Computer Graphics (CS7.302)

Output Primitives

Raghavendra G S

Apr 12th, 2025

Output Primitives

Output Primitives

Set of basic geometric entities which is used in combination to represent the output of graphics processing.

- Points.

Output Primitives

Set of basic geometric entities which is used in combination to represent the output of graphics processing.

- Points.
- Lines.

Output Primitives

Set of basic geometric entities which is used in combination to represent the output of graphics processing.

- Points.
- Lines.
- Circles/Ellipses.

Output Primitives

Set of basic geometric entities which is used in combination to represent the output of graphics processing.

- Points.
- Lines.
- Circles/Ellipses.
- Polygons.

Output Primitives

Set of basic geometric entities which is used in combination to represent the output of graphics processing.

- Points.
- Lines.
- Circles/Ellipses.
- Polygons.
- Other Curves.

Rasterisation

- It is the step after geometric processing involving transformation, assignment of colors clipping etc.
- It converts the information which is in 3D to 2D and generate fragments which has information for potential pixel along with color attributes etc.
- It contains z information too to facilitate hidden surface removal.

Drawing Points

Drawing Points

Done by setting a specific screen position/pixel to 1 in the frame buffer.

Drawing Points

Done by setting a specific screen position/pixel to 1 in the frame buffer.

- Every Graphics system will contain an API to set pixel.

Drawing Points

Done by setting a specific screen position/pixel to 1 in the frame buffer.

- Every Graphics system will contain an API to set pixel.
- setPixel(x,y)

Drawing Points

Done by setting a specific screen position/pixel to 1 in the frame buffer.

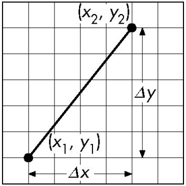
- Every Graphics system will contain an API to set pixel.
- setPixel(x,y)
- getPixel(x,y)

Drawing Lines

Calculate intermediate points between the end points.
Utilizes the line equation.

Drawing Lines

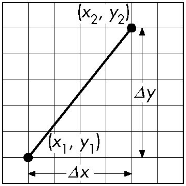
Calculate intermediate points between the end points.
Utilizes the line equation.



$$y = mx + c$$

Drawing Lines

Calculate intermediate points between the end points.
Utilizes the line equation.

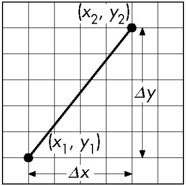


$$y = mx + c$$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta y}{\Delta x}$$

Drawing Lines

Calculate intermediate points between the end points.
Utilizes the line equation.



$$y = mx + c$$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta y}{\Delta x}$$

$$\Delta y = m.\Delta x$$

Drawing Lines

Drawing Lines

- DDA Algorithm (*Digital Differential Analyser*).

Drawing Lines

- DDA Algorithm (*Digital Differential Analyser*).
- Bressenham's Algorithm.

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

- sample x intervals i.e. $\Delta x = 1$.

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

- sample x intervals i.e. $\Delta x = 1$.
- when $\Delta x = 1$, $\Delta y = ?$.

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

- sample x intervals i.e. $\Delta x = 1$.
- when $\Delta x = 1$, $\Delta y = ?$.
- $\Delta y = m$.

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

- sample x intervals i.e. $\Delta x = 1$.
- when $\Delta x = 1$, $\Delta y = ?$.
- $\Delta y = m$.
- successively do $y_{i+1} = y_i + m$.
  - $i = 1$ for first endpoint and $i = n$ for last endpoint.

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

- sample x intervals i.e. $\Delta x = 1$.
- when $\Delta x = 1$, $\Delta y = ?$.
- $\Delta y = m$.
- successively do $y_{i+1} = y_i + m$.
  - $i = 1$ for first endpoint and $i = n$ for last endpoint.

