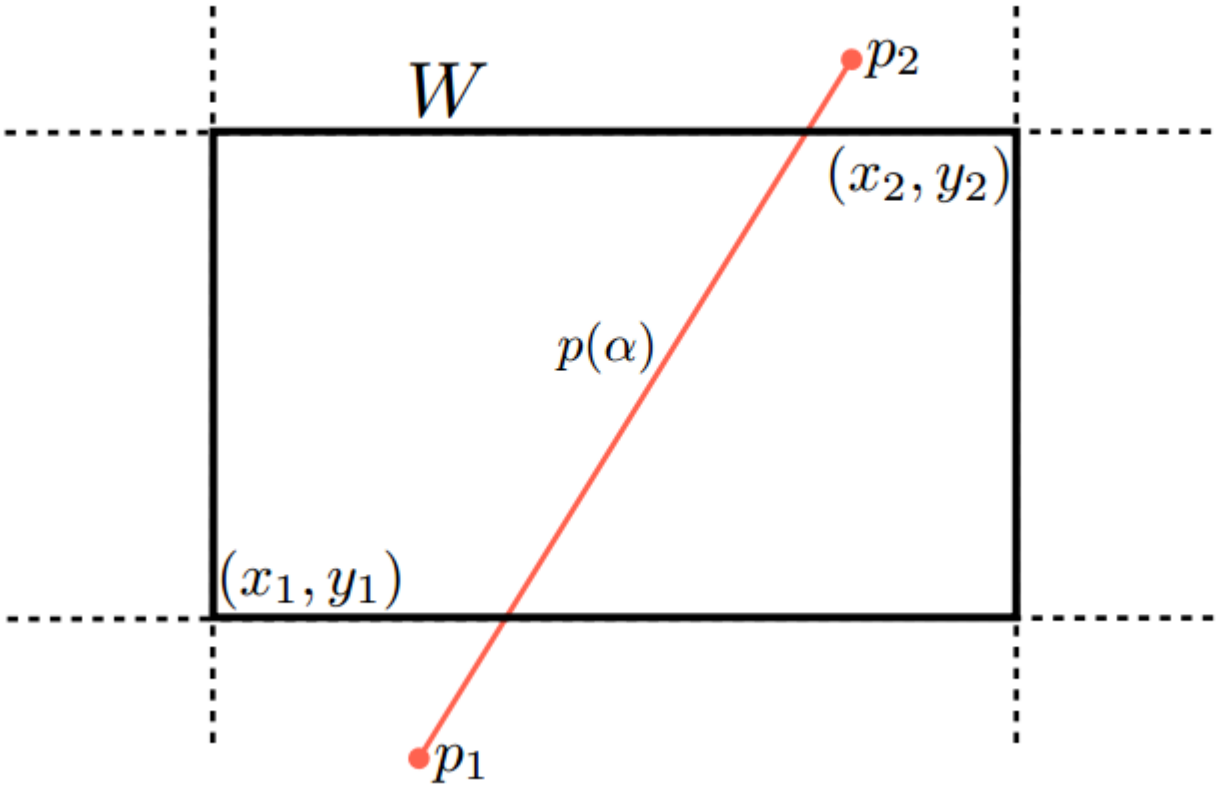
