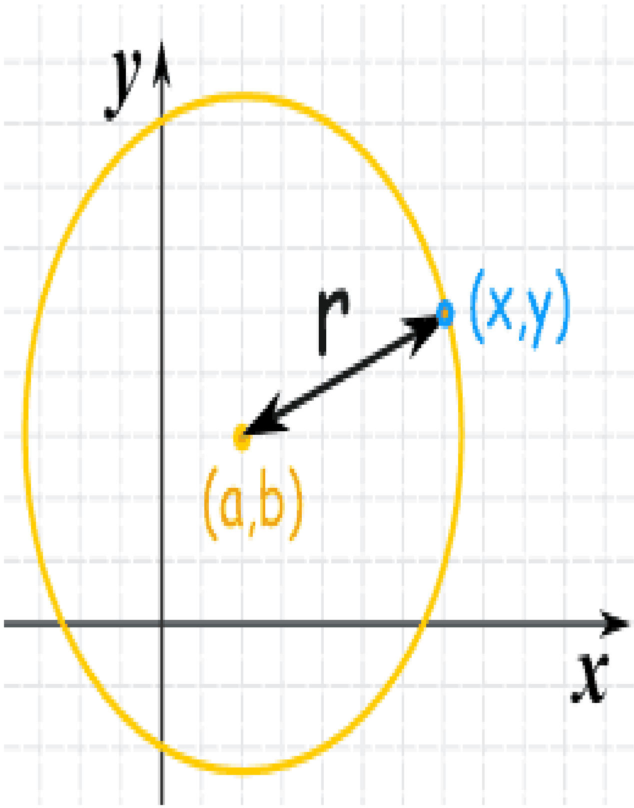
Mid-Point Circle Drawing Algorithm

Strathmore
UNIVERSITY

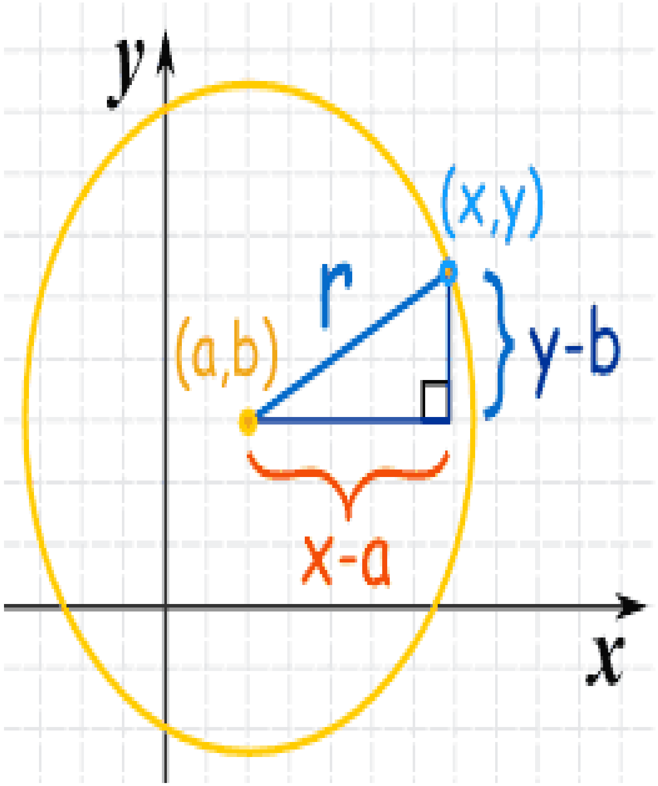
What is a circle?

- The set of all points on a plane that are a fixed distance from a center.
- Let us put that center at **(a,b)**.
- So the circle is **all the points (x,y)** that are "**r**" away from the center **(a,b)**.



Standard formula of a circle

- Now we can work out **exactly** where all those points are by making a right-angled triangle (as shown), and then use Pythagoras theorem ($a^2 + b^2 = c^2$):
- $(x-a)^2 + (y-b)^2 = r^2$



Class Exercise

- What is the center of the circle $(x - 5)^2 + (y + 3)^2 = 49$?

- What is the radius of the circle $(x + 2)^2 + (y - 4)^2 = 36$?

