MCA101 : COMPUTER GRAPHICS

2D GEOMETRY REPRESENTATION

Raghav B. Venkataramaiyer

CSED TIET Patiala India.

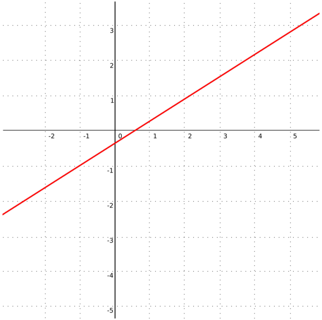
September 5, 2024

OUTLINE

OUTLINE

$y = mx + c$

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



PARAMETRIC FORM

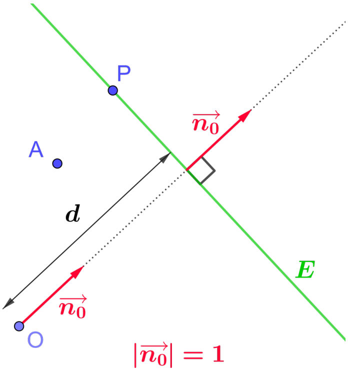
For any two vectors $\mathbf{u}, \mathbf{v} \in V$, a point on the line segment joining them is given parameterised by $t \in [0, 1]$, as

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

PARAMETRIC FORM

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,

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



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.

OUTLINE

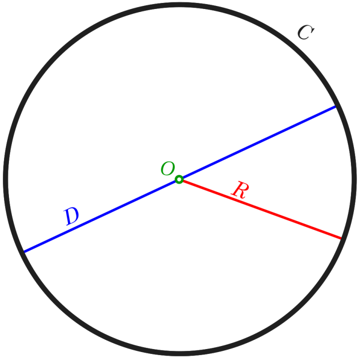


FIGURE: Image Courtesy: Wikipedia

Implicit Form:

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

Parametric Form:

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

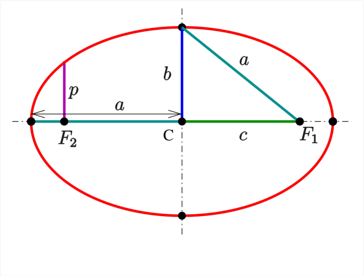


FIGURE: Image Courtesy: Wikipedia

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

OUTLINE

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.

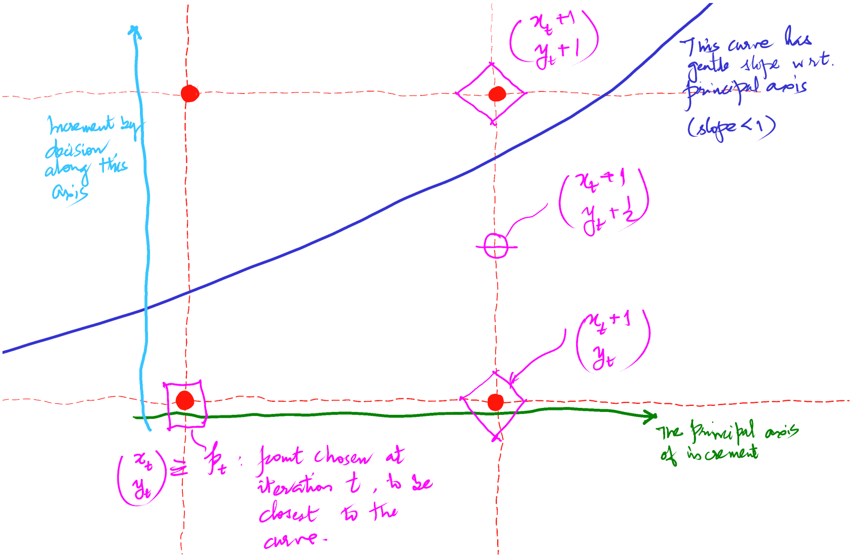
METHOD

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.

