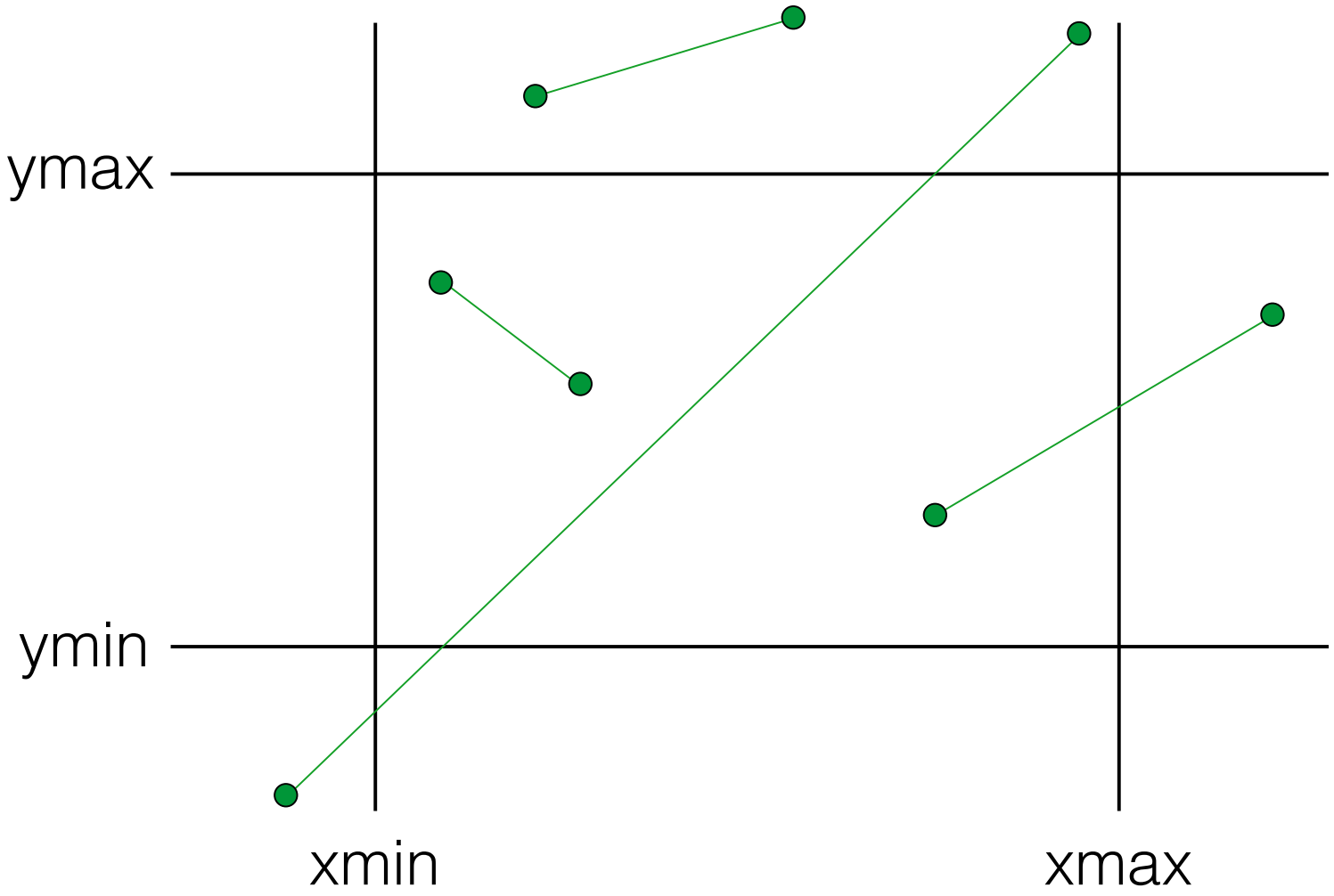
Testar o próximo semi-espaço.

Se um vértice está “dentro” e outro “fora”:

1) Computar o ponto de interseção Q
2) Continuar o algoritmo com o segmento recortado.

