

# MCA101 : COMPUTER GRAPHICS

## 2D GEOMETRY REPRESENTATION

Raghav B. Venkataramaiyer

CSED TIET Patiala India.

September 3, 2024

## 1 2D GEOMETRY — INTRODUCTION

## 2 MID-POINT ALGORITHM

## 1 2D GEOMETRY — INTRODUCTION

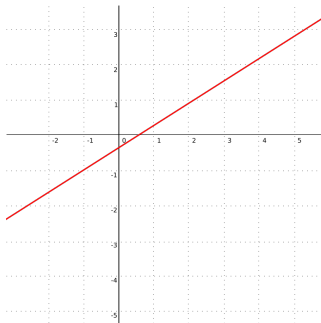
- Straight Lines

- Conics

## 2 MID-POINT ALGORITHM

$$y = mx + c$$

$$y = f(x) = mx + c$$



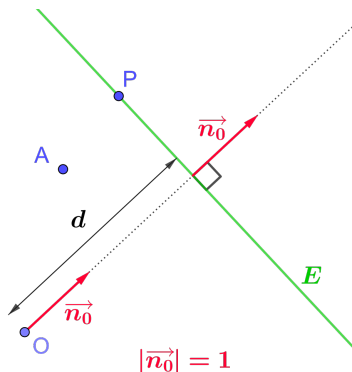
For any two vectors  $\mathbf{u}, \mathbf{v} \in V$ , a point on the line segment joining them is given parameterised by  $t \in [0, 1]$ , as

$$\mathbf{p} = f(t) = (1 - t)\mathbf{u} + t\mathbf{v}$$

Any point on a line in the direction of unit vector  $\mathbf{u} : \|\mathbf{u}\|_2^2 = 1$ , and an incident point  $\mathbf{p}_0$  may be given parameterised by  $t \in \mathbb{R}$  as,

$$\mathbf{p} = f(t) = \mathbf{p}_0 + t\mathbf{u}$$

## HESSE NORMAL FORM



Given,

Normal to the line  $\mathbf{n}_0 : \|\mathbf{n}_0\|_2^2 = 1$ , and  
its distance from origin,  $d$ ;

The point on the line is given implicitly as the locus  
of all points  $\mathbf{p}$  that satisfy,

$$\mathbf{n}_0 \cdot \mathbf{p} - d = 0$$

Distance from the origin  $O$  to the line  $E$   
calculated with the Hesse normal form.  
Normal vector in red, line in green, point  $O$   
shown in blue.

## 1 2D GEOMETRY — INTRODUCTION

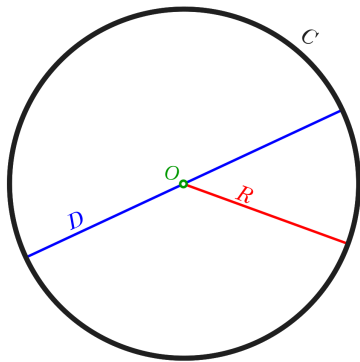
- Straight Lines

- Conics

## 2 MID-POINT ALGORITHM



# CIRCLE



Implicit Form:

$$f\begin{pmatrix} x \\ y \end{pmatrix} = x^2 + y^2 - r^2 = 0$$

Parametric Form:

$$f(r, t) = \begin{bmatrix} r \cos t \\ r \sin t \end{bmatrix}$$

FIGURE: Image Courtesy: [Wikipedia](#)

# ELLIPSE

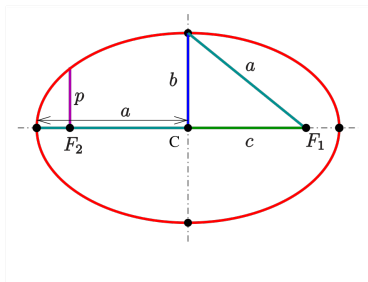


FIGURE: Image Courtesy: [Wikipedia](#)

Standard form

$$f\begin{pmatrix} x \\ y \end{pmatrix} = \frac{x^2}{a^2} + \frac{y^2}{b^2} - 1 = 0$$