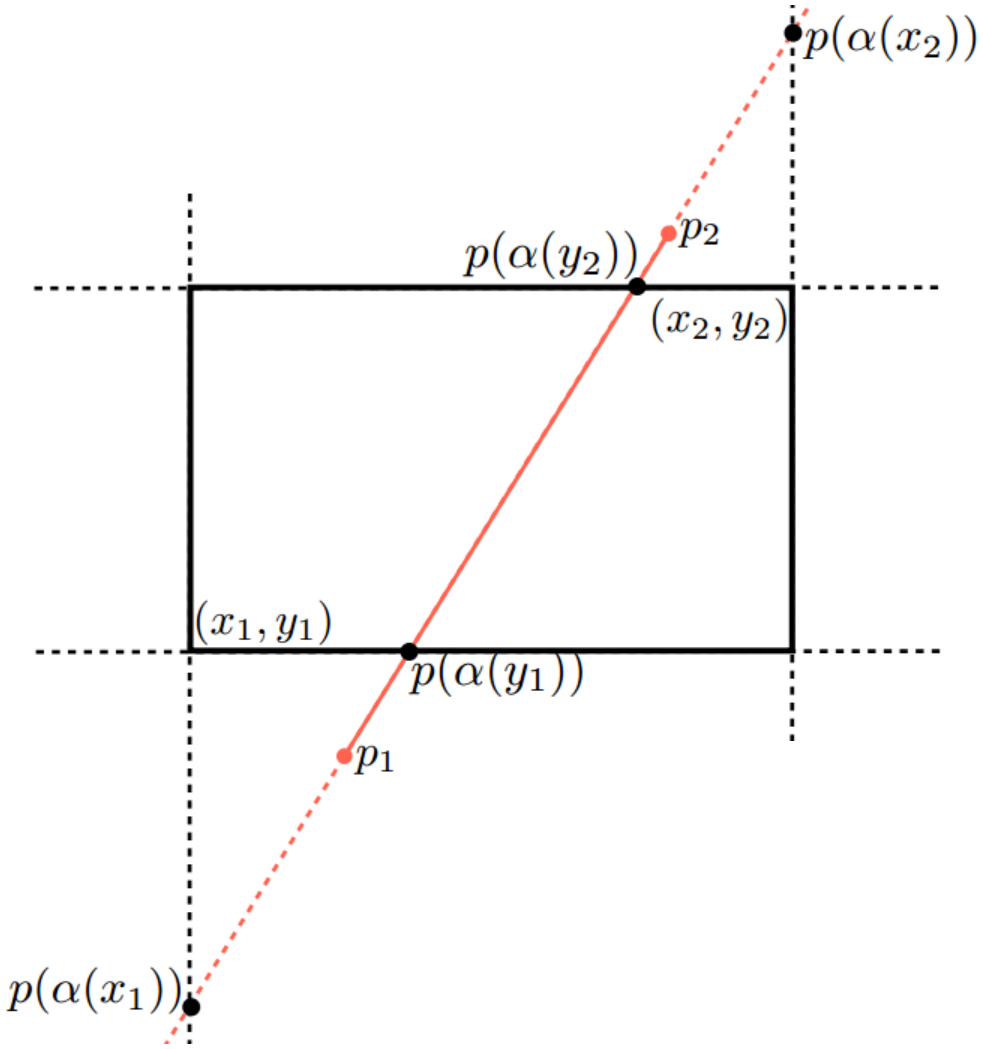
Clipping de segmentos de líneas



