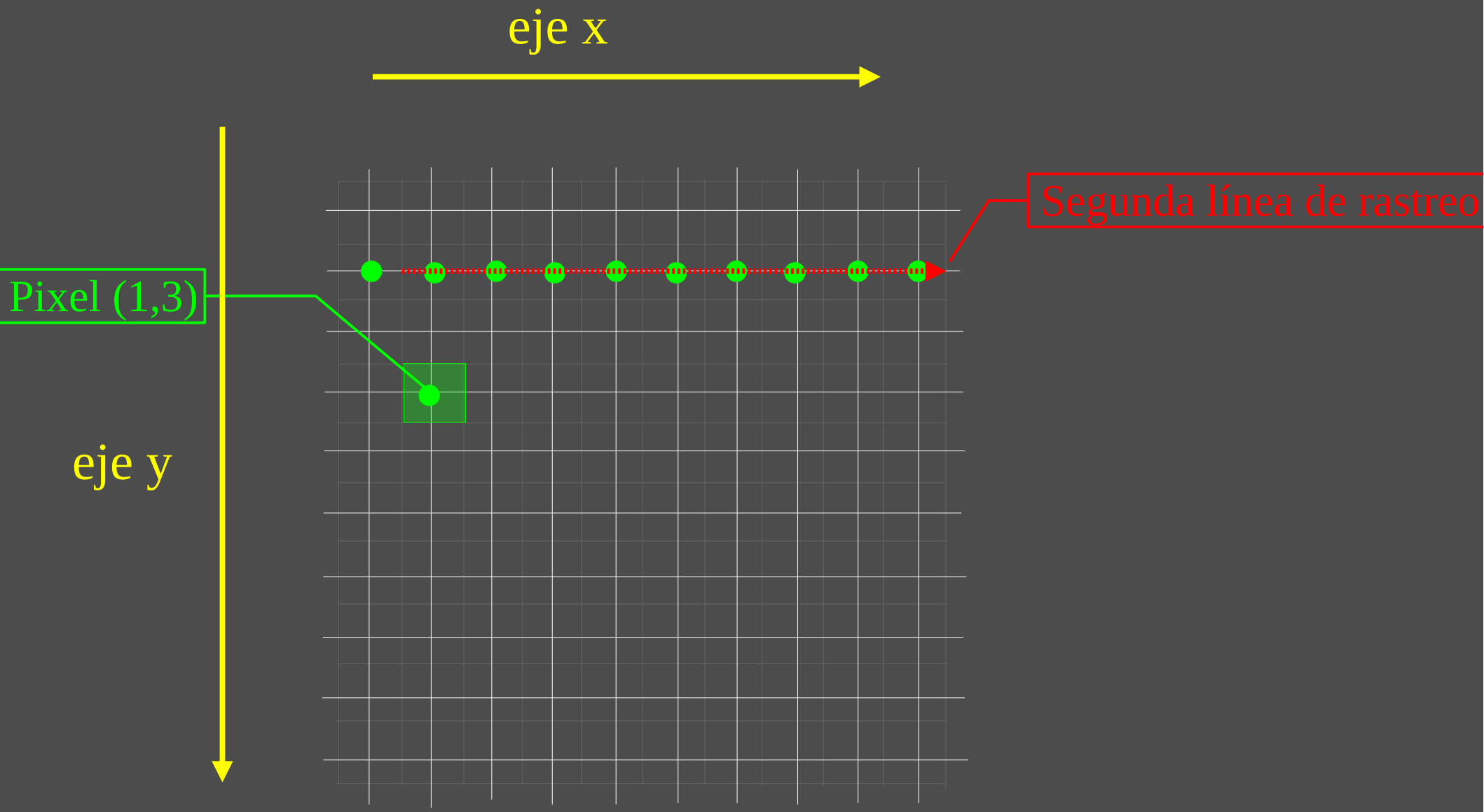
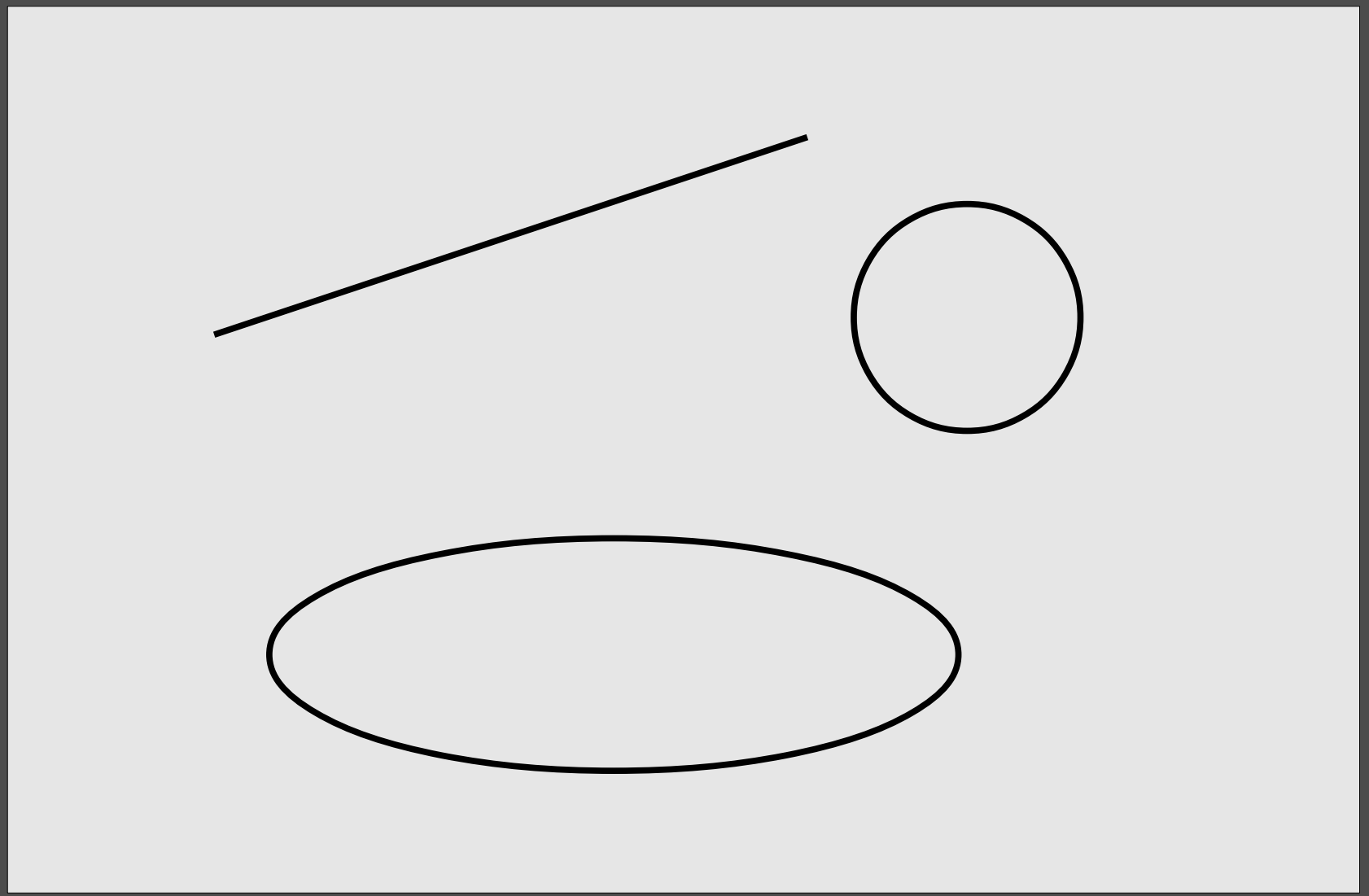
INDICE

