FICH
ICH

Computación Gráfica

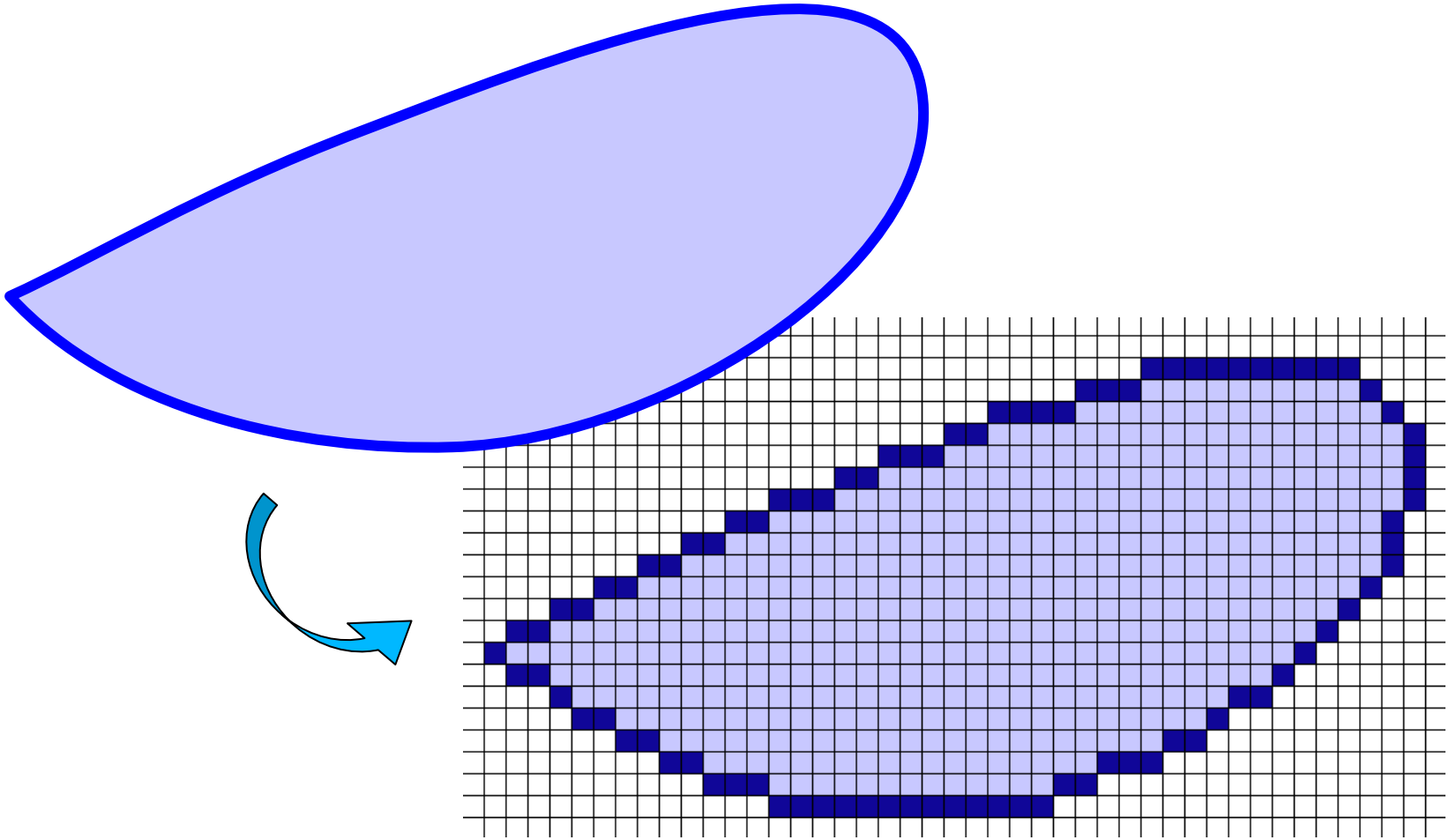
Rasterización

Docentes:

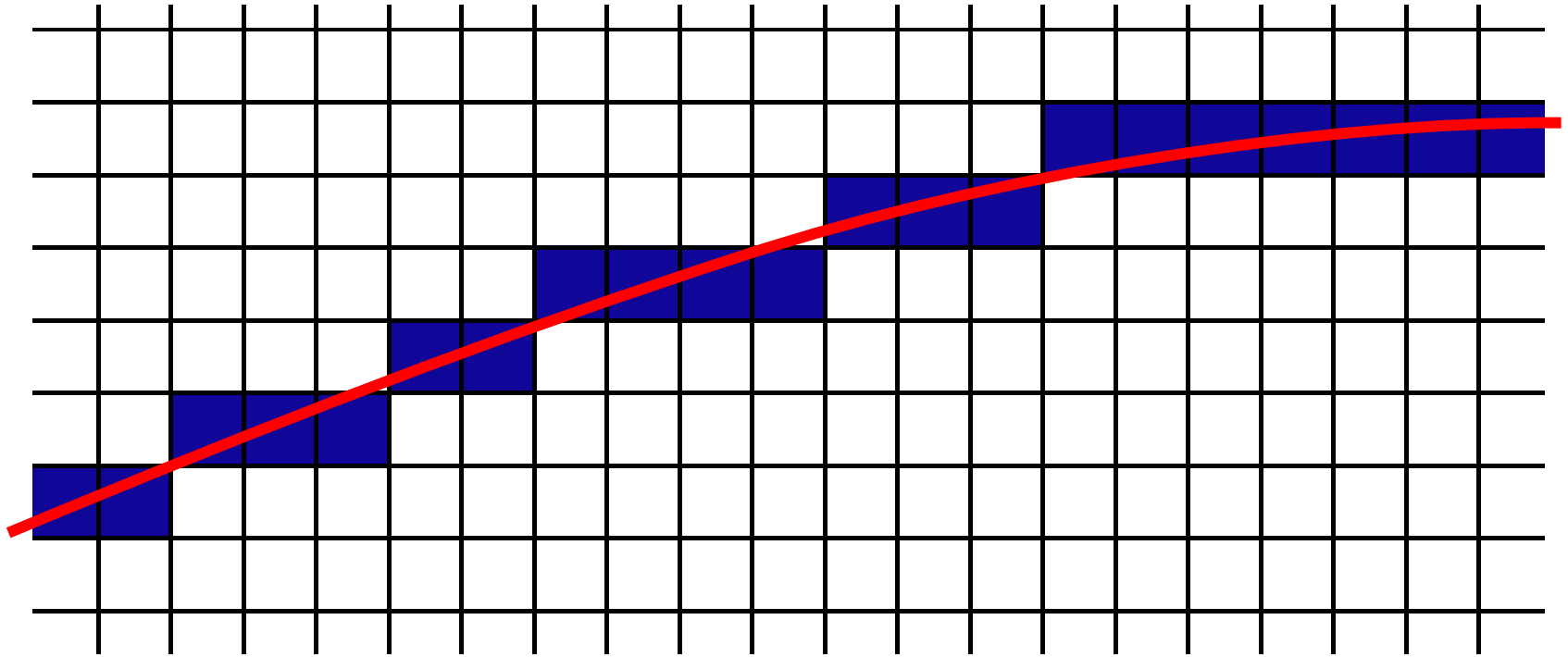
Nestor Calvo
Pablo Novara
Walter Sotil

2011

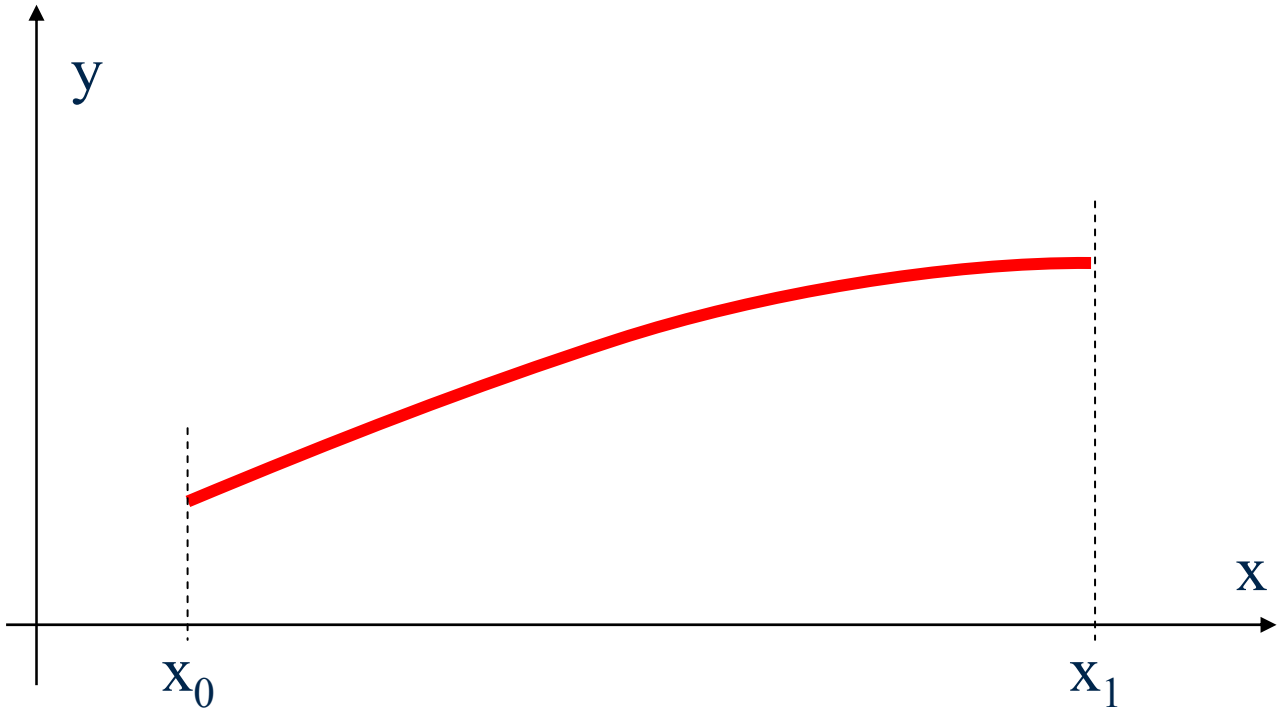
Rasterización o Scan-Conversion



Rasterización o Scan-Conversion

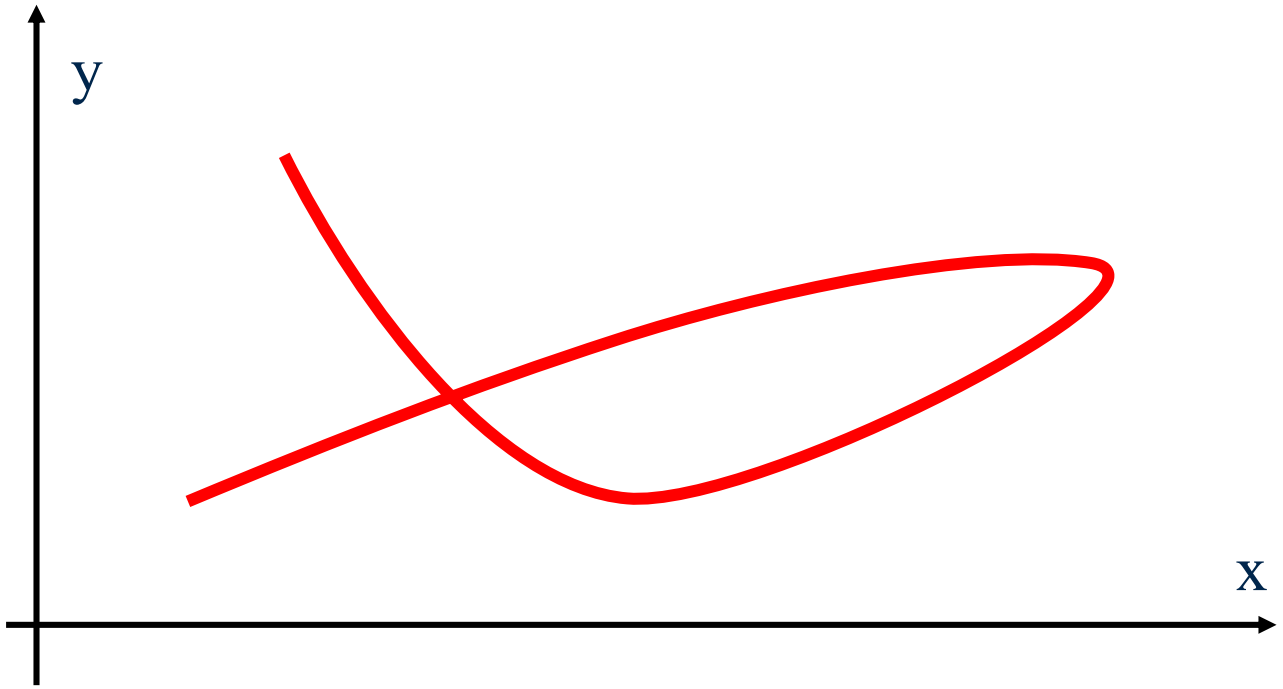


Curva



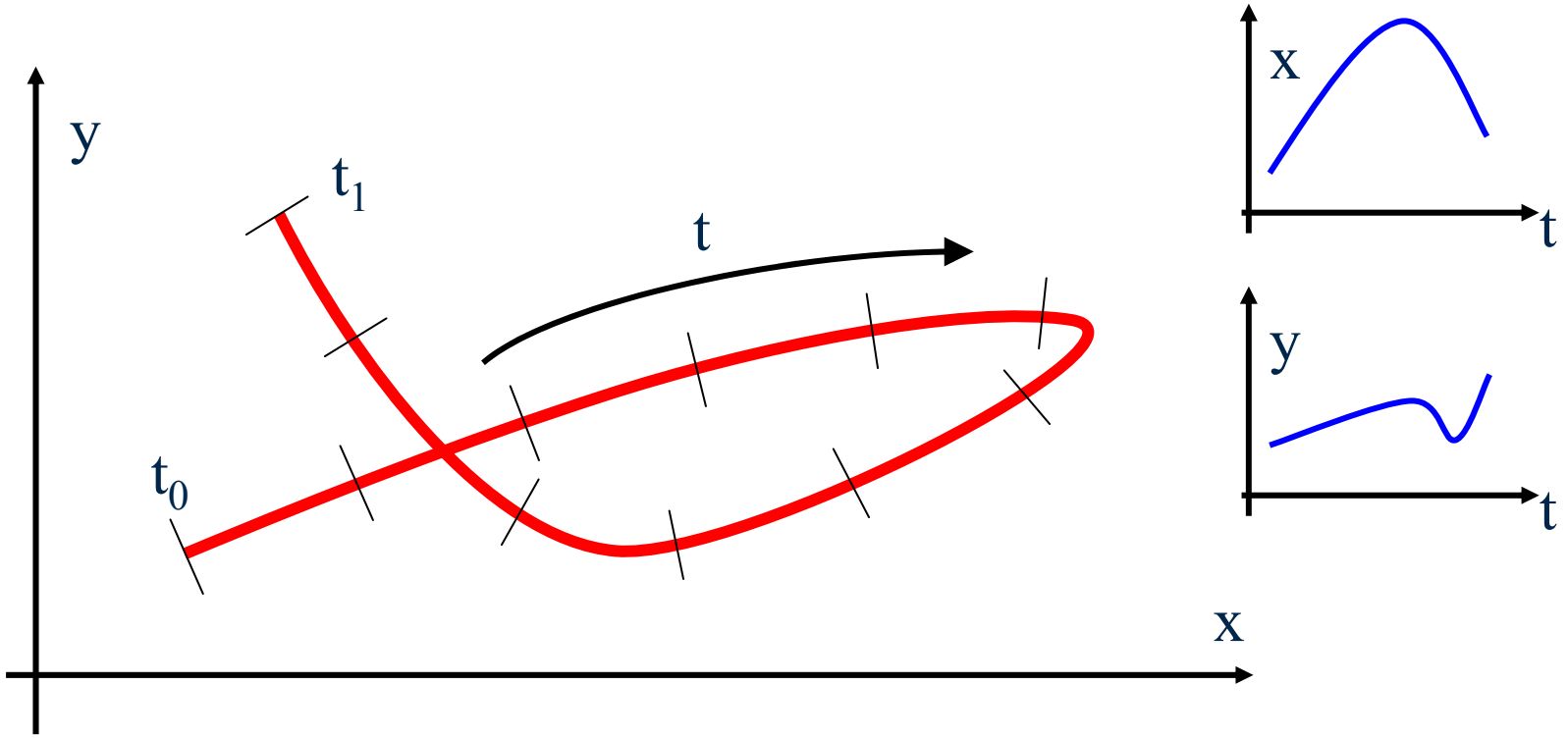