Cualquier punto del segmento tiene la forma

$$p(\alpha) = (1 - \alpha)p_1 + \alpha p_2$$

Clipping de segmentos de líneas



$$\alpha(x_1) = \frac{x_1 - x(p_1)}{x(p_2) - x(p_1)}$$

$$\alpha(x_2) = \frac{x_2 - x(p_1)}{x(p_2) - x(p_1)}$$

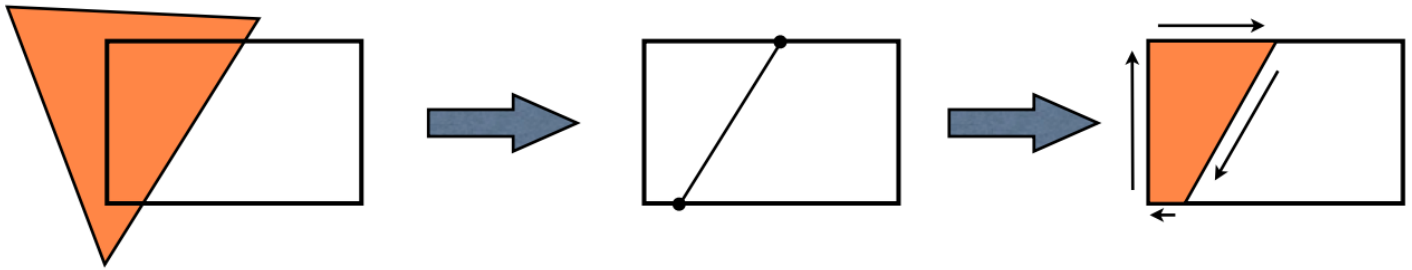
$$\alpha(y_1) = \frac{y_1 - y(p_1)}{y(p_2) - y(p_1)}$$