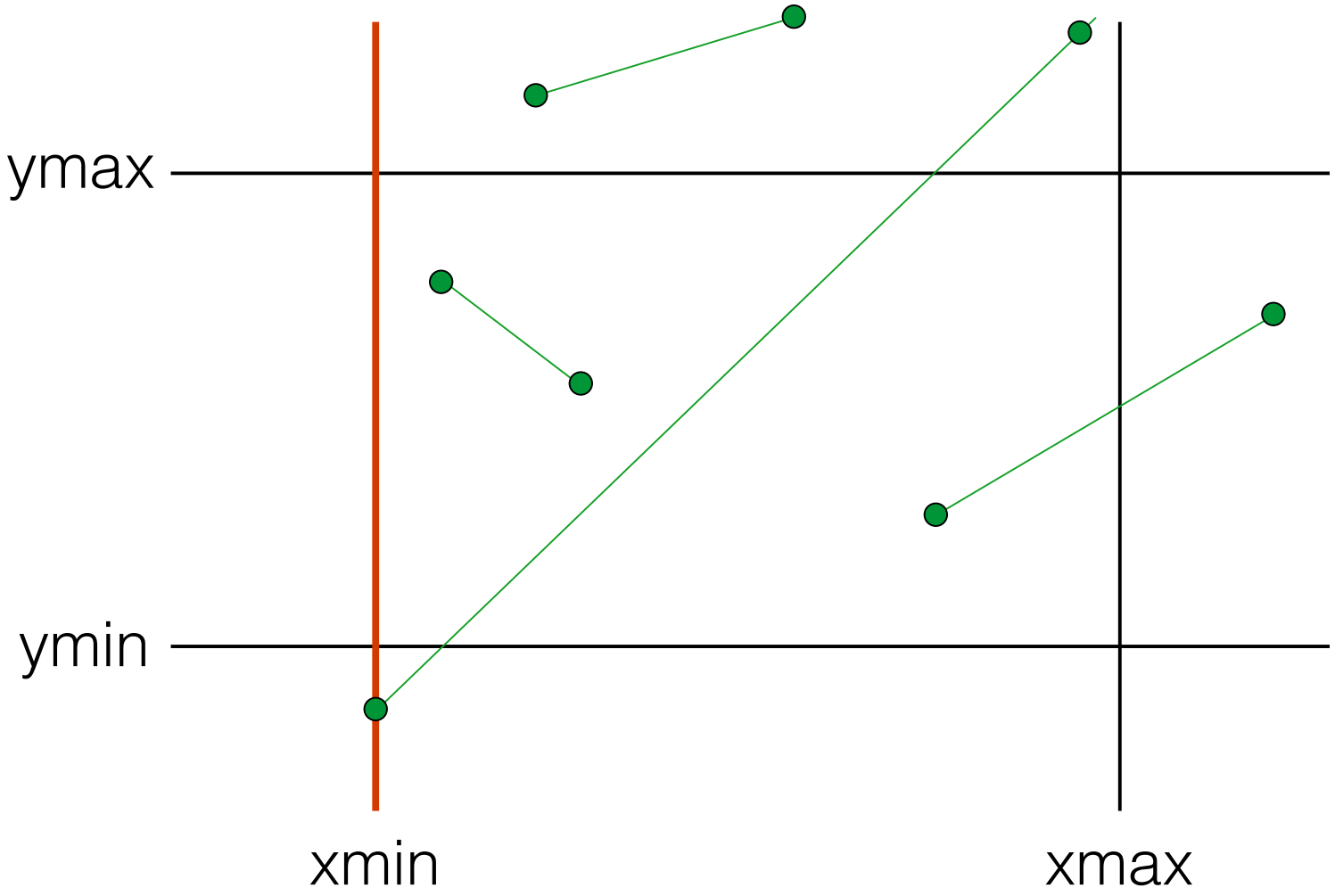
Clipping

Segmento de reta x retângulo



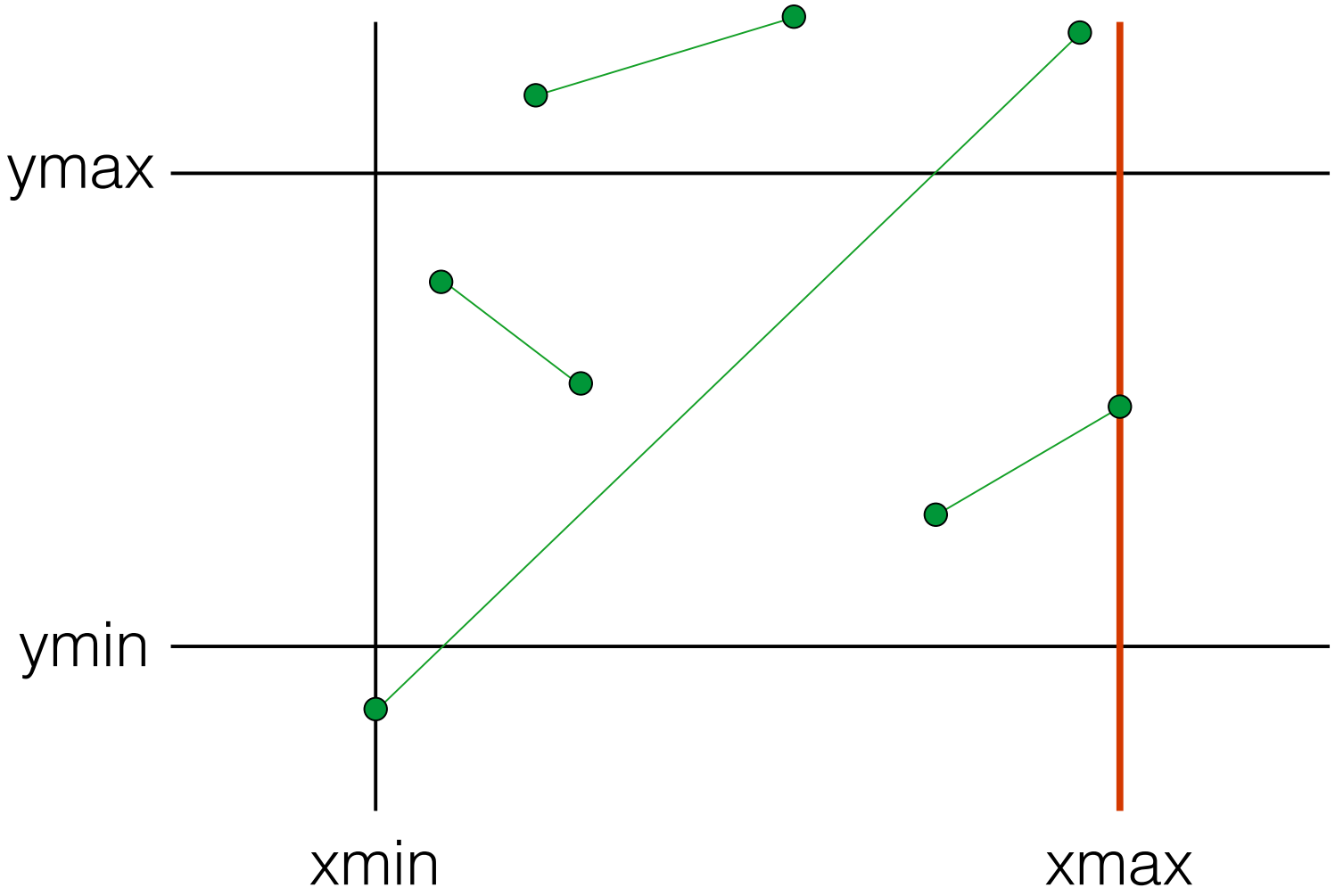
Clipping

Segmento de reta x retângulo



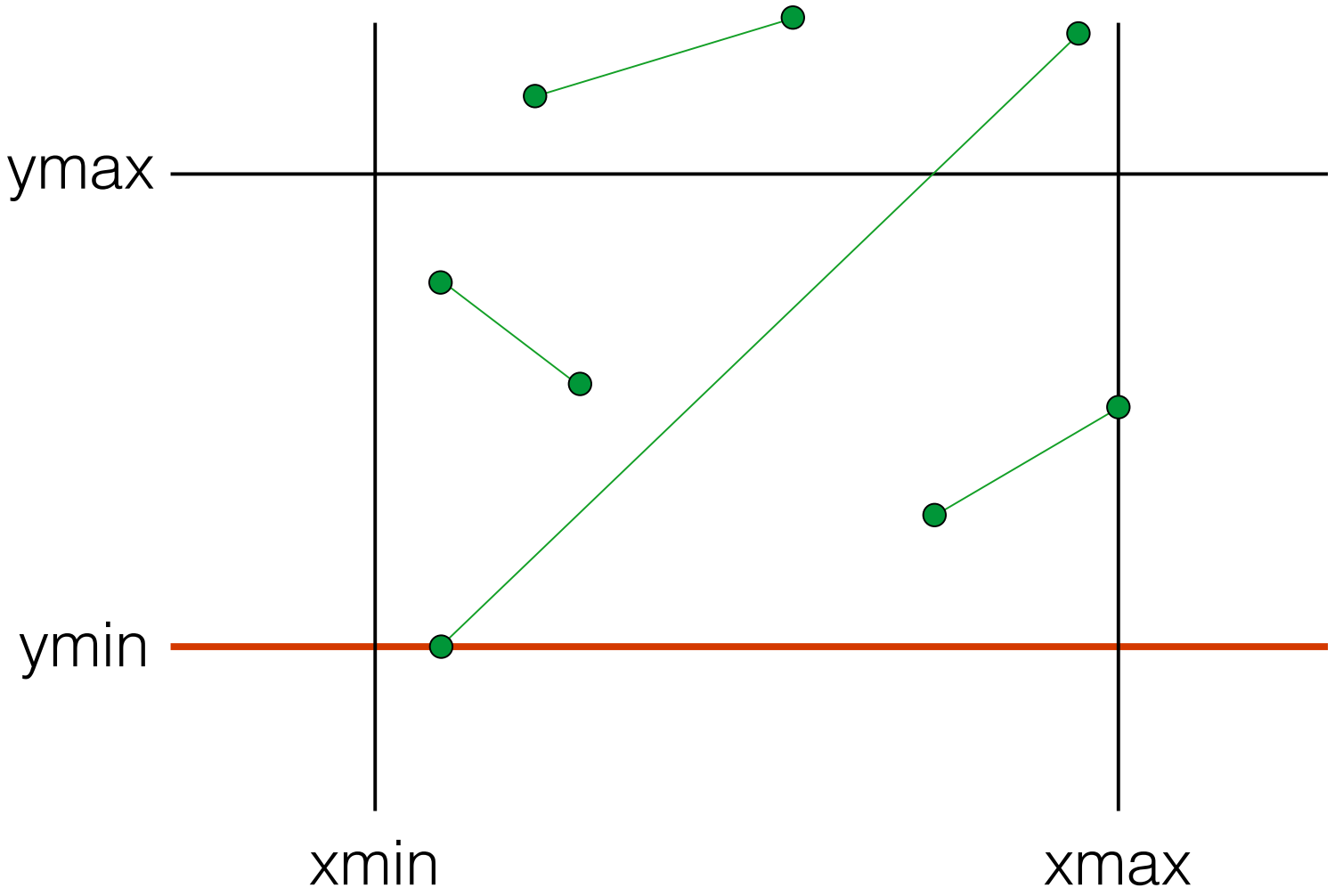
Clipping

Segmento de reta x retângulo



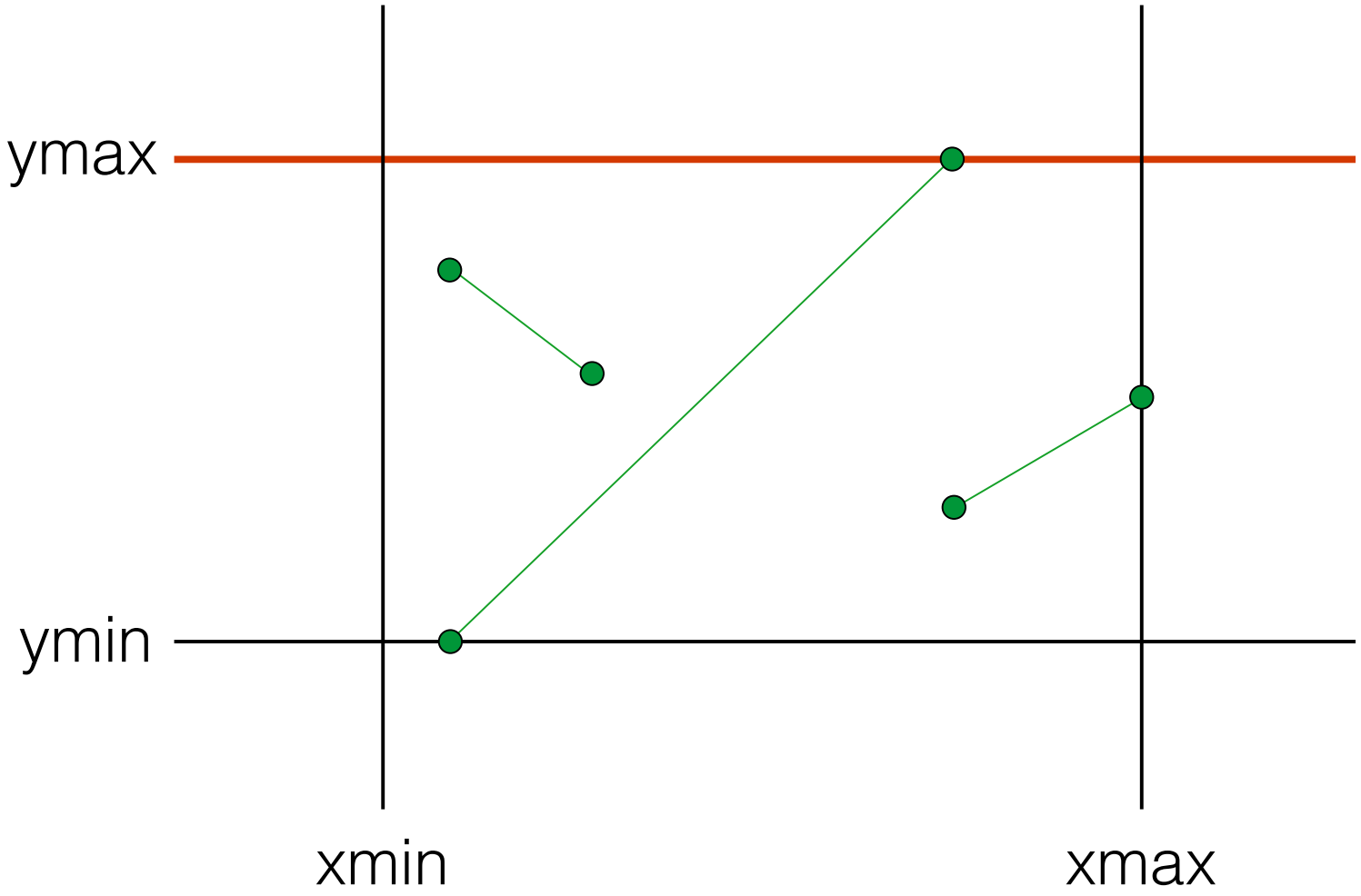
Clipping

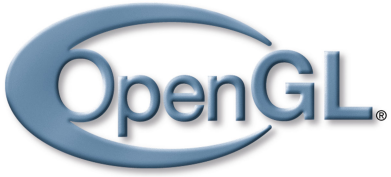
Segmento de reta x retângulo



Clipping

Segmento de reta x retângulo