- Routines for generating basic curves such as circles and ellipses are not included as primitive functions in OpenGL core library.

- Then how do we display curves?

1. We can generate a curve by <u>approximating its dimensions using a polyline.</u>

- This involves locating a set of points along the curve path and connecting the points with straight line segments.
- The more line sections we include, the smoother the curve/circle.

2. We can generate a curve by writing curve generation functions based on some algorithms. For example midpoint circle algorithm.

A Simple Circle Drawing Algorithm

The equation for a circle is:

$$x^2 + y^2 = r^2$$

where r is the radius of the circle

So, we can write a simple circle drawing algorithm by solving the equation for y at unit x intervals using:

$$y = \pm\sqrt{r^2 - x^2}$$

A Simple Circle Drawing Algorithm (cont…)

$$y_0 = \sqrt{20^2 - 0^2} \approx 20$$

$$y_1 = \sqrt{20^2 - 1^2} \approx 20$$

$$y_2 = \sqrt{20^2 - 2^2} \approx 20$$

$$\vdots$$

$$y_{19} = \sqrt{20^2 - 19^2} \approx 6$$

$$y_{20} = \sqrt{20^2 - 20^2} \approx 0$$

A Simple Circle Drawing Algorithm (cont…)

However, unsurprisingly this is not a brilliant solution!

Firstly, the resulting circle has large gaps where the slope approaches the vertical

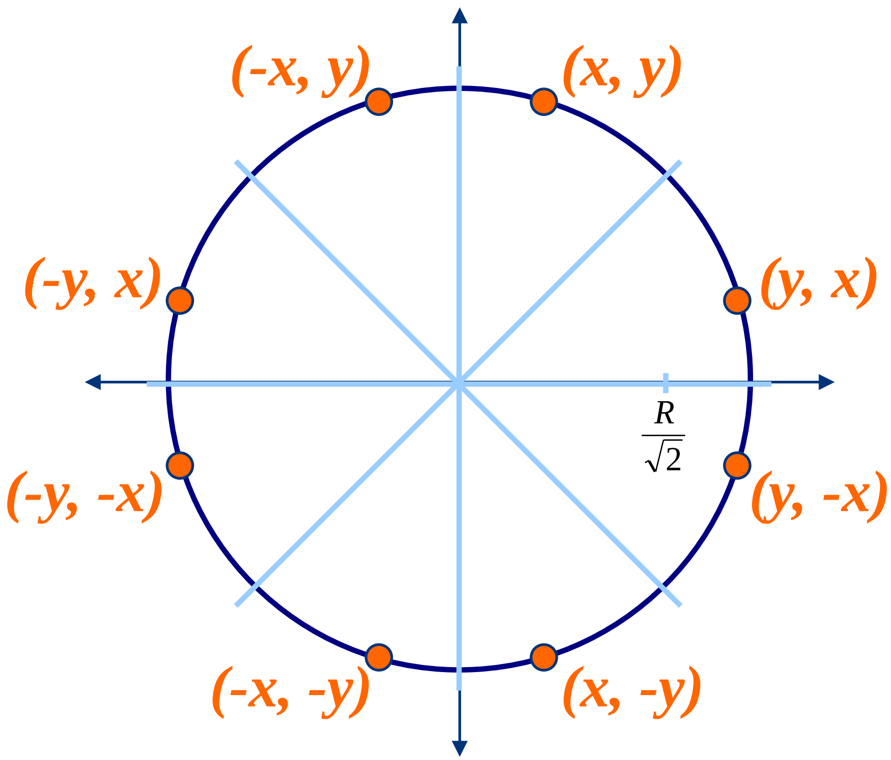
Secondly, the calculations are not very efficient

- The square (multiply) operations
- The square root operation – try really hard to avoid these!