$$\alpha(y_2) = \frac{y_2 - y(p_1)}{y(p_2) - y(p_1)}$$

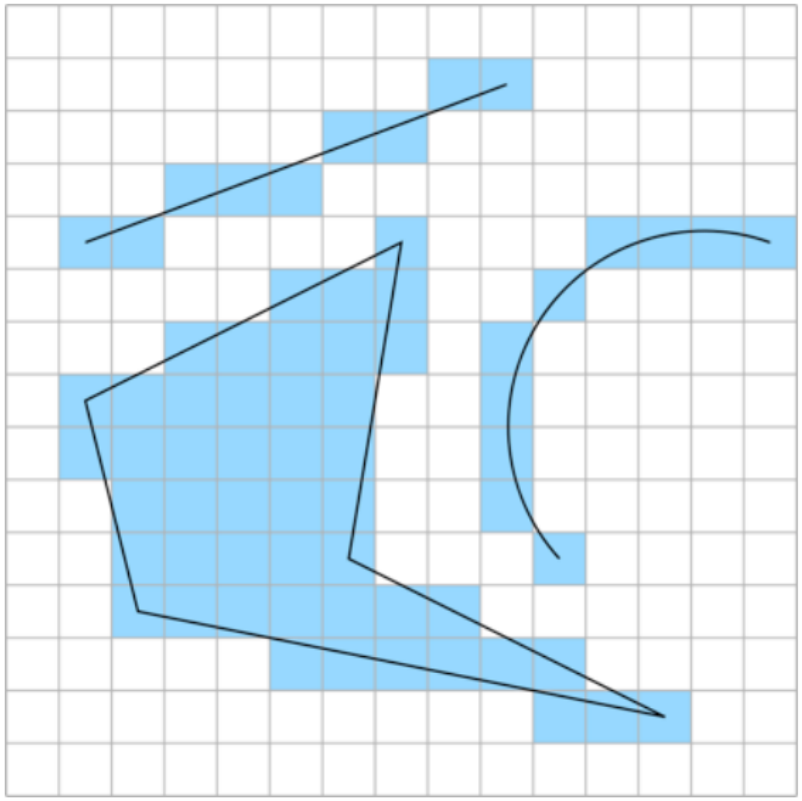
Algoritmo:

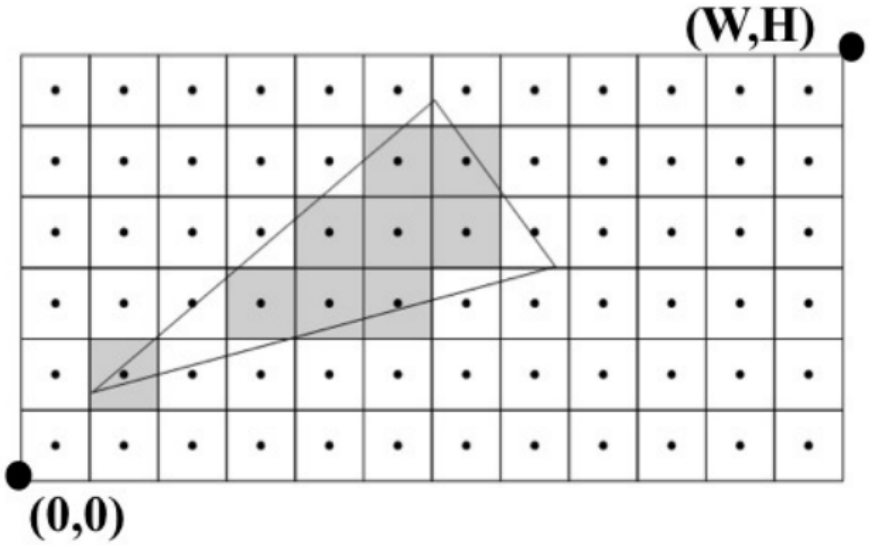
- Ordenar $\alpha(x1)\ \alpha(x2)\ \alpha(y1)\ \alpha(y2)$ por valor.
  Si el orden es
  XXYY - $\alpha(x^*)\ \alpha(x^*)\ \alpha(y^*)\ \alpha(y^*)$ o
  YYXX - $\alpha(x^*)\ \alpha(x^*)\ \alpha(y^*)\ \alpha(y^*)$
  el segmento esta fuera de W
- Sino el nuevo segmento esta definido por los 2 parámetros $\alpha()$ intermedios
- Tomar la intersección con el rango (0,1)
- Caso especial: segmentos alineados con X o Y

