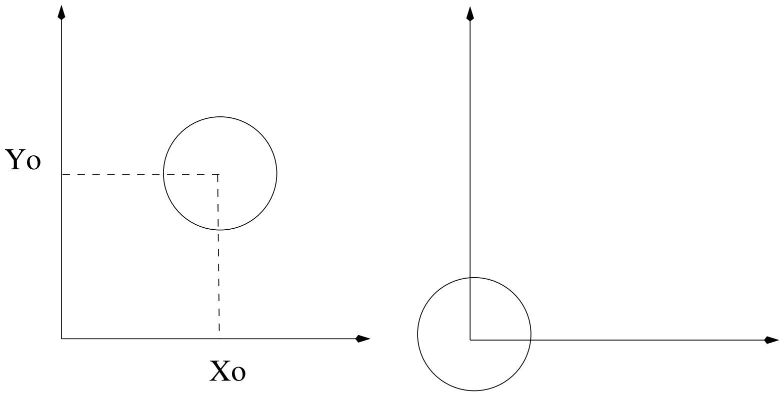
Scan Converting Circles

- **Equation of circle**

$$(x - x_0)^2 + (y - y_0)^2 = R^2$$

$$x^2 + y^2 = R^2$$



- **A simple algorithm**

- Draw only <u>one quarter</u> of the circle

- Use symmetry to draw the other quarters

step1: set x_1=0, $\Delta x = 1$

step2: calculate $y_i = \sqrt{R^2 - x_i^2}$

step3: round y_i: $\hat{y}_i$=round(y_i)

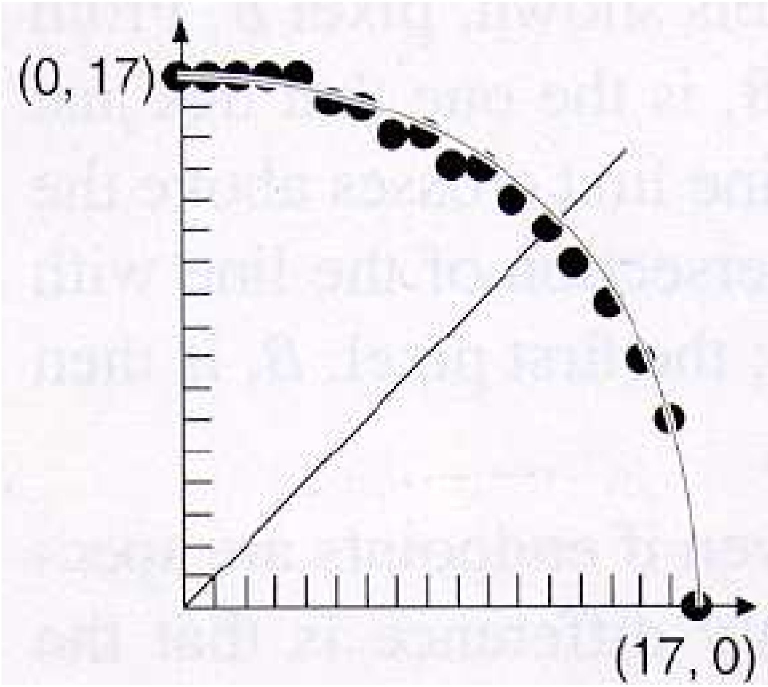
step4: draw (x_i, $\hat{y}_i$)

step5: x_i=x_i^{old}+Δx

step6: if $x_i \le R$, goto step2

- <u>Computational cost:</u> floating point multiplications, subtraction, square-root, rounding

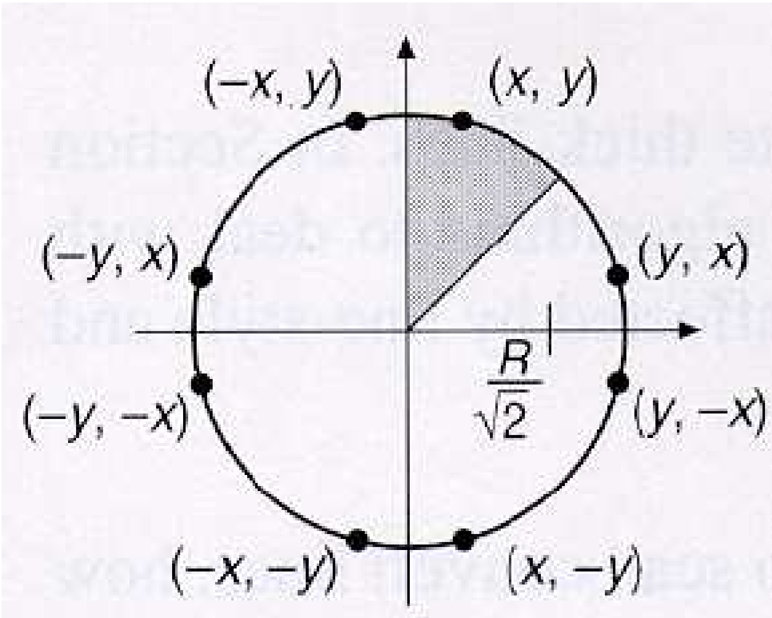
- <u>Problems:</u> large gaps for x values close to R