We need a more efficient, more accurate solution

Eight-Way Symmetry

The first thing we can notice to make our circle drawing algorithm more efficient is that circles centred at $(0, 0)$ have *eight-way symmetry*



Mid-Point Circle Algorithm

Similarly to the case with lines, there is an incremental algorithm for drawing circles – the *mid-point circle algorithm*

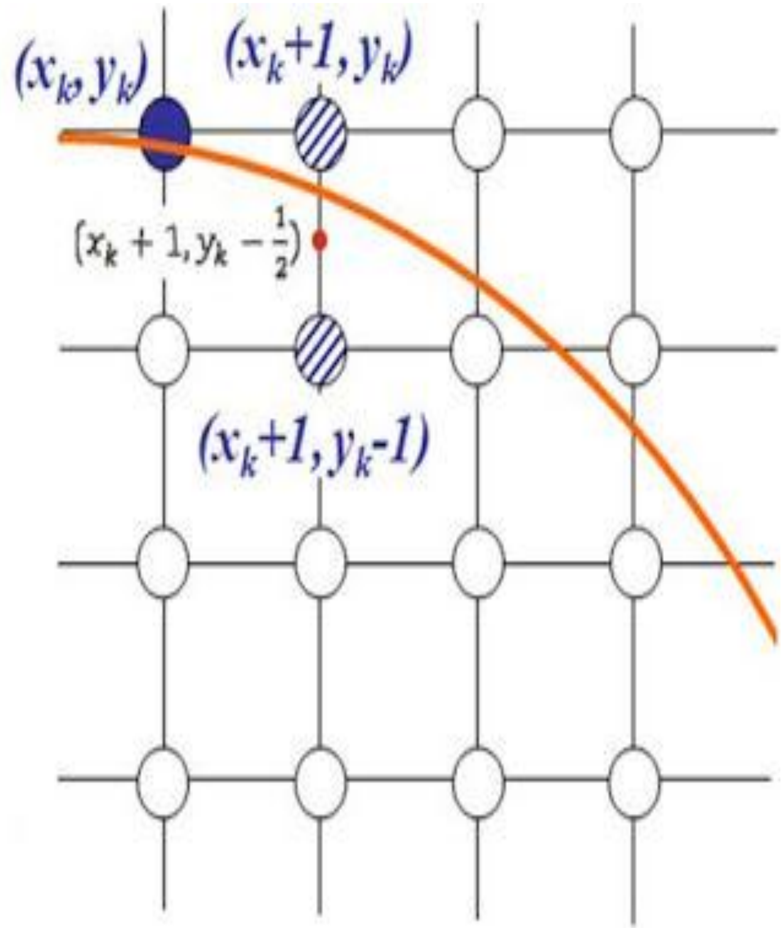
In the mid-point circle algorithm we use eight-way symmetry so only ever calculate the points for the top right eighth of a circle, and then use symmetry to get the rest of the points

The mid-point circle algorithm was developed by Jack Bresenham, Remember him?

Mid-Point Circle Algorithm (cont…)

Assume that we have just plotted point (x_k, y_k)