For $m > |1|$ sample y i.e. $\Delta y = 1$

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

- sample x intervals i.e. $\Delta x = 1$.
- when $\Delta x = 1$, $\Delta y =?$.
- $\Delta y = m$.
- successively do $y_{i+1} = y_i + m$.
  - $i = 1$ for first endpoint and $i = n$ for last endpoint.

For $m > |1|$ sample y i.e. $\Delta y = 1$

- when $\Delta y = 1$, $\Delta x =?$

DDA

Consider $m \leq |1|$. we can take care of others using symmetry.

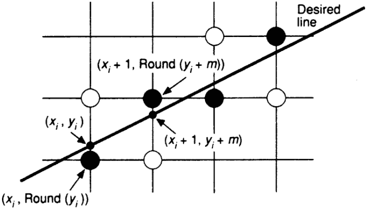
- sample x intervals i.e. $\Delta x = 1$.
- when $\Delta x = 1$, $\Delta y = ?$.
- $\Delta y = m$.
- successively do $y_{i+1} = y_i + m$.
  - $i = 1$ for first endpoint and $i = n$ for last endpoint.

For $m > |1|$ sample y i.e. $\Delta y = 1$

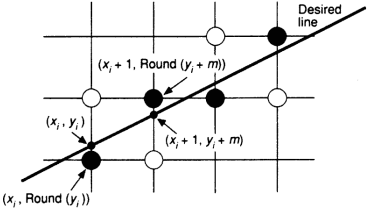
- when $\Delta y = 1$, $\Delta x = ?$
- $\Delta x = \frac{1}{m}$.

Here we assume we are processing form left to right.

DDA

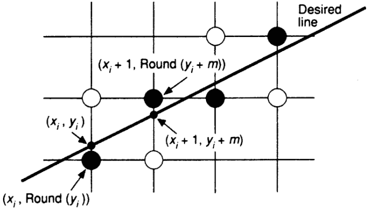


DDA



Disadvantages.

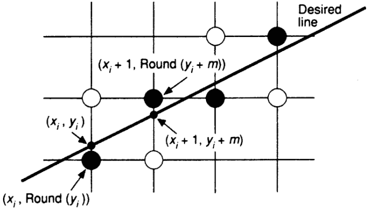
DDA



Disadvantages.

- m is a fraction.

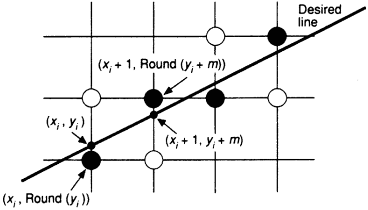
DDA



Disadvantages.

- m is a fraction.
- It involves floating point operations.

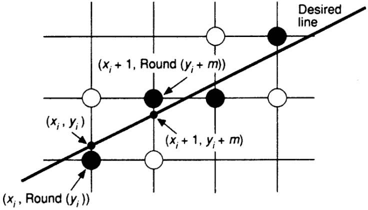
DDA



Disadvantages.

- m is a fraction.
- It involves floating point operations.
- It involves rounding operations.

DDA

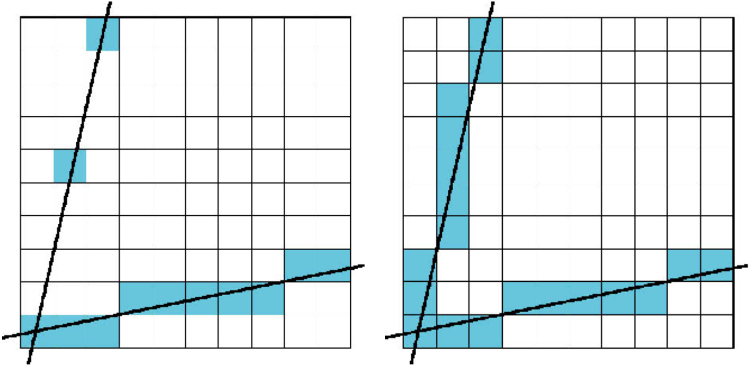


Disadvantages.