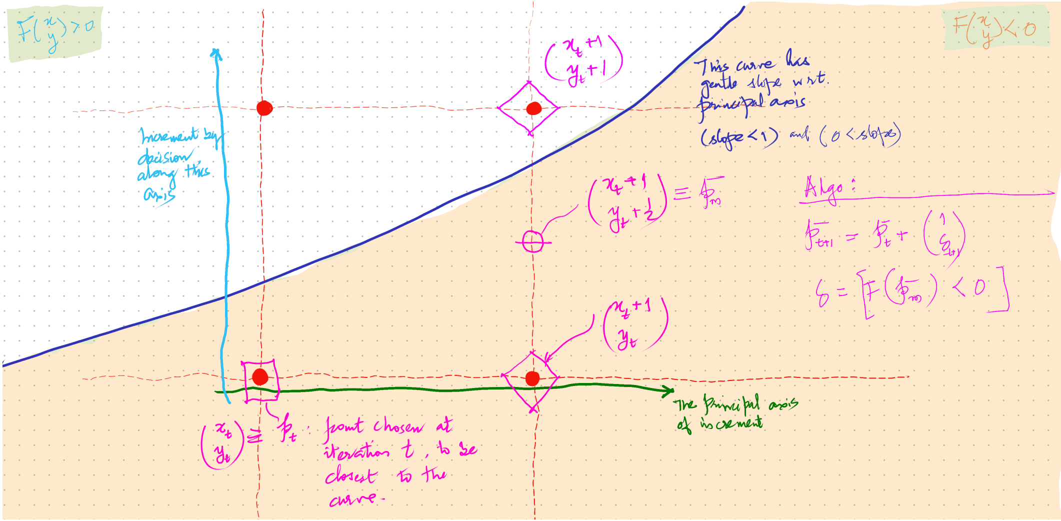
Increment if required.

EXAMPLE



$\begin{pmatrix} x_t+1 \\ y_t+1 \end{pmatrix}$
This curve has gentle slope w.r.t. principal axis (slope < 1)
Increment by decision along this axis
$\begin{pmatrix} x_t+1 \\ y_t+\frac{1}{2} \end{pmatrix}$
$\begin{pmatrix} x_t+1 \\ y_t \end{pmatrix}$
The principal axis of increment
$\begin{pmatrix} x_t \\ y_t \end{pmatrix} \equiv p_t$: point chosen at iteration t, to be closest to the curve.

EXAMPLE


$F\binom{x}{y} > 0$
$F\binom{x}{y} < 0$
This curve has gentle slope w.r.t. principal axis
(slope < 1) and (0 < slope)
Increment by decision along this axis
$\binom{x_t+1}{y_t+1}$
$\binom{x_t+1}{y_t+\frac{1}{2}} \equiv p_m$
$\binom{x_t+1}{y_t}$
Algo:
$p_{t+1} = p_t + \binom{1}{\delta_{t+1}}$
$\delta = [F(p_m) < 0]$
$\binom{x_t}{y_t} \equiv p_t$: point chosen at iteration t, to be closest to the curve.
The principal axis of increment

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,

$$0 \leqslant dy \leqslant dx$$

GENERIC ALGO

Algorithm 1: Generic Mid-point Algorithm

Input: $x_0, x_{\max} \in \mathbb{Z}$ — Start and end x-coordinates.

Input: $F : \mathbb{R}^2 \to \mathbb{R}$ — Signed Distance Function from the curve.

Output: $C \equiv \{\mathbf{p}_0, \ldots, \mathbf{p}_{\max}\} \subset \mathbb{Z}^2$ — Curve in discrete 2D space.

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

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

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

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

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

OUTLINE

CHARACTERISING STRAIGHT LINES

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

$$\begin{array}{rcll}
0 \leqslant \mathrm{d}y/\mathrm{d}x \leqslant 1 & \mapsto & 0 \leqslant A \leqslant B & \ldots\text{CASE } 1 \\
-1 \leqslant \mathrm{d}y/\mathrm{d}x \leqslant 0 & \mapsto & 0 \leqslant A \leqslant -B & \ldots\text{CASE } 2 \\
0 \leqslant \mathrm{d}x/\mathrm{d}y \leqslant 1 & \mapsto & 0 \leqslant B \leqslant A & \ldots\text{CASE } 3 \\
-1 \leqslant \mathrm{d}x/\mathrm{d}y \leqslant 0 & \mapsto & 0 \leqslant -B \leqslant A & \ldots\text{CASE } 4
\end{array}$$

Read more [...]