Parametric Form

$$f(t; a, b) = \begin{bmatrix} a \cos t \\ b \sin t \end{bmatrix}$$

# OUTLINE

1 2D GEOMETRY — INTRODUCTION

2 MID-POINT ALGORITHM

## 1 2D GEOMETRY — INTRODUCTION

## 2 MID-POINT ALGORITHM

- Fundamentals
- Straight Line
- Circle

## PROBLEM

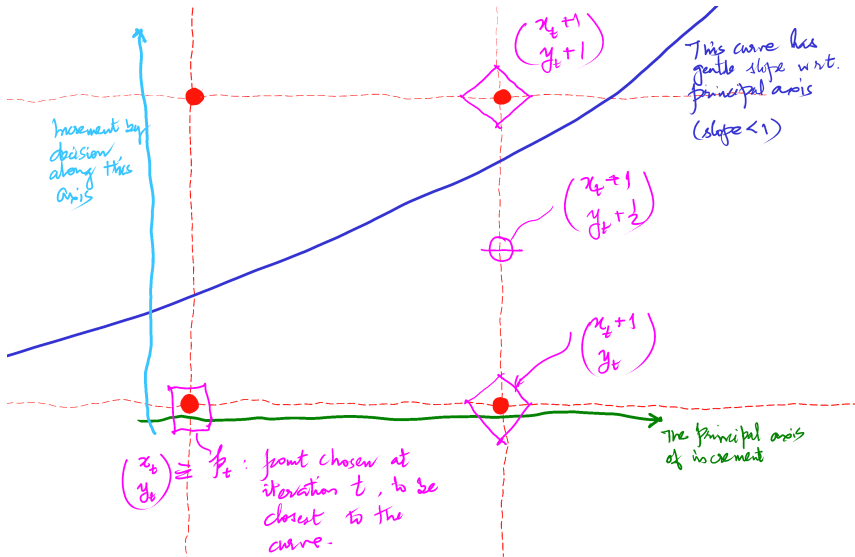
In a quantised (pixelated or discrete) 2d plane, find the set of points that visually approximate a given curve, say a straight line or a conic.

Iteratively, increment along one axes,  
with respect to which, the slope of the curve  
is gentle.

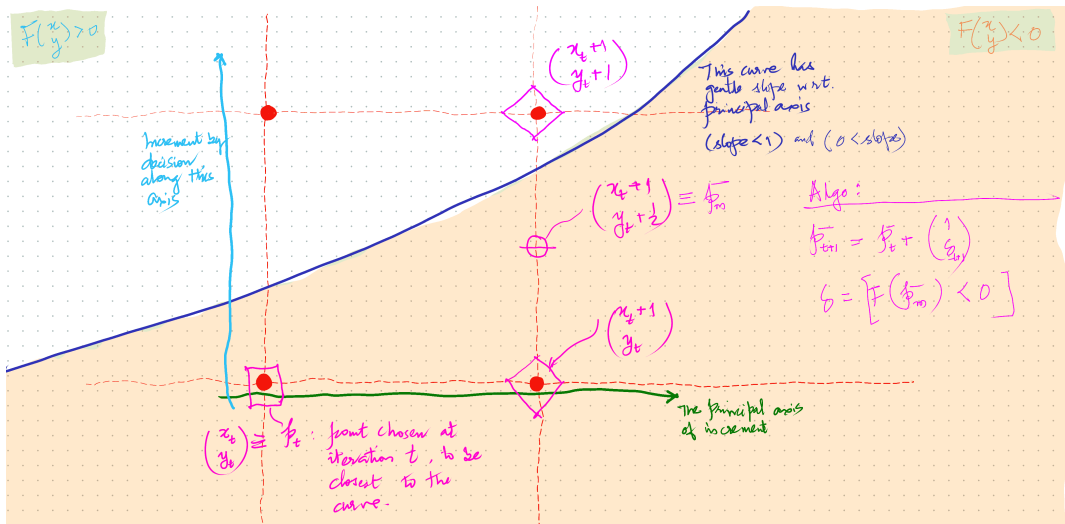
Decide whether it is required to increment  
along the perpendicular axis or not.

Increment if required.