Clipping

Segmento de reta x retângulo

O recorte só é necessário se houver um vértice dentro e outro fora em relação a um semi-plano.

Clipping

Segmento de reta x retângulo

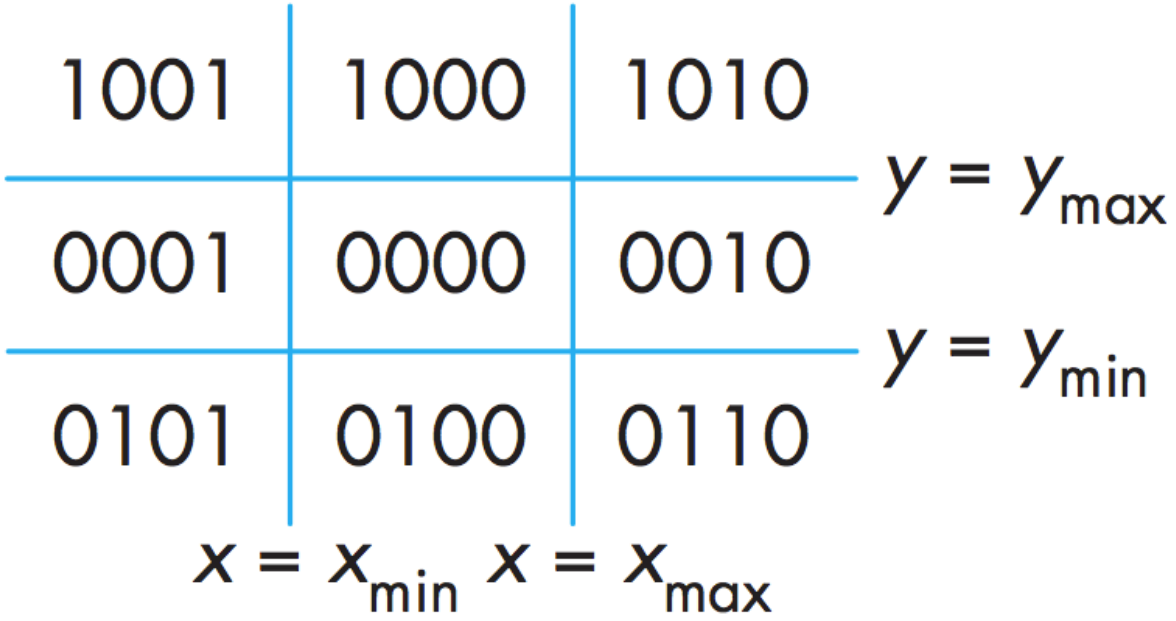
O recorte só é necessário se houver um vértice dentro e outro fora em relação a um semi-plano.

Classificação de cada vértice pode ser codificada usando 4 bits, um para cada semi-espaço.