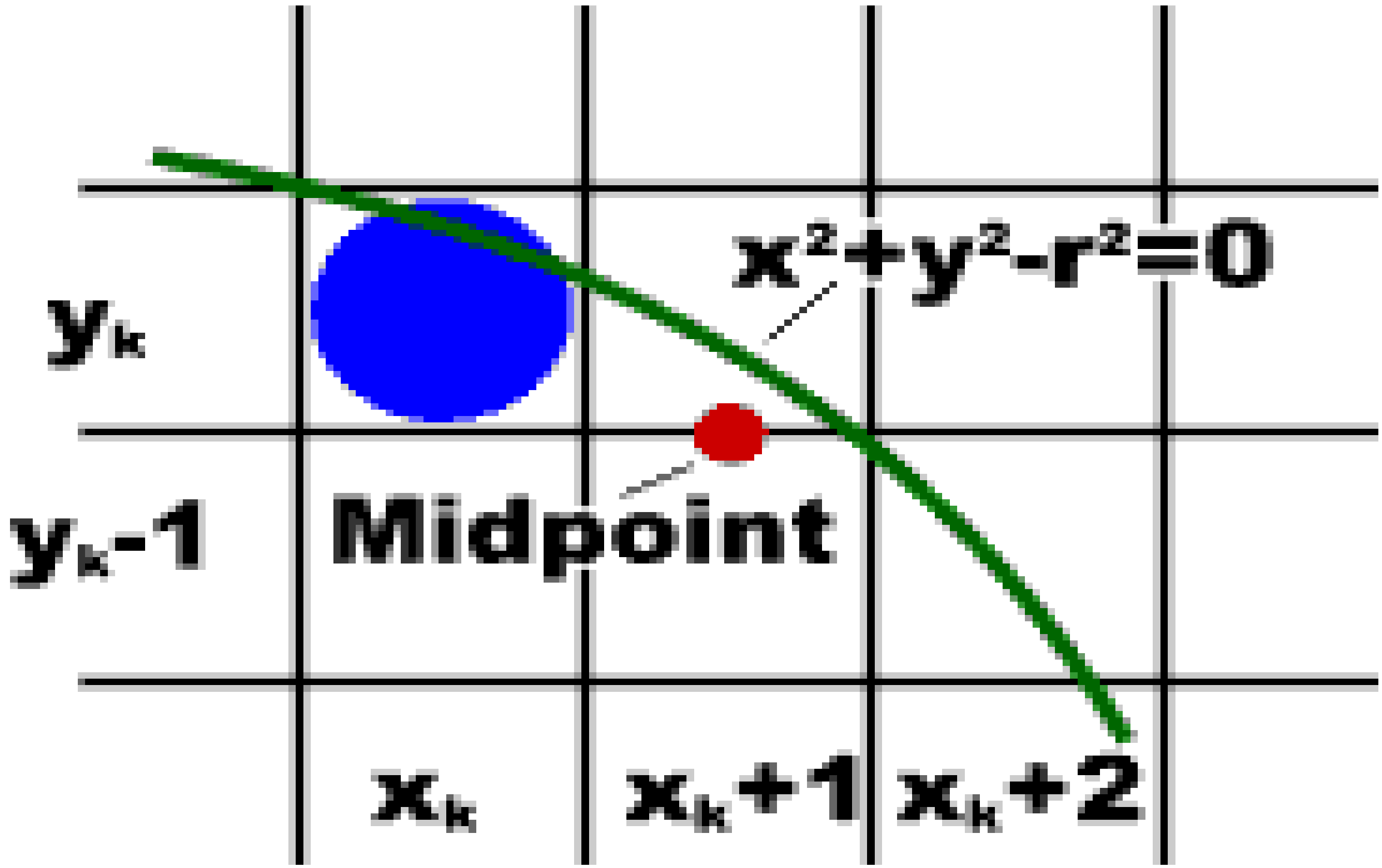
The next point is a choice between (x_k+1, y_k) and (x_k+1, y_k-1)



We would like to choose the point that is nearest to the actual circle

So how do we make this choice?

Midpoint Circle Algorithm



Mid-Point Circle Algorithm (cont…)

Let's re-jig the equation of the circle slightly to give us:

$$f_{circ}(x, y) = x^2 + y^2 - r^2$$

Any point (x, y) on the boundary of the circle with radius (r) satisfies the equation

$$f_{circle}(x,y) = x^2 + y^2 - r^2 = 0$$

Mid-Point Circle Algorithm (cont…)

The equation evaluates as follows:

$$f_{circ}(x, y)\begin{cases} < 0, & \text{if } (x, y) \text{ is inside the circle boundary} \\ = 0, & \text{if } (x, y) \text{ is on the circle boundary} \\ > 0, & \text{if } (x, y) \text{ is outside the circle boundary} \end{cases}$$

By evaluating this function at the midpoint between the candidate pixels we can make our decision

Mid-Point Circle Algorithm (cont…)

Assuming we have just plotted the pixel at (x_k, y_k) so we need to choose between (x_k+1, y_k) and (x_k+1, y_k-1)

Our decision variable can be defined as:

$$p_k = f_{circ}(x_k + 1, y_k - \tfrac{1}{2})$$
$$= (x_k + 1)^2 + (y_k - \tfrac{1}{2})^2 - r^2$$

If $p_k < 0$ the midpoint is inside the circle and the pixel at y_k is closer to the circle

Otherwise the midpoint is outside and y_k-1 is closer

Mid-Point Circle Algorithm (cont…)

- Successive decision parameters are obtained using incremental calculations, thus avoiding a lot of computation at each step. We obtain a recursive expression for the next decision parameter i.e. at the *k+1*th step, in the following manner.
- $$= (x_k + 1)^2 + (y_k - \tfrac{1}{2})^2 - r^2$$