$$y = y(x) \quad x \in [x_0, x_1]$$

Curva



$y \neq y(x)$

Curva

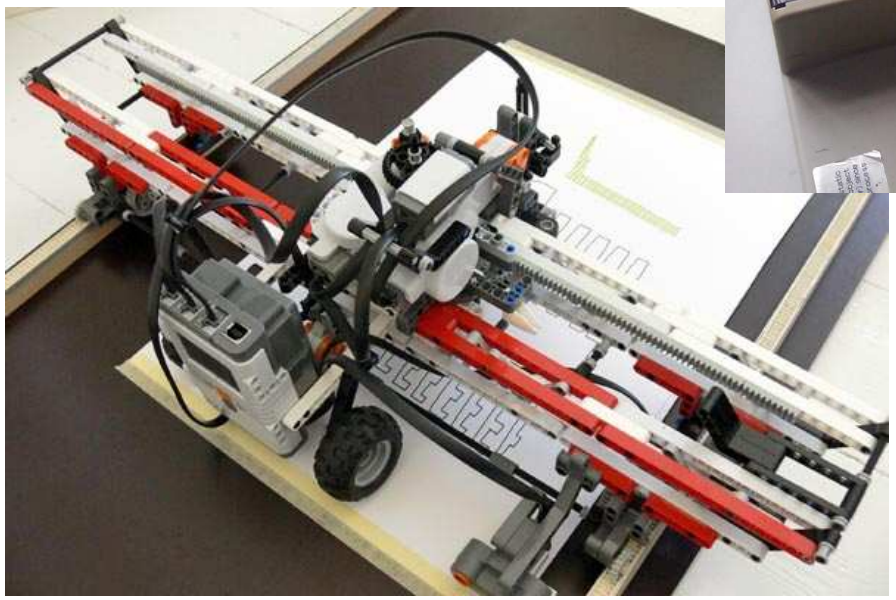


$$y = y(t) \quad x = x(t) \quad t \in [t_0, t_1]$$

definición de “curva” (plana)

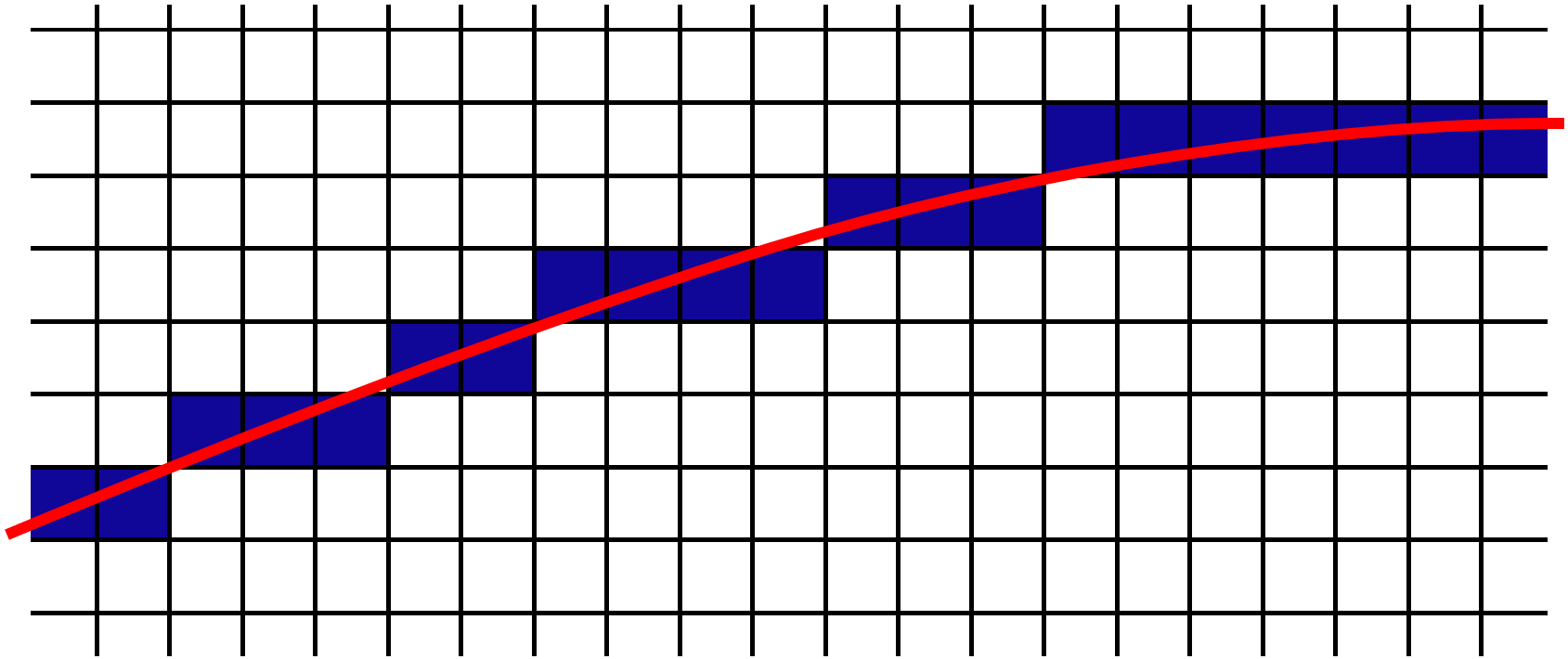
Trazado Vectorial

Trazado Vectorial



Trazado raster (scan-lines)