Dentro = 0 e Fora = 1

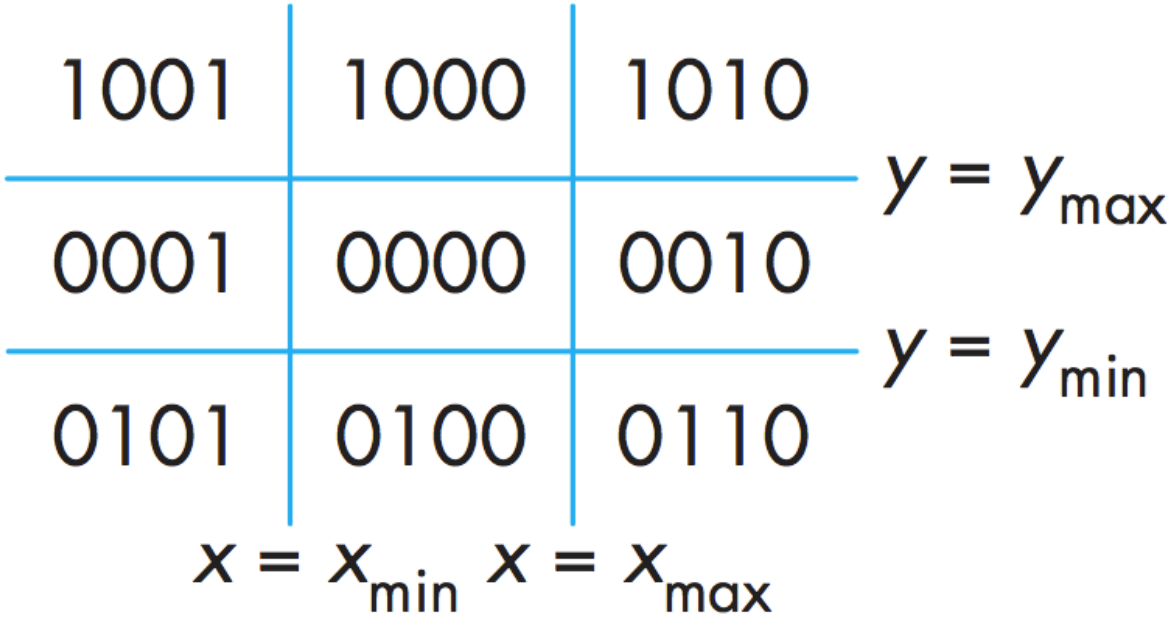
Clipping

Segmento de reta x retângulo



Clipping

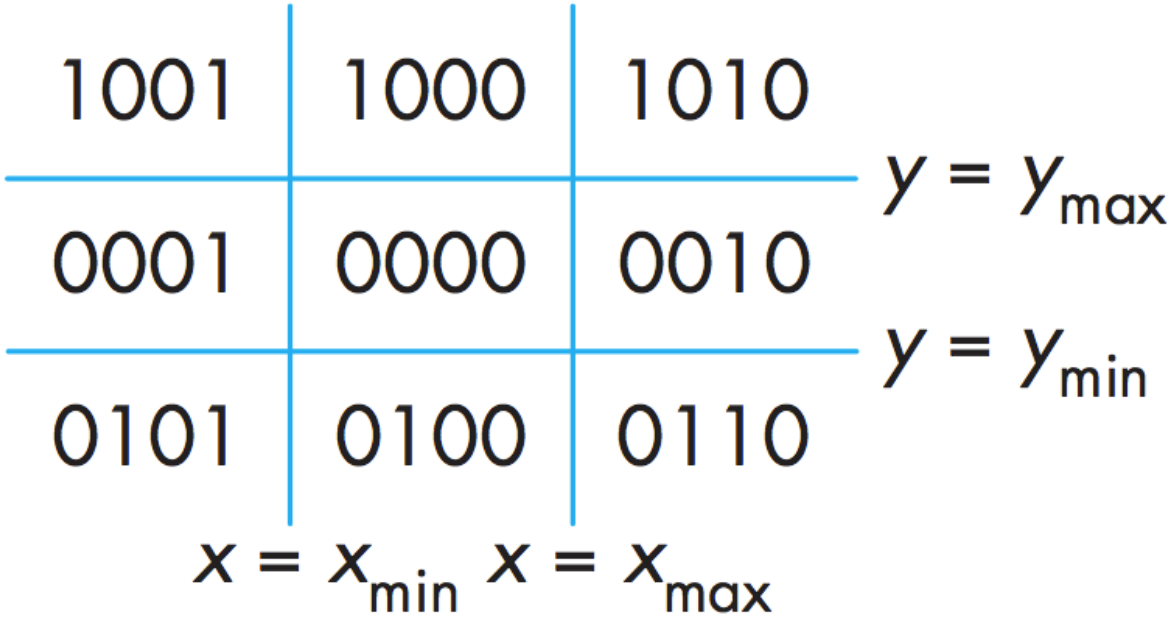
Segmento de reta x retângulo



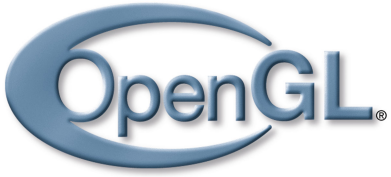
Rejeição trivial: Classifica(P1) and Classifica(P2) ≠ 0

Clipping

Segmento de reta x retângulo



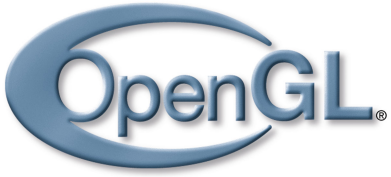
Aceitação trivial: Classifica(P1) or Classifica(P2) = 0



Clipping

Segmento de reta x retângulo

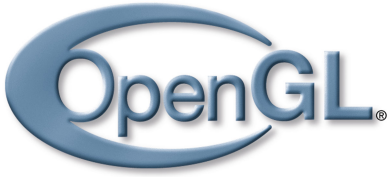
```
byte code(double    x, double    y,
          double xmin, double xmax, double ymin, double ymax)
{
   byte code=0;

   if (y > ymax) code += 8;
   if (y < ymin) code += 4;
   if (x > xmax) code += 2;
   if (x < xmin) code += 1;

   return code;
}
```



Clipping

Segmento de reta x retângulo

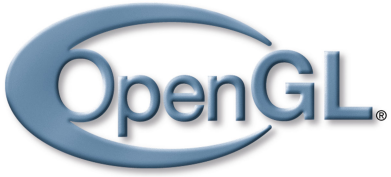
```
void CohenSutherlandLineClip(double   x0, double   y0, double   x1, double   y1,
                             double xmin, double xmax, double ymin, double ymax)
{
 byte outcode0, outcode1, outcodeOut;
 double x, y;  boolean accept = FALSE, done = FALSE;

 outcode0 = code(x0, y0, xmin, xmax, ymin, ymax);
 outcode1 = code(x1, y1, xmin, xmax, ymin, ymax);

 do {
   if ((outcode0 | outcode1) == 0) {
       accept = TRUE;  done = TRUE;                                  /* trivial draw and exit */
   } else if((outcode0 & outcode1) != 0) {
       done = TRUE;                                                  /* trivial reject and exit */
   } else {                                                          /* discart an out part */
     outcodeOut = (outcode0 != 0) ?  outcode0 : outcode1;            /* pick an out vertice */
     if (outcodeOut & 8) {                                         /* discart top */
       x = x0 + (x1 - x0) * (ymax - y0) / (y1 - y0);  y = ymax;
     } else if(outcodeOut & 4) {                                   /* discart bottom */
       x = x0 + (x1 - x0) * (ymin - y0) / (y1 - y0);  y = ymin;
     } else if(outcodeOut & 2) {                                   /* discart right */
       y = y0 + (y1 - y0) * (xmax - x0) / (x1 - x0);  x = xmax;
     } else if(outcodeOut & 1) {                                   /* discart left */
       y = y0 + (y1 - y0) * (xmin - x0) / (x1 - x0);  x = xmin;
     }

     if (outcodeOut == outcode0) {
        x0 = x; y0 = y; outcode0 = code(x0, y0, xmin, xmax, ymin, ymax);
     } else {
        x1 = x; y1 = y;  outcode1 = code(x1, y1, xmin, xmax, ymin, ymax);
     }
   }
  } while (!done);
   if (accept) DrawLineReal(x0, y0, x1, y1);
}
```



Clipping

Segmento de reta x retângulo

O algoritmo de Cohen-Sutherland calcula interseções desnecessárias, pois resolve o mesmo segmento várias vezes.

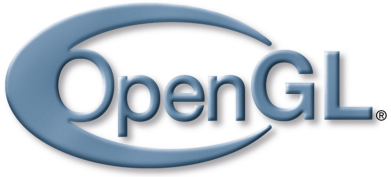


Clipping

Segmento de reta x retângulo

O algoritmo de Cohen-Sutherland calcula interseções desnecessárias, pois resolve o mesmo segmento várias vezes.

O algoritmo de Liang-Barsky utiliza uma estratégia que recorta com menor número de interseções.



Clipping

Segmento de reta x retângulo

Se baseia na equação paramétrica da reta.

$$P(t) = P_0 + t(P_1 - P_0)$$

Clipping

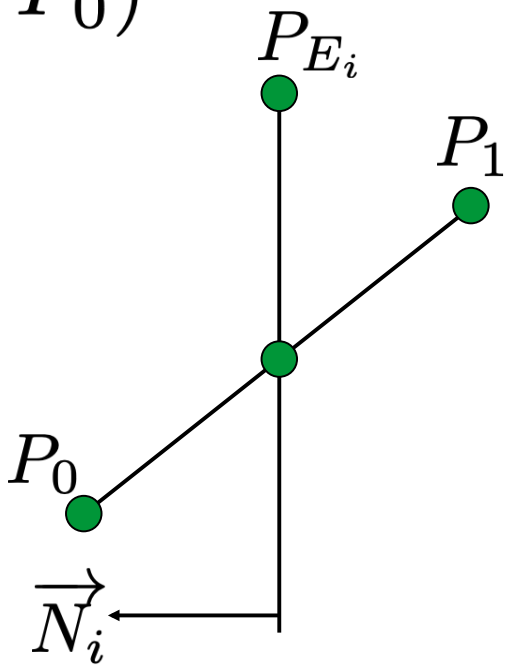
Segmento de reta x retângulo

Se baseia na equação paramétrica da reta.