MID-POINT ALGO FOR ST. LINE

Case 1. $0 < A < B$

Algorithm 2: Mid-Point Algorithm for Straight Line

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: $a, b, c \in \mathbb{Z} \vdash 0 \leqslant a < b$; b even — Coefficients: $F(x, y) = ax - by + c$.

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

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

$\delta_1 \leftarrow ax_1 - by_1 - \frac{b}{2} + c$ — $\frac{b}{2} \in \mathbb{Z}$ because b even.

for $t \in \{2, \ldots, N\}$ **do**

$\quad x_t \leftarrow x_{t-1} + 1$

$\quad \delta_t \leftarrow \delta_{t-1} + a - b \cdot I[0 \leqslant \delta_{t-1}]$

$\quad y_t \leftarrow y_{t-1} + I[0 \leqslant \delta_t]$

return $C \equiv \{(x_1, y_1), \ldots, (x_N, y_N)\}$

MID-POINT ALGO FOR ST. LINE

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: $a, b, c \in \mathbb{Z} \vdash 0 \leqslant a < b$; b even — Coefficients: $F(x, y) = ax - by + c$.

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

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

Init: $C \leftarrow \emptyset$ — Initialise as array.

Init: $(x, y, \delta) \leftarrow (x_1, y_1, 0)$ — Initialise as integers.

$\delta \leftarrow ax - by - \frac{b}{2} + c$ — $\frac{b}{2} \in \mathbb{Z}$ because b even.

$C \cdot \text{PUSH}\left((x, y)\right)$ — (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)

---

for $t \in \{2, \ldots, N\}$ **do**

- $x \leftarrow x + 1$ — Increment along x-axis.
- $\delta \leftarrow \delta + a - b \cdot I[0 \leqslant \delta]$ — Update decision param δ.
- $y \leftarrow y + I[0 \leqslant \delta]$ — Update along y-axis based on decision param.
- $C \cdot \textsc{push}\left((x, y)\right)$ — (x, y) is a tuple.

return C

MID-POINT ALGO FOR ST. LINE

Handle all cases (Case 2)

Algorithm 5: Mid Point Algorithm for Straight Lines (all cases) [...CONTD]

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

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: $a, b, c \in \mathbb{Z} \vdash a, b$ even — SDF $F(x, y) \triangleq ax - by + c$.

  Output: $\mathbb{C} \equiv \{(x_1, y_1), \ldots, (x_{\max}, y_{\max})\}$ — Collection of points on curve.

  $(y_1, y_{\max}) \leftarrow \left(\left\lceil \frac{ax_1+c}{b} \right\rceil, \left\lceil \frac{ax_{\max}+c}{b} \right\rceil\right)$ — Compute start and end y-coordinates.

  if $0 < a < -b$ **then** — Case 2. $F'(x, y) = F(x, -y)$

    $(x_1, y_1, N, a, b, c) \leftarrow (x_1, -y_1, x_{\max} - x_1 + 1, a, -b, c)$ — Flip about x-axis.

    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]

---

else if $0 < b < a$ **then** — Case 3. $F'(x, y) = F(-y, -x)$

    $(x_1, y_1, N, a, b, c) \leftarrow (-y_{\max}, -x_{\max}, -y_1 + y_{\max} + 1, b, a, c)$ — Transpose.

    define: TRF $((x, y)) \mapsto (-y, -x)$ — Define inverse transformation.

else if $0 < -b < a$ **then** — Case 4. $F'(x, y) = F(-y, x)$

    $(x_1, y_1, N, a, b, c) \leftarrow (y_{\max}, -x_{\max}, y_1 - y_{\max} + 1, -b, a, c)$ — Both flip and transpose.

    define: TRF $((x, y)) \mapsto (-y, x)$ — Define inverse transformation.

MID-POINT ALGO FOR ST. LINE

Handle all cases (Case 1)

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

else — Case 1. $0 < a < b$

  $N \leftarrow x_{\max} - x_1 + 1$

  define: $\text{TRF}\,((x, y)) \mapsto (x, y)$ — Identity.

$C \leftarrow \text{MID-POINT-ALGO-ST-LINE-BASE}(x_1, y_1, N, a, b, c)$

$C \leftarrow \text{MAP}(\text{TRF}, C)$

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, δ, $I[0 \leqslant \delta]$ for each iteration from 1 to N. Subsequently compute TRF (x, y).
Showing here for part 4.

x'_1	y'_1	N	a'	b'	c'	δ_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
$c_t \leftarrow I[0 \leqslant \delta_t]$	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?