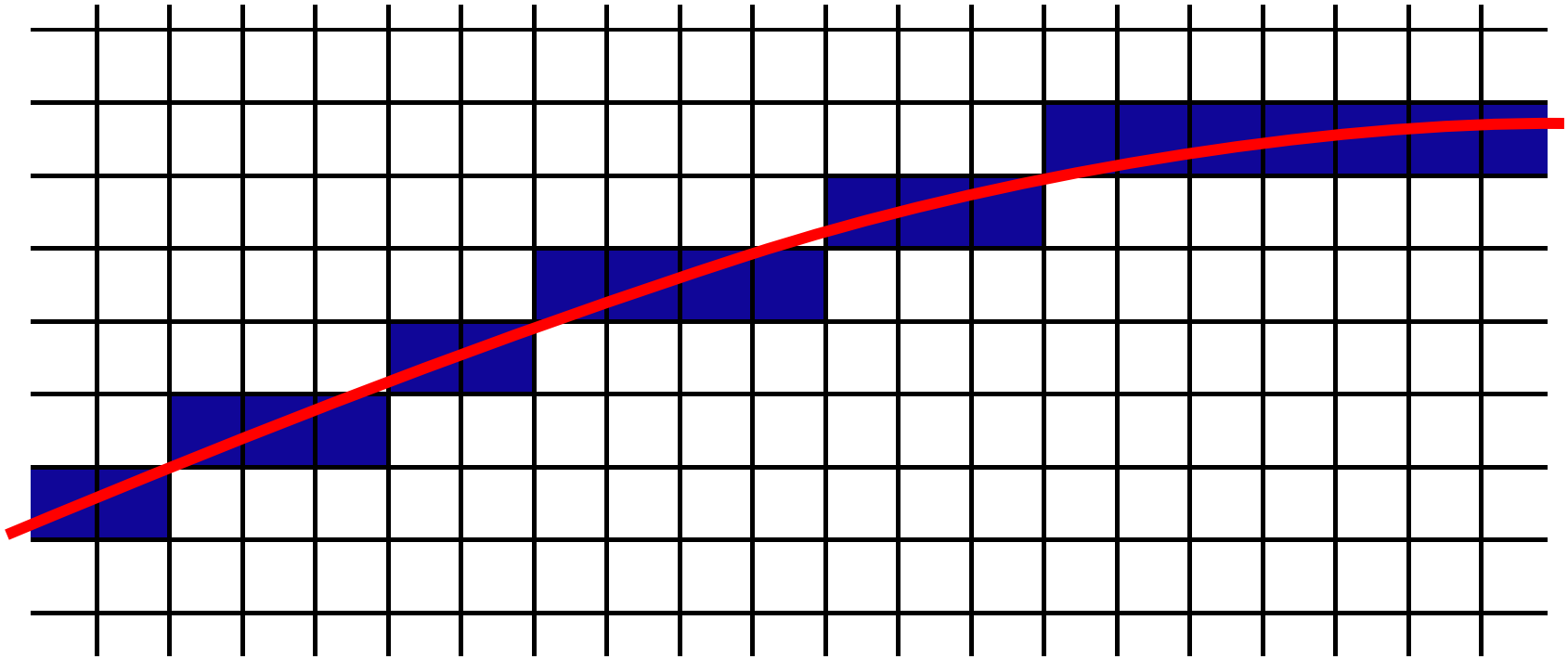
Rasterización de curvas



Pixeles contiguos: $\max(|\Delta x|, |\Delta y|)=1$

Coordenadas enteras: $\{int(x(t)+0.5), int(y(t)+0.5)\}$

Rasterización de curvas



$$y = y(t) \quad x = x(t) \quad t \in [t0, t1]$$

$$\Delta t \equiv dt \qquad \Delta x \approx dx = dx/dt\ dt \qquad \Delta y \approx dy = dy/dt\ dt$$

$$\max(|\Delta x|, |\Delta y|) = 1 \Leftrightarrow dt = \min(1/|x'|,1/|y'|) = 1/\max(|x'|,|y'|)$$

Curva DDA

```
static int redondea(float a) {return int(a+.5);}

static void intercambia (float &a, float &b) {float tmp=a; a=b; b=tmp;}

void evalua_curva(float t, float &x, float &y, float &dx, float &dy){
  // de acuerdo a la curva en cuestión evalúa x e y en t y las derivadas
  // ej: elipse de semiejes 100 y 200 centrada en 500,300
  x=100*cos(t)+500; y=200*sin(t)+300; dx=100*sin(t); dy=-200*cos(t);
}

void curvaDDA(float t0, float t1) {
  if (t1<t0) intercambia(t0,t1);
  float t=t0,x,y,dx,dy;
  do{
    evalua_curva(t,x,y,dx,dy); pinta(redondea(x),redondea(y));
    if (fabs(dx)≥fabs(dy)) t+=1/dx; else t+=1/dy;
  } while(t≤t1);
}
```

Línea DDA

```
void linea_DDA(float x1, float y1, float x2, float y2) {
  float dx=x2–x1, dy=y2–y1, m;
  if (redondea(dx)==0 && redondea(dy)==0) { // solo un punto
    pinta(redondea(x1), redondea(y1)); return;
  }
  if (fabs(dx)>=fabs(dy)){ // horizontal
    if (dx<0) {intercambia(x1,x2); intercambia(y1,y2); dx=–dx; dy=–dy;} // x avanza
    for (m=dy/dx, y=y1, x=x1; x<= x2; x++, y+=m) pinta(redondea(x), redondea(y));
  }
  else { // vertical
    if (dy<0) {intercambia(x1,x2); intercambia(y1,y2); dx=–dx; dy=–dy;} // y avanza
    for (m=dx/dy, x=x1, y=y1; y<=y2; y++, x+=m) pinta(redondea(x), redondea(y));
  }
}
```

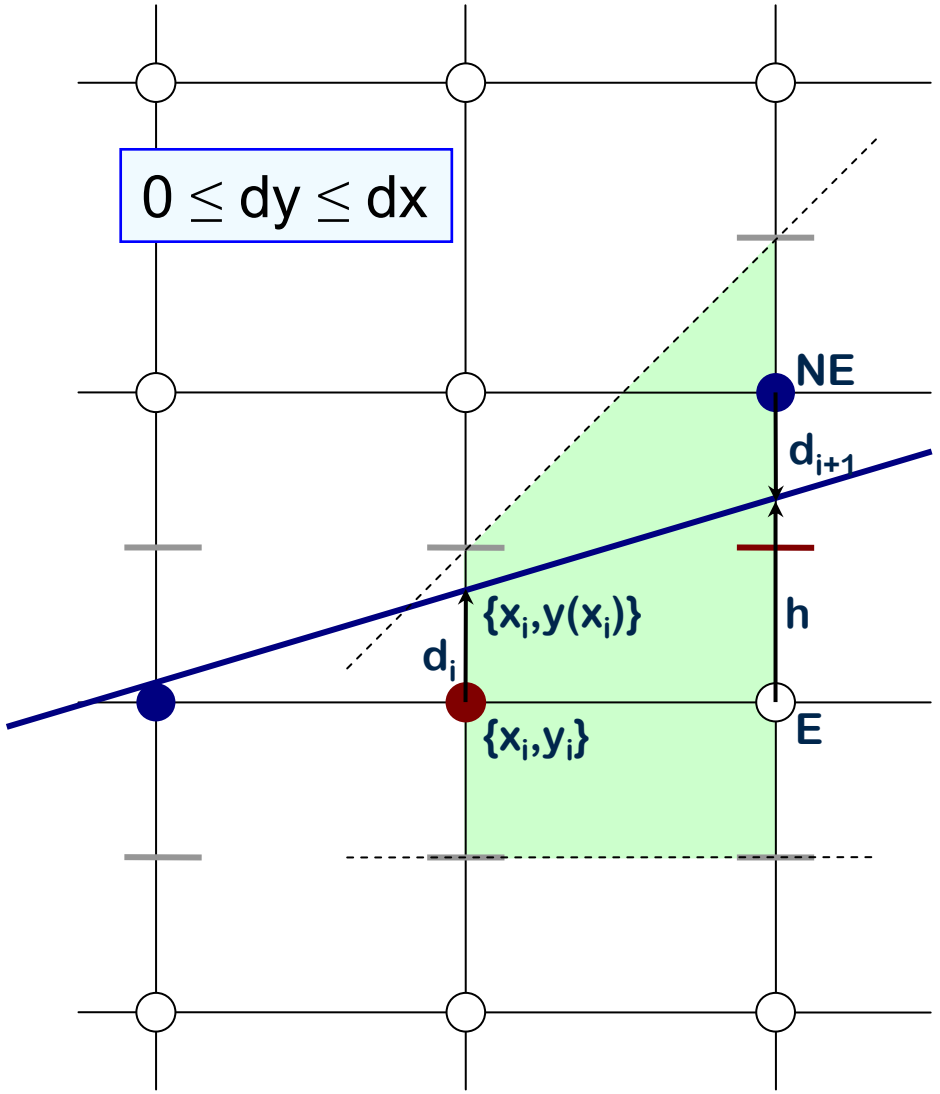
Línea DDA

```
void linea_DDA(float x1, float y1, float x2, float y2) {
  float dx=x2–x1, dy=y2–y1, m;
  if (redondea(dx)==0 && redondea(dy)==0) { // solo un punto
    pinta(redondea(x1), redondea(y1)); return;
  }
  if (fabs(dx)>=fabs(dy)){ // horizontal
    if (dx<0) {intercambia(x1,x2); intercambia(y1,y2); dx=–dx; dy=–dy;} // x avanza
    x1=redondea(x1); x2=redondea(x2);
    for (m=dy/dx, y=y1, x=x1; x<= x2; x++, y+=m) pinta(x, redondea(y));
  }
  else { // vertical
    if (dy<0) {intercambia(x1,x2); intercambia(y1,y2); dx=–dx; dy=–dy;} // y avanza
    y1=redondea(y1); y2=redondea(y2);
    for (m=dx/dy, x=x1, y=y1; y<=y2; y++, x+=m) pinta(redondea(x), y);
  }
}
```