# EXAMPLE



# EXAMPLE





## CONDITIONS FOR APPLICATION OF MID-POINT ALGORITHM

Mid-point algorithm is applicable to a curve within a given finite interval, **iff**

- 1 The curve increases monotonically;
- 2 The curve increases gradually.

In other words,

$$0 \leq dy \leq dx$$

---

## Algorithm 1: Generic Mid-point Algorithm

---

**Input:**  $x_0, x_{\max} \in \mathbb{Z}$

Start and end x-coordinates.

**Input:**  $F: \mathbb{R}^2 \rightarrow \mathbb{R}$

Signed Distance Function from the curve.

**Output:**  $C \equiv \{\mathbf{p}_0, \dots, \mathbf{p}_{\max}\} \subset \mathbb{Z}^2$

Curve in discrete 2D space.

1  $\mathbf{p}_0 \leftarrow \begin{bmatrix} x_0 \\ \lceil y_0 \rceil \end{bmatrix} \vdash F\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = 0$

2 **for**  $t \in \{1, \dots, \max\}$  **do**

3      $\mathbf{p}_{\text{mid}} \leftarrow \mathbf{p}_{t-1} + \begin{pmatrix} 1 \\ \frac{1}{2} \end{pmatrix}$

4      $\delta_t \leftarrow I[F(\mathbf{p}_{\text{mid}}) < 0]$

5      $\mathbf{p}_t \leftarrow \mathbf{p}_{t-1} + \begin{pmatrix} 1 \\ \delta_t \end{pmatrix}$

---

## 1 2D GEOMETRY — INTRODUCTION

## 2 MID-POINT ALGORITHM

- Fundamentals
- Straight Line
- Circle

## CHARACTERISING STRAIGHT LINES

$$F(x, y) = Ax - By + C$$

$$0 \leq dy/dx \leq 1 \quad \mapsto \quad 0 \leq A \leq B$$

...CASE 1

$$-1 \leq dy/dx \leq 0 \quad \mapsto \quad 0 \leq A \leq -B$$

...CASE 2

$$0 \leq dx/dy \leq 1 \quad \mapsto \quad 0 \leq B \leq A$$

...CASE 3

$$-1 \leq dx/dy \leq 0 \quad \mapsto \quad 0 \leq -B \leq A$$

...CASE 4

Read more [...]

# MID-POINT ALGO FOR ST. LINE

Case 1.  $0 < A < B$

---

## Algorithm 2: Mid-Point Algorithm for Straight Line

---

1 **Function** MID-POINT-ALGO-ST-LINE-BASE  $(x_1, y_1, N, a, b, c)$  **is** Base case.  
    **Input:**  $x_1, y_1, N \in \mathbb{Z} \vdash 0 < N$  Start coordinates and num points.  
    **Input:**  $a, b, c \in \mathbb{Z} \vdash 0 \leq a < b; b \text{ even}$  Coefficients:  $F(x, y) = ax - by + c$ .  
    **Output:**  $C \equiv \{(x_1, y_1), \dots, (x_N, y_N)\} \subset \mathbb{Z}^2$   
    An ordered sequence; a curve in discrete 2D space.  
2  $\delta_1 \leftarrow ax_1 - by_1 - \frac{b}{2} + c$   $\frac{b}{2} \in \mathbb{Z}$  because  $b$  even.  
3 **for**  $t \in \{2, \dots, N\}$  **do**  
4      $x_t \leftarrow x_{t-1} + 1$   
5      $\delta_t \leftarrow \delta_{t-1} + a - b \cdot I[0 \leq \delta_{t-1}]$   
6      $y_t \leftarrow y_{t-1} + I[0 \leq \delta_t]$   
7 **return**  $C \equiv \{(x_1, y_1), \dots, (x_N, y_N)\}$

---

Case 1.  $0 < A < B$

---

**Algorithm 3:** Mid-Point Algorithm for Straight Line (alternate)

---

[...CONTD]

**Input:**  $x_1, y_1, N \in \mathbb{Z} \vdash 0 < N$

Start and end x-coordinates.

**Input:**  $a, b, c \in \mathbb{Z} \vdash 0 \leq a < b; b \text{ even}$

Coefficients:  $F(x, y) = ax - by + c$ .