$$P(t) = P_0 + t(P_1 - P_0)$$

Neste caso:

$$\left\langle \overrightarrow{N_i}, P(t) - P_{E_i} \right\rangle = 0$$



Clipping

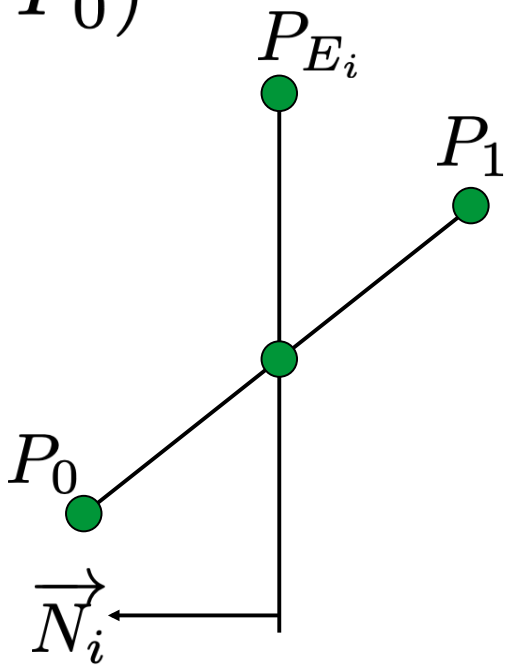
Segmento de reta x retângulo

Se baseia na equação paramétrica da reta.

$$P(t) = P_0 + t(P_1 - P_0)$$

Neste caso:

$$t = -\frac{\left\langle \overrightarrow{N_i}, P_0 - P_{E_i} \right\rangle}{\left\langle \overrightarrow{N_i}, P_1 - P_0 \right\rangle}$$



Clipping

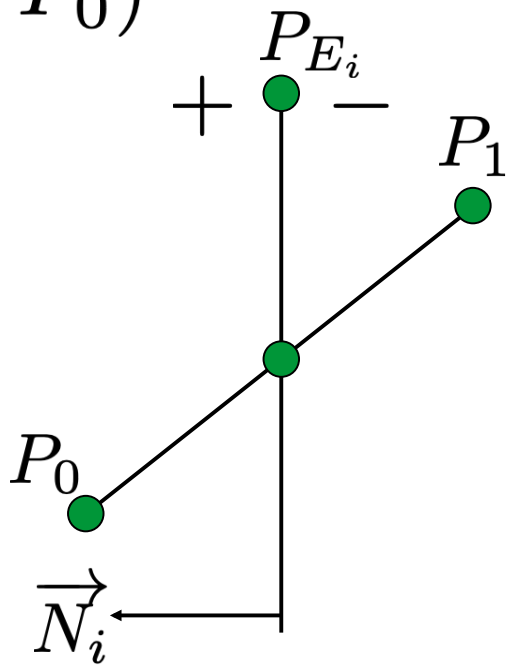
Segmento de reta x retângulo

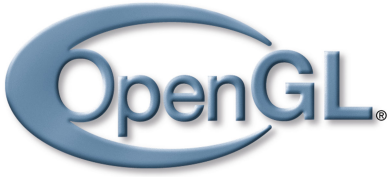
Se baseia na equação paramétrica da reta.

$$P(t) = P_0 + t(P_1 - P_0)$$

Neste caso:

$$t = -\frac{\left\langle \overrightarrow{N_i}, P_0 - P_{E_i} \right\rangle}{\left\langle \overrightarrow{N_i}, P_1 - P_0 \right\rangle}$$

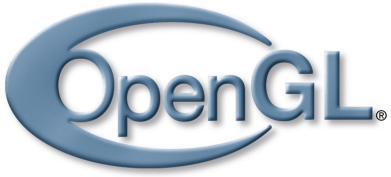




Clipping

Segmento de reta x retângulo

Dado um segmento, sua reta suporte intersecta as quatro retas laterais do retângulo em quatro pontos (caso geral).

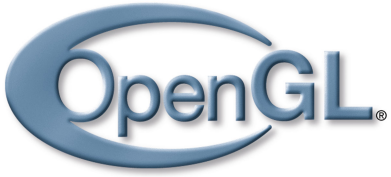


Clipping

Segmento de reta x retângulo

Dado um segmento, sua reta suporte intersecta as quatro retas laterais do retângulo em quatro pontos (caso geral).

Usando a equação paramétrica podemos calcular o parâmetro t no qual a reta intersecta uma das quatro retas laterais do retângulo de recorte.



Clipping

Segmento de reta x retângulo

Dado um segmento, sua reta suporte intersecta as quatro retas laterais do retângulo em quatro pontos (caso geral).

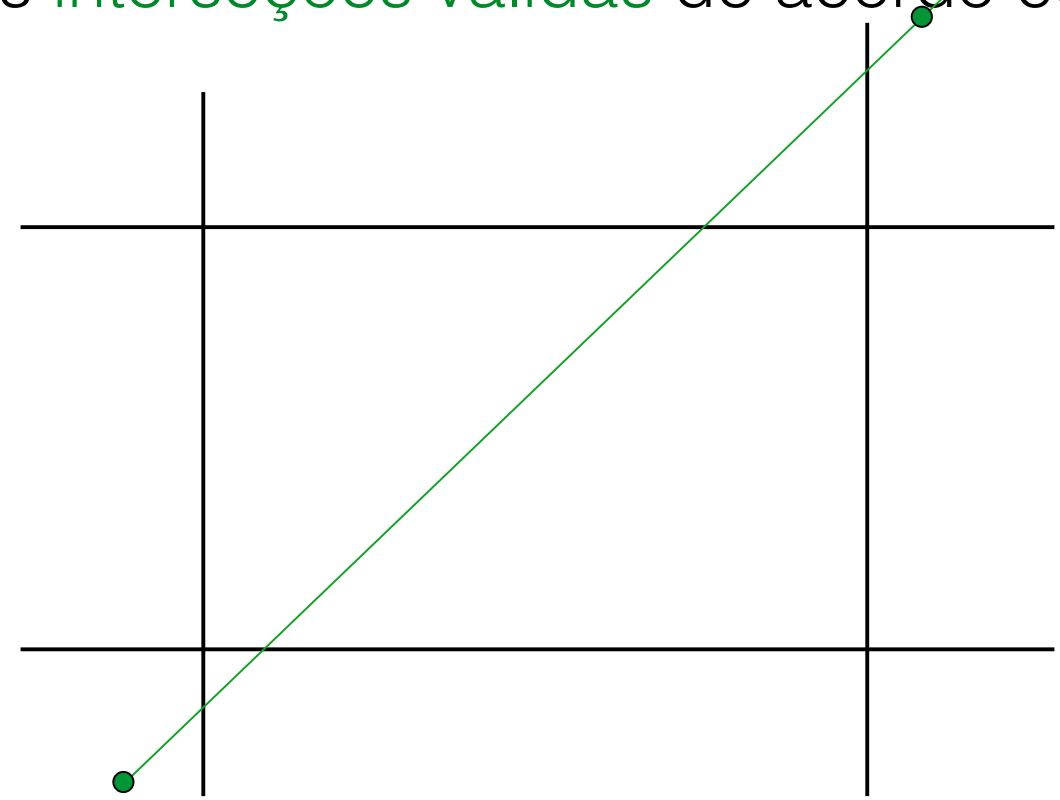
Usando a equação paramétrica podemos calcular o parâmetro t no qual a reta intersecta uma das quatro retas laterais do retângulo de recorte.

Descartamos os valores de t fora do intervalo [0,1].