- m is a fraction.
- It involves floating point operations.
- It involves rounding operations.
- rounding errors accumalate.

DDA

Examples



(a) (b)

Figure: (a) case when $m > 1$. (b) handling the case $m > 1$.

Bressenham

Goal:

Bressenham

Goal:

- fast decision of the next pixel.

Bressenham

Goal:

- fast decision of the next pixel.
- avoid floating point operations.

Bressenham

Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Bressenham

Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Criterion:

Bressenham

Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Criterion:

- position of the midpoint M w.r.t intersection point Q

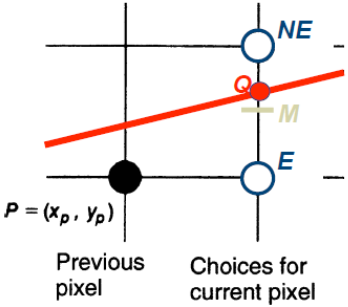
Bressenham

Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Criterion:

- position of the midpoint M w.r.t intersection point Q



Bressenham

Let

$$y = mx + b \tag{1}$$

Bressenham

Let

$$y = mx + b \tag{1}$$

- Use implicit form of straight line.

$$F(x, y) = ax + by + c = 0 \tag{2}$$

Bressenham

Let

$$y = mx + b \tag{1}$$

- Use implicit form of straight line.

$$F(x, y) = ax + by + c = 0 \tag{2}$$

- From (1) we have $m = \Delta y / \Delta x$ we get,

$$a = \Delta y, b = -\Delta x, c = b.\Delta x$$

and

$$\Delta y.x - \Delta x.y + b.\Delta x = 0 \tag{3}$$

Bressenham

Let

$$y = mx + b \tag{1}$$

- Use implicit form of straight line.

$$F(x, y) = ax + by + c = 0 \tag{2}$$

- From (1) we have $m = \Delta y / \Delta x$ we get,

$$a = \Delta y, b = -\Delta x, c = b.\Delta x$$

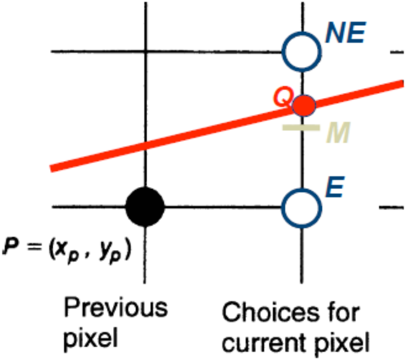
  and

$$\Delta y.x - \Delta x.y + b.\Delta x = 0 \tag{3}$$

- Evaluate at midpoint M

$$d = F(M) = F(x_m, y_m) = F(x_p + 1, y_p + \frac{1}{2}) \tag{4}$$

Decision Criterion



$d > 0 \Rightarrow$ select pixel ***NE***

$d < 0 \Rightarrow$ select pixel ***E***

Updating the Decision Criterion E

- Fast incremental update of Decision Variable

$$d_{old} = F(x_p + 1, y_p + \frac{1}{2}) = a(x_p + 1) + b(y_p + \frac{1}{2}) + c \qquad (5)$$

- Now if E is applied

$$d_{new} = F(x_p + 2, y_p + \frac{1}{2}) = a(x_p + 2) + b(y_p + \frac{1}{2}) + c \qquad (6)$$

- difference Δd is given by (20)-(19)

$$d_E = d_{new} - d_{old} = a = \Delta y \qquad (7)$$

Updating the Decision Criterion NE

- Fast incremental update of Decision Variable

$$d_{old} = F(x_p + 1, y_p + \frac{1}{2}) = a(x_p + 1) + b(y_p + \frac{1}{2}) + c \tag{8}$$