**Output:**  $C \equiv \{(x_1, y_1), \dots, (x_N, y_N)\} \subset \mathbb{Z}^2$

An ordered sequence; a curve in discrete 2D space.

1 **Init:**  $C \leftarrow \emptyset$

Initialise as array.

2 **Init:**  $(x, y, \delta) \leftarrow (x_1, y_1, 0)$

Initialise as integers.

3  $\delta \leftarrow ax - by - \frac{b}{2} + c$

$\frac{b}{2} \in \mathbb{Z}$  because  $b$  even.

4  $C \cdot \text{PUSH}((x, y))$

$(x, y)$  is a tuple.

---

## MID-POINT ALGO FOR ST. LINE

Case 1.  $0 < A < B$

---

**Algorithm:** [CONTD ...] Mid-Point Algorithm for Straight Line (alternate)

---

```
7 for  $t \in \{2, \dots, N\}$  do
8    $x \leftarrow x + 1$                                      Increment along x-axis.
9    $\delta \leftarrow \delta + a - b \cdot I[0 \leq \delta]$        Update decision param  $\delta$ .
10   $y \leftarrow y + I[0 \leq \delta]$                            Update along y-axis based on decision param.
11   $C \cdot \text{PUSH}((x, y))$                                     $(x, y)$  is a tuple.
12 return  $C$ 
```

---

# MID-POINT ALGO FOR ST. LINE

Handle all cases (Case 2)

---

## Algorithm 5: Mid Point Algorithm for Straight Lines (all cases)

---

[...CONTD]

1 **Function** MID-POINT-ALGO-ST-LINE ( $x_1, x_{\max}, a, b, c$ ) **is**

2 A wrapper around MID-POINT-ALGO-ST-LINE-BASE

**Input:**  $x_1, x_{\max} \in \mathbb{Z} \vdash x_1 < x_{\max}$

Start and end x-coordinates.

**Input:**  $a, b, c \in \mathbb{Z} \vdash a, b \text{ even}$

SDF  $F(x, y) \triangleq ax - by + c$ .

**Output:**  $\mathbb{C} \equiv \{(x_1, y_1), \dots, (x_{\max}, y_{\max})\}$

Collection of points on curve.

3    $(y_1, y_{\max}) \leftarrow (\lceil \frac{ax_1+c}{b} \rceil, \lceil \frac{ax_{\max}+c}{b} \rceil)$

Compute start and end y-coordinates.

4   **if**  $0 < a < -b$  **then**

Case 2.  $F'(x, y) = F(x, -y)$

5      $(x_1, y_1, N, a, b, c) \leftarrow (x_1, -y_1, x_{\max} - x_1 + 1, a, -b, c)$

Flip about x-axis.

6     **define:** TRF  $((x, y)) \mapsto (x, -y)$

Define inverse transformation.

---



## MID-POINT ALGO FOR ST. LINE

Handle all cases (Case 3, 4)

---

**Algorithm:** [CONTD...] Mid Point Algorithm for Straight Lines (all cases) [...CONTD]

---

```
7
8   else if  $0 < b < a$  then                                     Case 3.  $F'(x, y) = F(-y, -x)$ 
9        $(x_1, y_1, N, a, b, c) \leftarrow (-y_{\max}, -x_{\max}, -y_1 + y_{\max} + 1, b, a, c)$       Transpose.
10      define:  $\text{TRF}((x, y)) \mapsto (-y, -x)$                                 Define inverse transformation.
11  else if  $0 < -b < a$  then                                     Case 4.  $F'(x, y) = F(-y, x)$ 
12       $(x_1, y_1, N, a, b, c) \leftarrow (y_{\max}, -x_{\max}, y_1 - y_{\max} + 1, -b, a, c)$       Both flip and transpose.
13      define:  $\text{TRF}((x, y)) \mapsto (-y, x)$                                 Define inverse transformation.
```

---

Handle all cases (Case 1)

---

**Algorithm:** [CONTD...] Mid Point Algorithm for Straight Lines (all cases)

---

```

14
15  else Case 1.  $0 < a < b$ 
16     $N \leftarrow x_{\max} - x_1 + 1$ 
17    define:  $\text{TRF}((x, y)) \mapsto (x, y)$  Identity.
18     $C \leftarrow \text{MID-POINT-ALGO-ST-LINE-BASE}(x_1, y_1, N, a, b, c)$ 
19     $C \leftarrow \text{MAP}(\text{TRF}, C)$ 
20  return  $C$ 