Algoritmo de línea de Bresenham

¿$h = d_i + dy/dx > .5$?
Si:
  pintar NE
  hacer: $d_{i+1} = h - 1 = d_i + dy/dx - 1$
No:
  pintar E
  hacer: $d_{i+1} = h = d_i + dy/dx$

¿$2\ dx\ d_i + 2\ dy - dx > 0$?
Si:
  pintar NE
  hacer: $2\ dx\ d_{i+1} = 2\ dx\ d_i + 2\ dy - 2\ dx$
No:
  pintar E
  hacer: $2\ dx\ d_{i+1} = 2\ dx\ d_i + 2\ dy$

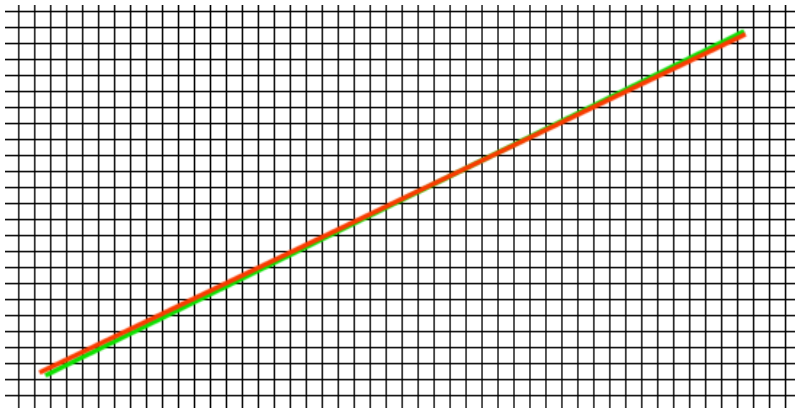


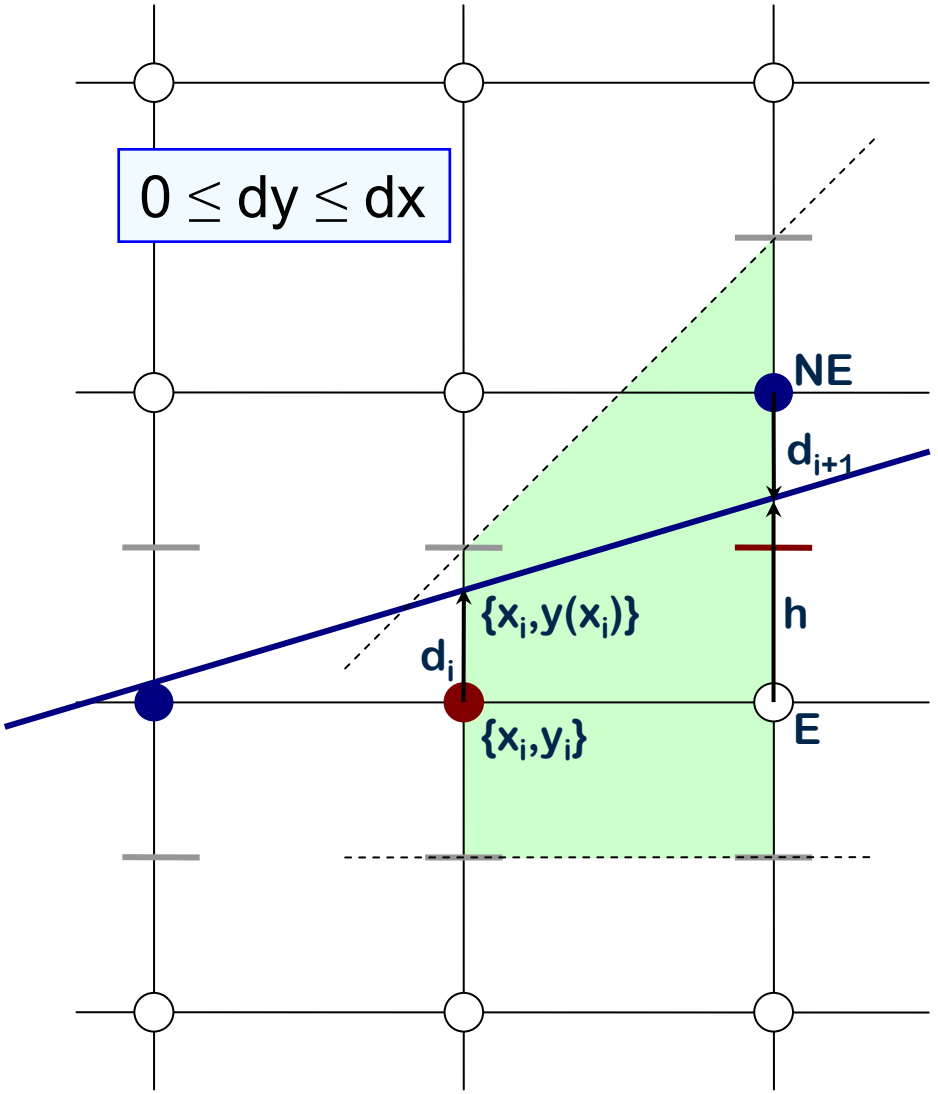
Algoritmo de línea de Bresenham

¿2 dx d_i + 2 dy – dx > 0?
Si:
  pintar NE
  hacer: 2 dx d_{i+1} = 2 dx d_i + 2 dy – 2 dx
No:
  pintar E
  hacer: 2 dx d_{i+1} = 2 dx d_i + 2 dy

$D_i = 2\,dx\,d_i + 2\,dy - dx$ variable de decisión
$E = 2\,dy$ incremento general
$NE = -2\,dx$ incremento adicional NE

$D_0 = 2\,dy - dx$ ($d_0 = 0 \Rightarrow D_0 = E - dx$)