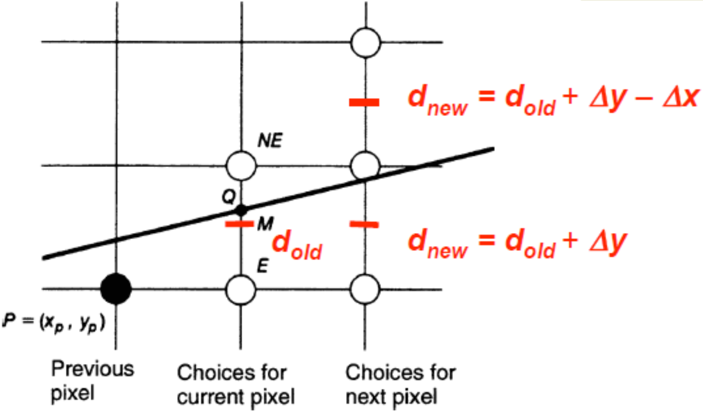
- Now if NE is applied

$$d_{new} = F(x_p + 2, y_p + \frac{3}{2}) = a(x_p + 2) + b(y_p + \frac{3}{2}) + c \tag{9}$$

- difference Δd is given by (23)-(22)

$$d_{NE} = d_{new} - d_{old} = a + b = \Delta y - \Delta x \tag{10}$$

Update Criterion



Eliminating Floating Point Arithmetic

- Initialisation of Decision Criterion.

$$d_{start} = F(x_0 + 1, y_0 + \frac{1}{2}) = a(x_0 + 1) + b(y_0 + \frac{1}{2}) + c \quad (11)$$

$$d_{start} = a.x_0 + b.y_0 + c + a + \frac{b}{2} \quad (12)$$

$$d_{start} = F(x_0, y_0) + a + \frac{b}{2} \quad (13)$$

- Now as (x_0, y_0) is on the line $F(x_0, y_0) = 0$ substituting this in (27) we get

$$d_{start} = a + \frac{b}{2} = \Delta y - \Delta x/2 \quad (14)$$

- Now we have division by 2 eliminate it by multiplying by 2 on both sides of (28)

$$d_0 = 2 * d_{start} = 2.\Delta y - \Delta x \quad (15)$$

- Also

$$2.F(x, y) = 2(a.x + b.y + c) \quad (16)$$

Bressenham

Actual Algorithm

1. Input two endpoints store left one as (x_0, y_0).
2. Plot first point (x_0, y_0).
3. Calculate $\Delta x, \Delta y, 2.\Delta y, 2.\Delta y - 2.\Delta x$, obtain first decision parameter

$$d_0 = 2.\Delta y - \Delta x$$

4. for each x_p from $p = 0$ if $d_p < 0$ the next point to plot is $(x_p + 1, y_p)$ and

$$d_{p+1} = d_p + 2.\Delta y$$

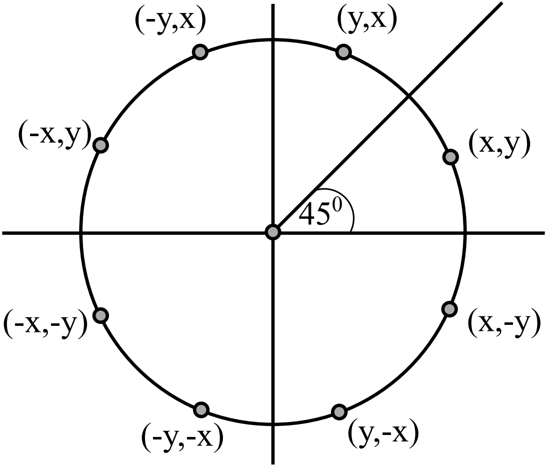
5. if $d_p > 0$ plot $(x_p + 1, y_p + 1)$ and

$$d_{p+1} = d_p + 2.\Delta y - 2.\Delta x$$

Recap of Circle

- Equation of a circle with centre (x_c, y_c) and radius r is $(x - x_c)^2 + (y - y_c)^2 = r^2$
- If centre is $(0, 0)$ then it will be $x^2 + y^2 = r^2$
- Parametric form $x = x_c + r.cos\,\theta$ and $y = y_c + r.sin\,\theta$
- Eight-fold symmetry*
    - if you can find one point by symmetry you can find seven other points lying on the circle.