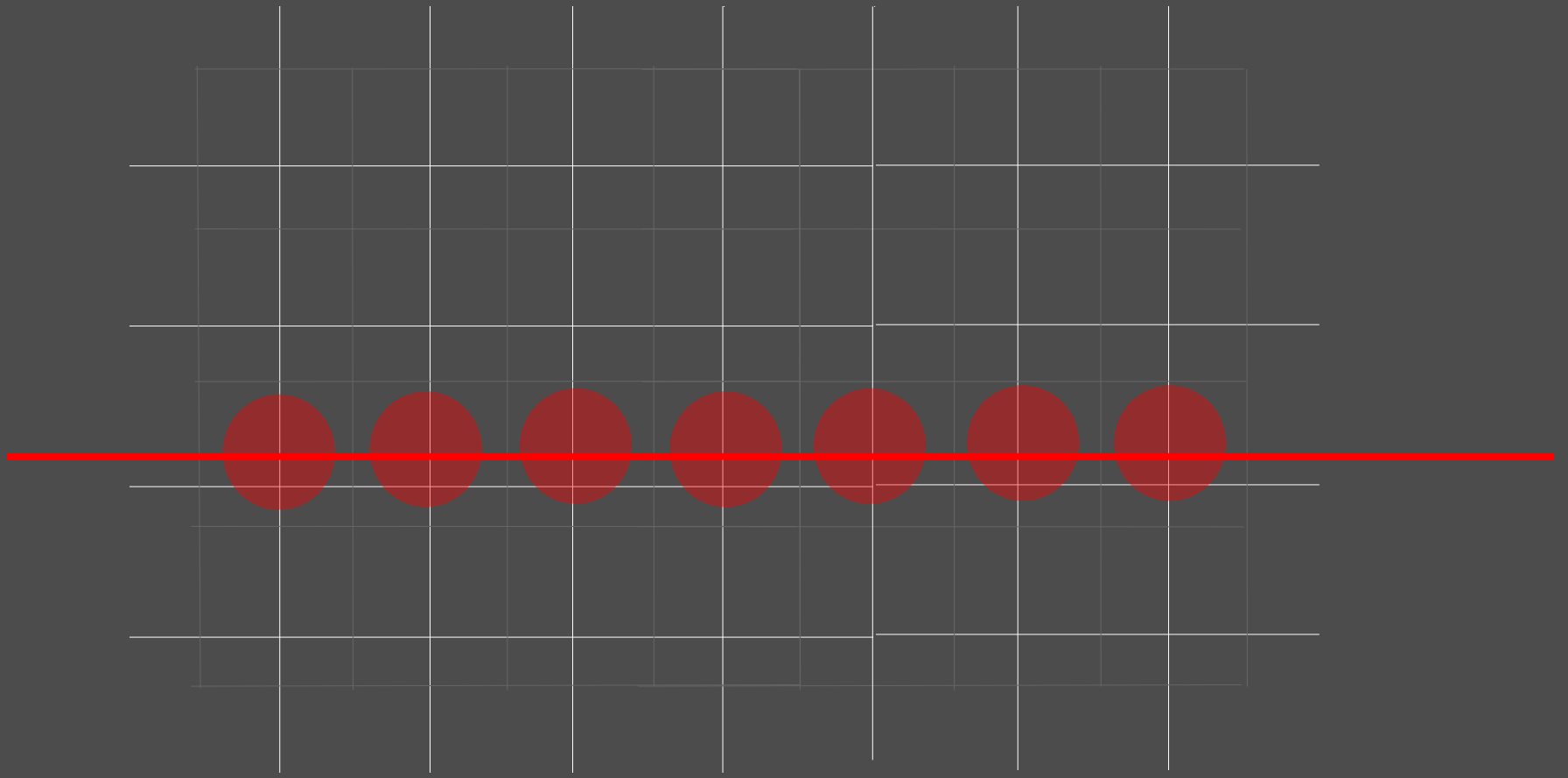
Intro / Convención



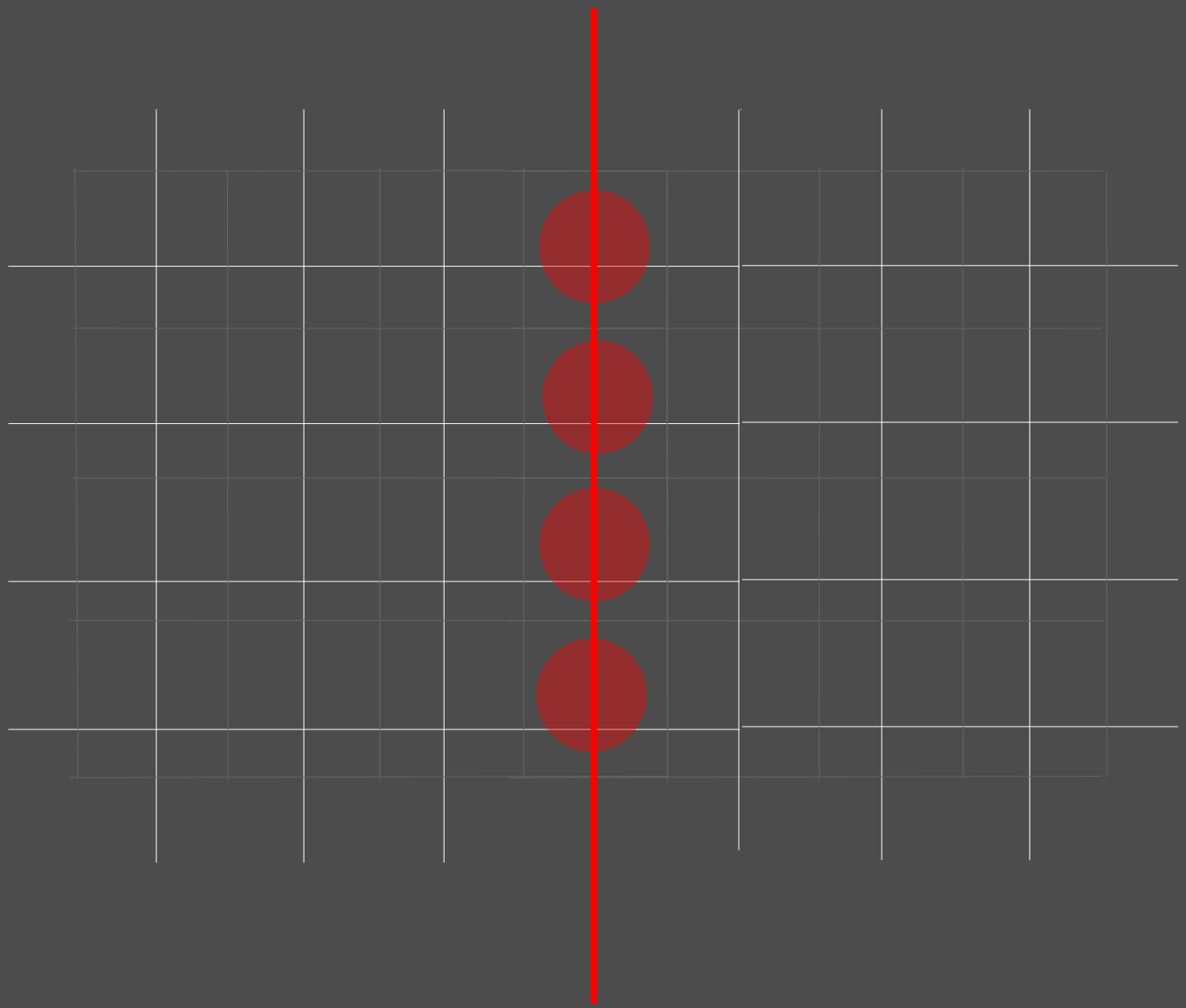
INTRO / Problemática



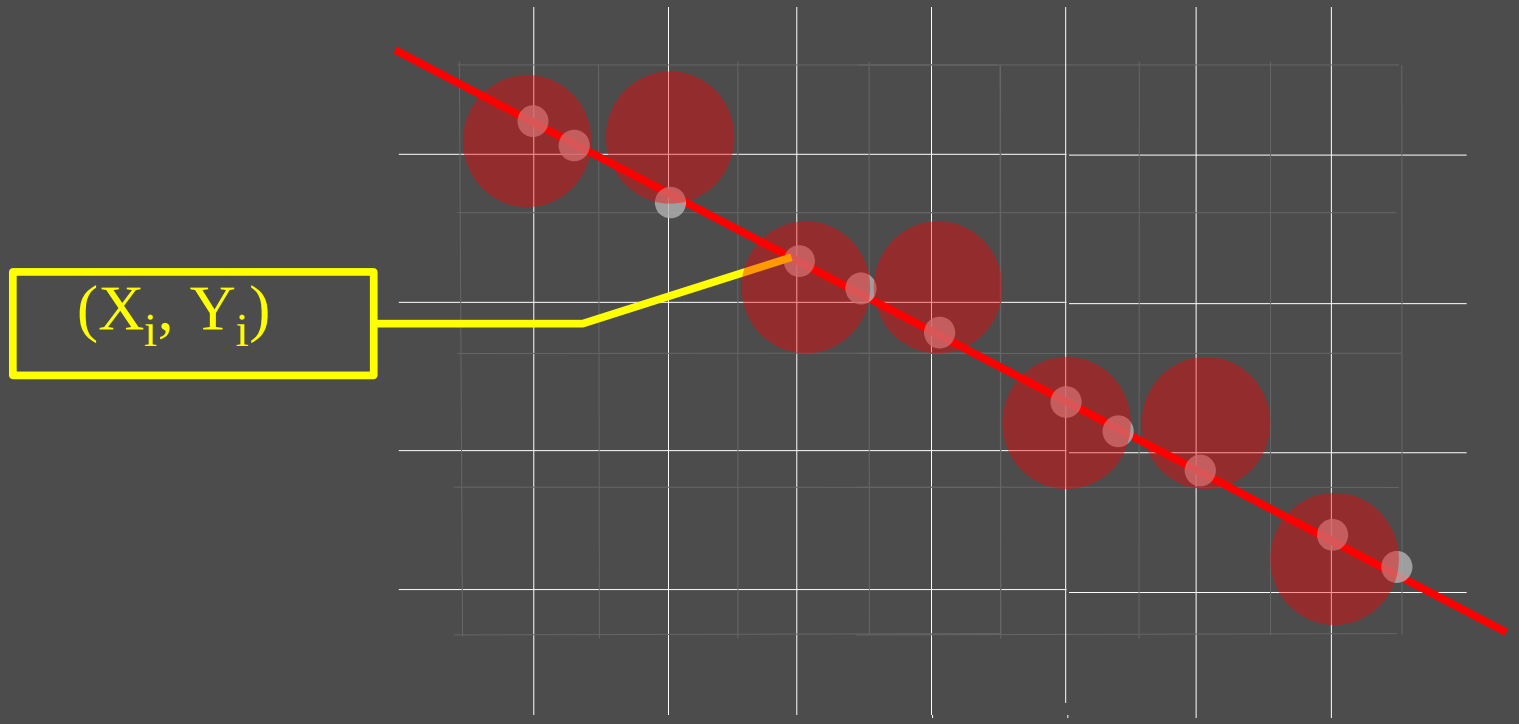
INTRO / Problemática



INTRO / Problemática



INTRO / Problemática



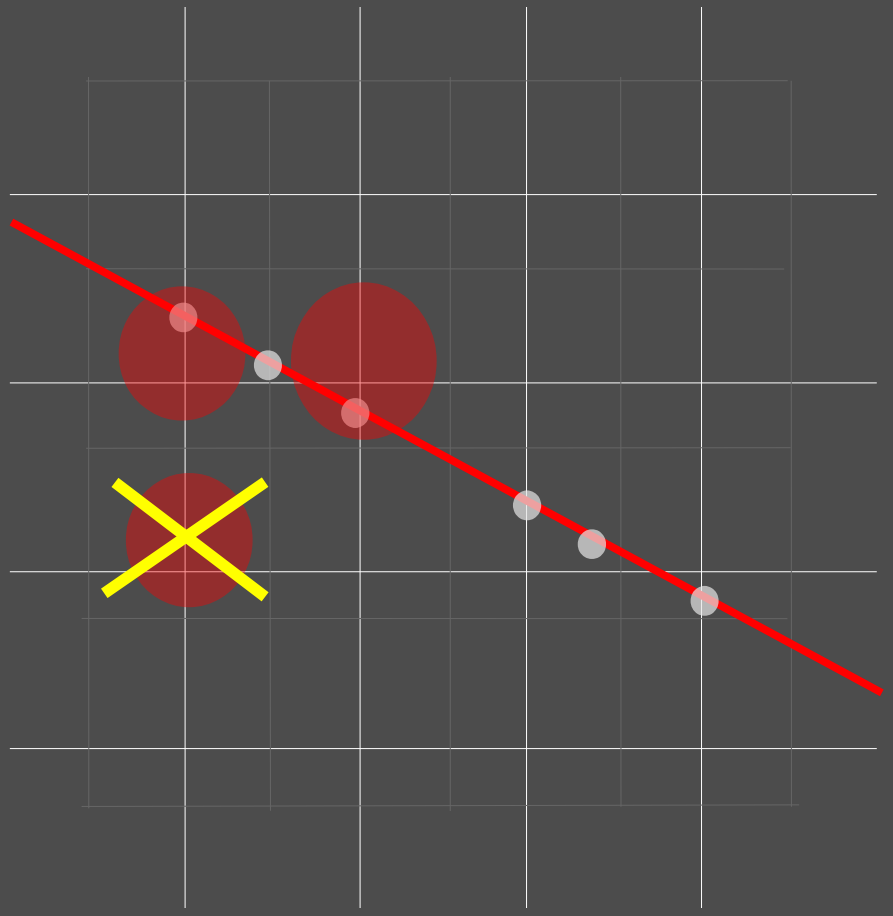
i-ésima intersección de la línea ideal con la malla

El problema consiste en encontrar las pixeles (coordenadas enteras de la malla) que mejor aproximen las intersecciones de la recta ideal con la malla.

ALGORITMOS PARA EL TRAZADO DE LINEAS

Observe que si m <= 1 ->

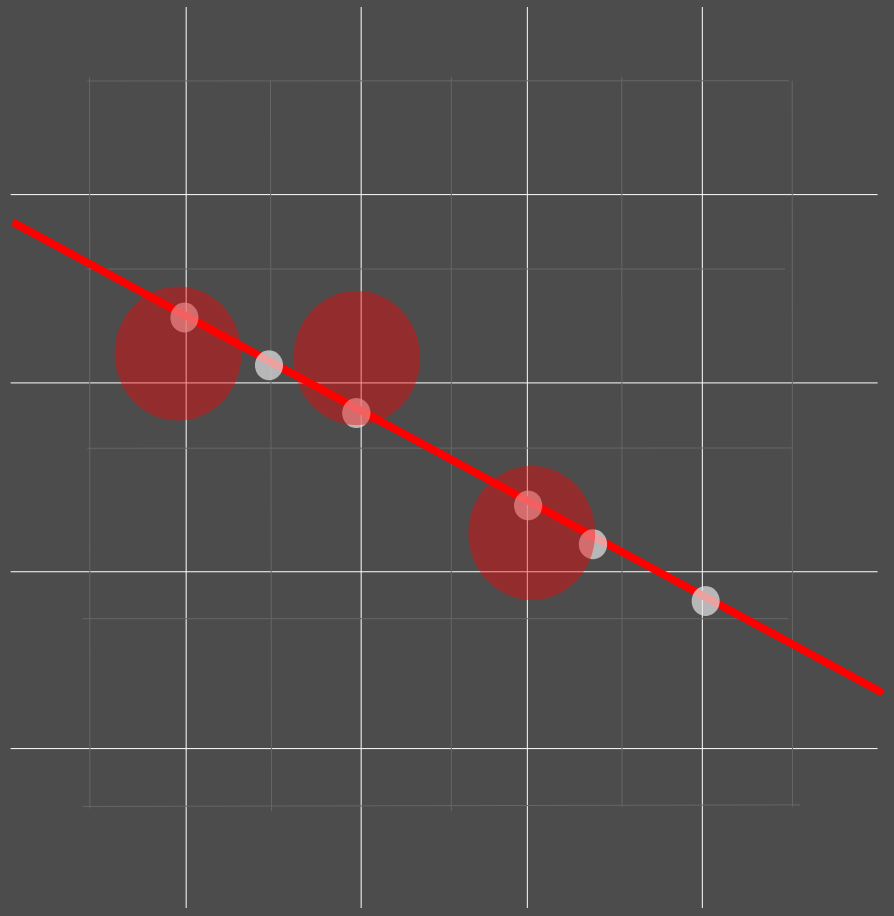
Una vez que se asigne un pixel,



el siguiente, no puede quedar en la misma columna

ALGORITMOS PARA EL TRAZADO DE LINEAS

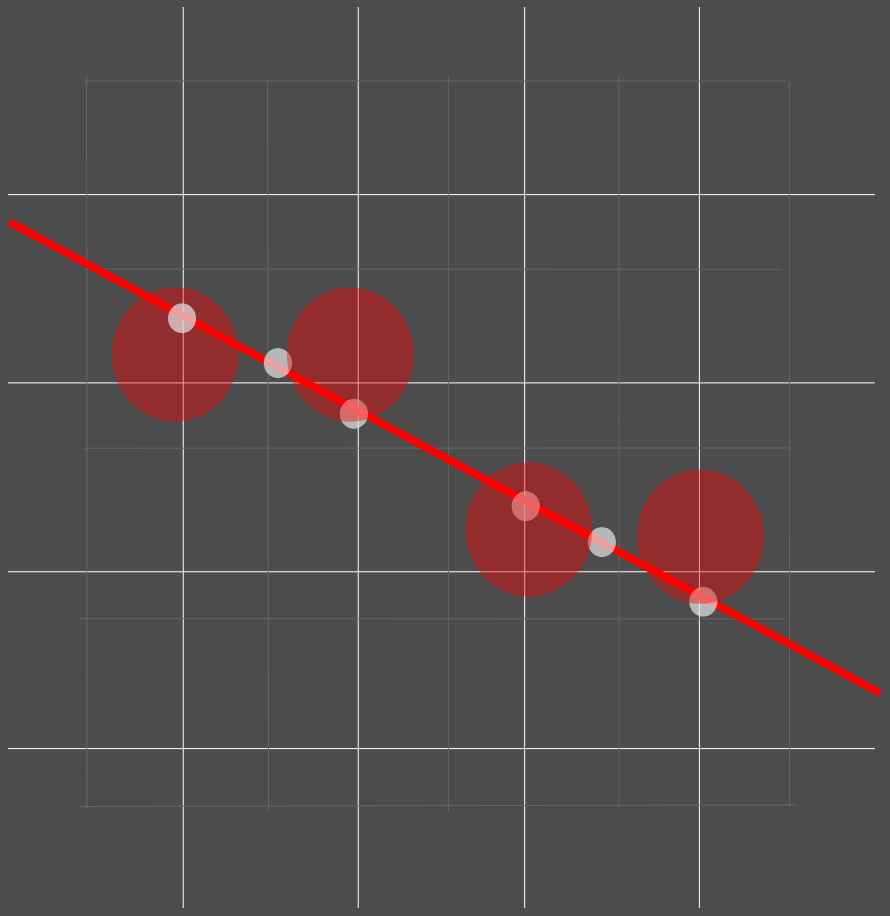
Una vez que se asigne un pixel,



el siguiente, no puede quedar después de una o más columnas

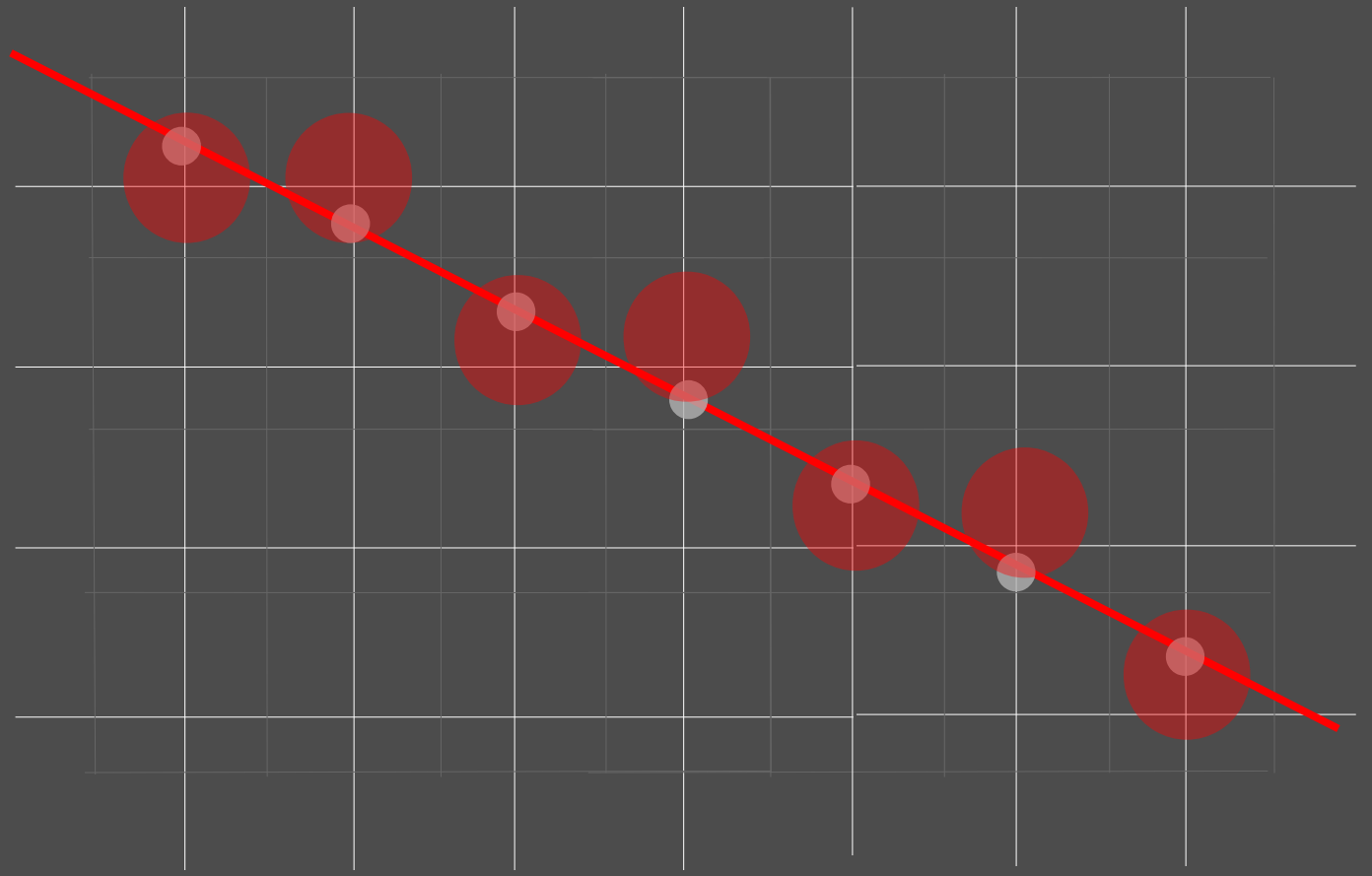
ALGORITMOS PARA EL TRAZADO DE LINEAS

Los dos resultados anteriores muestran que si m<=1 debe haber exactamente 1 pixel en cada columna,



por lo que interesan solo las intersecciones de la línea ideal con los columnas

ALGORITMOS PARA EL TRAZADO DE LINEAS



Se mostró que si $m <= 1$ -> $X_{i+1} = X_i + 1$

(para calcular el pixel a colorear basta con redoncear la intersección en Y)

ALGORITMOS PARA EL TRAZADO DE LINEAS / DDA

Cálculo de la Intersección en Y

Se puede cálcular Y_i como $mX_i + b$ Esto implica una multiplicación y una suma de punto flotante. Podemos evitar la multiplicación:

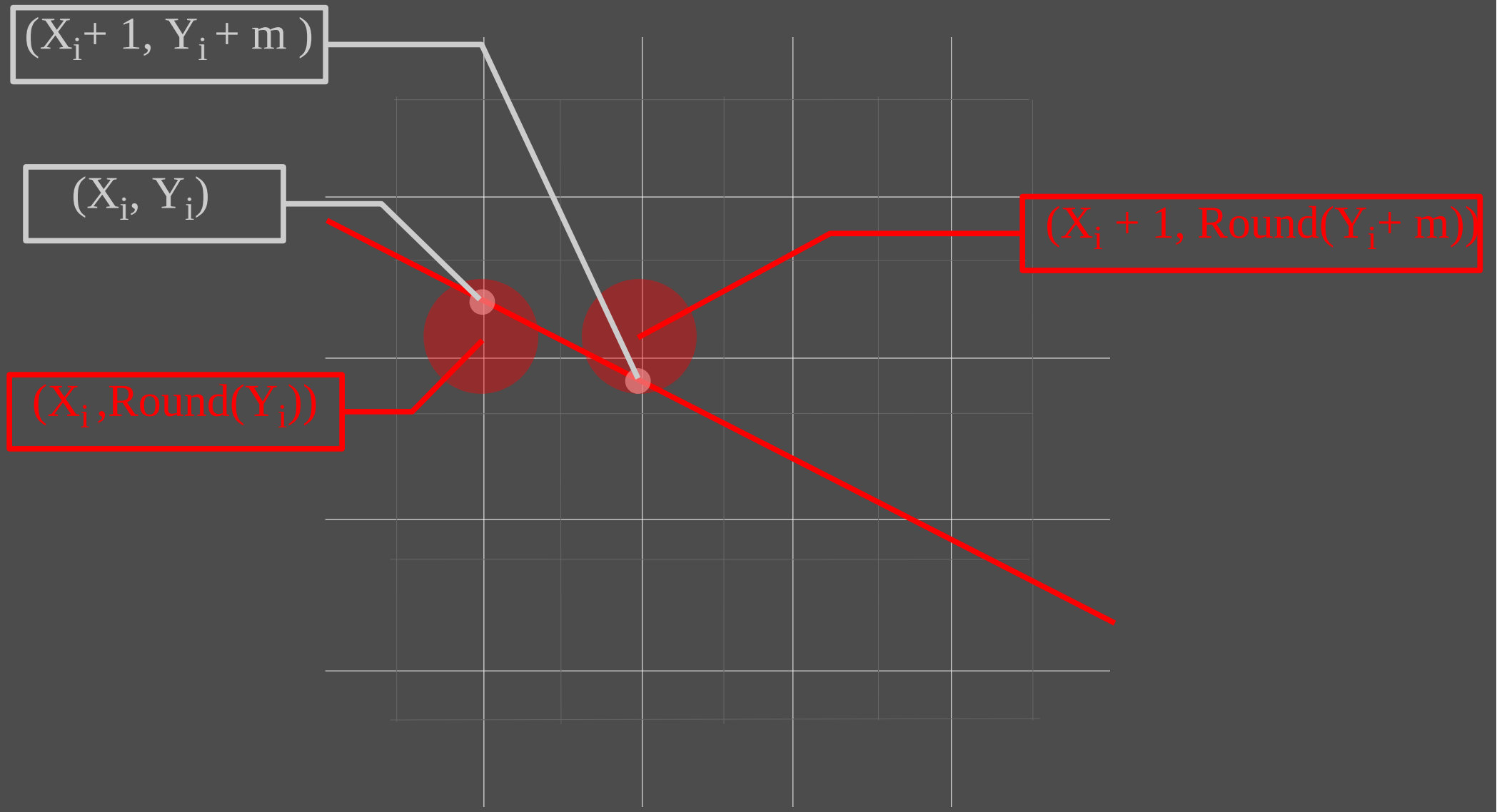
DDA (Digital Differential Analyzer)

Tenemos: $Y_i = m\ X_i + b$

Como $Y_{i+1} = m\ X_{i+1} + b$ y $X_{i+1} = X_i + 1$

entonces $Y_{i+1} = m\ (X_i + 1) + b = m\ X_i + b + m = Y_i + m$

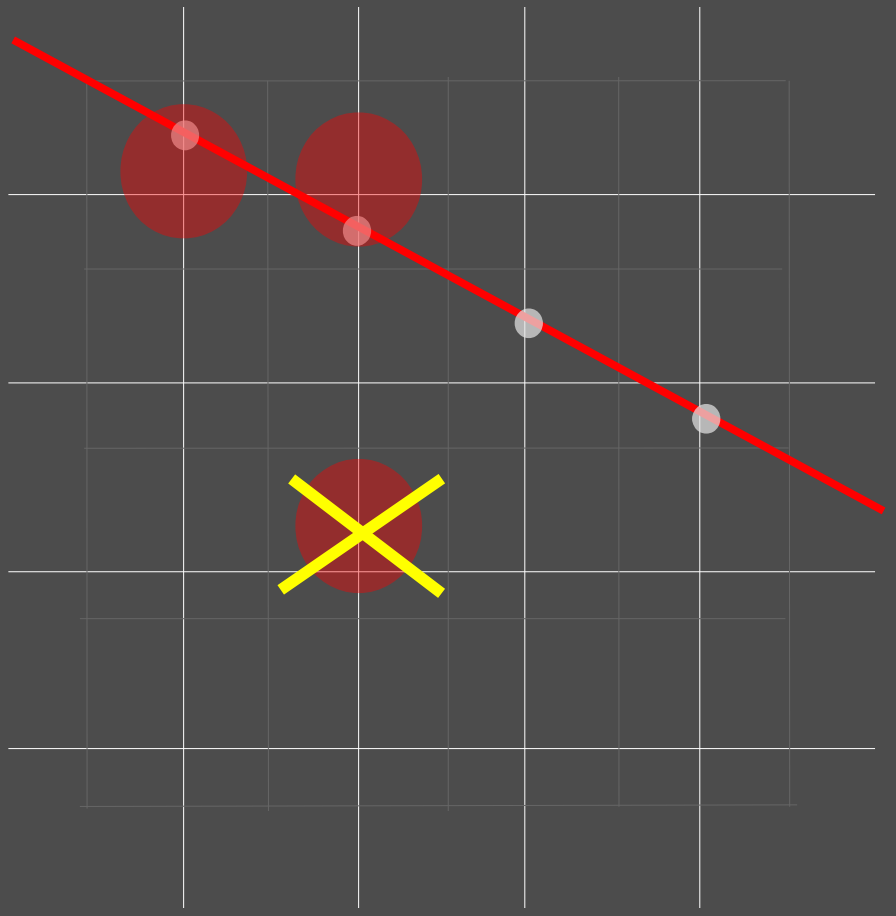
ALGORITMOS PARA EL TRAZADO DE LINEAS / DDA



ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Observe adicionalmente que si m <= 1 ->

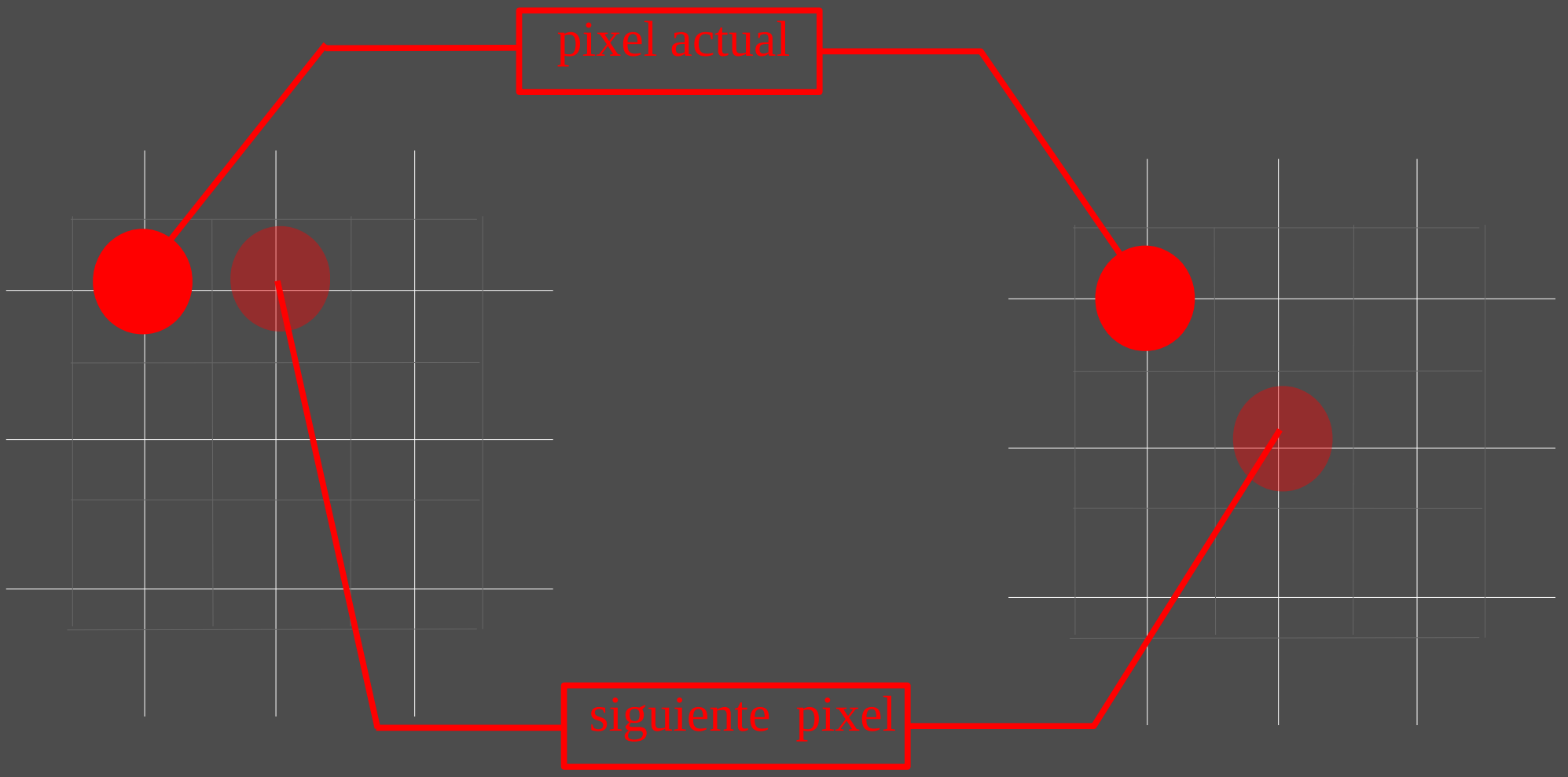
Una vez que se asigne un pixel,



el de la siguiente columna, no puede quedar alejado 2 o más renglones

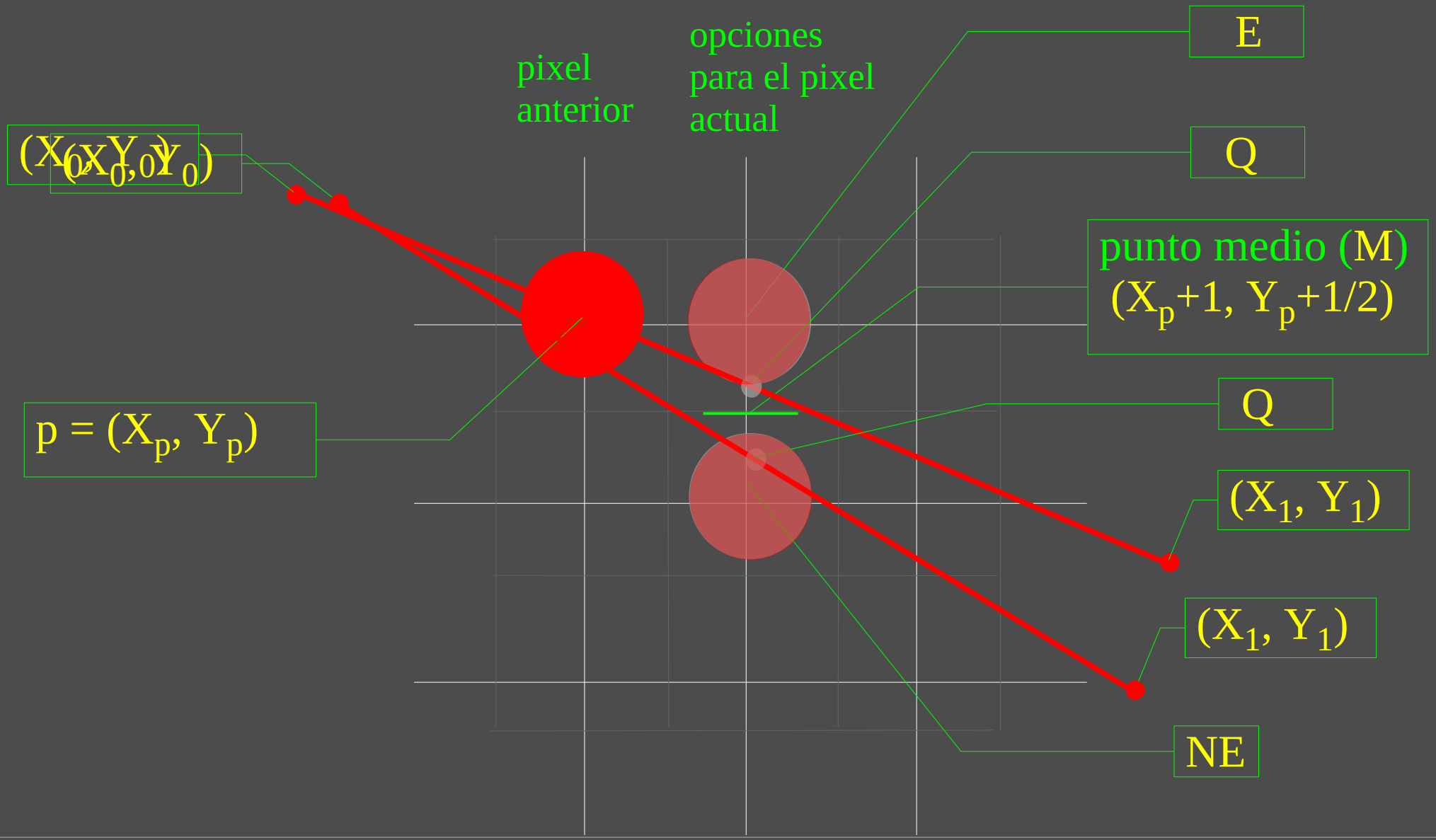
ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Patrones de relleno para el siguiente pixel



ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Factor de decisión a partir de los patrones:



ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Algoritmo de línea de punto medio (parte 1)

Se puede escribir la ecuación de la recta como $F(x,y) = ax + by + c = 0$

Si $dy = y_1 - y_0$ y $dx = x_1 - x_0$ -> $y = (dy/dx)x + B$

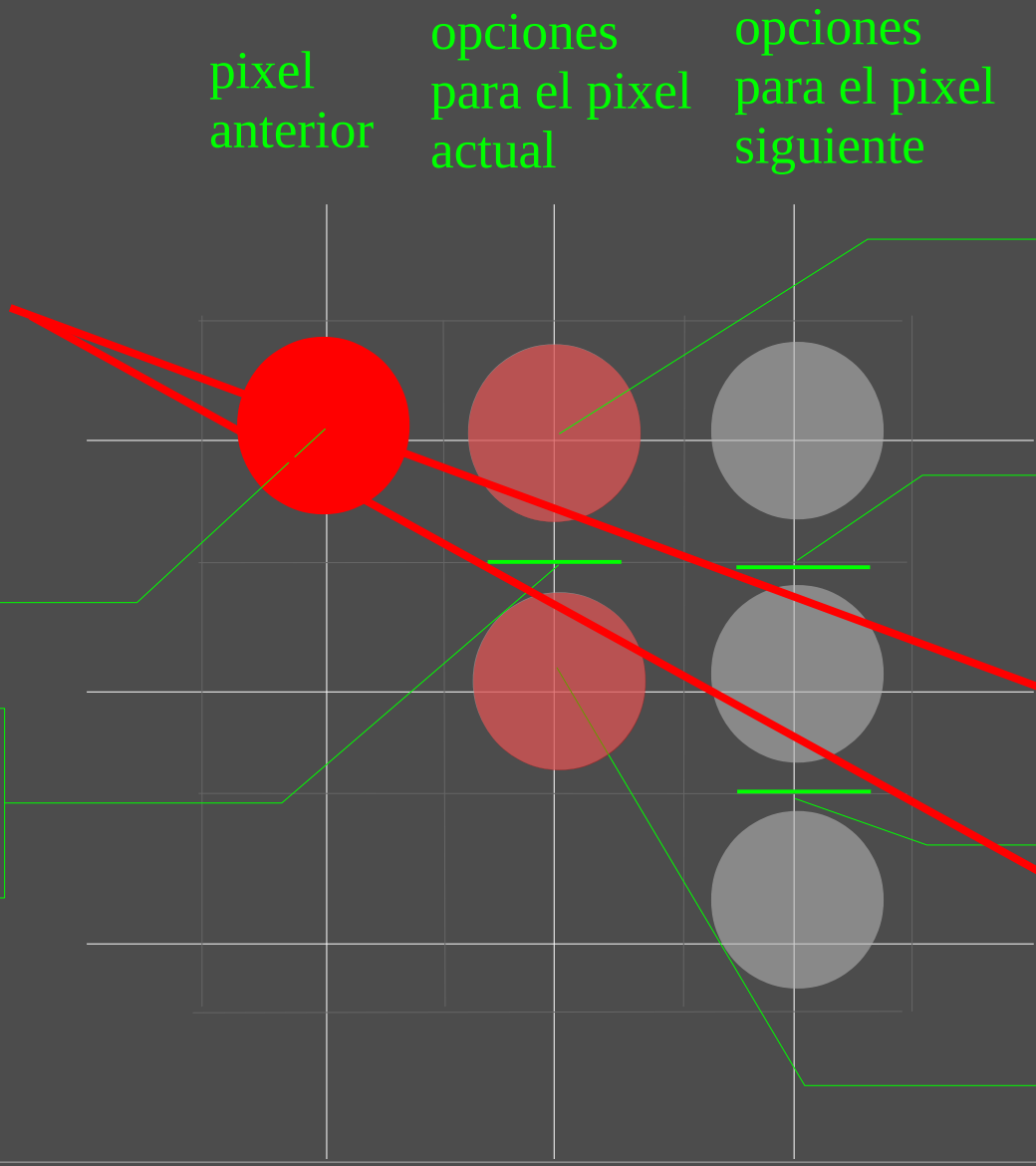
Si $y = (dy/dx)x + B$ -> $0 = (dy)\,x - (dx)\,y + B(dx) = F(x,y)$
con $a = dy$, $b = -dx$ y $c = Bdx$

Definimos una variable de decisión d, para evaluar F en el punto medio (Esta es 0 en la línea, positiva por debajo y negativa por encima).