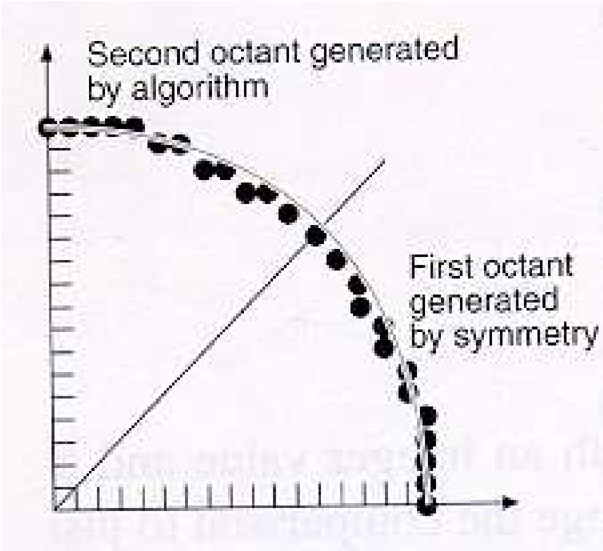


- **Improvements using the eight-way symmetry**

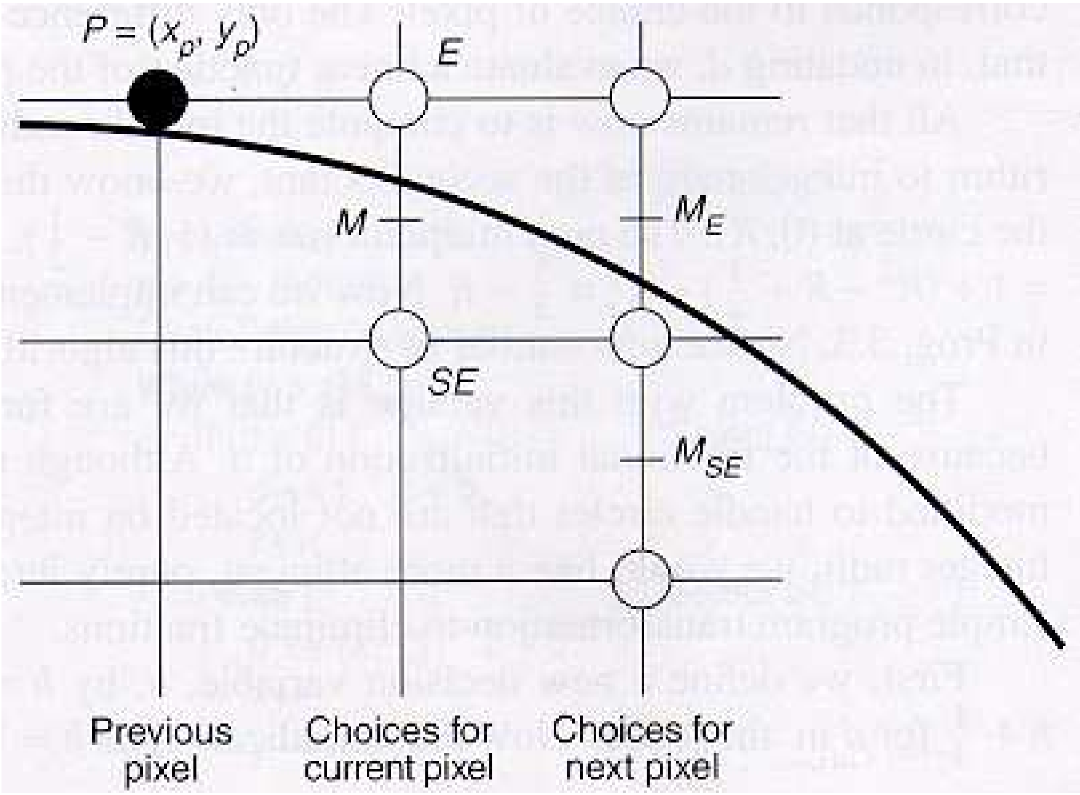
  - Draw only one octant of the circle

  - *x* assumes values from 0 to $R/\sqrt{2}$





- **Midpoint circle algorithm**



* Assume that P has been selected and we must choose between E and SE

* If M is inside the circle, then pixel E is closer to the circle

* If M is outside the circle, then pixel SE is closer to the circle

Implicit representation of circle

$$F(x, y) = x^2 + y^2 - R^2 = 0$$

* Sign of $F(x, y)$

$F(x, y) = 0$, for points on the circle
$F(x, y) > 0$, for points outside the circle
$F(x, y) < 0$, for points inside the circle