SOLUTION 2

[Link to Spreadsheet]

OUTLINE

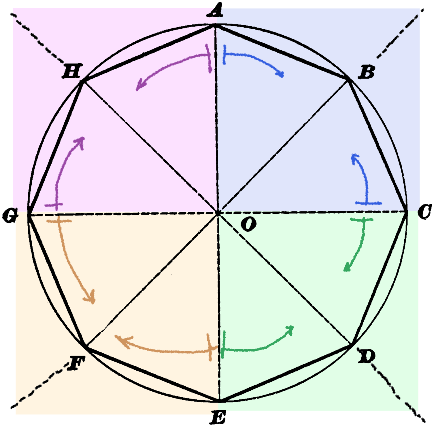


FIGURE: We choose 8-way symmetry of the circle for computational efficiency.

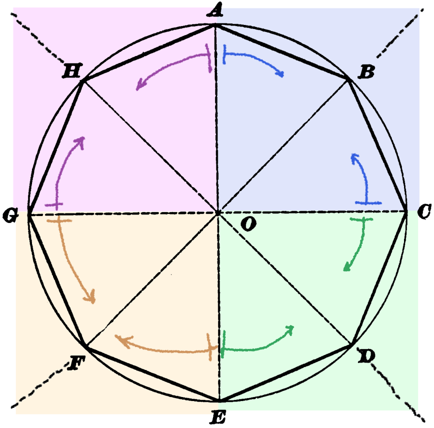
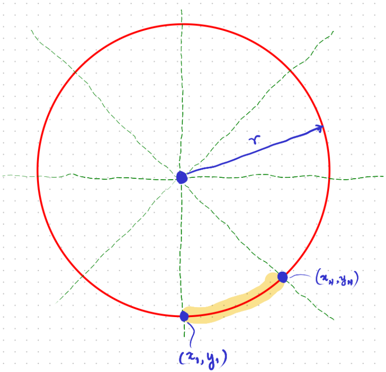


FIGURE: We choose 8-way symmetry of the circle for computational efficiency.

Computing the points on any one of the eight symmetrical sectors (octant), gives us the points on the other seven.
Let (x, y) be the point on one octant, the other seven are given as,

1. $(x, -y)$
2. $(-x, y)$
3. $(-x, -y)$
4. (y, x)
5. $(y, -x)$
6. $(-y, x)$
7. $(-y, -x)$

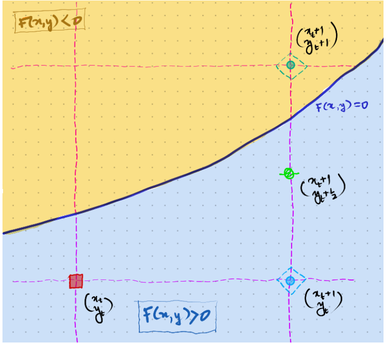


Given a circle with radius r, centred at origin, we intend to compute the points on (or nearest to) the circle within the octant defined by end points,

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} 0 \\ -r \end{bmatrix}$$

$$\begin{bmatrix} x_N \\ y_N \end{bmatrix} = \begin{bmatrix} \left\lceil \frac{r}{\sqrt{2}} \right\rceil \\ -\left\lceil \frac{r}{\sqrt{2}} \right\rceil \end{bmatrix}$$

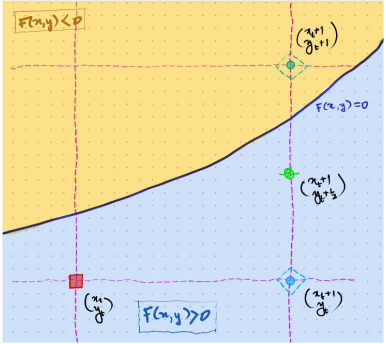
AT ITERATION t



$$F(x, y) = x^2 + y^2 - r^2$$

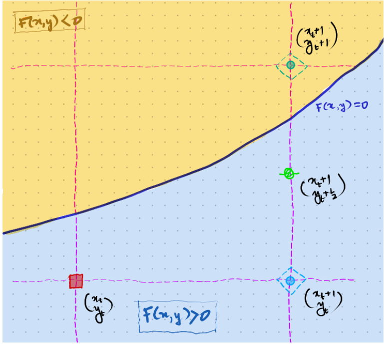
At timestep t, let (x_t, y_t) be the point closest to the curve given by $F(x, y) = 0$.