Mid-Point Circle Algorithm (pk+1)

$$Pk = (x_k + 1)^2 + (y_k - 1/2)^2 - r^2$$

$$p_{k+1} = f\left(x_{k+1} + 1, y_{k+1} - \frac{1}{2}\right) = (x_{k+1} + 1)^2 + (y_{k+1} - \frac{1}{2})^2 - r^2$$

Or, $p_{k+1} = (x_k + 1 + 1)^2 + (y_{k+1} - \frac{1}{2})^2 - r^2$

Or, $p_{k+1} = (x_k + 2)^2 + (y_{k+1} - \frac{1}{2})^2 - r^2$ --------------------------(5)

(5)-(4) gives

$$p_{k+1} - p_k = (x_k + 2)^2 - (x_k + 1)^2 + (y_{k+1} - \frac{1}{2})^2 - (y_k - \frac{1}{2})^2 - r^2 + r^2$$

Or, $p_{k+1} = p_k + (2x_k + 3).1 + (y_{k+1} + y_k - 1)(y_{k+1} - y_k)$ -------(6)

Mid-Point Circle Algorithm (pk<=0)

Now if P_k <=0, then the midpoint of the two possible pixels lies within the circle, thus north pixel is nearer to the theoretical circle. Hence, $Y_{k+1} = Y_k$. Substituting this value in the previous equation we have ;

$$p_{k+1} = p_k + (2x_k + 3) + (y_k + y_k - 1)(y_k - y_k)$$

$$\text{Or, } p_{k+1} = p_k + (2x_k + 3)$$

Mid-Point Circle Algorithm (pk>0)

If $p_{k\,>}$ 0 then the midpoint of the two possible pixels lies outside the circle, thus south pixel is nearer to the theoretical circle. Hence, Y_{k+1}= Y_k-1. we have . Substituting this value in the previous equation we have ;

$$p_{k+1} = p_k + (2x_k + 3) + (y_k - 1 + y_k - 1)(y_k - y_k - 1)$$

$$\text{Or, } p_{k+1} = p_k + 2(x_k - y_k) + 5$$

Mid-Point Circle Algorithm (Starting value)

For the boundary condition, we have x=0, y=r. Substituting these values , we have

$$p_k = f\left(x_k + 1, y_k - \frac{1}{2}\right) = (x_k + 1)^2 + (y_k - \frac{1}{2})^2 - r^2$$

$$p_0 = (0+1)^2 + (r - \frac{1}{2})^2 - r^2 = 1 + r^2 + \frac{1}{4} - r - r^2 = \frac{5}{4} - r$$

Midpoint Circle Algorithm exercise

To see the mid-point circle algorithm in action lets use it to draw a circle centred at (0,0) with radius 8.

Midpoint Circle Algorithm exercise

K	X previous	Y previous	Decision parameter (p)	X new	Y new
0			1-r= -7	1	8
1	0	8	-7+0+3=-4	2	8
2	1	8	-4+2+3=1	3	7
3	2	8	1+(4-16)+5=-6	4	7
4	3	7	-6+6+3=3	5	6
5	4	7	3+(8-14)+5=2	6	5
				(Stop when x>=y)	

Class exercise

- Calculate the required points in the first quadrant to plot a circle with a radius of 10 and centered at the origin using the midpoint circle algorithm.
- Complete all the other points in other quadrants

Solution

K	X $_{previous}$	Y $_{previous}$	Decision parameter (p)	X $_{new}$	Y $_{new}$
0			1-10= -9	1	10
1	0	10	-9+0+3=-6	2	10
2	1	10	-6+2+3=-1	3	10
3	2	10	-1+4+3=6	4	9
4	3	10	6+(6-20)+5=-3	5	9
5	4	9	-3+8+3=8	6	8
6	5	9	8+(10-18)+5=5	7	7

Class exercise

- Calculate the required points in the first quadrant to plot a circle with a radius of 10 and centered at (3, 4)using the midpoint circle algorithm.
- Complete all the other points in other quadrants

Strathmore

UNIVERSITY

Ole Sangale Road, Madaraka Estate. PO Box 59857-00200, Nairobi, Kenya
Tel: (+254) (0)703 034000/200/300 Fax : +254 (0)20 607498
Email: info@strathmore.edu Website: www.strathmore.edu