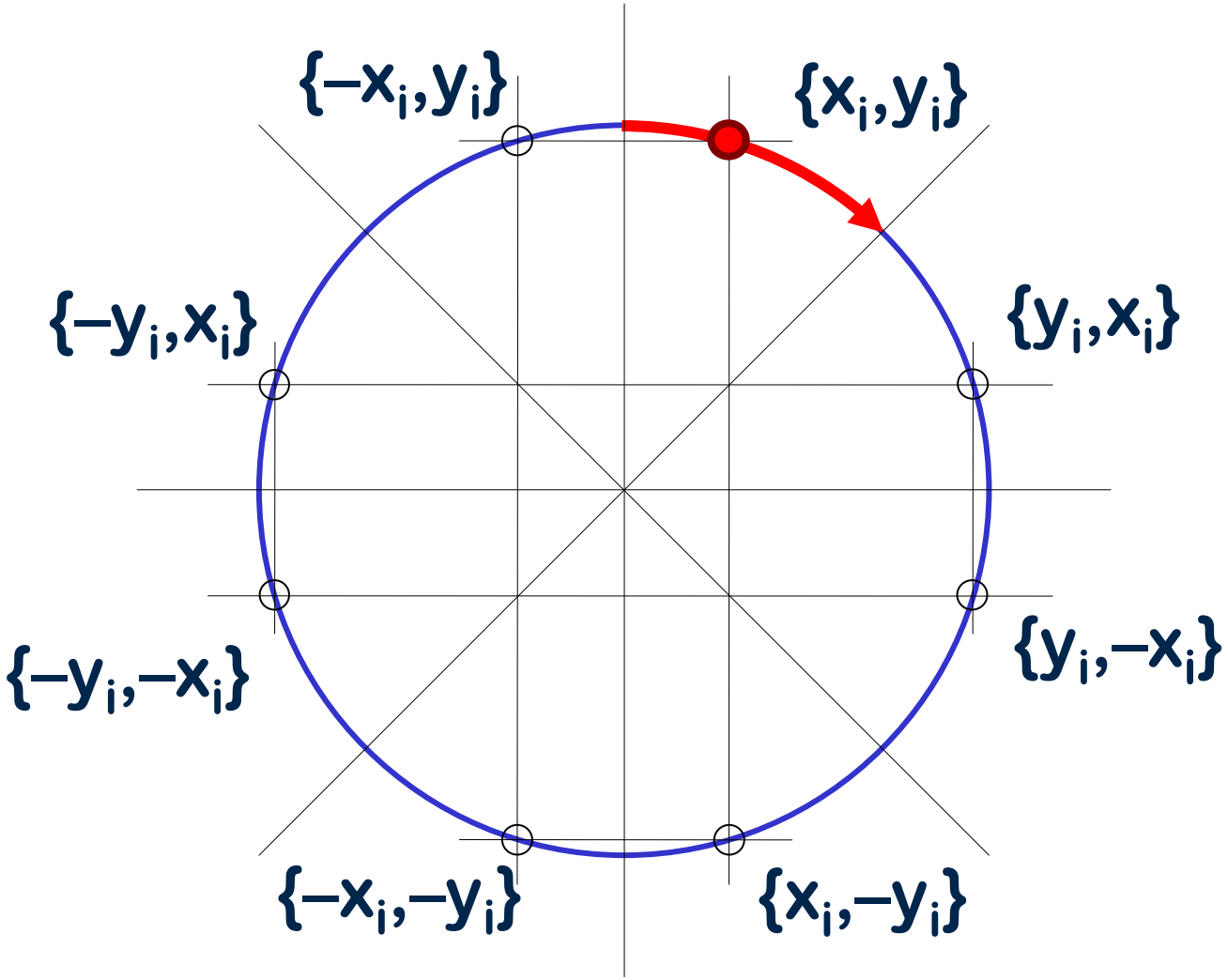


Algoritmo de línea de Bresenham

```
void linea_Bresenham(int x1, int y1, int x2, int y2) {
  int dx=x2-x1, dy=y2-y1, x,y,NE,E,D,xstep,ystep;
  if (!(dx|dy)) {pinta(x1,y1); return;}
  if (abs(dx)>=abs(dy)) {
    if (dx>=0) {
      x=x1; y=y1; NE=dx<<1; // (<<1 = *2, shift de un bit)
      ystep=1; if (dy<0) {ystep=-1; dy=-dy;}
      E=dy<<1; D=E-dx;
      while (x<x2) {
        pinta(x,y);
        if (D>0) {y+=ystep; D-=NE;}
        x++; D+=E;
      }
    }
  }
  pinta(x, y); // punto final
}

void linea_Bresenham(float x1, float y1, float x2, float y2) {
  linea_Bresenham(redondea(x1), redondea(x2), redondea(y1), redondea(y2));
}
```

Circunferencia



Circunferencia

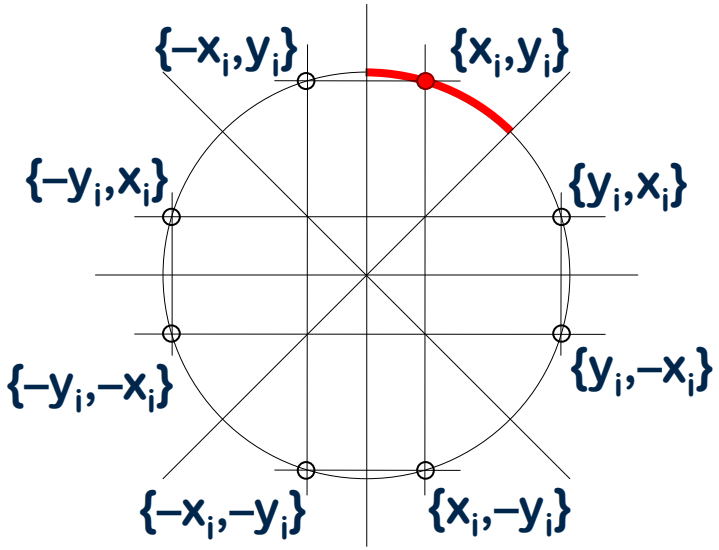
¿$(x_i + 1)^2 + (y_i - .5)^2 - r^2 > 0$?
Si: pintar SE
No: pintar E

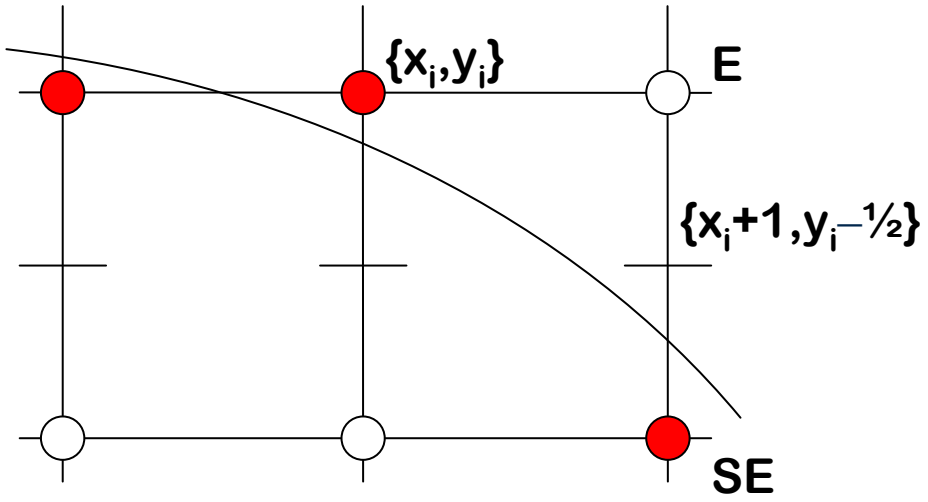
$d_i = (x_i + 1)^2 + (y_i - 1/2)^2 - r^2$
$\quad = x_i^2 + 2\,x_i + y_i^2 - y_i + 5/4 - r^2$
$E_i = 2\,x_i + 3$
$SE_i = 2\,(x_i - y_i) + 5$

$d_0 = d(0,r) = 5/4 - r$
$h = d - 1/4$
$d > 0 \Leftrightarrow h > -1/4 \Leftrightarrow h \geq 0$