```

---

## EXERCISE 1

Using Bresenham's/ Mid-point Algorithm,  
compute the points along the following  
lines, between

1  $(2, 0) \rightarrow (6, 2)$

2  $(0, 1) \rightarrow (6, 13)$

3  $(0, 1) \rightarrow (6, -2)$

4  $(0, 4) \rightarrow (6, -8)$

# SOLUTION 1 STEP 1

For each part, tabulate  $a, b, c, x_1, x_{\max}, y_1, y_{\max}, a', b', c', x'_1, x'_{\max}, y'_1, y'_{\max}, N$ .

Part	$x_1$	$y_1$	$x_{\max}$	$y_{\max}$	$a$	$b$	$c$	$a'$	$b'$	$c'$	$x'_1$	$y'_1$	$x'_{\max}$	$y'_{\max}$	$N$
1	2	0	6	2	1	2	-2	1	2	-2	2	0	6	2	5
2	0	1	6	13	2	1	1	1	2	1	-13	-6	-1	0	13
3	0	1	6	-2	1	-2	-2	1	2	-2	0	-1	6	2	7
4	0	4	6	-8	2	-1	-4	1	2	-4	-8	-6	4	0	13

## SOLUTION 1 STEP 2

Compute the table of  $x, y, \delta, I[0 \leq \delta]$  for each iteration from 1 to  $N$ . Subsequently compute  $\text{TRF}(x, y)$ .

Showing here for part 4.

$x'_1$	$y'_1$	$N$	$a'$	$b'$	$c'$	$\delta_1$
-8	-6	13	1	2	-4	-1

Iteration	1	2	3	4	5	6	7	8	9	10	11	12	13
$x'_t \leftarrow x'_{t-1} + 1$	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4
$\delta_t \leftarrow \delta_{t-1} + a' - b'c_{t-1}$	-1	0	-1	0	-1	0	-1	0	-1	0	-1	0	-1
$c_t \leftarrow I[0 \leq \delta_t]$	0	1	0	1	0	1	0	1	0	1	0	1	0
$y'_t \leftarrow y'_{t-1} + c_t$	-6	-5	-5	-4	-4	-3	-3	-2	-2	-1	-1	0	0
$x \leftarrow -y'$	6	5	5	4	4	3	3	2	2	1	1	0	0
$y \leftarrow x'$	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4

## EXERCISE 2 (PRACTICAL)

- 1 Can the calculations be done using an online spreadsheet with formulae?
- 2 Can the complete algorithm be encoded on a spreadsheet?

## 1 2D GEOMETRY — INTRODUCTION

## 2 MID-POINT ALGORITHM

- Fundamentals
- Straight Line
- Circle

---

## Algorithm 8: Generic Mid-point Algorithm

---

**Input:**  $x_0, x_{\max} \in \mathbb{Z}$

Start and end x-coordinates.

**Input:**  $F: \mathbb{R}^2 \rightarrow \mathbb{R}$

Signed Distance Function from the curve.

**Output:**  $C \equiv \{\mathbf{p}_0, \dots, \mathbf{p}_{\max}\} \subset \mathbb{Z}^2$

Curve in discrete 2D space.

1  $\mathbf{p}_0 \leftarrow \begin{bmatrix} x_0 \\ \lceil y_0 \rceil \end{bmatrix} \vdash F\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = 0$

2 **for**  $t \in \{1, \dots, \max\}$  **do**

3      $\mathbf{p}_{\text{mid}} \leftarrow \mathbf{p}_{t-1} + \begin{pmatrix} 1 \\ \frac{1}{2} \end{pmatrix}$

4      $\delta_t \leftarrow I[F(\mathbf{p}_{\text{mid}}) < 0]$

5      $\mathbf{p}_t \leftarrow \mathbf{p}_{t-1} + \begin{pmatrix} 1 \\ \delta_t \end{pmatrix}$

---



Coming Soon