$d = F(M) = F(X_p+1, Y_p+1/2)$, en resumen:
Si $d>0$ se elige el pixel NE
Si $d<0$ se elige el pixel E
Si $d=0$ se puede elegir NE o E. Por convención se elige el pixel E

ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Análisis del punto medio (M)



Algoritmo de línea de punto medio (parte 2)

Recuerde que d = F(M). Al desear evaluar d en el nuevo punto medio, se tienen dos casos:

CASO 1 Se elije E

$d_{nuevo} = F(X_p+2, Y_p+1/2) = a(X_p+2) + b(Y_p+1/2) + c$

como

$d_{viejo} = a(X_p+1) + b(Y_p+1/2) + c$

entonces

$d_{nuevo} = d_{viejo} + a$ (donde $a = dy = Y_1-Y_0$)

Algoritmo de línea de punto medio (parte 2)

CASO 2 Se elije NE

$d_{nuevo} = F(X_p+2, Y_p+3/2) = a(X_p+2) + b(Y_p+3/2) + c$

como

$d_{viejo} = a(X_p+1) + b(Y_p+1/2) + c$

entonces

$d_{nuevo} = d_{viejo} + a + b$ (donde $a = dy = Y_1-Y_0$ y $b=-dx=X_0-X_1$)

ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

d_{inicio}

Como el primer punto es (X_0, Y_0) entonces el primer punto medio se encuentra en $(X_0+ 1, Y_0+ ½)$ y podemos evaluar d, así:

$$F(X_0+ 1, Y_0+ ½) = a (X_0+ 1) + b (Y_0+ ½) + c$$
$$= a X_0 + b Y_0 + c + a + b/2$$
$$= F(X_0, Y_0) + a + b/2$$
$$= a + b/2 = dy - dx/2$$

Para eliminar la fracción, se redefine F (al multiplicar la función original por 2), así:

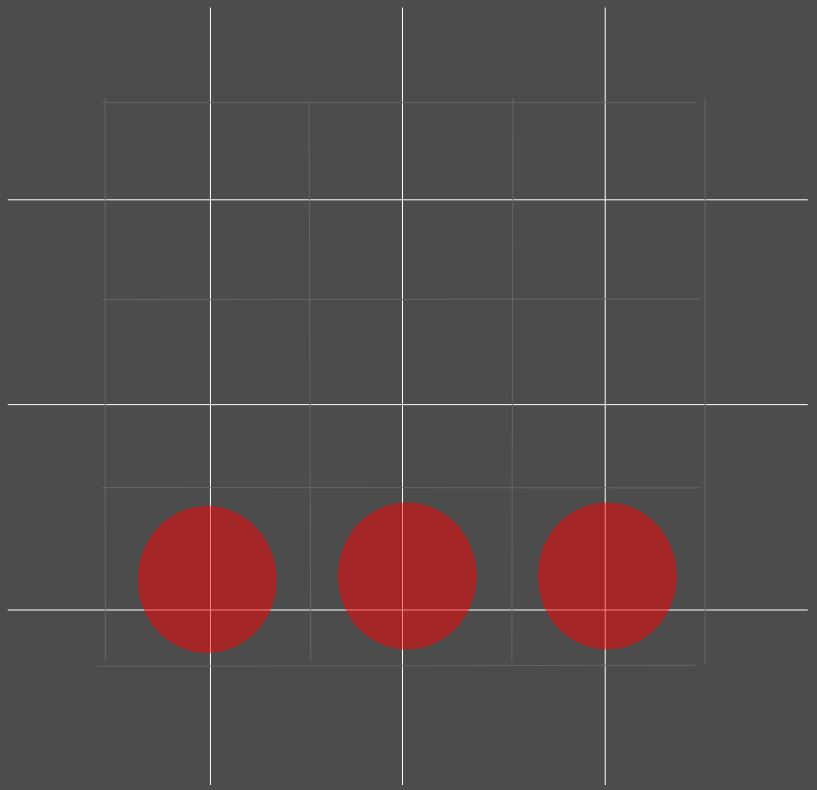
$F(x,y) = 2(ax + by + c)$

Observe que esto no afecta el signo de d. Finalmente se tiene:

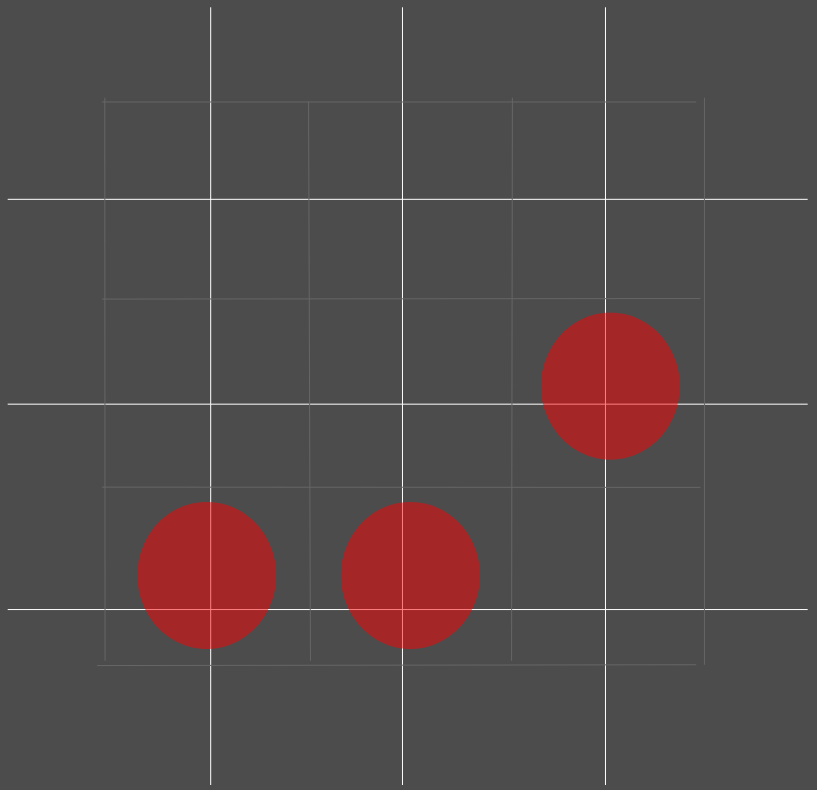
$d_{inicio} = 2dy - dx$

ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Algoritmo de doble paso



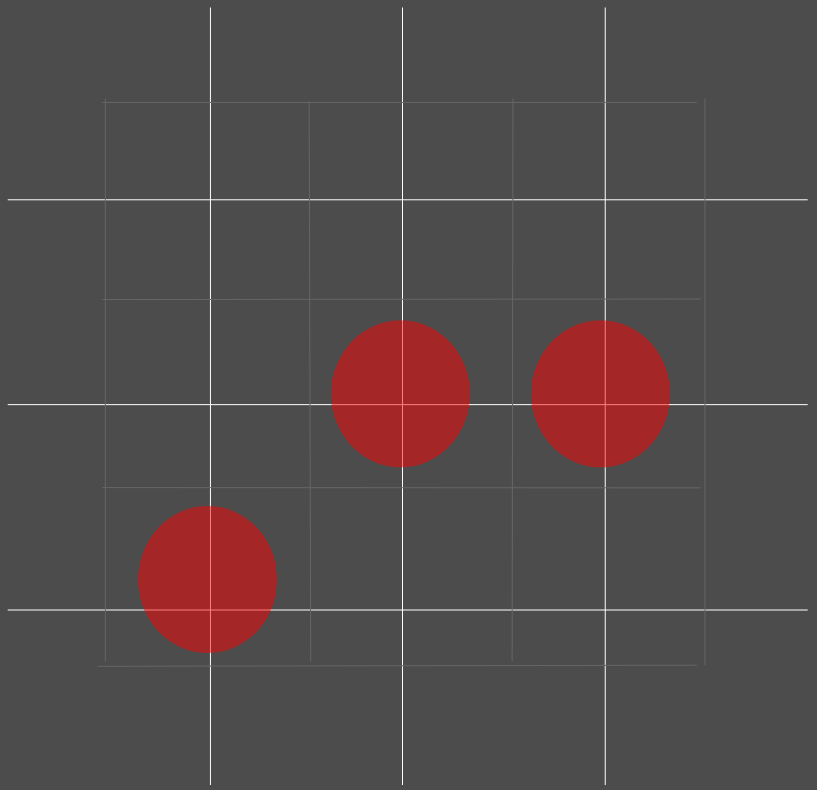
Patrón 1



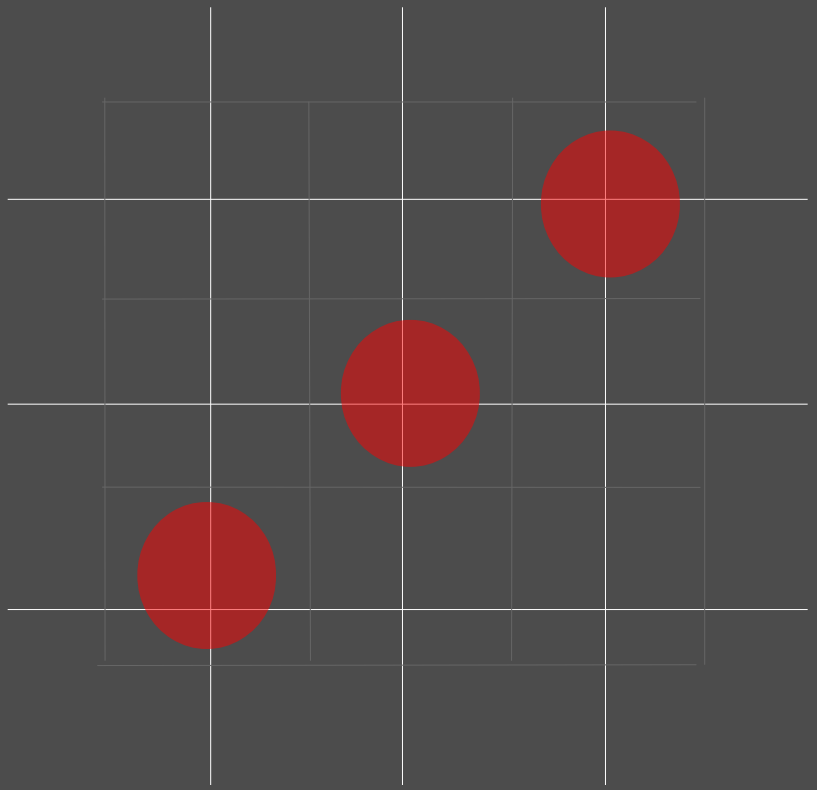
Patrón 2

ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Algoritmo de doble paso



Patrón 3



Patrón 4

ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Algoritmo de doble paso

Observaciones:
1. El patrón 1 y el 4 no puden ocurrir en la misma línea.
2. Si $m > ½$ no puede ocurrir el patrón 1.
3. Si $m < ½$ no puede ocurrir el patrón 4.

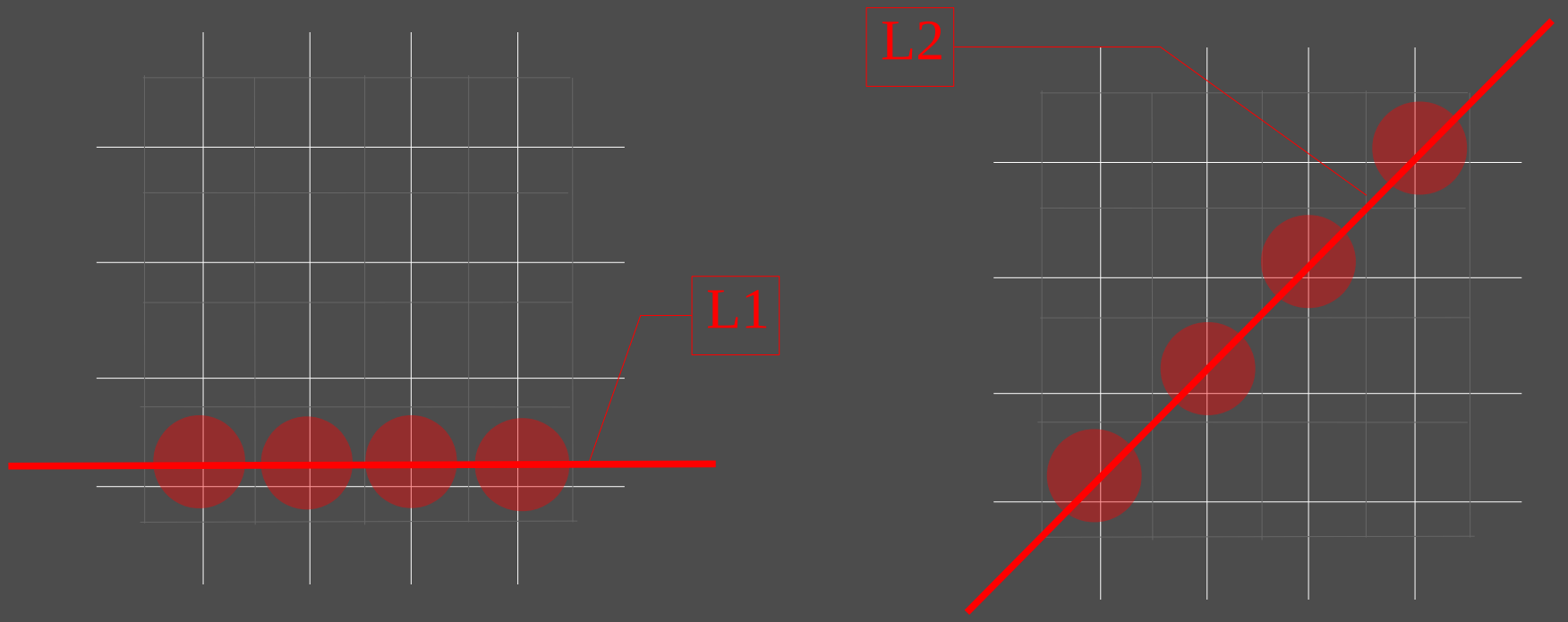
Aplicando un análisis similar al del caso simple, se obtiene, para $m < ½$:

Si $d < 0$ -> se emplea el patrón 1
Si $d >= 0$ -> si $d < 2dy$ -> se emplea el patrón 2
si $d >= 2dy$ -> se emplea el patrón 3

$d_{inicio} = 4\,dy - dx$
$d_{i+1} = d_i + 4\,dy$ (patrón 1)
$d_{i+1} = d_i + 4\,dy - 2\,dx$ (patrón 2 o 3)

ALGORITMOS PARA TRAZADO DE LINEAS / BRESENHAM

Brillo como función de la tangente de la recta



En ambas líneas se emplea el mismo número de pixeles (4), sin embargo L2 es SQRT(2) veces más larga que L1.

Si la intensidad de cada pixel es I, entonces la intensidad por unidad de longitud de L1 es I, mientras que para L2 es I/SQRT(2).

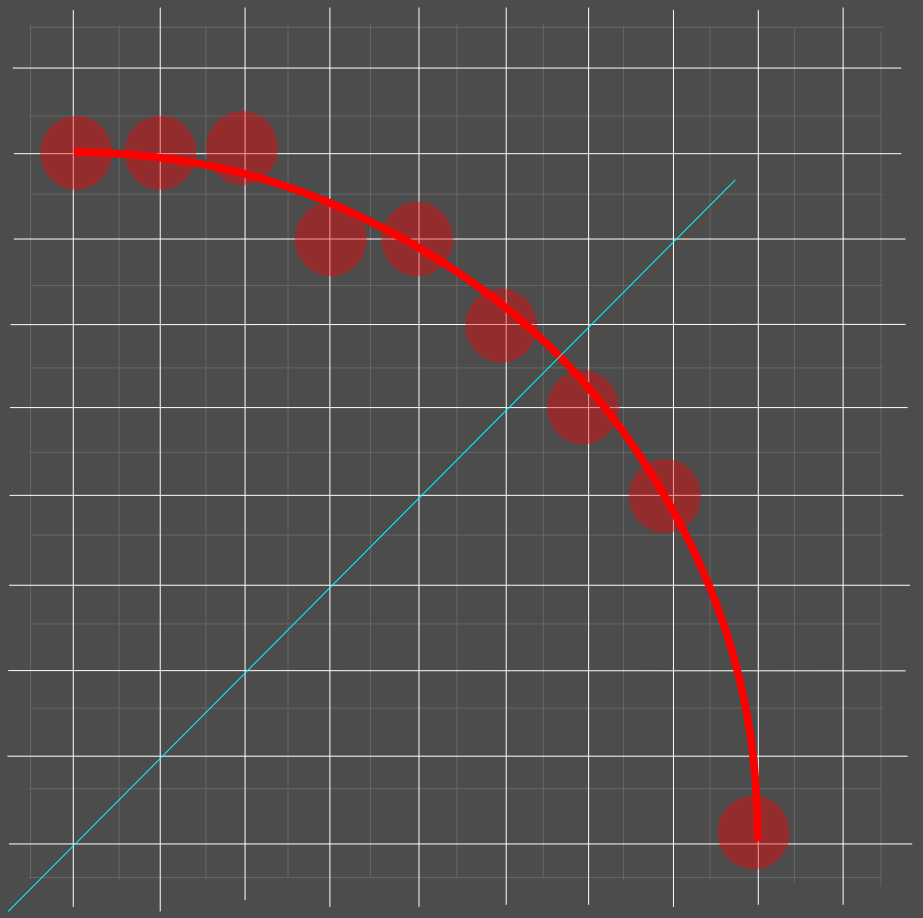
Se debe aplicar los ajustes de brillo de modo que la intensidad por unidad de longitud sea la misma (es decir, independiente de la tangente de la recta).