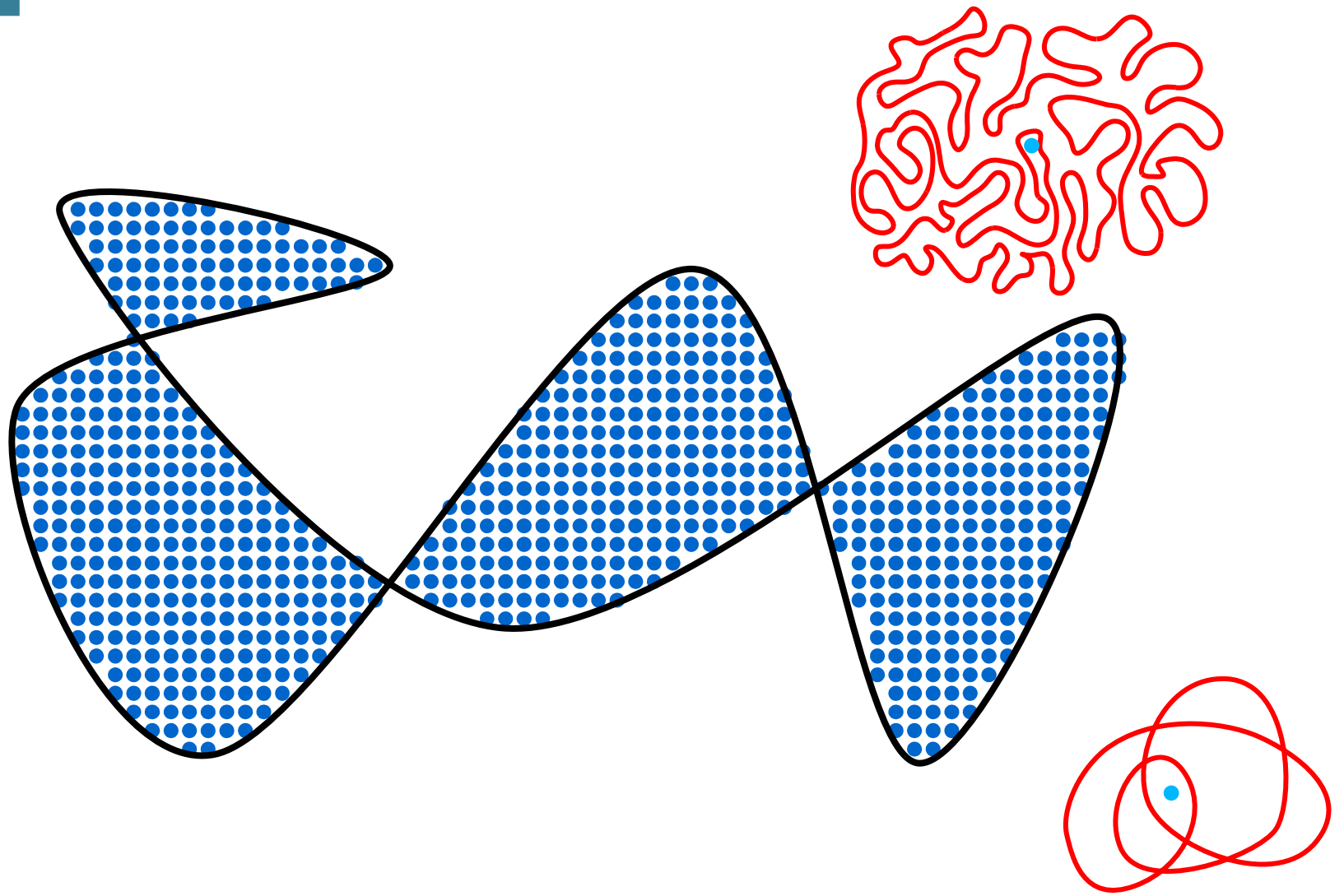
$h_0 = d_0 - 1/4 = 1 - r$
$E_0 = 3$
$SE_0 = 5 - 2\,r$

