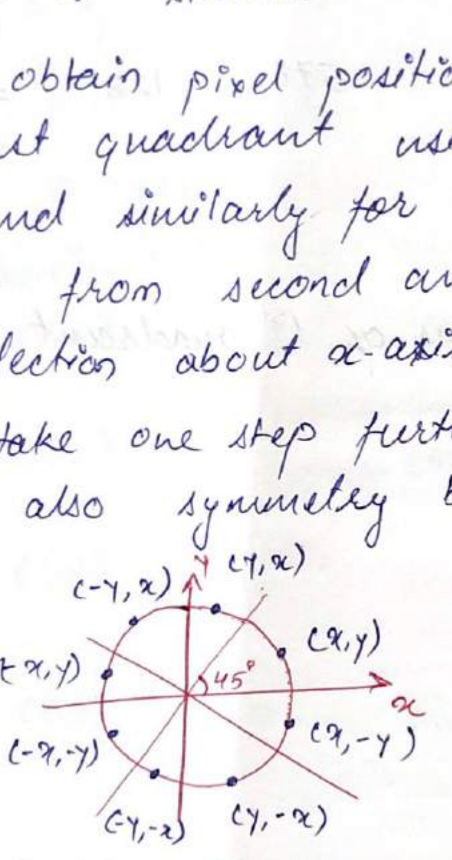
Mid point circle drawing Algorithm.

- A circle is defined as the set of points that are all at a given distance r from a center position say (x_c, y_c)
- To reduce computation we consider symmetry property of circle as the shape of circle is similar in each quadrant.
- we can obtain pixel position in second quadrant from first quadrant using reflection about y-axis and similarly for third and forth quadrant from second and first resp. using reflection about x-axis.
- we can take one step further and note that there is also symmetry between octants.



Symmetry of circle.

- similar to raster line algorithm we sample at unit interval and determine the closest pixel position to the specified circle path at each step.

- Given radius 'r' and center (x_c, y_c)
- we first setup our algorithm to calculate circular path co-ordinates for center (0,0). And then we will transfer calculated pixel position to center (x_c, y_c) by adding x_c to x & y_c to y.

- Along the circle section from $x=0$ to $x=y$ in the first quadrant, the slop of the curve varies from 0 to -1. so we can step unit step in positive x direction over this octant and use a decision parameter to determine which of the two possible y position is closer to the circular path.
- for decision parameter we use circle f.n as

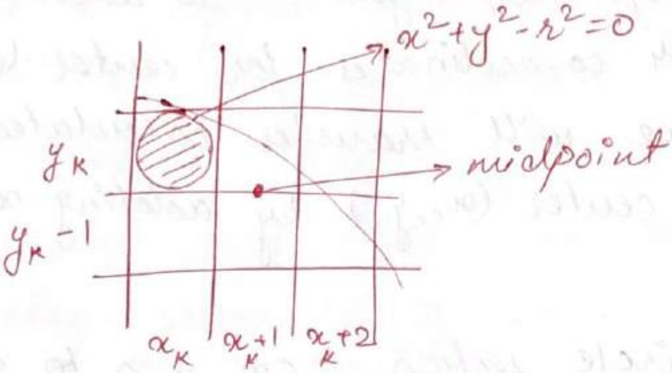
$$f_{circle}(x,y) = x^2+y^2-r^2$$

- for any point (x,y) we have 3 options given as follows

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

- Above eq.n we calculate for mid position b/w pixels near the circular path at each sampling step

- Below figure shows the midpoint b/w the two candidate pixels at sampling position x_k+1



- Assuming we have plotted the pixel at (x_k, y_k) and next we need to determine whether the pixel at position (x_k+1, y_k) or the one at position (x_k+1, y_k-1) is closer to circle boundary.
- So for finding which pixel is more closer we use decision parameter evaluated at the midpoint b/w two candidate pixel as below

(x_k+1, y_K) (x_k+1, y_k-1)

$$x_{mid} = \frac{x_k+1+x_k+1}{2} = \frac{2x_k+2}{2} = x_k+1$$

$$y_{mid} = \frac{y_k+y_k-1}{2} = \frac{2y_k-1}{2} = y_k-\frac{1}{2}$$

$$P_k = f_{circle}\left(x_k+1, y_k-\frac{1}{2}\right)$$

$$= (x_k+1)^2 + \left(y_k-\frac{1}{2}\right)^2 - r^2 \quad \text{—} \textcircled{1}$$

- If $P_K < 0$ this ~~mo~~ point is inside the circle and the pixel on the scan line y_K is closer to circle boundary.
  otherwise midpoint is outside or on the boundary and we select the scan line $y_K - 1$
- successive decision parameters are obtain using incremental calculation as follows:

$$P_{K+1} = f_{circle}(x_{K+1}+1,\ y_{K+1} - \tfrac{1}{2})$$

$$= (x_{K+1}+1)^2 + (y_{K+1} - \tfrac{1}{2})^2 - r^2$$

$$= [(x_K+1)+1]^2 + (y_{K+1} - \tfrac{1}{2})^2 - r^2 \quad — \text{②}$$

Now we subtrack P_K from P_{K+1}

$$P_{K+1} - P_K = [\{(x_K+1)+1\}^2 + (y_{K+1} - \tfrac{1}{2})^2 - r^2] - [(x_K+1)^2 + (y_K - \tfrac{1}{2})^2 - r^2]$$

$$= \cancel{(x_K+1)^2} + 2(x_K+1) + 1 + y_{K+1}^2 - y_{K+1} + \cancel{\tfrac{1}{4}} - \cancel{r^2} - \cancel{(x_K+1)^2} - y_K^2 + y_K - \cancel{\tfrac{1}{4}} + \cancel{r^2}$$

$$= 2(x_K+1) + 1 + y_{K+1}^2 - y_{K+1} - y_K^2 + y_K$$

$$P_{K+1} = P_K + 2(x_K+1) + (y_{K+1}^2 - y_K^2) - (y_{K+1} - y_K) + 1$$

- In the above eq.^{n} y_{K+1} is either y_K or $y_K - 1$ depending on sign of P_K

- The initial decision parameter is obtained by evaluating the circle function at the start position $(x_0, y_0) = (0, r)$ as follows.

$$P_0 = f_{circle}(0+1, r-\tfrac{1}{2})$$

$$= 1^2 + (r-\tfrac{1}{2})^2 - r^2$$

$$= 1 + r^2 - r + \tfrac{1}{4} - r^2$$

$$= \tfrac{5}{4} - r \approx 1-r$$

Algorithm for midpoint circle generation

1. I/P radius r and circle center (x_c, y_c) and obtain the first point on the circumference of a circle centered on the origin as

$$(x_0, y_0) = (0, r)$$

2. Calculate the initial value of decision parameter as

$$P_0 = 1-r$$

3. At each x_k position, starting at $k=0$, perform the following test.

If $P_k < 0$, the next point to plot is (x_k+1, y_k) &

$$P_{k+1} = P_k + 2x_{k+1} + 1$$

otherwise, the next point to plot is (x_k+1, y_k-1)

$$P_{k+1} = P_k + 2x_{k+1} + 1 - 2y_{k+1}$$

4. Determine symmetry points in other seven octents.

5. Move each calculated pixel position (x,y) onto the circular path centered on (x_c, y_c) and plot the co-ordinate values:

$$x = x + x_c, \quad y = y + y_c$$

6. Repeat steps 3 to 5 until $x \geq y$.