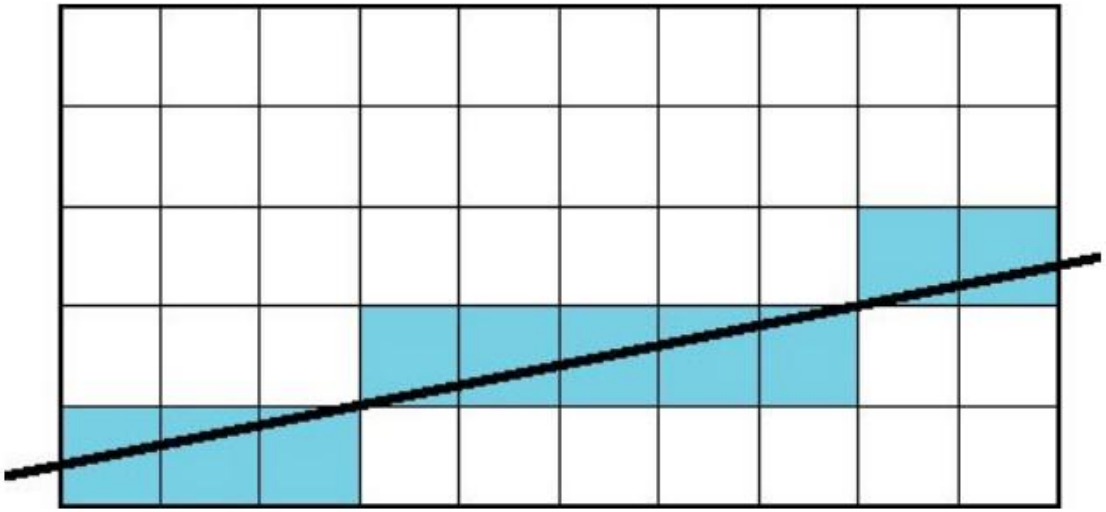
Clipping de polígonos



Raterización







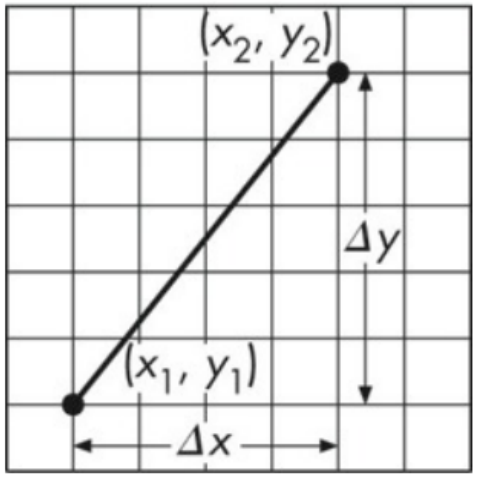
Discretización líneas -DDA

Ventajas

- Fácil implementación

Desventajas

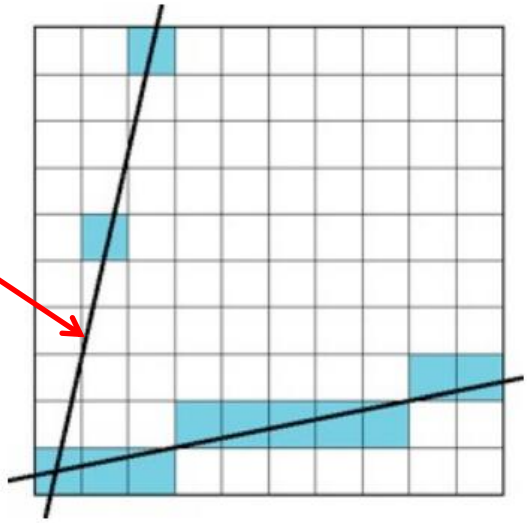
- Acumulación de error de redondeo
- Utiliza aritmética de punto flotante



$$
\begin{aligned}
&m = \tfrac{y_2 - y_1}{x_2 - x_1};\\
&y = y_1;\\
&\text{for}\big(x = x_1;\ x \le x_2;\ x++\big)\{\\
&\qquad \text{fill}(x,\text{round}(y));\\
&\qquad y = y + m;\\
&\}
\end{aligned}
$$

Discretización líneas -DDA

Submuestreo



Condiciones

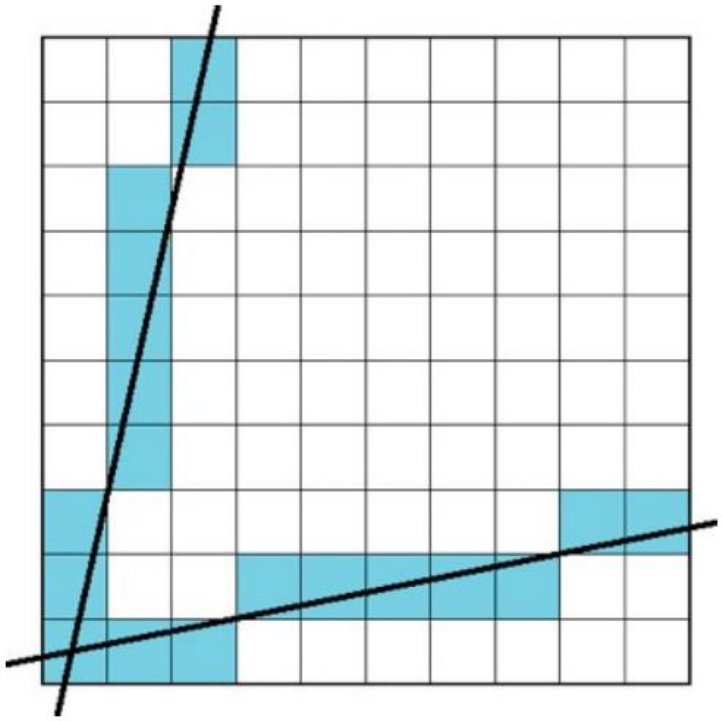
$\Delta x \geq 0, \Delta y \leq 0, \Delta x \leq \Delta y$

(Primer octante)

$$
\begin{aligned}
&m = \tfrac{y_2 - y_1}{x_2 - x_1};\\
&y = y_1;\\
&\text{for}\big(x = x_1;\ x \leq x_2;\ x++\big)\{\\
&\qquad \text{fill}(x,\text{round}(y));\\
&\qquad y = y + m;\\
&\}
\end{aligned}
$$