Recap of Circle



How to draw?

- You can use the equation $(x - x_c)^2 + (y - y_c)^2 = r^2$
  - rearranging terms we get $y = y_c \pm \sqrt{r^2 - (x - x_c)^2}$
  - evaluate from $x_c - r$ to $x_c + r$

How to draw?

- You can use the equation $(x - x_c)^2 + (y - y_c)^2 = r^2$
  - rearranging terms we get $y = y_c \pm \sqrt{r^2 - (x - x_c)^2}$
  - evaluate from $x_c - r$ to $x_c + r$
- Disadvantages
  - floating point operations
  - rounding errors

Bressenham

Again we have Bressenham to rescue.
Goal:

Bressenham

Again we have Bressenham to rescue.
Goal:

- fast decision of the next pixel.

Bressenham

Again we have Bressenham to rescue.
Goal:

- fast decision of the next pixel.
- avoid floating point operations.

Bressenham

Again we have Bressenham to rescue.
Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Bressenham

Again we have Bressenham to rescue.
Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Criterion:

Bressenham

Again we have Bressenham to rescue.

Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Criterion:

- position of the midpoint M w.r.t intersection point Q

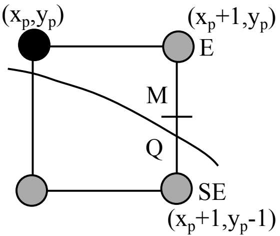
Bressenham

Again we have Bressenham to rescue.
Goal:

- fast decision of the next pixel.
- avoid floating point operations.
- eliminate rounding off errors.

Criterion:

- position of the midpoint M w.r.t intersection point Q



Bressenham

Assume centre to be $(0, 0)$

- Use implicit form of Circle.

$$F(x, y) = x^2 + y^2 - r^2 = 0 \tag{17}$$

Bressenham

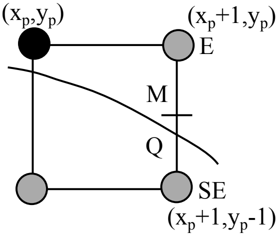
Assume centre to be $(0, 0)$

- Use implicit form of Circle.

$$F(x, y) = x^2 + y^2 - r^2 = 0 \tag{17}$$

- Given a point (x, y) we know that
  - $F(x, y) < 0$ if (x, y) lies inside the circle
  - $F(x, y) = 0$ if (x, y) lies on the circle
  - $F(x, y) > 0$ if (x, y) lies outside the circle

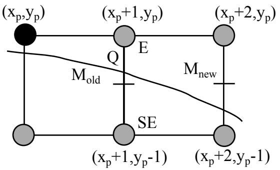
Decision Criterion



$$d = F(M) = F(x_p + 1, y_p - \frac{1}{2}) \tag{18}$$

- if $F(M) < 0$ choose E
- if $F(M) \geq 0$ choose SE

Updating the Decision Criterion E



- Fast incremental update of Decision Variable

$$d_{old} = F(x_p + 1, y_p - \frac{1}{2}) = (x_p + 1)^2 + (y_p - \frac{1}{2})^2 - r^2 \tag{19}$$

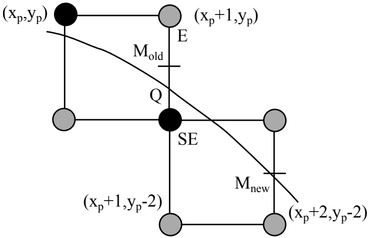
- Now if E is applied

$$d_{new} = F(x_p + 2, y_p - \frac{1}{2}) = (x_p + 2)^2 + (y_p - \frac{1}{2})^2 - r^2 \tag{20}$$