Clipping

Segmento de reta x retângulo

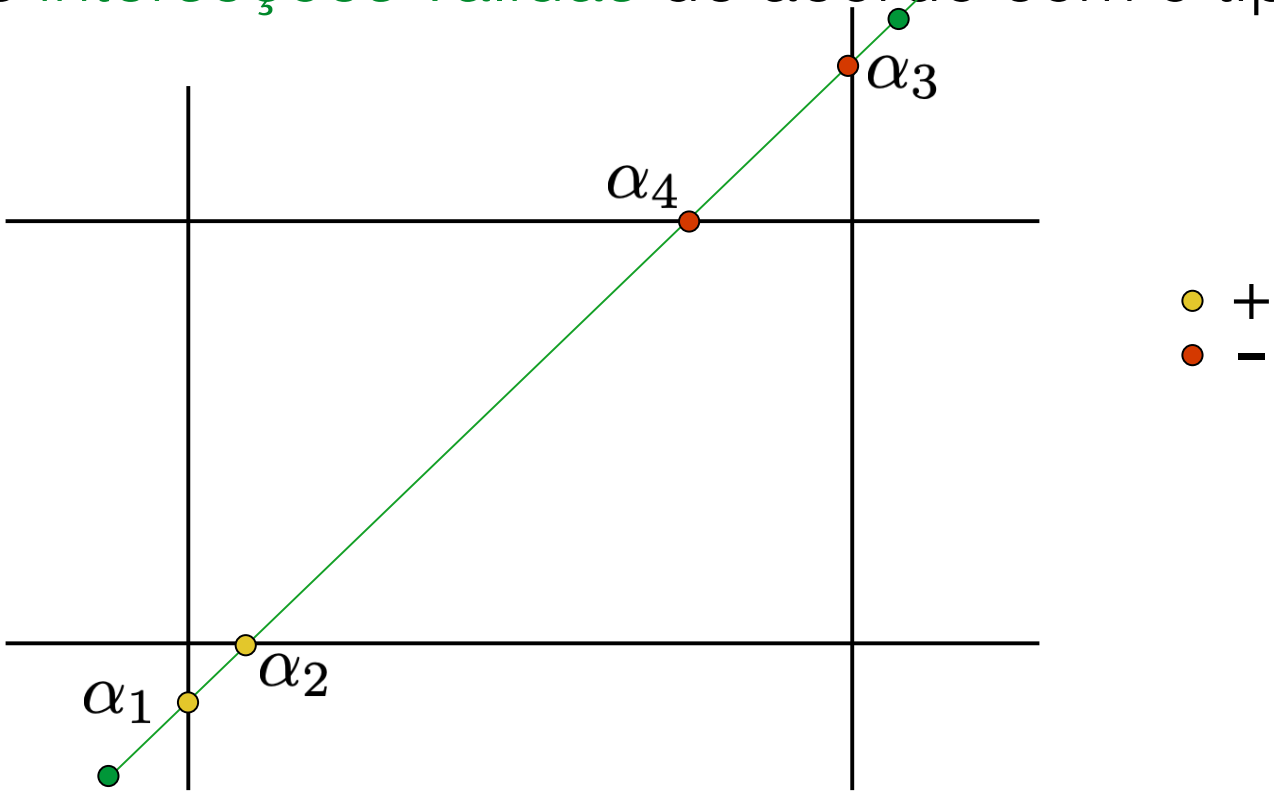
Marcamos as interseções válidas de acordo com o tipo:



Clipping

Segmento de reta x retângulo

Marcamos as interseções válidas de acordo com o tipo:

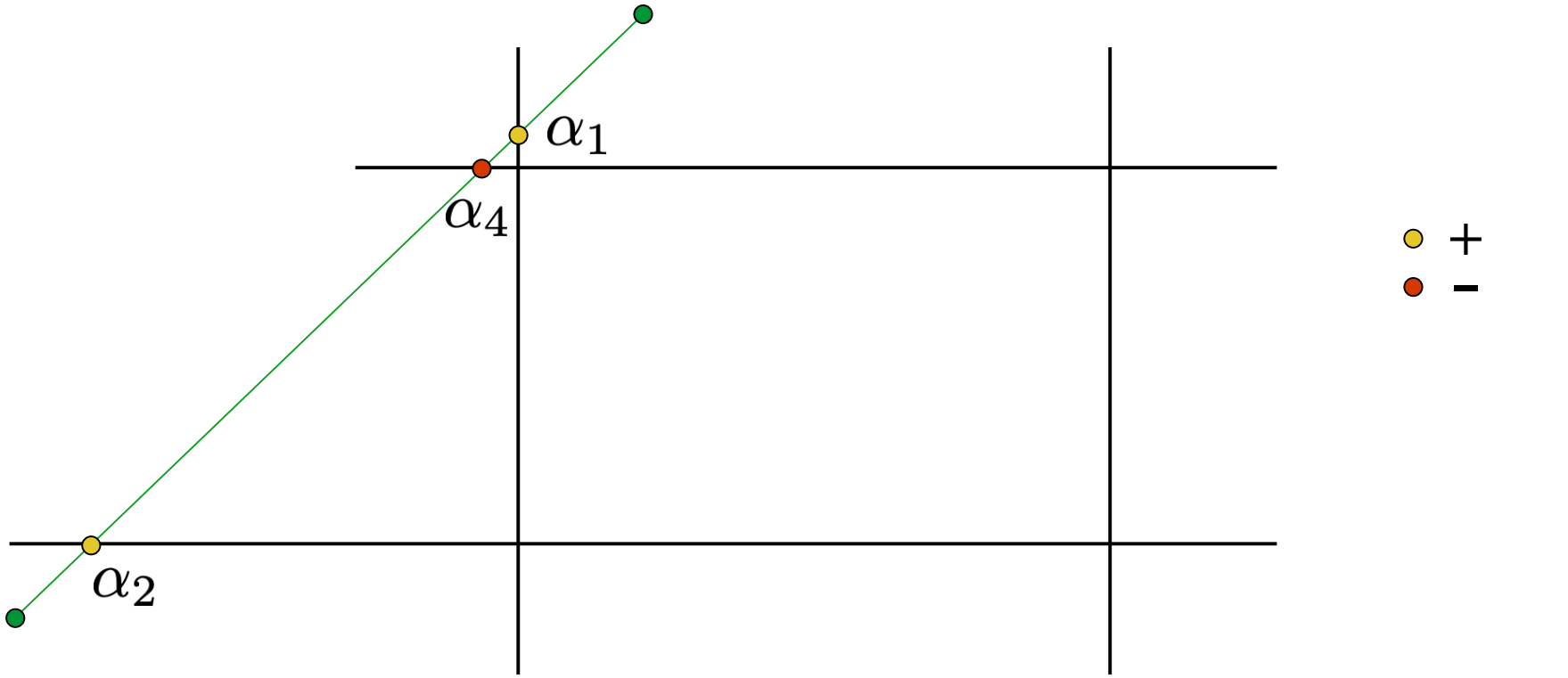


$$0 < \alpha_1 < \alpha_2 < \alpha_4 < \alpha_3 < 1$$

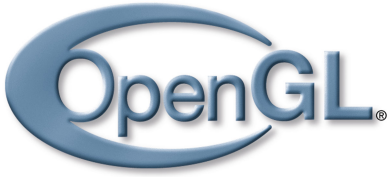
Clipping

Segmento de reta x retângulo

Marcamos as interseções válidas de acordo com o tipo:



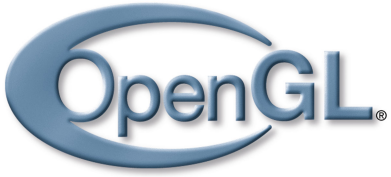
$$0 < \alpha_2 < \alpha_4 < \alpha_1 < 1$$



Clipping

Polígono x retângulo

Podemos pensar em um algoritmo de clipping de polígono baseado no recorte de segmentos.

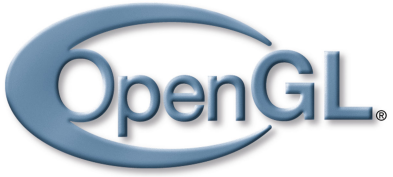


Clipping

Polígono x retângulo

Podemos pensar em um algoritmo de clipping de polígono baseado no recorte de segmentos.

Basta recortarmos as arestas do polígono sucessivamente.