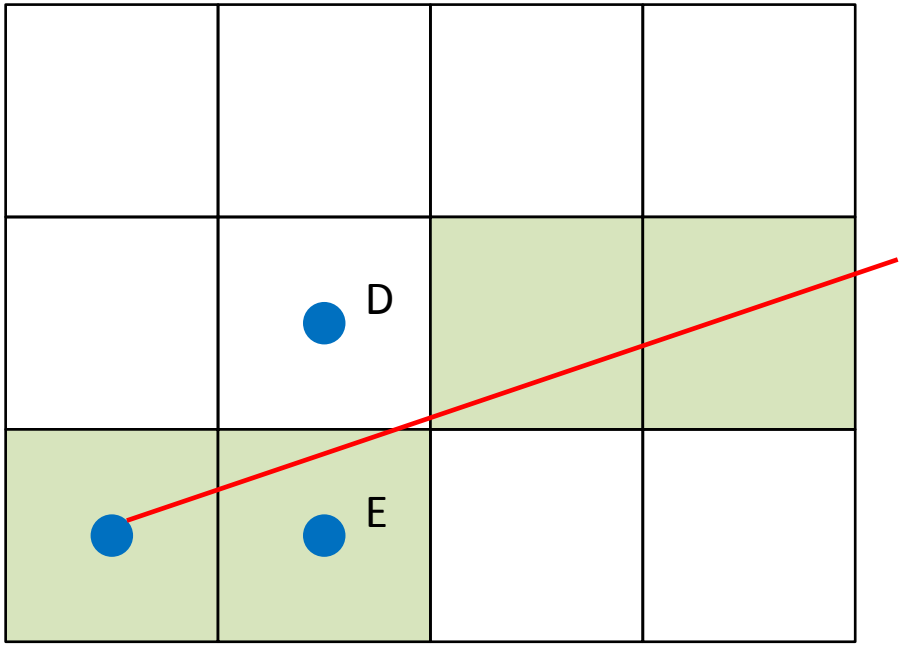
Discretización líneas -DDA

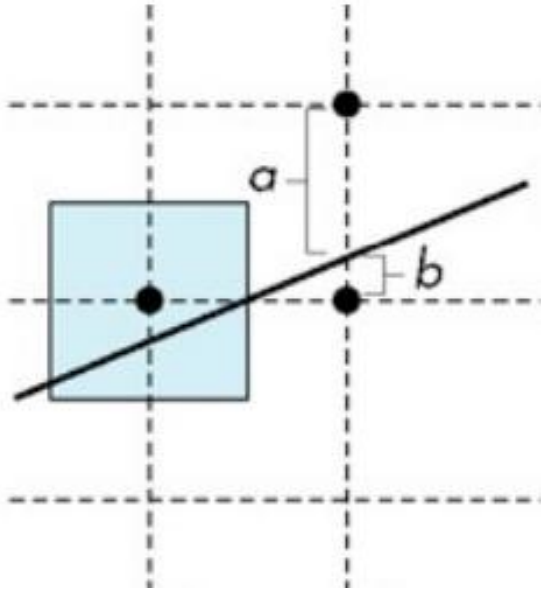
```
if (y2 − y1 ≤ x2 − x1){
    m = (y2−y1)/(x2−x1);
    y = y1;
    for(x = x1; x ≤ x2; x++){
        fill(x,round(y));
        y = y + m;
    }
}
else {
    m = (x2−x1)/(y2−y1);
    x = x1;
    for(y = y1; y ≤ y2; y++)
        fill(round(x),y);
        x = x + m;
    }
}
```