ALGORITMOS PARA TRAZADO DE CIRCUNFERENCIAS

$Y = +/- SQR (R^2 - X^2)$ Círculo centrado en el origen de radio R

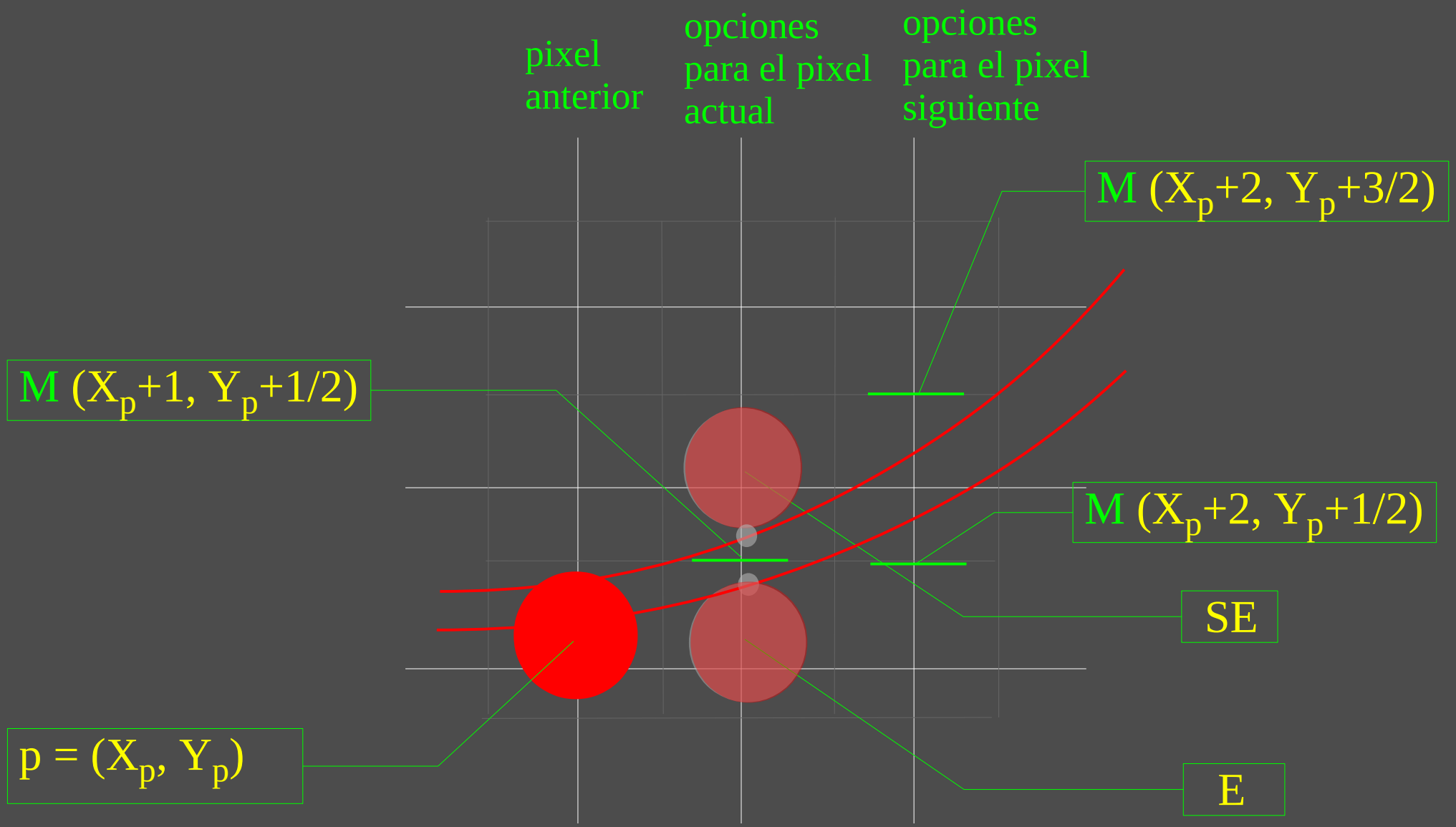
Al realizar simetría por cuadrante se tiene un efecto de separación de pixeles (producido al incrementar cada vez x en 1).

Se puede entonces emplear simetría de "medio cuadrante".



ALGORITMOS PARA TRAZADO DE CIRCUNFERENCIAS / BRES

Análisis en el segundo medio cuadrante en 0<=x<=(y=R/sqr(2))



ALGORITMOS PARA TRAZADO DE CIRCUNFERENCIAS / BRES

Algoritmo de círculo de punto medio (parte 1)

Se puede escribir la ecuación del círculo como $F(x,y) = x^2 + y^2 - R^2 = 0$

Definimos una variable de decisión d, para evaluar F en el punto medio (Esta es 0 en la círculo, positiva fuera y negativa dentro).

$d = F(M) = F(X_p+1, Y_p+1/2)$, entonces teniendo en cuenta el nuevo punto medio (y d_{nuevo}) tenemos:

Caso 1:

Si $d_{viejo} < 0$ se elige el pixel E. Expresando d_{nuevo} en función de d_{viejo} se tiene:

$$d_{nuevo} = d_{viejo} + (2X_p+3)$$

ALGORITMOS PARA TRAZADO DE CIRCUNFERENCIAS / BRES

Algoritmo de círculo de punto medio (parte 2)

Caso 2:

Si $d_{viejo} >= 0$ se elige el pixel SE. Expresando d_{nuevo} en función de d_{viejo} se tiene:

$$d_{nuevo} = d_{viejo} + (2X_p - 2Y_p + 5)$$

ALGORITMOS PARA TRAZADO DE CIRCUNFERENCIAS / BRES

d_{inicio}

Como el primer punto es (0, R) entonces el primer punto medio se encuentra en (1, R - ½) y podemos evaluar d, así:

$F(1, R - ½) = 1 + (R^2 - R + 1/4) - R^2$

$= 5/4 - R$

Para eliminar la fracción, definimos una nueva variable de decisión h, así:
h = d-1/4 (y se sustituye h+1/4 por d)

Esto produce:

$h_{inicio} = 1 - R$

La comparación d<0 se converte en h<1/4. Ahora, como h comienza con un valor entero y es incrementado por valores enteros, h<1/4 se puede cambiar por h<0.

Diferencias de Segundo Orden

Observe que los incrementos ΔE y ΔSE en:

Caso 1 (se elige el pixel E) $d_{nuevo} = d_{viejo} + (2X_p+3)$ (ΔE)

y

Caso 2 (se elige el pixel SE) $d_{nuevo} = d_{viejo} + (2X_p-2Y_p+5)$ (ΔSE)

son funciones lineales en (x, y). La idea es evaluar diferencias de segundo orden para Δ, para ambos casos:

Caso 1 (se elige el pixel E)

El punto de evaluación se mueve de (X_p, Y_p) a (X_p+1, Y_p), entonces:

ΔE_{nuevo} en $(X_p+1, Y_p) = 2(X_p+1) + 3$, es decir, $\Delta E_{nuevo} = \Delta E_{viejo} + 2$

ΔSE_{nuevo} en $(X_p+1, Y_p) = 2(X_p+1) -2Y_p + 5$, es decir, $\Delta SE_{nuevo} = \Delta SE_{viejo} + 2$

Caso 2 (se elige el pixel SE)

El punto de evaluación se mueve de (X_p, Y_p) a (X_p+1, Y_p-1), entonces:

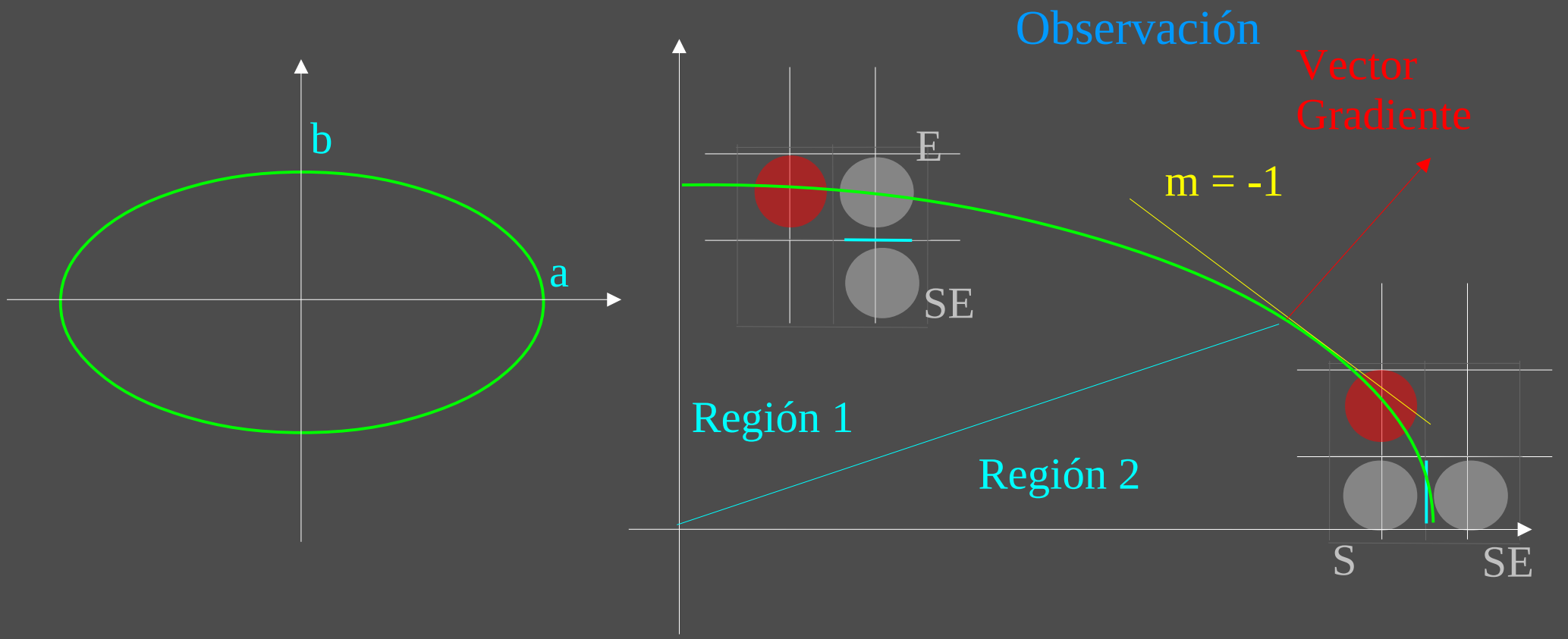
ΔE_{nuevo} en $(X_p+1, Y_p-1) = 2(X_p+1) + 3$, es decir, $\Delta E_{nuevo} = \Delta E_{viejo} + 2$

ΔSE_{nuevo} en $(X_p+1, Y_p-1) = 2(X_p+1) -2(Y_p-1)+ 5$, es decir, $\Delta SE_{nuevo} = \Delta SE_{viejo} + 4$

ALGORITMOS PARA TRAZADO DE ELIPSES

Algoritmo de elipse de punto medio

Se puede escribir la ecuación de la elipse como $F(x,y) = b^2x^2 + a^2y^2 - a^2b^2 = 0$



ALGORITMOS PARA TRAZADO DE ELIPSES

Algoritmo de elipse de punto medio (algebra)

Cambio de Región

Grad $F(x, y) = \partial F/\partial x\ \mathbf{i} + \partial F/\partial y\ \mathbf{j} = 2b^2x\ \mathbf{i} + 2a^2y\ \mathbf{j}$

Esto da que si para el siguiente punto medio se tiene:

Si $a^2(y_p - 1/2) \leq b^2(x_p + 1)$ entonces pasamos de la región 1 a la 2

Región 1 (Variable de decisión: d = F(x,y) se evalúa en $(x_p+1, y_p-1/2)$)

Si se escoge E entonces $d_{nuevo} = d_{viejo} + b^2(2x_p+3)$

Si se escoge SE entonces $d_{nuevo} = d_{viejo} + b^2(2x_p+3) + a^2(-2y_p+2)$

Región 2 (Variable de decisión: h = F(x,y) se evalúa en $(x_p+1/2, y_p-1)$)

Si se escoge SE entonces $h_{nuevo} = h_{viejo} + b^2(2x_p+2) + a^2(-2y_p+3)$

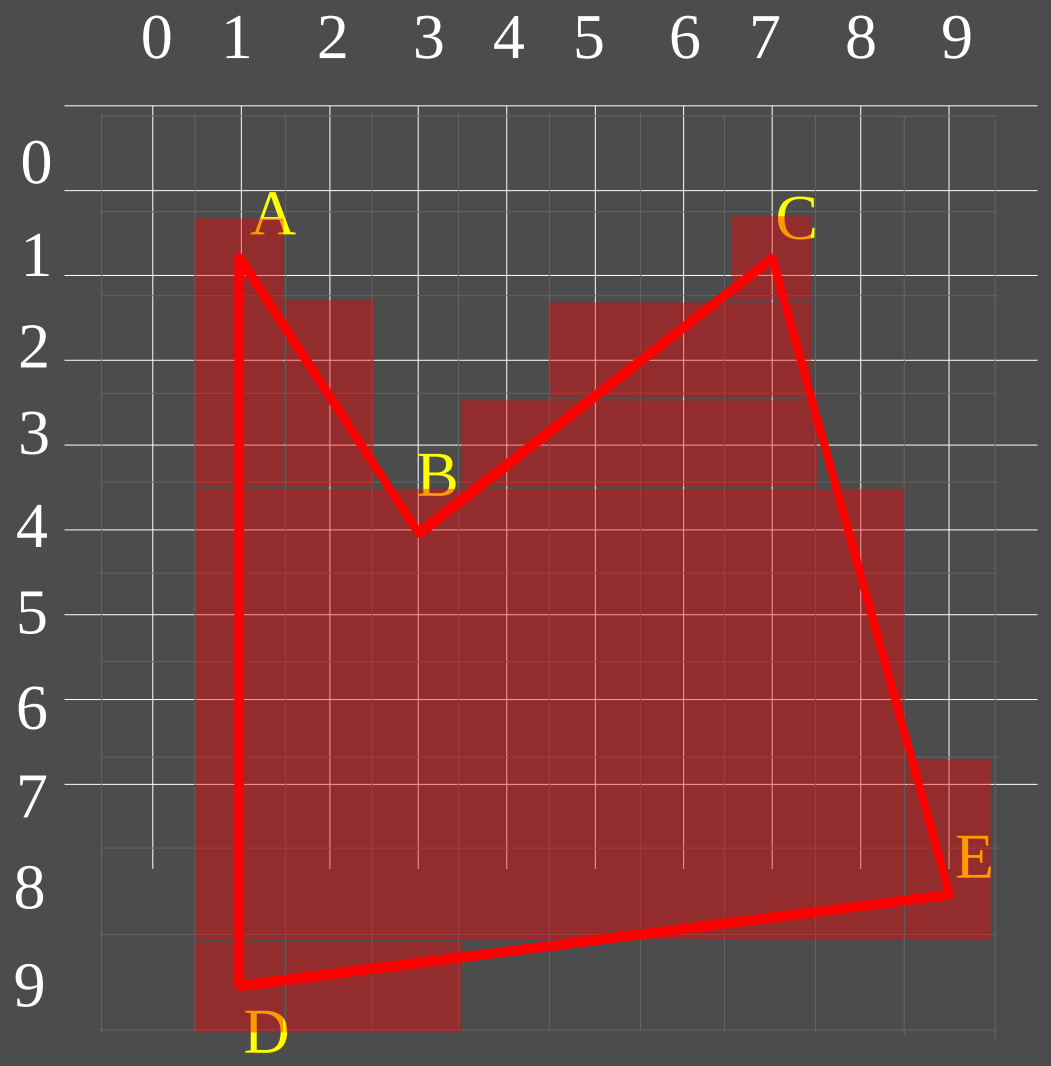
Si se escoge S entonces $h_{nuevo} = h_{viejo} + a^2(-2y_p+3)$