- difference Δd is given by (20)-(19)

$$d_E = d_{new} - d_{old} = 2x_p + 3 \tag{21}$$

Updating the Decision Criterion SE



- Fast incremental update of Decision Variable

$$d_{old} = F(x_p + 1, y_p - \frac{1}{2}) = (x_p + 1)^2 + (y_p - \frac{1}{2})^2 - r^2 \qquad (22)$$

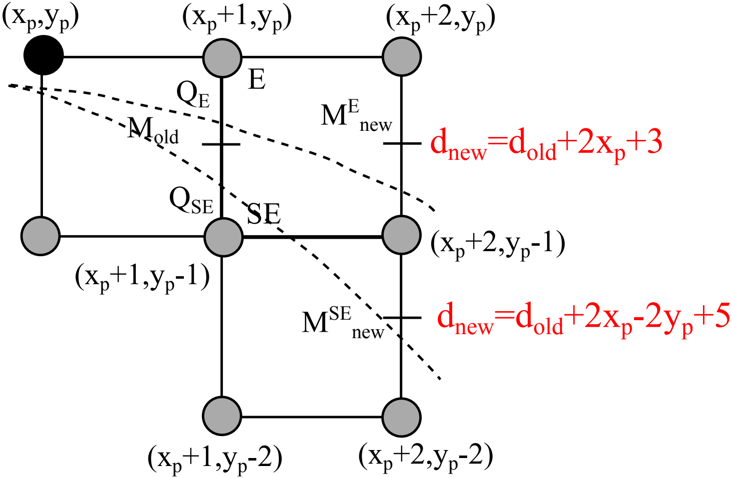
- Now if SE is applied

$$d_{new} = F(x_p + 2, y_p - \frac{3}{2}) = (x_p + 2)^2 + (y_p - \frac{3}{2})^2 - r^2 \qquad (23)$$

- difference Δd is given by (23)-(22)

$$d_{SE} = d_{new} - d_{old} = 2x_p - 2y_p + 5 \qquad (24)$$

Update Criterion



Eliminating Floating Point Arithmetic

- Initialisation of Decision Criterion.

$$d_{start} = F(x_0 + 1, y_0 - \frac{1}{2}) = (x_0 + 1)^2 + (y_0 - \frac{1}{2})^2 - r^2 \tag{25}$$

$$d_{start} = x_0^2 + y_0^2 - r^2 + 2.x_0 + 1 - y_0 + \frac{1}{4} \tag{26}$$

$$d_{start} = F(x_0, y_0) + 2.x_0 + 1 - y_0 + \frac{1}{4} \tag{27}$$

- Now as (x_0, y_0) is on the line $F(x_0, y_0) = 0$ also $x_0 = 0$,$y_0 = r$ substituting this in (27) we get

$$d_{start} = \frac{5}{4} - r \tag{28}$$

- Now we have division by 4 eliminate it in (28) by making substitution $h = d - 1/4$ or $d = h + 1/4$ and $h_{start} = 1 - r$
- Now we have comparisons like $h > -1/4$. Can be replaced by $h > 0$ as we have only integer values.

Bressenham

Actual Algorithm

1. Input store it as $(x_0, y_0) = (0, r)$.
2. Plot first point $(0, r)$.
3. Calculate h_{start}, thus obtain first decision parameter

$$h_{start} = 1 - r$$

4. Till $y > x$ from $p = 0$ if $h_p < 0$ the next point to plot is $(x_p + 1, y_p)$ and

$$h_{p+1} = h_p + 2.x_p + 3$$

5. if $h_p > 0$ plot $(x_p + 1, y_p - 1)$ and

$$h_{p+1} = h_p + 2.x_p - 2.y_p + 5$$

6. plot symmetric points.

In the next class

Output primitives (continued)

The End