EE = 2
ESE = 2
SEE = 2
SESE = 4

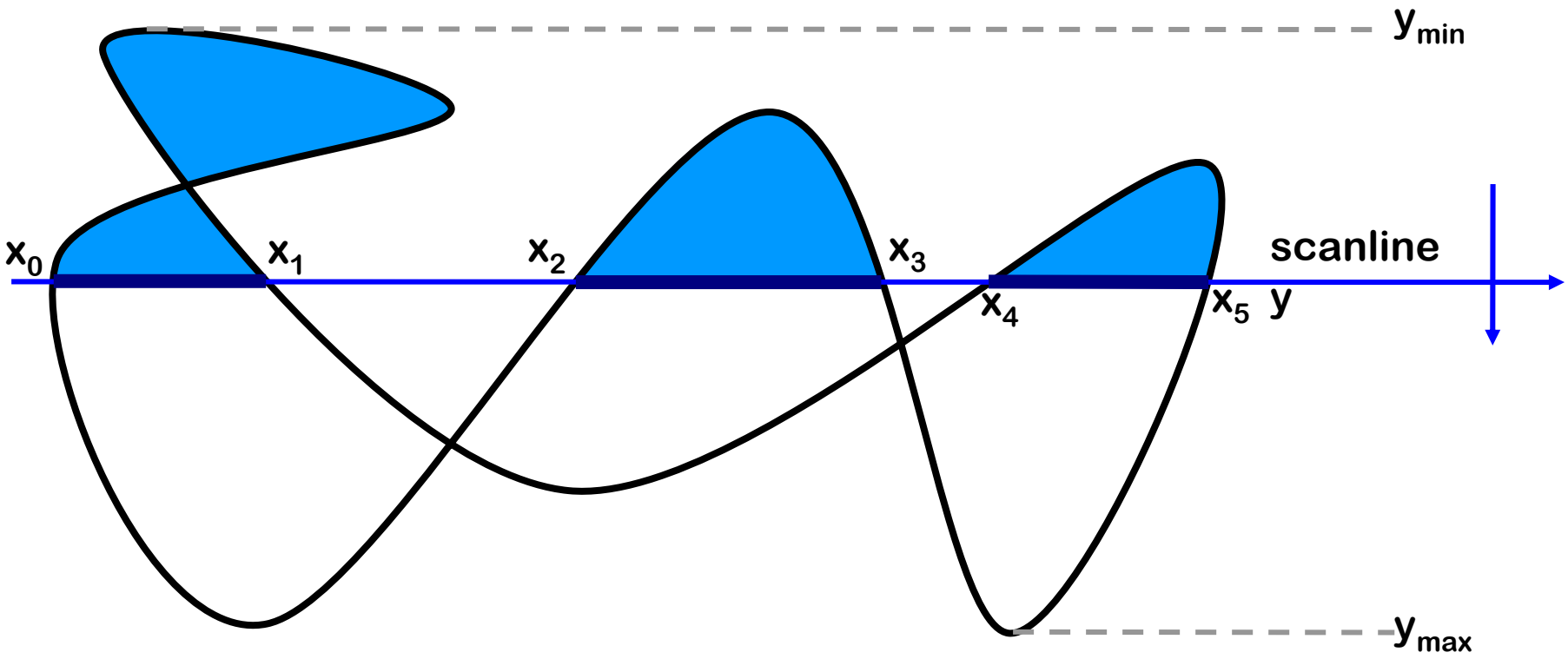




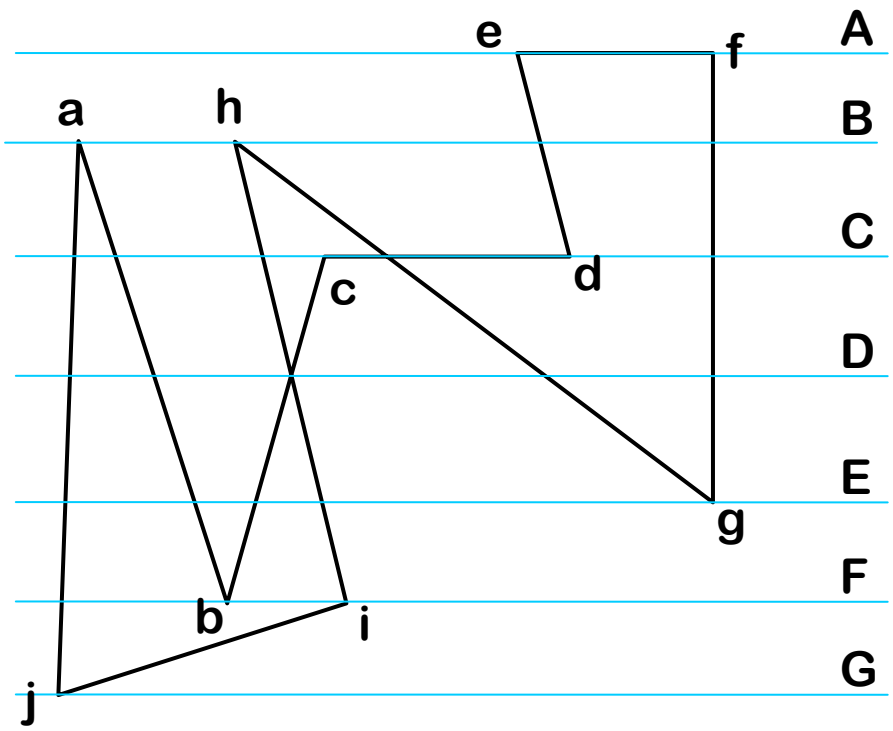
Rasterización de Figuras Cerradas

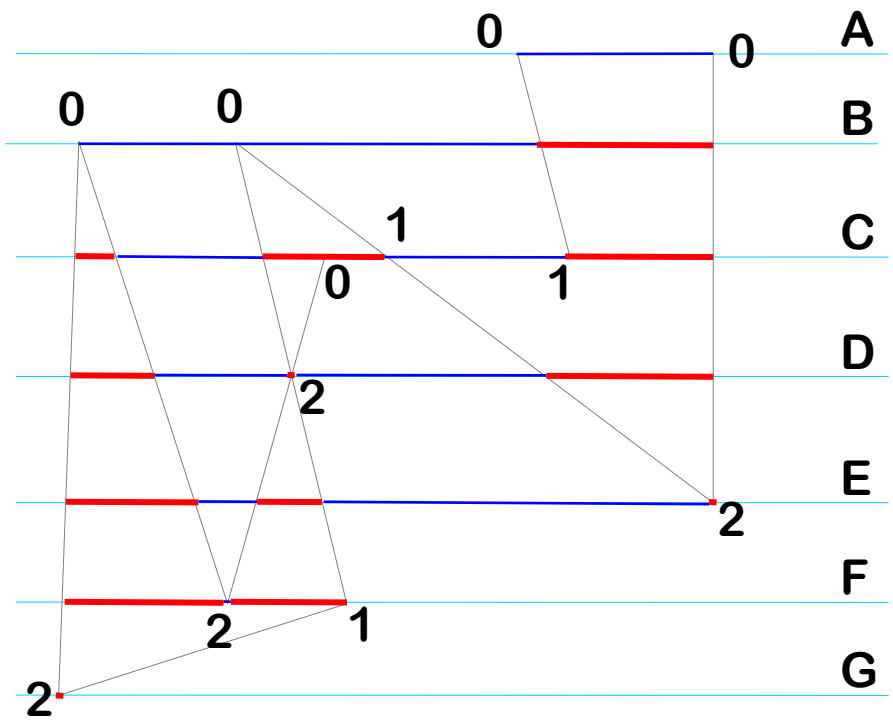


Rasterización de Figuras Cerradas



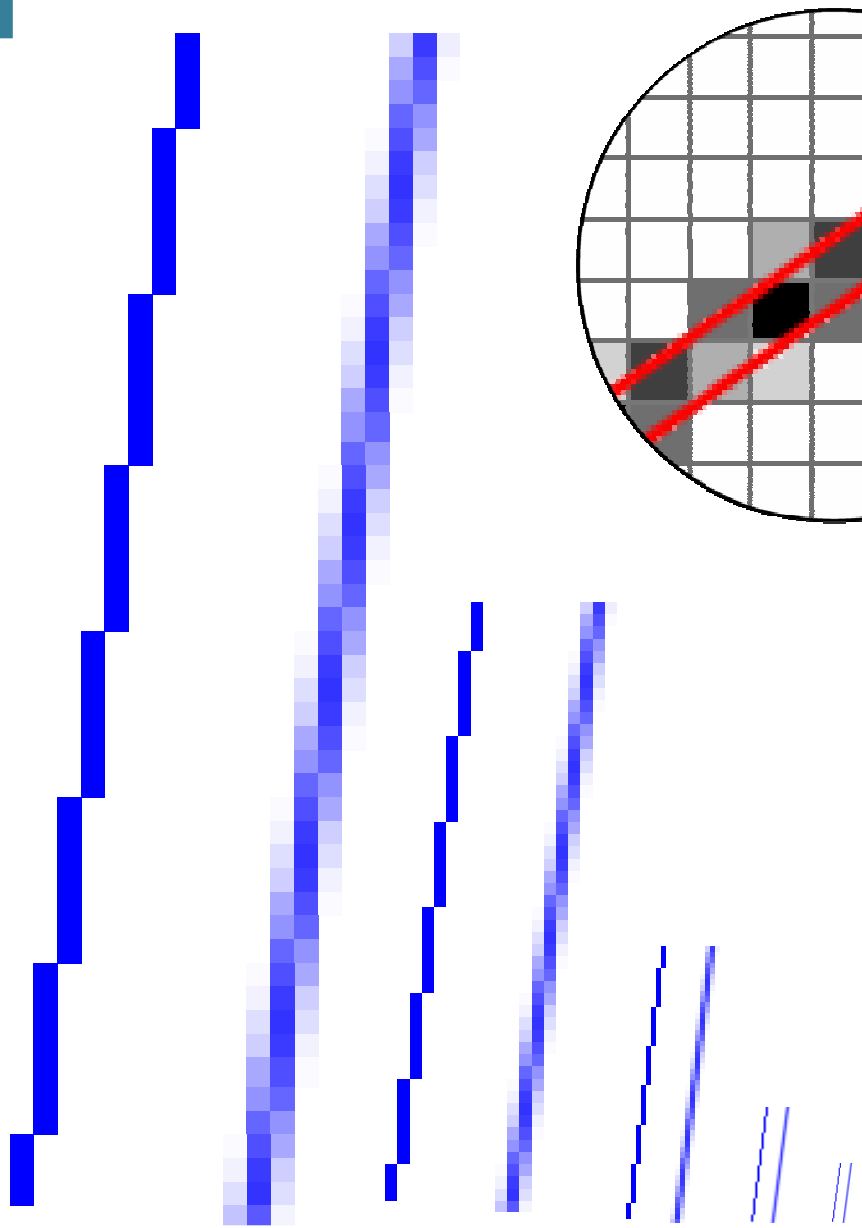
Rasterización de Poligonales Cerradas

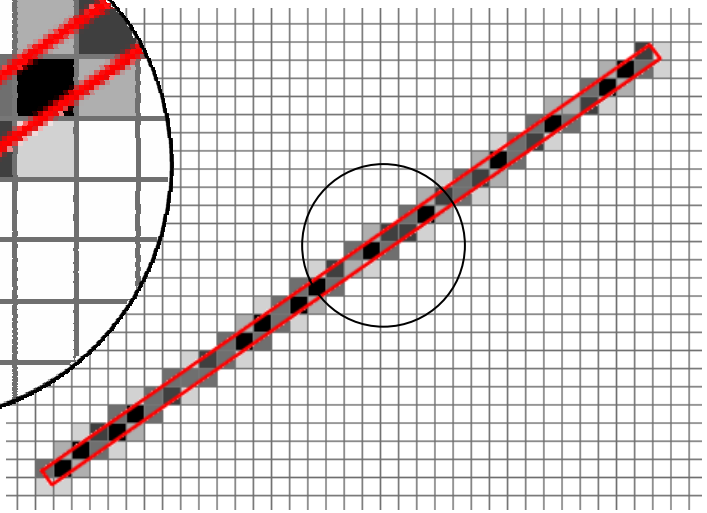


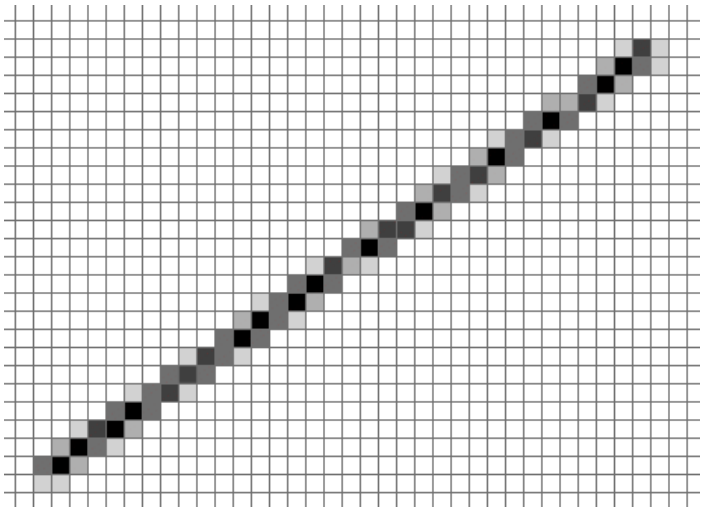


Pasos horizontales no se agregan a la lista.
Vértices: sólo los mínimos estrictos de su segmento.

Antialiasing







Bibliografía

- Hearn-Baker (cap 3 y 4)
- Angel (cap 7)