AT ITERATION $t+1$



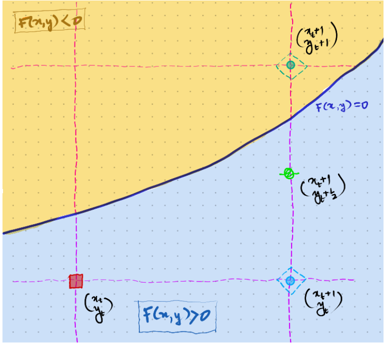
$$
\begin{aligned}
{\color{red}x_{t+1}} &= x_t + 1\\
{\color{red}y_{t+1}} &= y_t + I[\delta_{t+1} > 0]
\end{aligned}
$$

AT ITERATION $t + 1$



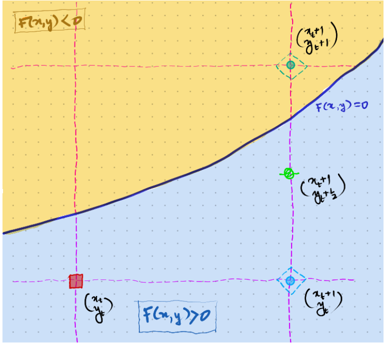
$$x_{t+1} = x_t + 1$$
$$y_{t+1} = y_t + I[\delta_{t+1} > 0]$$
$$\delta_{t+1} = F(x_t + 1, y_t + \frac{1}{2})$$
$$= x_t^2 + 2x_t + 1 + y_t^2 + y_t + \frac{1}{4} - r^2$$

AT ITERATION $t+1$



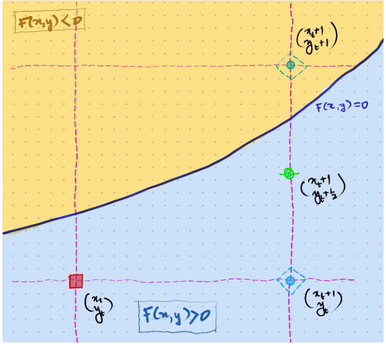
$$
\begin{aligned}
x_{t+1} &= x_t + 1 \\
y_{t+1} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} &= F(x_t + 1, y_t + \frac{1}{2}) \\
&= x_t^2 + 2x_t + 1 + y_t^2 + y_t + \frac{1}{4} - r^2 \\
\delta_t &= F(x_{t-1} + 1, y_{t-1} + \frac{1}{2}) = F(x_t, y_{t-1} + \frac{1}{2}) \\
&= x_t^2 + y_{t-1}^2 + y_{t-1} + \frac{1}{4} - r^2
\end{aligned}
$$

AT ITERATION $t+1$



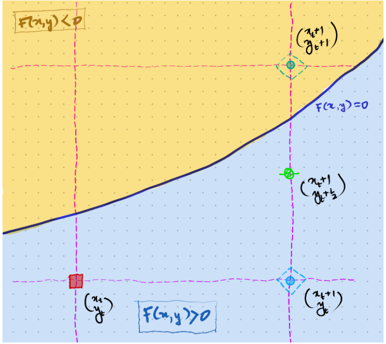
$$
\begin{aligned}
\color{red}{x_{t+1}} &= x_t + 1 \\
\color{red}{y_{t+1}} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} &= F(x_t + 1, y_t + \frac{1}{2}) \\
&= x_t^2 + 2x_t + 1 + y_t^2 + y_t + \frac{1}{4} - r^2 \\
\delta_t &= F(x_{t-1} + 1, y_{t-1} + \frac{1}{2}) = F(x_t, y_{t-1} + \frac{1}{2}) \\
&= x_t^2 + y_{t-1}^2 + y_{t-1} + \frac{1}{4} - r^2 \\
\delta_{t+1} - \delta_t &= 2x_t + 1 + (y_t - y_{t-1})(y_t + y_{t-1} + 1)
\end{aligned}
$$