Condición Inicial (la elipse inicia en (0, b) y el primer punto medio es (1, b-1/2))

$F(1, b-1/2) = b^2 + a^2(-b+1/4)$

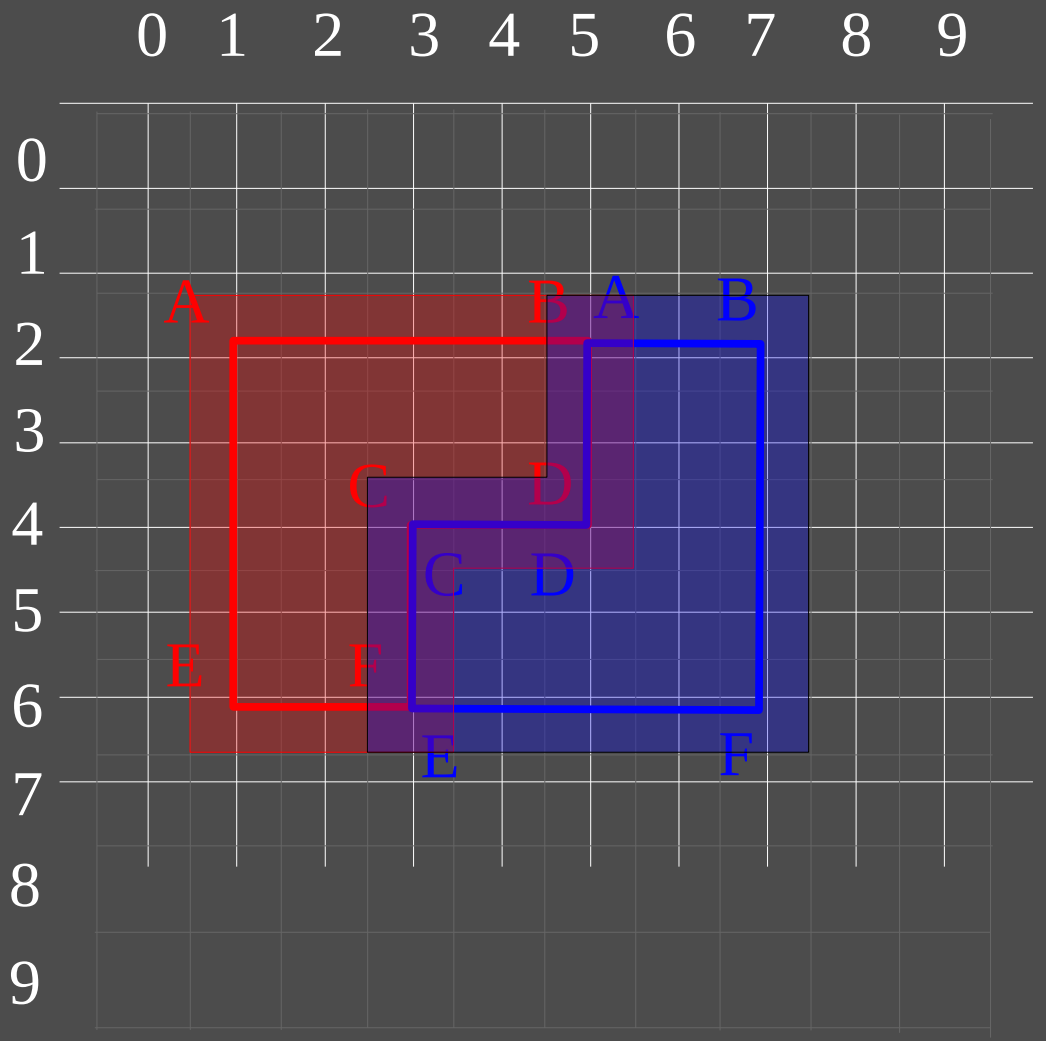
Rellenado de Polígonos

Problemática / Rellenado Mediante Línea de Rastreo



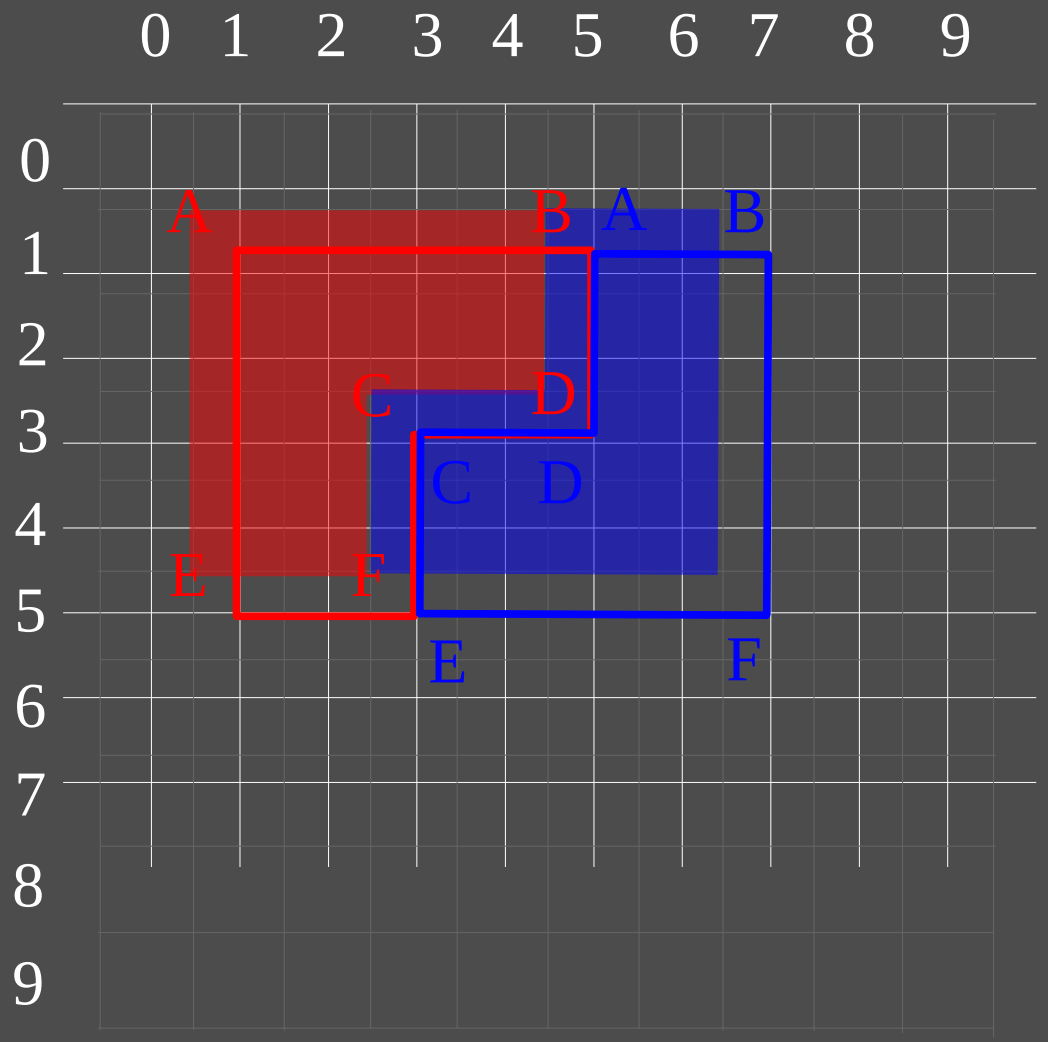
Rellenado de Polígonos

Problemática / Coherencia - Líneas Horizontales y Verticales



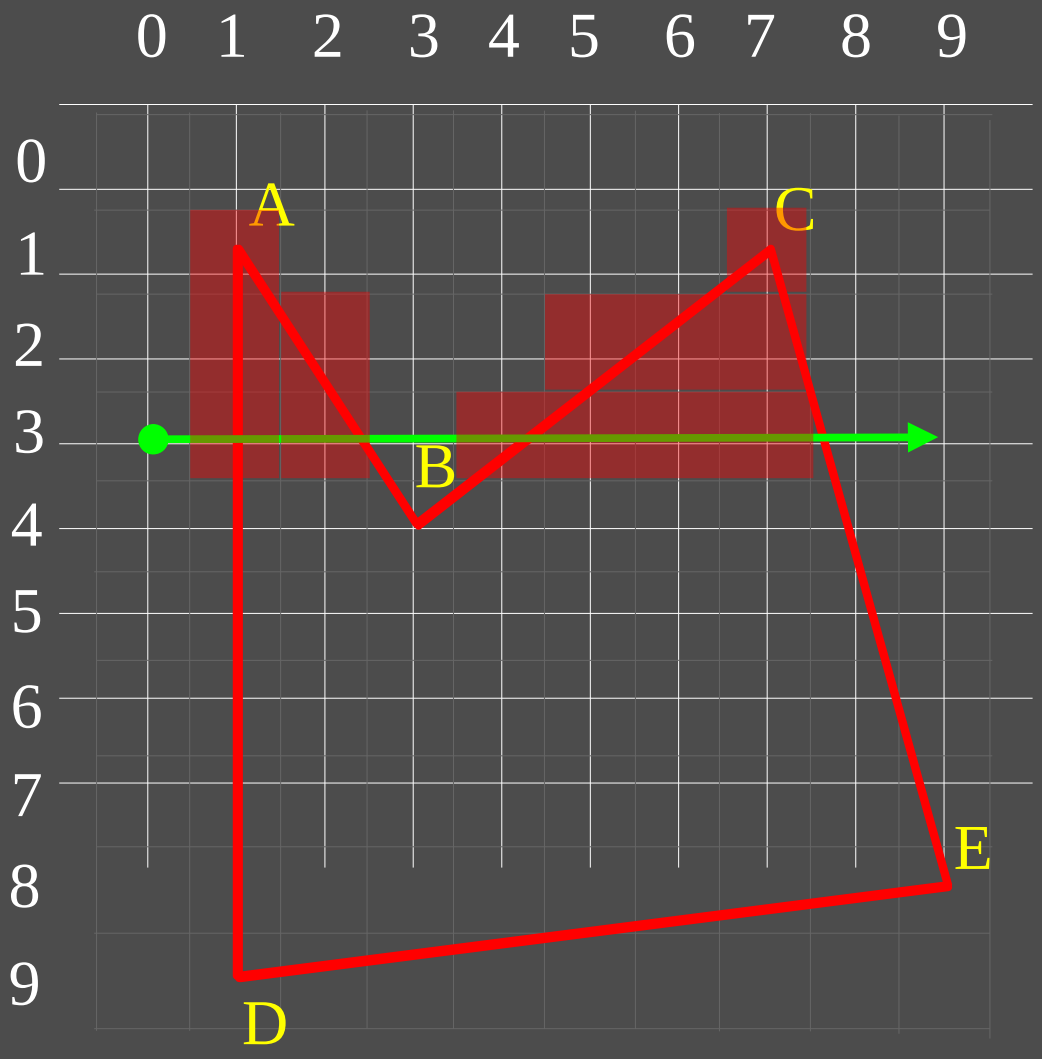
Rellenado de Polígonos

Problemática / Coherencia - Líneas Horizontales y Verticales



Rellenado de Polígonos

Algoritmo



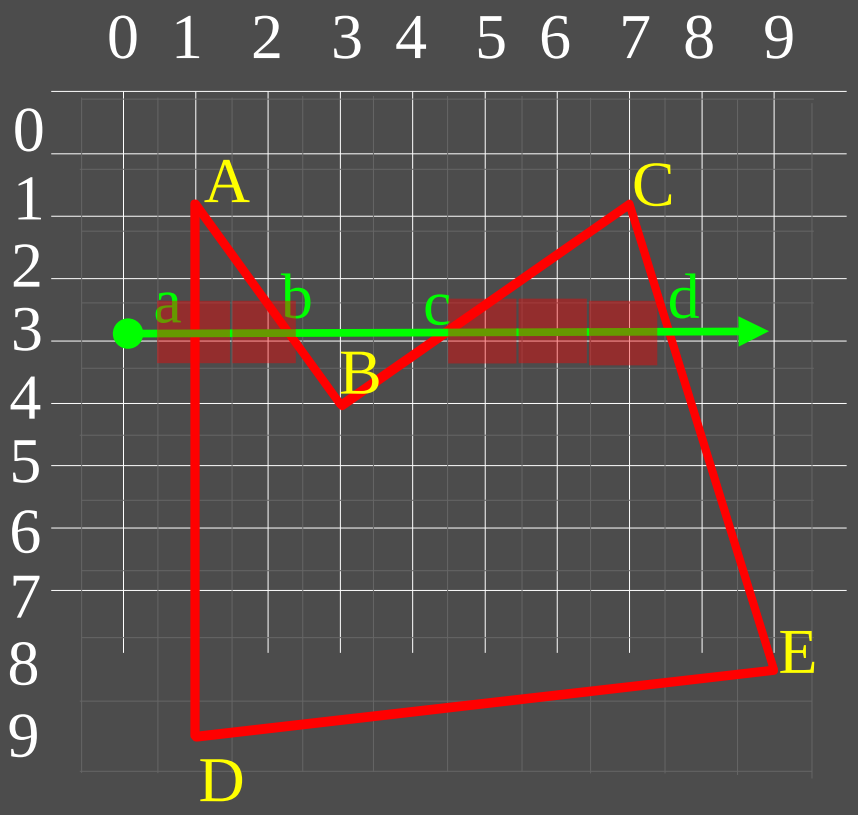
1. Hallar intersecciones de la línea de rastreo con todas las aristas del polígono.

2. Ordenar las intersecciones ascendentemente en la coordenada x.

3. Rellenar los pixeles entre pares de intersecciones que se encuentren dentro del polígono, empleando un *bit* de paridad.

Rellenado de Polígonos

Paso 3:



Del cálculo de intersecciones:

Coordenas de a: (1, 3)
Coordenas de b: (7/3, 3)
Coordenas de c: (13/3, 3)
Coordenas de d: (53/7, 3)

POLITICAS A EMPLEAR

Fraccionarios:

1. Si nos aprox. hacia una intersección fraccionaria y estamos dentro del polígono, redondeamos hacia abajo la coordenada de x. Si estamos afuera redondeamos hacia arriba.

Enteros:

2. Extremo izquierdo se incluye, extremo derecho no.

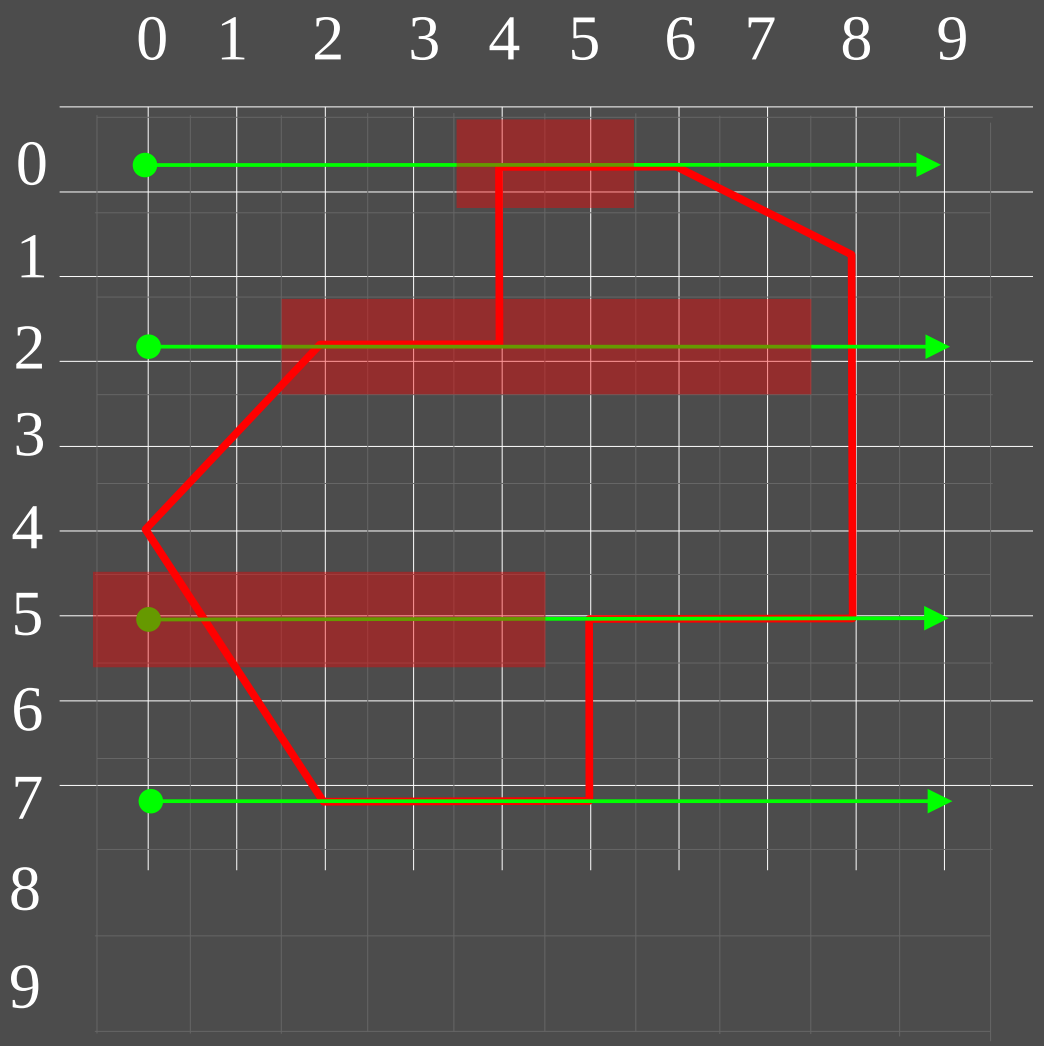
Vértices compartidos:

3. los vértices Ymax (de una arista) no se cuentan el cálculo de paridad. Así, si el vértice es el Ymin de una arista y el Ymax de otra, contamos solo el vértice Ymin en el cálculo de paridad.