Resuelve casos de primer y segundo octante

Discretización líneas - Bresenham

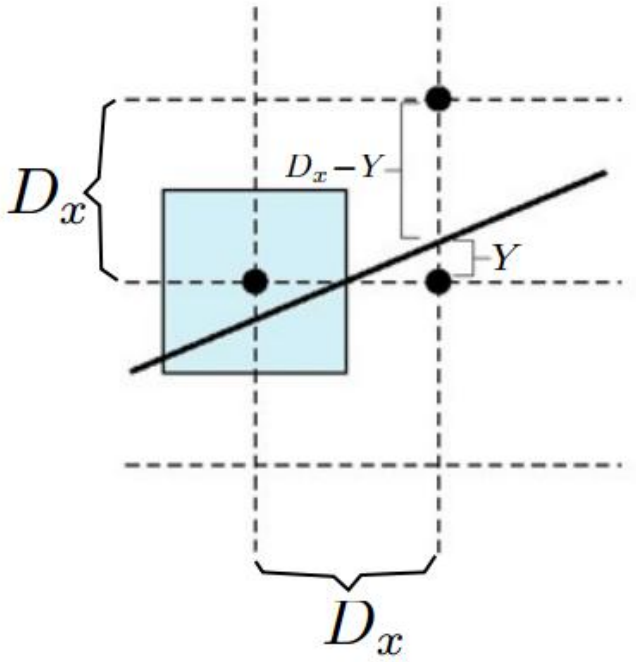




Minimizar error: Si (b<a) elijo E sino elijo D

Discretización líneas -Bresenham

```
Y = 0;
for (x=x1; x ≤ x2; x++){
if (Y ≤ Dx − Y){
        fill(x,y);
}
else {
        y++;
        fill(x,y);
        Y = Y − Dx;
        }
Y = Y + Dy;
}
```



Discretización líneas -Bresenham

- Condición de decisión binaria:

error + o error –

- Variable de acumulación entera

Ventajas:

- utiliza aritmética entera + simple

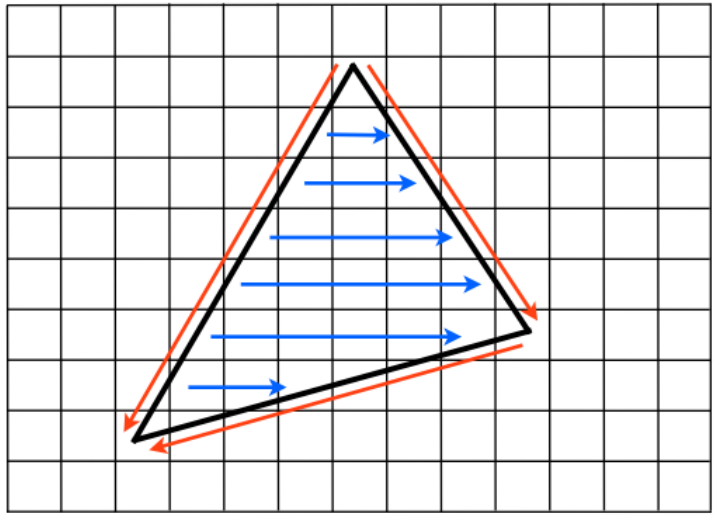
para implementar en hardware

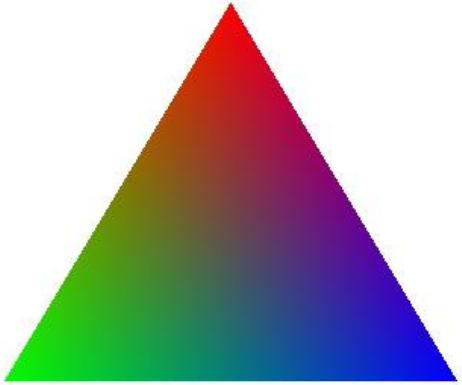
Discretización líneas -Bresenham

```
plot(x0,y0, x1,y1)
dx=x1-x0
dy=y1-y0
D = 2*dy – dx
plot(x0,y0)
y=y0
for x from x0+1 to x1 {
    if D > 0 y = y+1
            plot(x,y)
            D = D + (2*dy-2*dx)
    else
            plot(x,y)
            D = D + (2*dy)
```

Relleno de polígonos - Scanline

```
y = y_first;
initialize-left-segment();
initialize-right-segment();
while(y≤ y_last){
      if(y == end-of-left-segment)
            switch-to-next-left-segment();
      if(y == end-of-right-segment)
            switch-to-next-right-segment();
      x_left = evaluate-left-segment();
      x_right = evaluate-right-segment();
      for(x = x_left;x ≤ x_right; x++)
            fill(x,y);
      y++;
}
```





CASO CONVEXO