AT ITERATION $t+1$



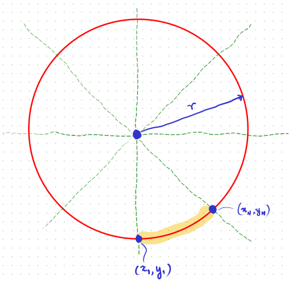
$$
\begin{aligned}
{\color{red}x_{t+1}} &= x_t + 1 \\
{\color{red}y_{t+1}} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} - \delta_t &= 2x_t + 1 + (y_t - y_{t-1})(y_t + y_{t-1} + 1) \\
{\color{red}\delta_{t+1}} &= \delta_t + 2x_t + 1 + I[\delta_t > 0] \cdot (2y_t)
\end{aligned}
$$

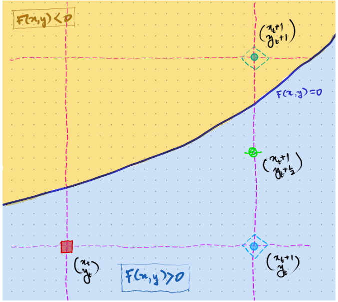
AT ITERATION $t + 1$



$$
\begin{aligned}
\color{red}{x_{t+1}} &= x_t + 1 \\
\color{red}{y_{t+1}} &= y_t + I[\delta_{t+1} > 0] \\
\color{red}{\delta_{t+1}} &= \delta_t + 2x_t + 1 + I[\delta_t > 0] \cdot (2y_t)
\end{aligned}
$$

BOUNDARY CONDITIONS





$$\begin{bmatrix} x_1 & x_N \\ y_1 & y_N \end{bmatrix} = \begin{bmatrix} 0 & \lceil \frac{r}{\sqrt{2}} \rceil \\ -r & -\lceil \frac{r}{\sqrt{2}} \rceil \end{bmatrix}$$

$$\begin{aligned} \delta_2 &= F(x_2, y_1 + \frac{1}{2}) && = F(1, -r + \frac{1}{2}) \\ &= 1 - r + \frac{1}{4} && \approx 1 - r \end{aligned}$$

GENERIC ALGO

Algorithm 8: Generic Mid-point Algorithm

Input: $x_0, x_{\max} \in \mathbb{Z}$ — Start and end x-coordinates.

Input: $F : \mathbb{R}^2 \to \mathbb{R}$ — Signed Distance Function from the curve.

Output: $C \equiv \{\mathbf{p}_0, \ldots, \mathbf{p}_{\max}\} \subset \mathbb{Z}^2$ — Curve in discrete 2D space.

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

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

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

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

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

Algorithm 9: Mid-Point Algorithm for Circle Octant

Function MID-POINT-ALGO-OCTANT (r) **is** — Base case.

  Input: $r \in \mathbb{Z} \vdash 0 < r$ — Radius of the circle.

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

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

  $C \leftarrow \emptyset$ — Initialise array.

  $(x, y, \delta, N) \leftarrow (0, -r, 1 - r, \lceil \frac{r}{\sqrt{2}} \rceil)$ — Initialise with boundary condition.

  $C \cdot \text{PUSH}((x, y))$ — (x, y) is a tuple.

  for $x \in \{1, \ldots, N\}$ **do** — Iterate over x

    $y \leftarrow y + I[\delta > 0]$

    $C \cdot \text{PUSH}((x, y))$ — (x, y) is a tuple.

