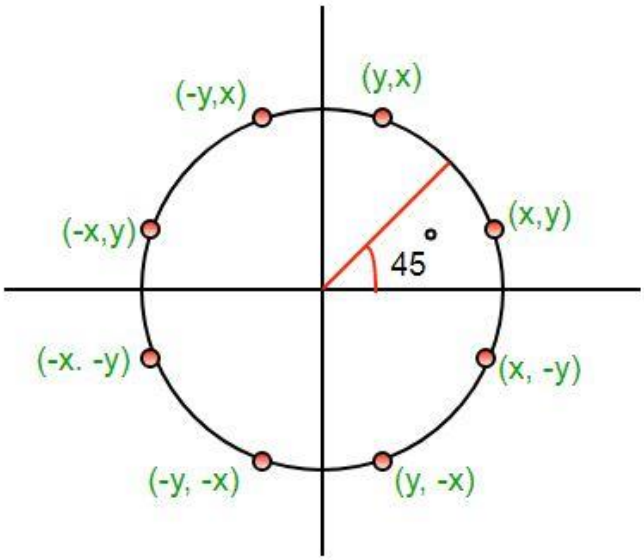
Experiment No 4

Aim: To implement Midpoint Circle Algorithm.

Theory: In computer graphics, the mid-point circle drawing algorithm is used to calculate all the perimeter points of a circle. In this algorithm, the mid-point between the two pixels is calculated which helps in calculating the decision parameter. The value of the decision parameter will decide which pixel should be chosen for drawing the circle. This algorithm only calculates the points for one octant and the points for other octants are generated using the eight-way symmetry for the circle.

The mid-point circle drawing algorithm is an algorithm used to determine the points needed for rasterizing a circle. We use the mid-point algorithm to calculate all the perimeter points of the circle in the first octant and then print them along with their mirror points in the other octants. This will work because a circle is symmetric about its centre.



This algorithm draws all eight octants simultaneously, starting from each cardinal direction (0°, 90°, 180°, 270°) and extends both ways to reach the nearest multiple of 45° (45°, 135°, 225°, 315°). It can determine where to stop because when $y = x$, it has reached 45°. The reason for using these angles is shown in the above picture: As x increases, it does not skip nor repeat any x value until reaching 45°. So during the while loop, x increments by 1 each iteration, and y decrements by 1 on occasion, never exceeding 1 in one iteration. This changes at 45° because that is the point where the tangent is rise=run. Whereas rise>run before and rise<run after.

The second part of the problem, the determinant, is far trickier. This determines when to decrement y. It usually comes after drawing the pixels in each iteration, because it

never goes below the radius on the first pixel. Because in a continuous function, the function for a sphere is the function for a circle with the radius dependent on *z* (or whatever the third variable is), it stands to reason that the algorithm for a discrete(voxel) sphere would also rely on this Midpoint circle algorithm.

The steps involved in Midpoint circle algorithm are:

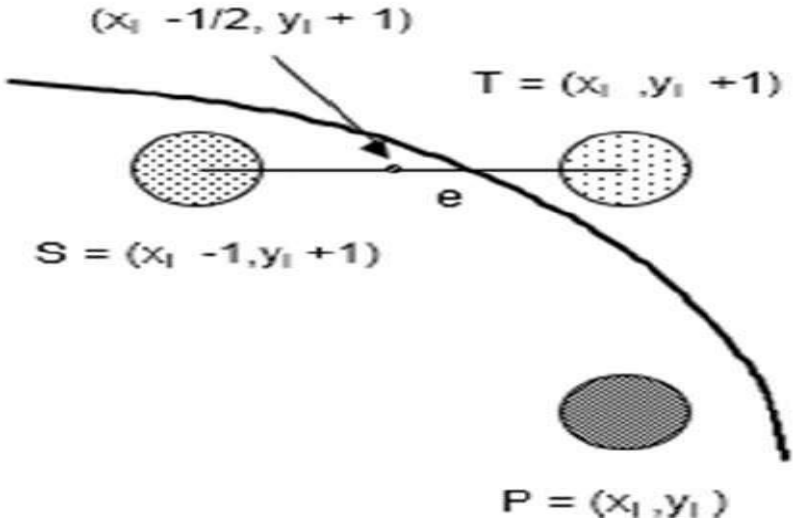
Step 1 − Input radius **r** and circle center (xc,yc)and obtain the first point on the circumference of the circle centered on the origin as