Nota: En el caso de las aristas horizontales sus vértices no se cuentan ni como Ymin ni Ymax

Rellenado de Polígonos

Ejemplo de Aristas Horizontales y vértices compartidos



3. Los vértices Ymax (de una arista) no se cuentan el cálculo de paridad. Así, si el vértice es el Ymin de una arista y el Ymax de otra, contamos solo el vértice Ymin en el cálculo de paridad.

Nota: En el caso de las aristas horizontales sus vértices no se cuentan ni como Ymin ni Ymax

Rellenado de Polígonos

Observaciones

1. Evitar la técnica burda de evaluar cada arista del polígono para detectar la intesección con cada línea de rastreo.

2. Varias de las aristas intersecadas por la línea de rastreo i, son intersecadas por la línea de rastreo i+1

3. Además se puede aprovechar la coherencia de intersecciones de la línea i a la i+1, para el cálculo de la coordenada x, así:

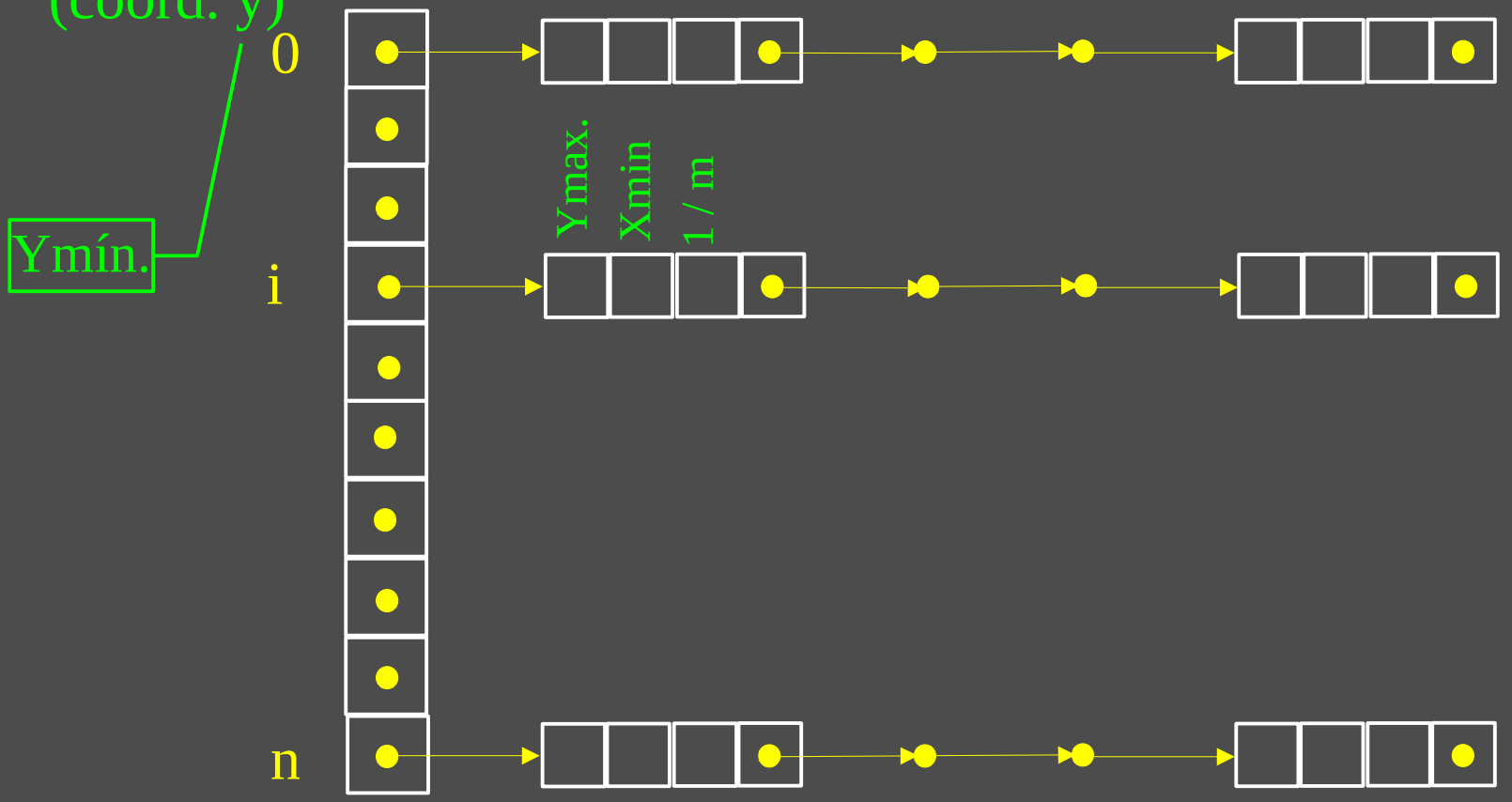
$X_{i+1} = X_i + 1/m$ (donde m es la pendiente de la arista) ya que $Y_{i+1} = Y_i + 1$

Rellenado de Polígonos

Estructuras de Datos

ET (Edge Table)

Aristas con el extremo inferior en la línea de rastreo ordenadas ascendentemente de acuerdo a:
1ro: x
2do: 1/m



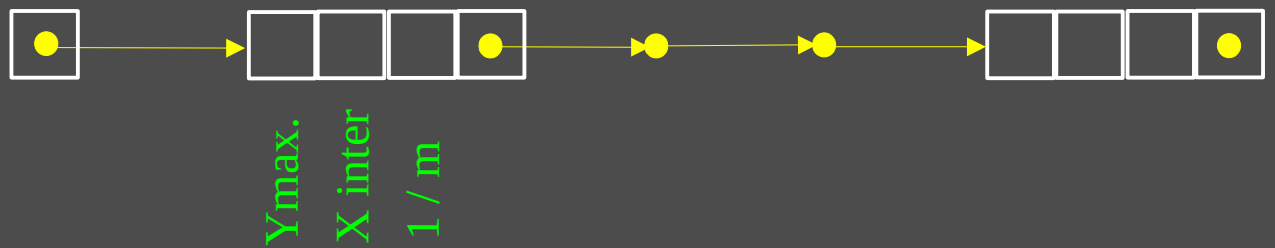
Rellenado de Polígonos

Estructuras de Datos

AET (Active Edge Table)

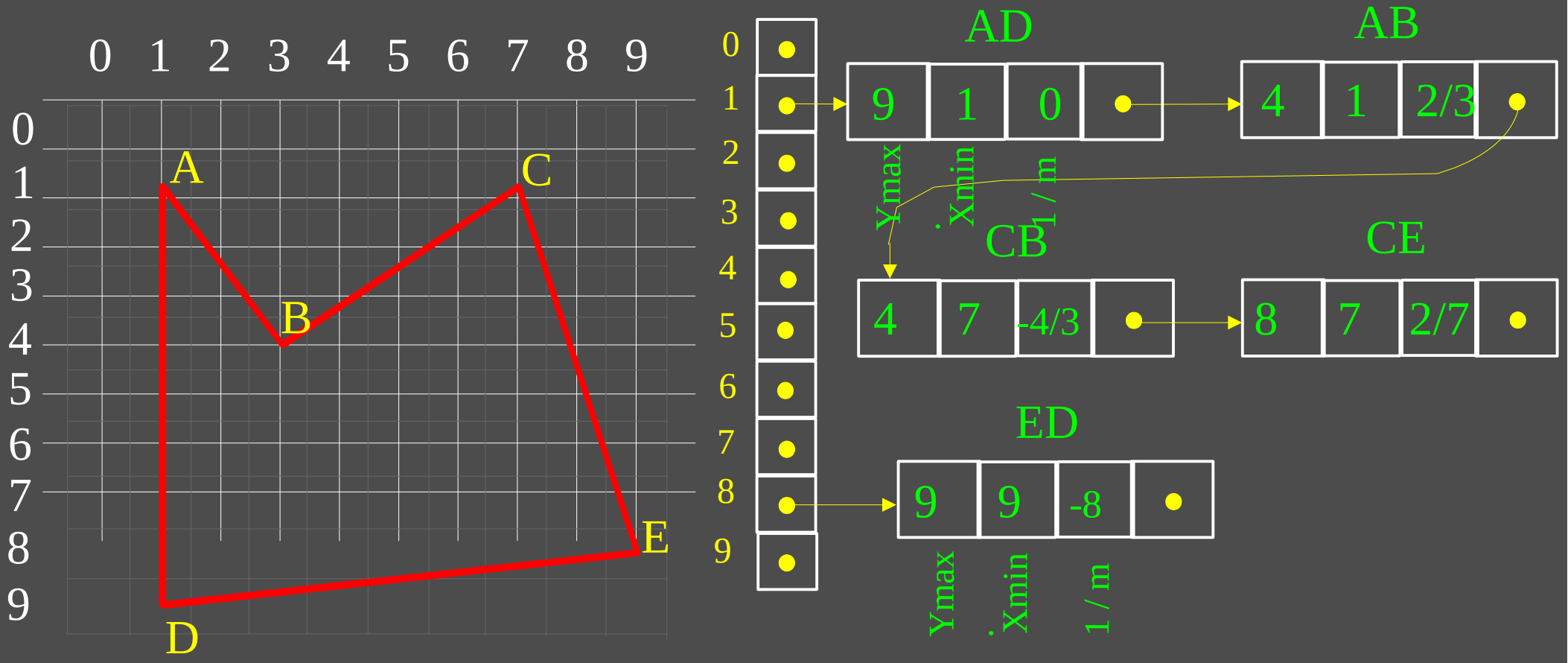
Línea de rastreo i-ésima

Aristas que intersecan la línea de rastreo



Rellenado de Polígonos

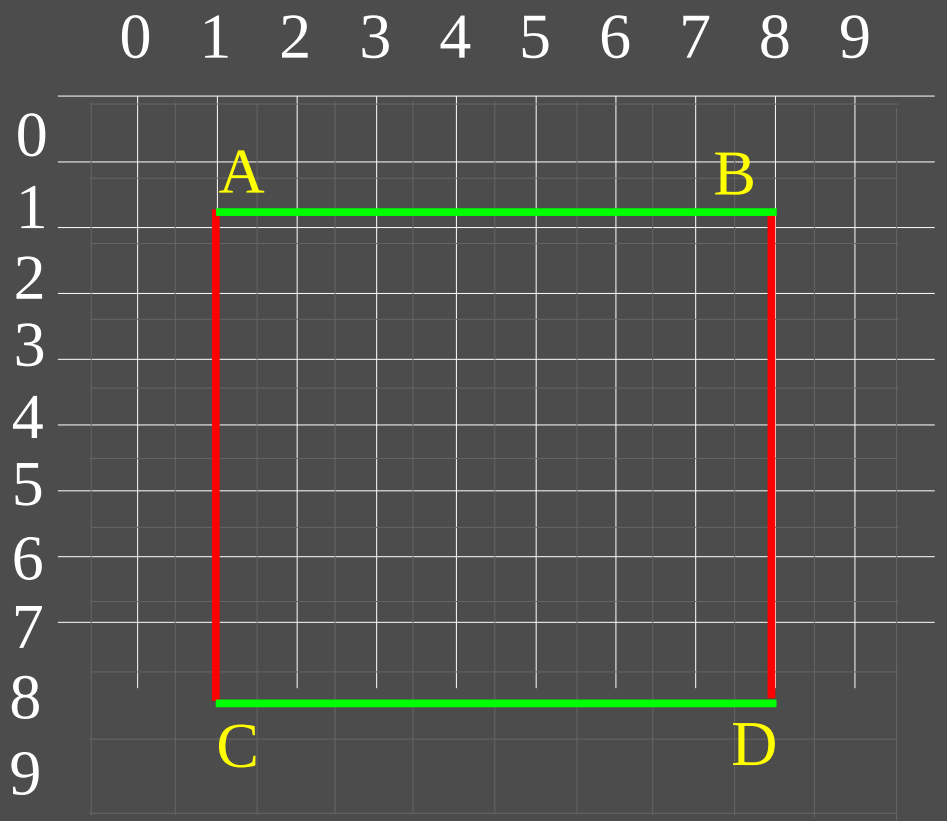
Ejemplo de ET (Edge Table)

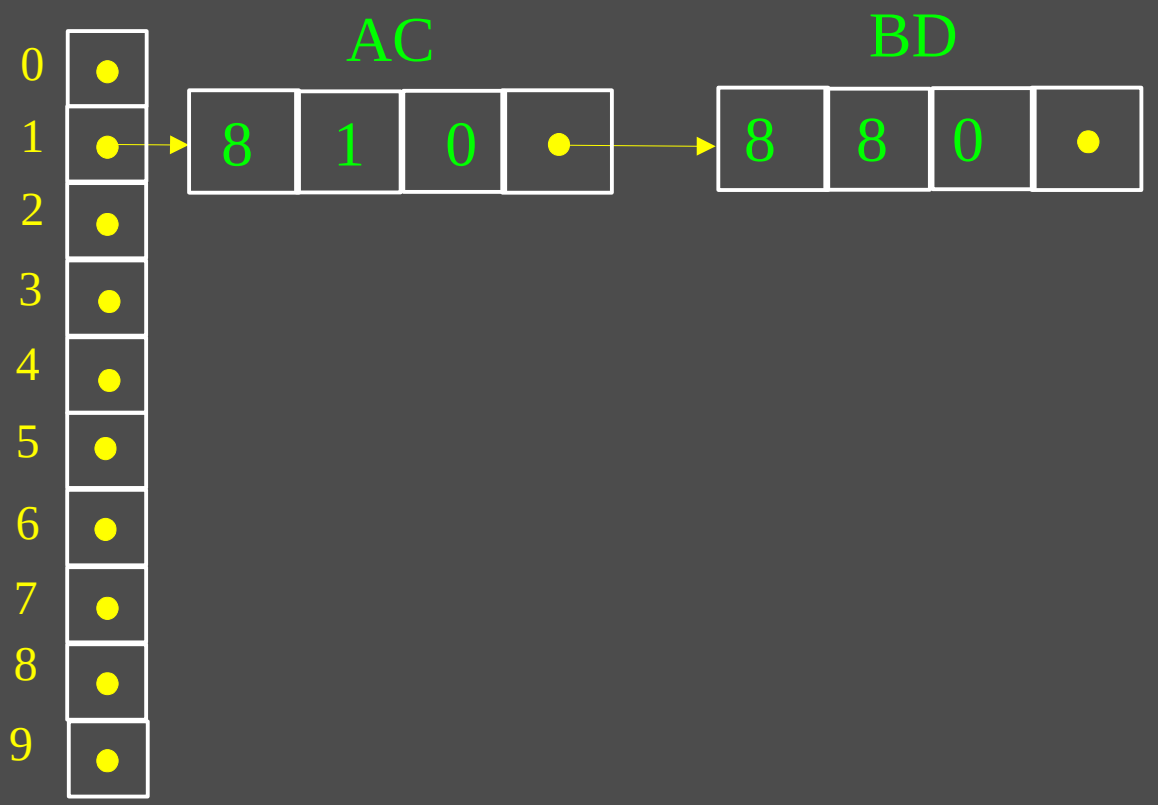


Rellenado de Polígonos

Ejemplo de ET (Edge Table)

Observación: No hace falta mantener información respecto de las aristas horizontales. Además se mantiene la coherencia respecto del cálculo de paridad de vértices compartidos.