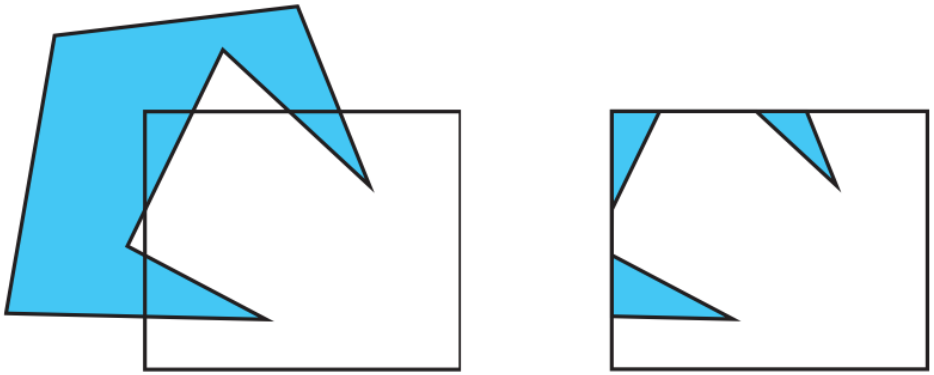


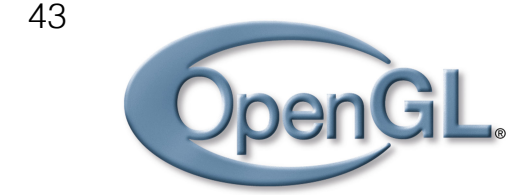
Clipping

Polígono x retângulo

Podemos pensar em um algoritmo de clipping de polígono baseado no recorte de segmentos.

Basta recortarmos as arestas do polígono sucessivamente.

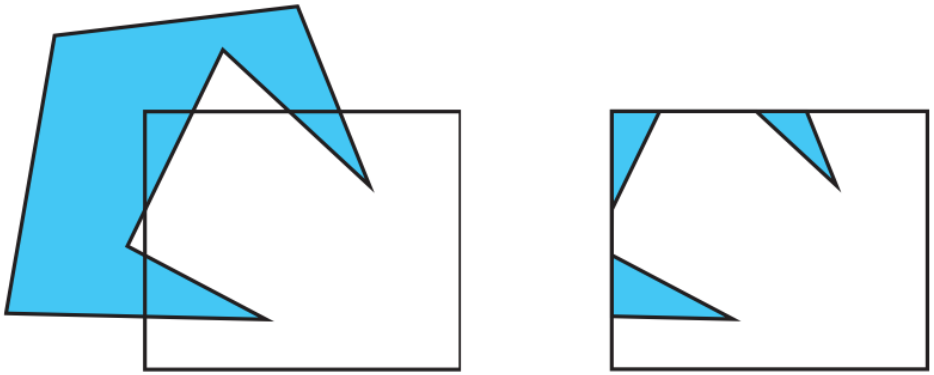


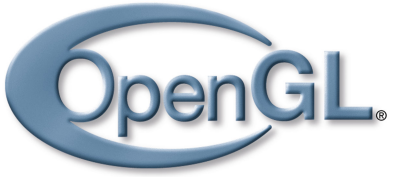


Clipping

Polígono x retângulo

Polígonos são objetos com um interior.



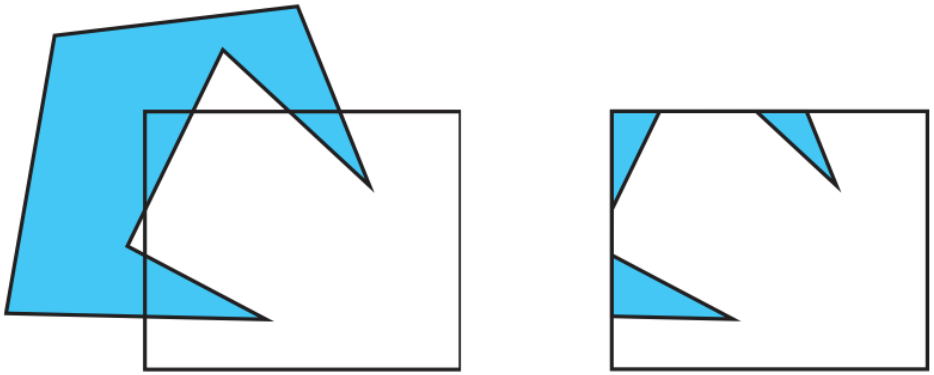


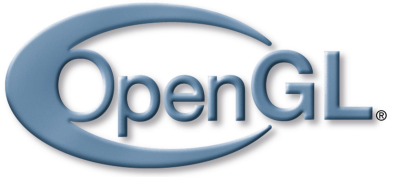
Clipping

Polígono x retângulo

Polígonos são objetos com um interior.

O recorte de um polígono não convexo pode gerar mais de um polígono.



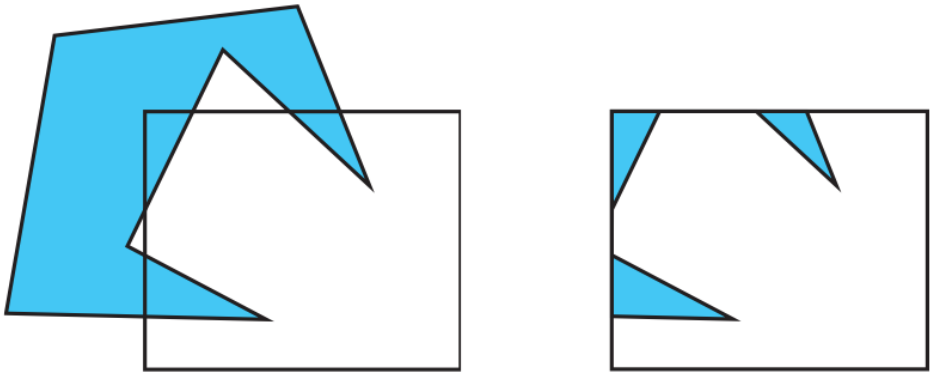


Clipping

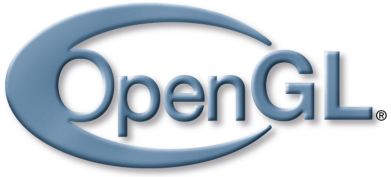
Polígono x retângulo

Polígonos são objetos com um interior.

O recorte de um polígono não convexo pode gerar mais de um polígono.



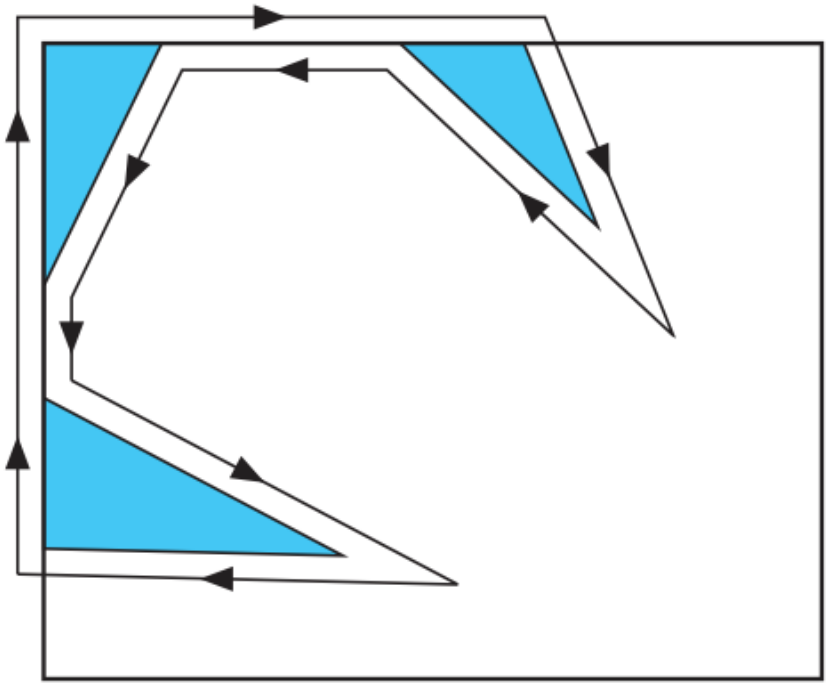
Problema!