- Check the sign of $d = F(x_p + 1, y_p - 1/2)$ to find in which region M lies

if $d > 0$, choose pixel SE
if $d < 0$, choose pixel E
if $d = 0$, pick either E or SE (we pick SE)

- **How d should be computed ? Incrementally !!**

$$d = (x_p + 1)^2 + (y_p - 1/2)^2 - R^2$$

* If E is chosen, then we have:

$d_{new} = F(M_E) = F(x_p + 2, y_p - 1/2) = (x_p + 2)^2 + (y_p - 1/2)^2 - R^2$

Also, $d_{old} = (x_p + 1)^2 + (y_p - 1/2)^2 - R^2$

Take the difference: $\Delta_E = d_{new} - d_{old} = 2x_p + 3$ or $d_{new} = d_{old} + (2x_p + 3)$

* If SE is chosen, then we have:

$d_{new} = F(M_{SE}) = F(x_p + 2, y_p - 3/2) = (x_p + 2)^2 + (y_p - 3/2)^2 - R^2$

Also, $d_{old} = (x_p + 1)^2 + (y_p - 1/2)^2 - R^2$

Take the difference: $\Delta_{SE} = d_{new} - d_{old} = 2x_p - 2y_p + 5$ or $d_{new} = d_{old} + (2x_p - 2y_p + 5)$

* Note that Δ_E, Δ_{SE} are not constants any more !!

- **Summary of the midpoint algorithm**

  - Choose between E and SE, based on the sign of d

  - Update d by adding either Δ_E or Δ_{SE}

- **How should we compute d_{start} ?**

  - The first midpoint will be at $(1, R - 1/2)$

    $F(1, R - 1/2) = 1 + (R - 1/2)^2 - R^2$ or $d_{start} = 5/4 - R$

- **A pure integer-arithmetic algorithm**

    * Define $h = d - 1/4$ (replace d by $h + 1/4$)

    * $h_{start} = 1 - R$

    (Note: $d < 0$ becomes $h < -1/4$ which is equivalent to $h < 0$)

- **Improvement: computing Δ_E and Δ_{SE} incrementally**

  - If E is chosen: (x_p, y_p) --> $(x_p + 1, y_p)$

    $\Delta_{Enew} = 2(x_p + 1) + 3$

    Also, $\Delta_{Eold} = 2x_p + 3$

    Take the difference: $\Delta_{Enew} - \Delta_{Eold} = 2$ or $\Delta_{Enew} = \Delta_{Eold} + 2$

    $\Delta_{SEnew} = 2(x_p + 1) - 2y_p + 5$

    Also, $\Delta_{SEold} = 2x_p - 2y_p + 5$

    Take the difference: $\Delta_{SEnew} - \Delta_{SEold} = 2$ or $\Delta_{SEnew} = \Delta_{SEold} + 2$

- If SE is chosen: (x_p, y_p) --> $(x_p + 1, y_p - 1)$

  $\Delta_{Enew} = 2(x_p + 1) + 3$

  Also, $\Delta_{Eold} = 2x_p + 3$

  Take the difference: $\Delta_{Enew} - \Delta_{Eold} = 2$ or $\Delta_{Enew} = \Delta_{Eold} + 2$

  $\Delta_{SEnew} = 2(x_p + 1) - 2(y_p - 1) + 5$

  Also, $\Delta_{SEold} = 2x_p - 2y_p + 5$

  Take the difference: $\Delta_{SEnew} - \Delta_{SEold} = 4$ or $\Delta_{SEnew} = \Delta_{SEold} + 4$

- How should Δ_E and Δ_{SE} be initialized ?

  * Use the starting point (0, R)

  * $\Delta_{Estart} = 2x_p + 3 = 2.0 + 3 = 3$

  * $\Delta_{SEstart} = 2x_p - 2y_p + 5 = 2.0 - 2R + 5 = 5 - 2R$

• **Summary of the revised algorithm**

- Choose between E and SE, based on the sign of d

- Update d by adding either Δ_E or Δ_{SE}

- Update Δ_E and Δ_{SE}

```
d=5/4-R; Xp=0; Yp=R;
De=3; Dse=5-2R;
draw(Xp,Yp);
while(Xp > Yp) {
  if(d < 0) { /* choose E */
    draw(Xp+1,Yp);
    d=d+De; De=De+2; Dse=Dse+2;
    Xp=Xp+1;
  }
  else { /* choose SE */
    draw(Xp+1,Yp-1);
    d=d+Dse; De=De+2; Dse=Dse+4;
    Xp=Xp+1; Yp=Yp-1;
  }
}
```