    $\delta \leftarrow \delta + 2x + 1 + I[\delta > 0] \cdot (2y)$

  return C

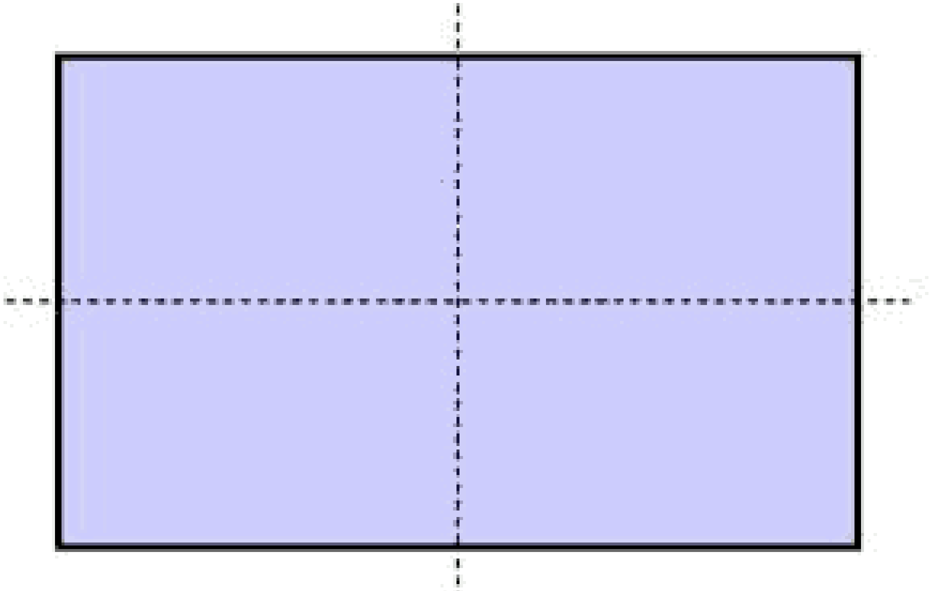
Algorithm 10: Mid-Point Algorithm for Circle

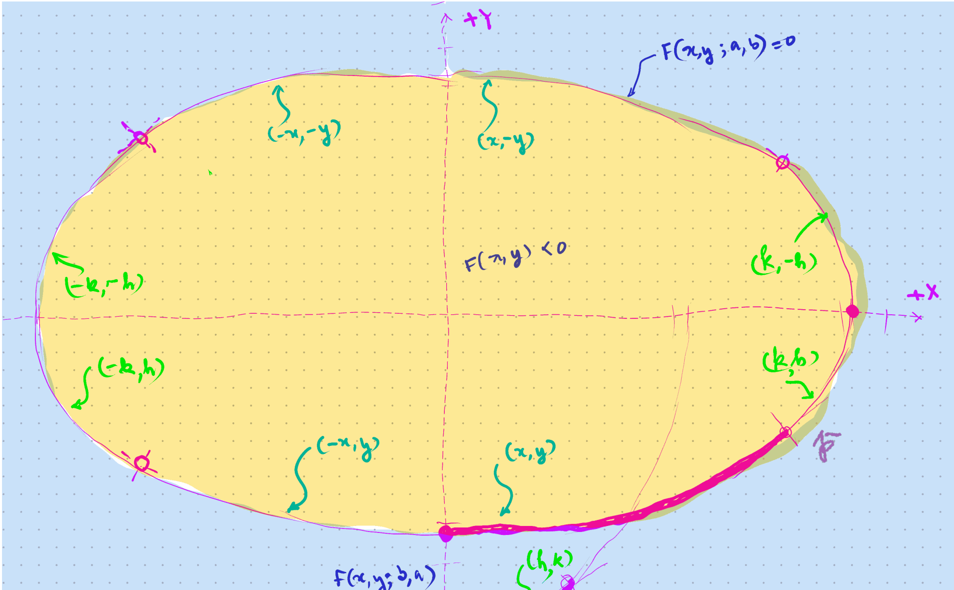
Function MID-POINT-ALGO-CIRCLE (r) **is** — All cases.

- **Input:** $r \in \mathbb{Z} \vdash 0 < r$ — Radius of the circle.
- **Output:** $C \equiv \{(x_1, y_1), \ldots, (x_N, y_N)\} \subset \mathbb{Z}^2$
  An ordered sequence; a curve in discrete 2D space.
- $C \leftarrow$ MID-POINT-ALGO-CIRCLE (r) — Initialise with octant points.
- **define:** QUAD-SYM $((x, y)) \mapsto [(x, y), (x, -y), (-x, y), (-x, -y)]$
- **define:** OCT-SYM $((x, y)) \mapsto$ CONCAT (QUAD-SYM $((x, y))$, QUAD-SYM $((y, x))$)
- $C \leftarrow$ MAP-CONCAT (OCT-SYM, C)
- **return** C

OUTLINE

SYMMETRY





+Y
F(x,y;a,b)=0
(-x,-y)
(x,-y)
F(x,y) <0
(k,-h)
(-k,-h)
+X
(-k,h)
(k,h)
(-x,y)
(x,y)
F(x,y;b,a)
(h,k)