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

Step 2 − Calculate the initial value of decision parameter as

$P0 = 5/4 – r$

$f(x, y) = x^2 + y^2 - r^2 = 0$
$f(x_i - 1/2 + e, y_i + 1)$
$= (x_i - 1/2 + e)^2 + (y_i + 1)^2 - r^2$
$= (x_i - 1/2)^2 + (y_i + 1)^2 - r^2 + 2(x_i - 1/2)e + e^2$
$= f(x_i - 1/2, y_i + 1) + 2(x_i - 1/2)e + e^2 = 0$



Let $d_i = f(x_i - 1/2, y_i + 1) = -2(x_i - 1/2)e - e^2$
Thus,
If $e < 0$ then $di > 0$ so choose point $S = (x_i - 1, y_i + 1)$.
$d_{i+1} = f(x_i - 1 - 1/2, y_i + 1 + 1) = ((x_i - 1/2) - 1)^2 + ((y_i + 1) + 1)^2 - r^2$
$= d_i - 2(x_i - 1) + 2(y_i + 1) + 1$
$= d_i + 2(y_{i+1} - x_{i+1}) + 1$

If $e >= 0$ then $di <= 0$ so choose point $T = (x_i, y_i + 1)$
$d_{i+1} = f(x_i - 1/2, y_i + 1 + 1)$

$= d_i + 2y_{i+1} + 1$

The initial value of di is
$d_0 = f(r - 1/2, 0 + 1) = (r - 1/2)^2 + 1^2 - r^2$
$= 5/4 - r$ {1-r can be used if r is an integer}

When point $S = (x_i - 1, y_i + 1)$ is chosen then
$d_{i+1} = d_i + -2x_{i+1} + 2y_{i+1} + 1$

When point $T = (x_i, y_i + 1)$ is chosen then
$d_{i+1} = d_i + 2y_{i+1} + 1$

Step 3 − At each XK position starting at K=0, perform the following test −

If $P_K < 0$ then next point on circle (0,0) is (X_{K+1}, Y_K) and
$P_{K+1} = P_K + 2X_{K+1} + 1$
Else
$P_{K+1} = P_K + 2X_{K+1} + 1 – 2Y_{K+1}$

Where, $2X_{K+1} = 2X_{K+2}$ and $2Y_{K+1} = 2Y_{K-2}$.

Step 4 − Determine the symmetry points in other seven octants.

Step 5 − Move each calculate pixel position X,Y onto the circular path centered on (XC,YC) and plot the coordinate values.

$X = X + X_C, \quad Y = Y + Y_C$

Step 6 − Repeat step-3 through 5 until X >= Y.

Advantages:

1. It is a powerful and efficient algorithm.
2. The entire algorithm is based on the simple equation of circle X2 + Y2 = R2 .

Disadvantages:

1. Accuracy of the generating points is an issue in this algorithm.
2. The circle generated by this algorithm is not smooth.

Conclusion: In conclusion, the Circle Midpoint Algorithm is a reliable and efficient method for drawing circles in computer graphics. Its simplicity, speed, and integer-based calculations make it a popular choice in various applications. However, its use may be limited in cases where higher precision or mathematical accuracy is critical. One of the primary advantages of the Circle Midpoint Algorithm is its efficiency. The algorithm uses integer arithmetic and bitwise operations, making it faster than algorithms that involve floating-point calculations or trigonometric functions.