Rellenado de Polígonos

Algoritmo para el Rellenado de Polígonos

Al avanzar a la línea de rastreo Y+1 (viniendo de la Y) se actualiza AET así:

1. Se eliminan las aristas que no sean intersecadas por la línea Y+1, al leer el campo Ymax de la AET, cuando Ymax = Y

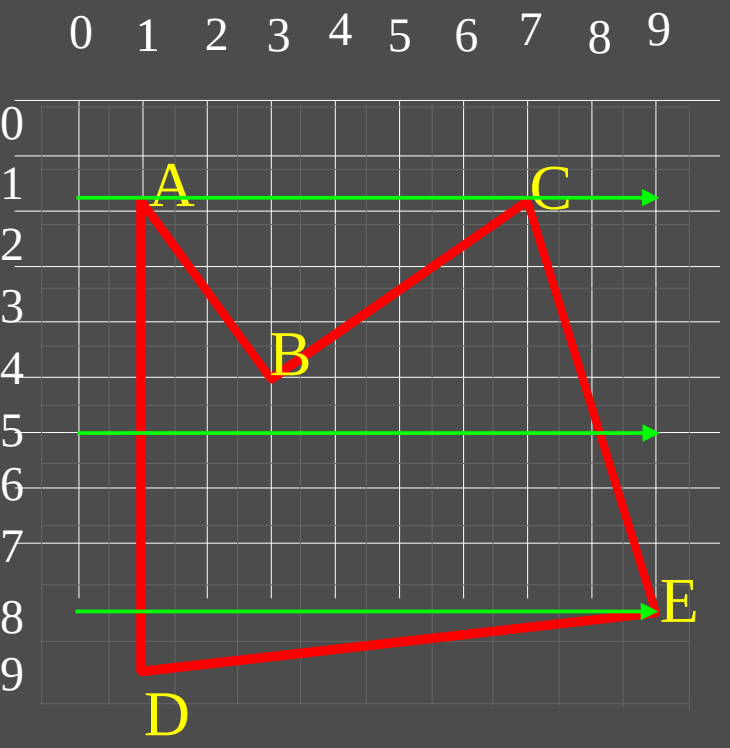
2. Se agregan las aristas nuevas intersecadas por la línea Y+1, al leer las entradas en Y+1 de ET, cuando Ymín = Y+1.
Esta operación se efectúa en orden ascendente, de acuerdo primero a la coordenada en X y luego a 1/m.

3. Se calcula las intersecciones con X de las aristas en AET, empleando el algoritmo incremental.

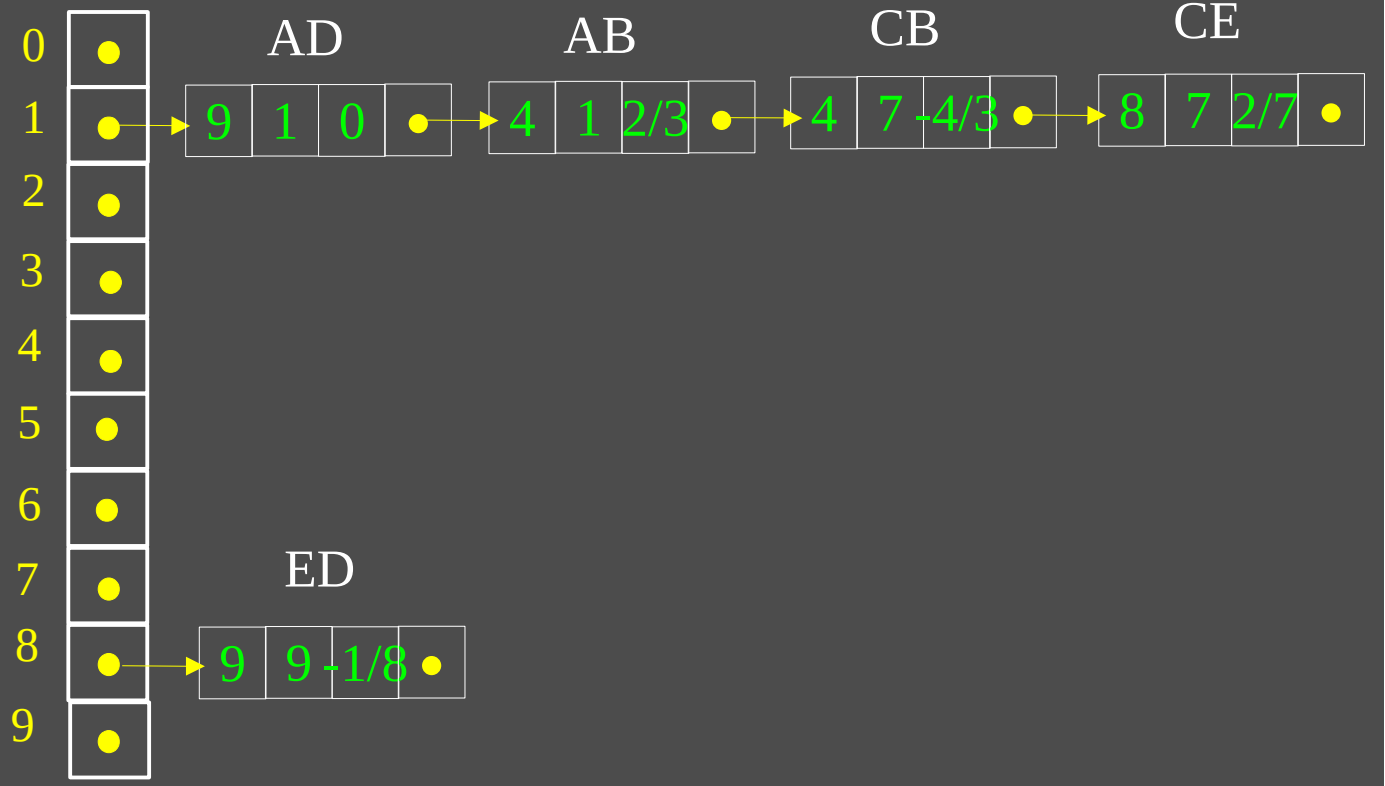
Finalmente, se dibujan aquellas porciones de la línea de rastreo de acuerdo con el bit de paridad

Rellenado de Polígonos

Ejemplo

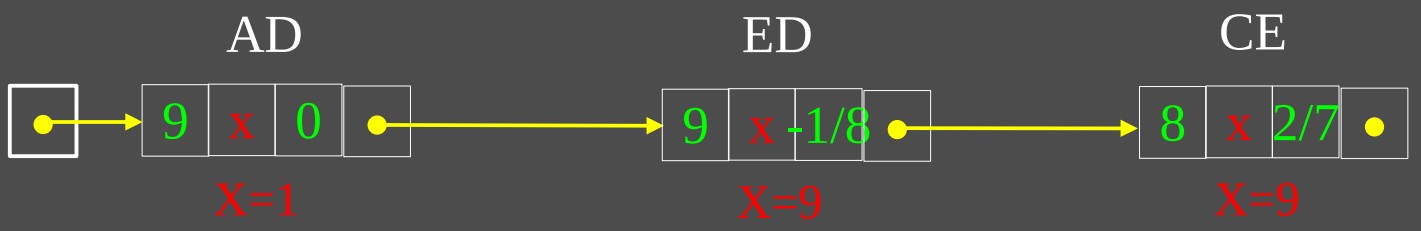


ET

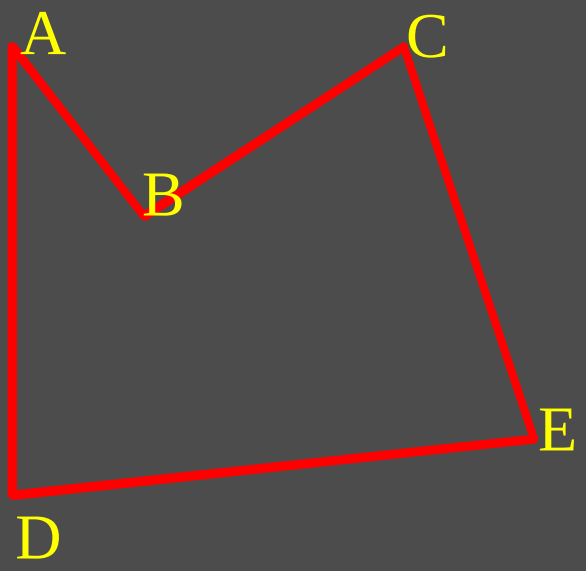


AET

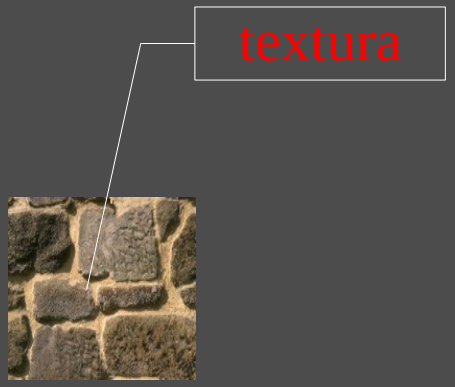
Línea de rastreo 8



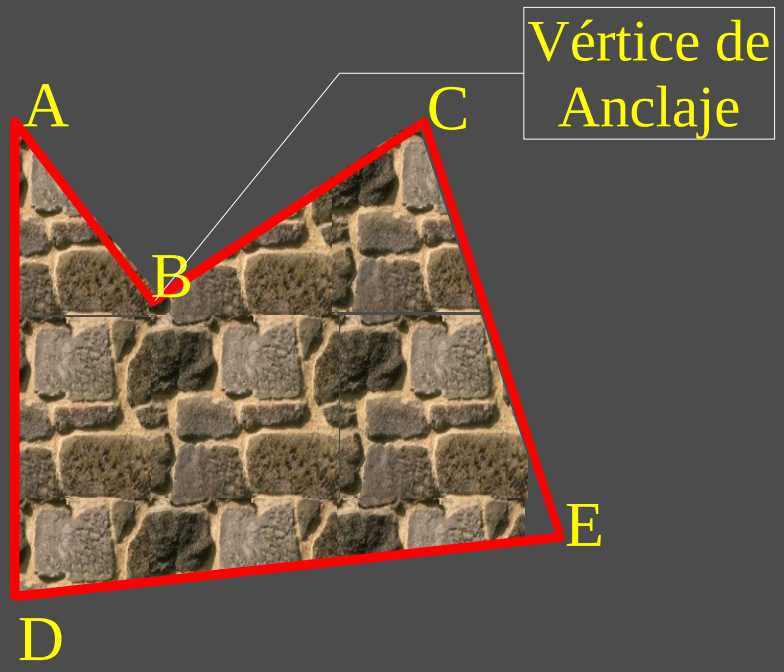
Rellenado con Patrones

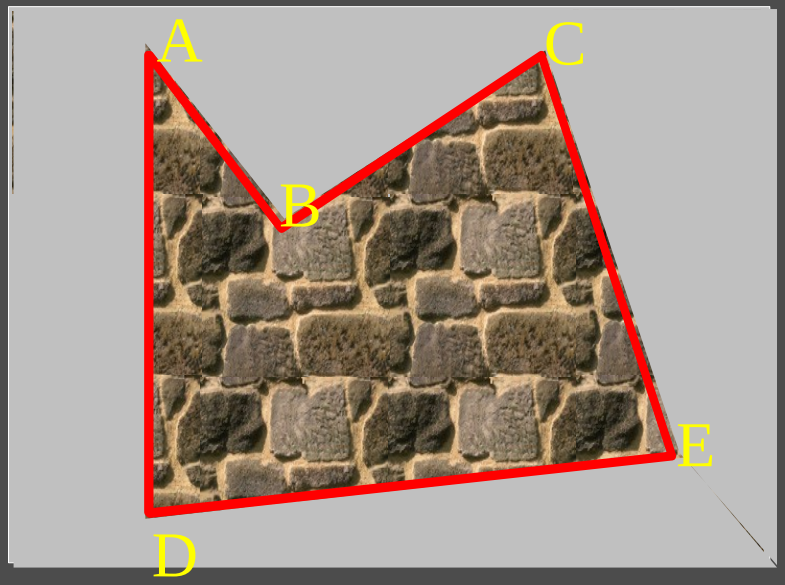


Anclaje Relativo



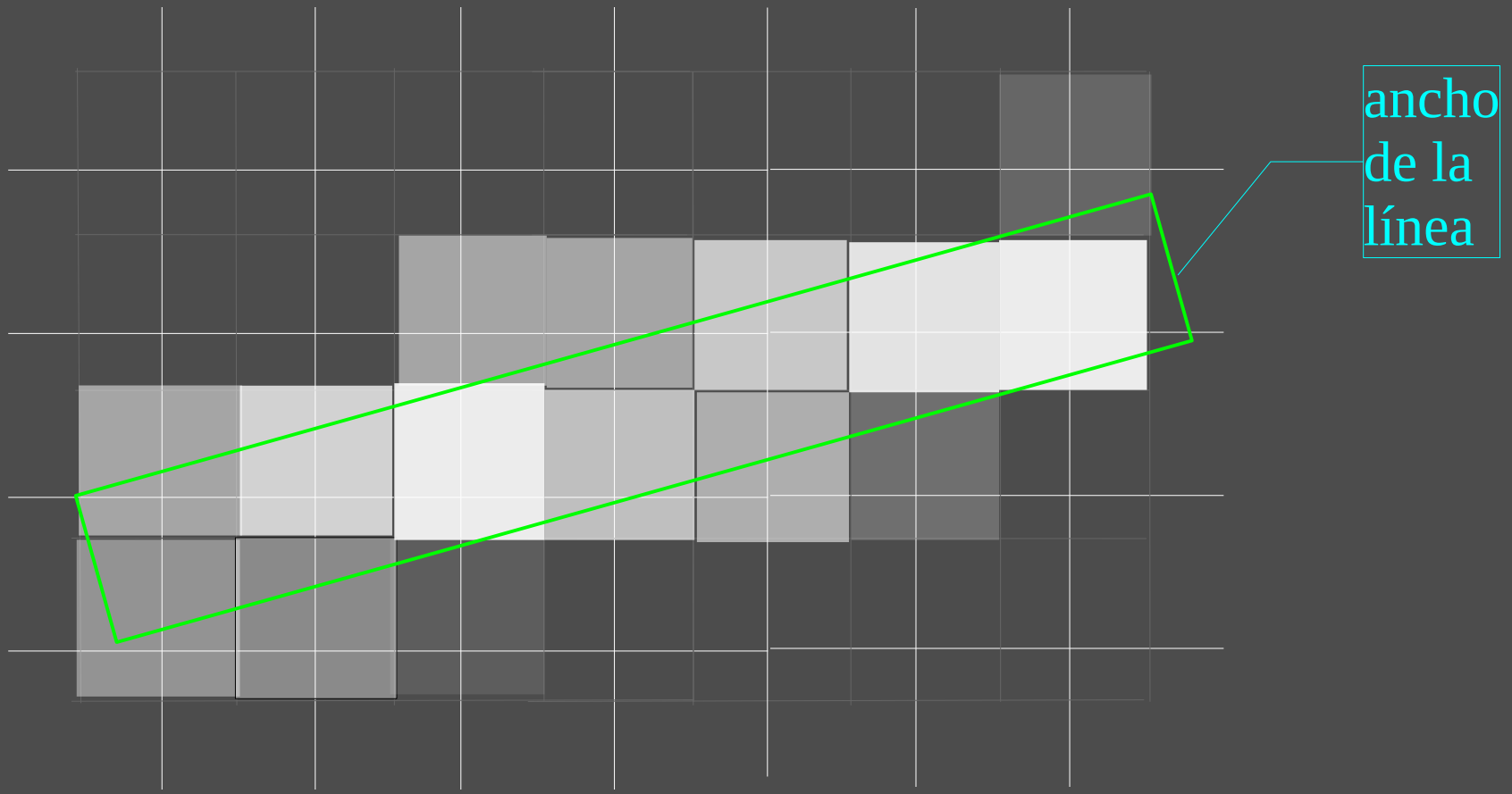
Anclaje Absoluto





Antialiasing

Idea



Antialiasing / Tipos

Muestreo no ponderado de área

Se define un ancho de línea.
El brillo del pixel es proporcional al área del pixel cubierta por la línea.
Iguales áreas contribuyen con el mismo brillo (independientemente de la distancia entre el centro del pixel y el área)

Muestreo ponderado de área

Se define un ancho de línea.
Se define una función de ponderación para determinar la influencia en la intensidad del pixel de un área dA de la primitiva.

Antialiasing / Función de Ponderación