Clipping

Polígono x retângulo

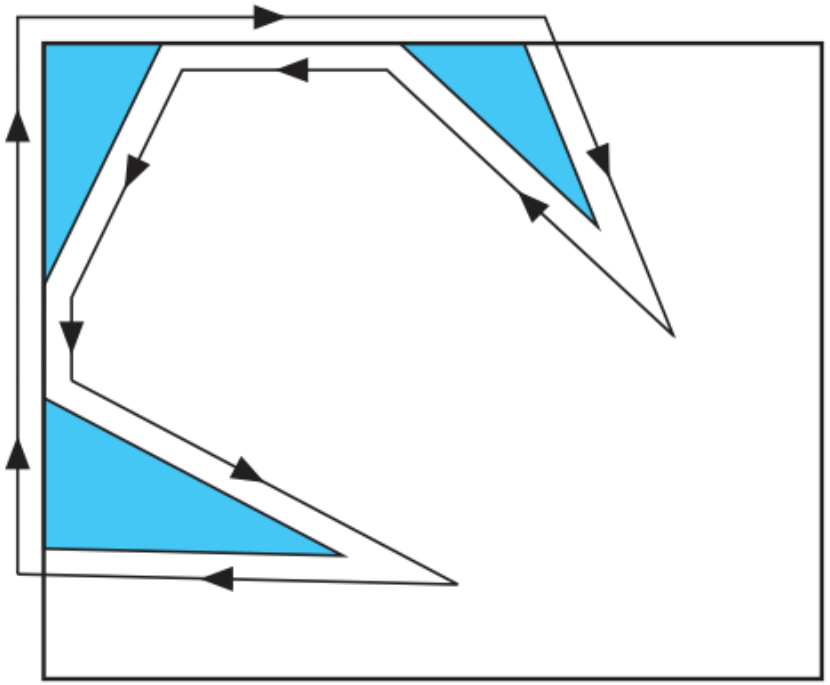
Como alternativa, podemos conectar as componentes através de regiões degeneradas.



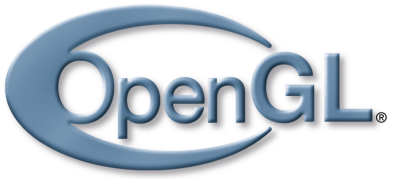
Clipping

Polígono x retângulo

Como alternativa, podemos conectar as componentes através de regiões degeneradas.



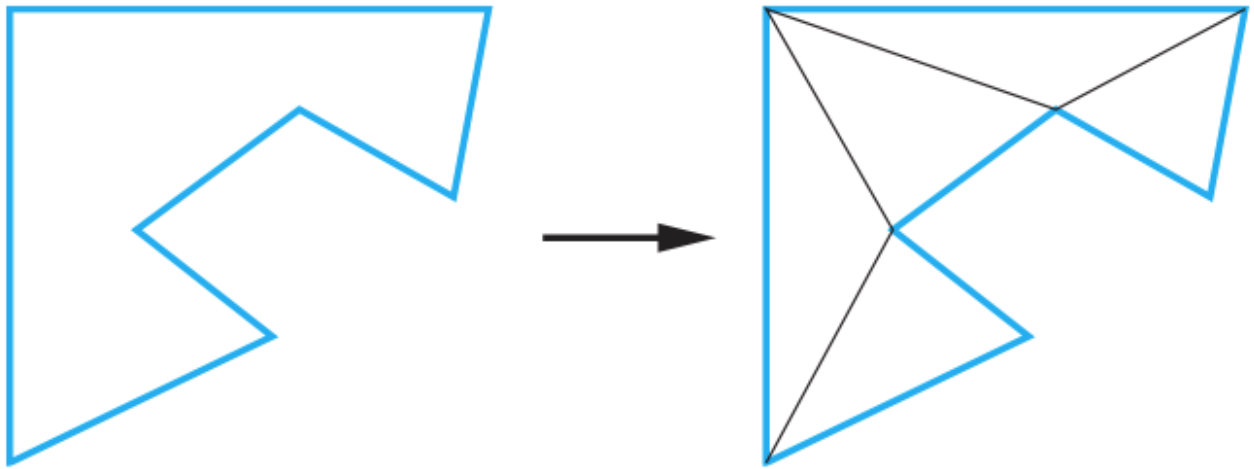
Ruim...

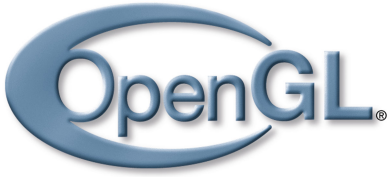


Clipping

Polígono x retângulo

A melhor alternativa é considerar apenas polígonos convexos!

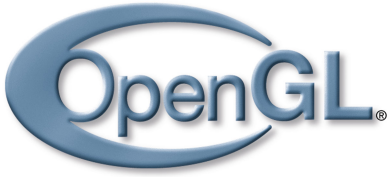




Clipping

Polígono x retângulo

Estudaremos o algoritmo de Sutherland-Hodgman

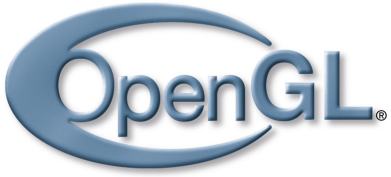


Clipping

Polígono x retângulo

Estudaremos o algoritmo de Sutherland-Hodgman

Um polígono é usualmente representado por uma lista circular de vértices.



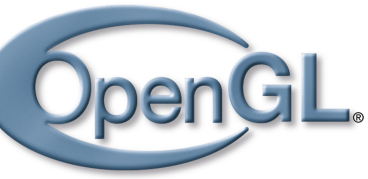
Clipping

Polígono x retângulo

Estudaremos o algoritmo de Sutherland-Hodgman

Um polígono é usualmente representado por uma lista circular de vértices.

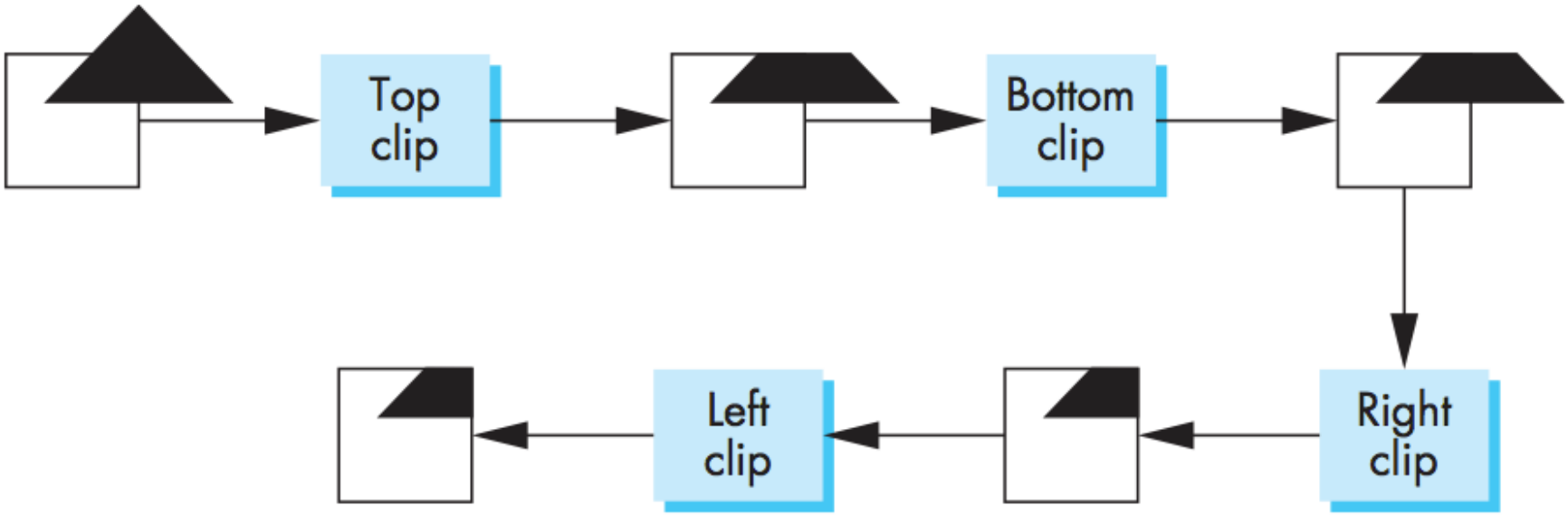
Os vértices e arestas do polígono são processados em sequência e classificados contra os semi-espaços do retângulo de recorte.



Clipping

Polígono x retângulo

Algoritmo de Sutherland-Hodgman



Computação Gráfica

TCC-00291

Assunto: Clipping