SIGNED DISTANCE FUNCTION (SDF) OF ELLIPSE

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

$$F(x, y; a, b) = b^2 x^2 + z^2 y^2 - a^2 b^2$$

POINT OF INFLECTION

$$
\begin{aligned}
\frac{x^2}{a^2} + \frac{y^2}{b^2} &= 1 \\
\frac{2x\mathrm{d}x}{a^2} + \frac{2y\mathrm{d}y}{b^2} &= 0 \\
\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{\mathbf{p}} = -\frac{b^2 x_p}{a^2 y_p} &= 1 \\
x_p &= -\frac{a^2 y_p}{b^2} \\
&= \pm\frac{a^2}{\sqrt{a^2 + b^2}}
\end{aligned}
$$

AT ITERATION t

Let point (x_t, y_t) be closest to the theoretical curve,

AT ITERATION $t+1$

$$x_{t+1} = x_t + 1$$
$$y_{t+1} = y_t + I[\delta_{t+1} > 0]$$

AT ITERATION $t+1$

$$
\begin{aligned}
\color{red}{x_{t+1}} &= x_t + 1 \\
\color{red}{y_{t+1}} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} &= F(x_t + 1, y_t + \frac{1}{2}) \\
&= b^2(x_t + 1)^2 + a^2(y_t + \frac{1}{2})^2 - a^2b^2
\end{aligned}
$$

AT ITERATION $t+1$

$$
\begin{aligned}
x_{t+1} &= x_t + 1 \\
y_{t+1} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} &= F(x_t + 1, y_t + \frac{1}{2}) \\
&= b^2(x_t + 1)^2 + a^2(y_t + \frac{1}{2})^2 - a^2b^2 \\
&= b^2x_t^2 + 2b^2x_t + b^2 + a^2y_t^2 + a^2y_t + a^2 - a^2b^2
\end{aligned}
$$

AT ITERATION $t+1$

$$
\begin{aligned}
x_{t+1} &= x_t + 1 \\
y_{t+1} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} &= F(x_t + 1, y_t + \frac{1}{2}) \\
&= b^2(x_t + 1)^2 + a^2(y_t + \frac{1}{2})^2 - a^2b^2 \\
&= b^2x_t^2 + 2b^2x_t + b^2 + a^2y_t^2 + a^2y_t + a^2 - a^2b^2 \\
\delta_t &= b^2x_t^2 + a^2y_{t-1}^2 + a^2y_{t-1} + a^2 - a^2b^2
\end{aligned}
$$

AT ITERATION $t+1$

$$
\begin{aligned}
x_{t+1} &= x_t + 1 \\
y_{t+1} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} &= F(x_t + 1, y_t + \frac{1}{2}) \\
&= b^2(x_t+1)^2 + a^2(y_t + \frac{1}{2})^2 - a^2b^2 \\
&= b^2x_t^2 + 2b^2x_t + b^2 + a^2y_t^2 + a^2y_t + a^2 - a^2b^2 \\
\delta_t &= b^2x_t^2 + a^2y_{t-1}^2 + a^2y_{t-1} + a^2 - a^2b^2 \\
\delta_{t+1} - \delta_t &= 2b^2x_t + b^2 + a^2(y_{t+1} - y_t)(y_{t+1} + y_t + 1)
\end{aligned}
$$

AT ITERATION $t + 1$

$$
\begin{aligned}
x_{t+1} &= x_t + 1 \\
y_{t+1} &= y_t + I[\delta_{t+1} > 0] \\
\delta_{t+1} - \delta_t &= 2b^2 x_t + b^2 + a^2 (y_{t+1} - y_t)(y_{t+1} + y_t + 1) \\
\delta_{t+1} &= \delta_t + 2b^2 x_t + b^2 + I[\delta_t > 0] \cdot (2a^2 y_t)
\end{aligned}
$$

at t=1, $x = 0$, $y = -b$
at t=2,

$$\begin{aligned}\delta_2 &= F(x_1 + 1, y_1 + \frac{1}{2}) \\ &= b^2 + a^2(-b + \frac{1}{2})^2 - a^2 b^2 \\ &= b^2 - a^2 b + \frac{a^2}{4}\end{aligned}$$