实验3:圆绘制算法
Circle Drawing Algorithm

华东师范大学计算机科学与技术学院

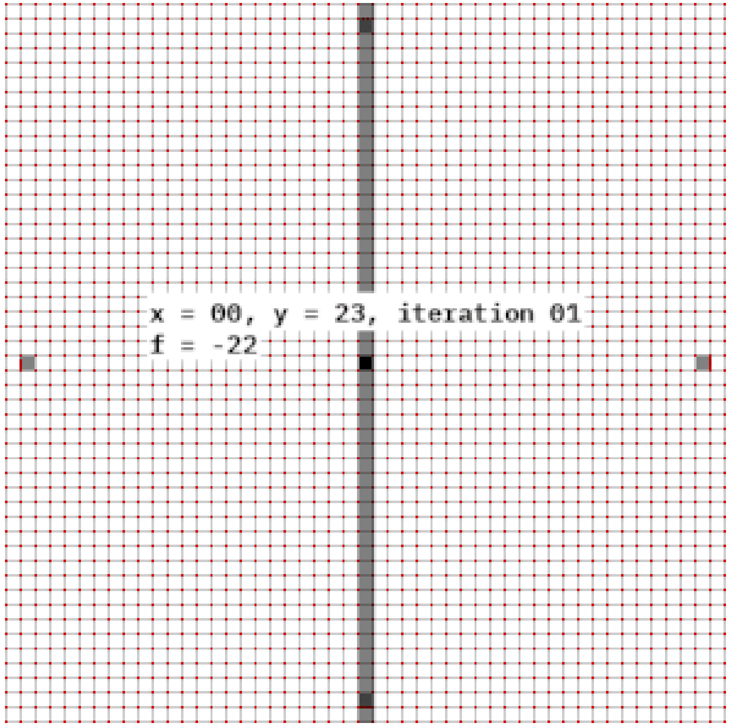
李晨 副研究员

cli@cs.ecnu.edu.cn

华东师范大学计算机科学与技术学院

Contents

- In today's lecture we'll have a look at:
  - Midpoint circle drawing algorithm
  - Bresenham's circle drawing algorithm
  - Exercise using Bresenham algorithm



A Simple Circle Drawing Algorithm

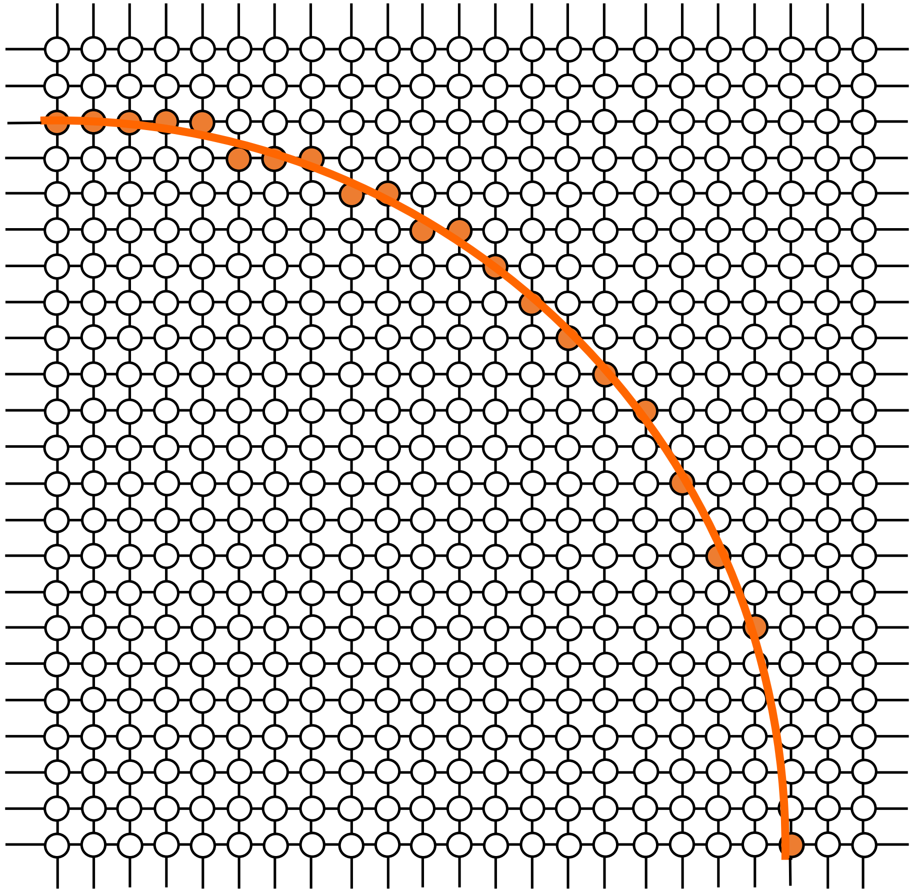
- The equation for a circle is:

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

- 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



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

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

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

$$\vdots$$

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

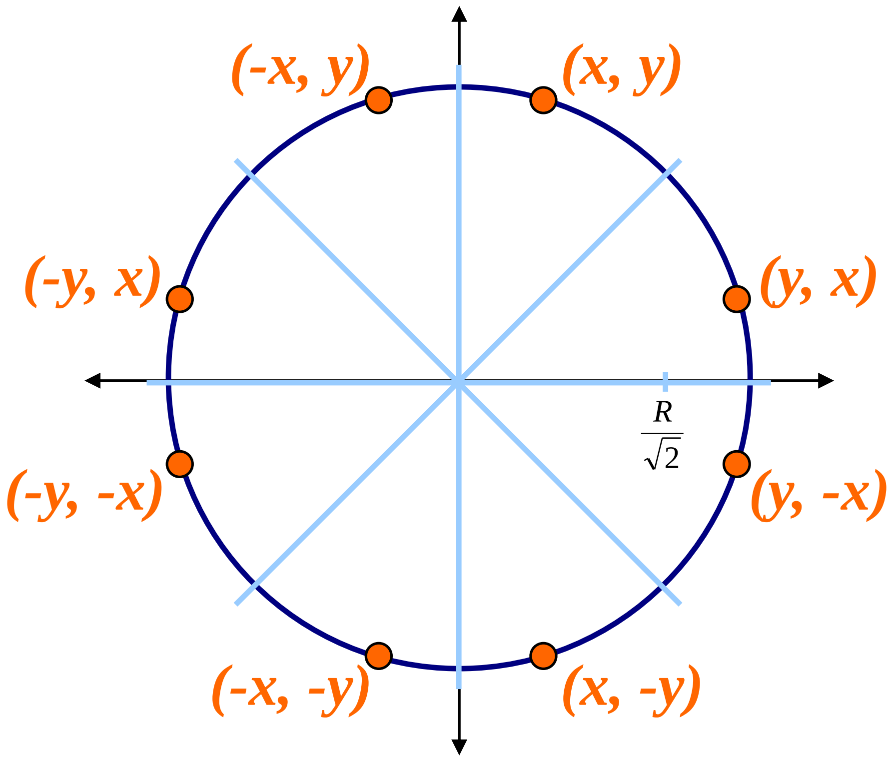
$$y_{20} = \sqrt{20^2 - 20^2} \approx 0$$

A Simple Circle Drawing Algorithm

- 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

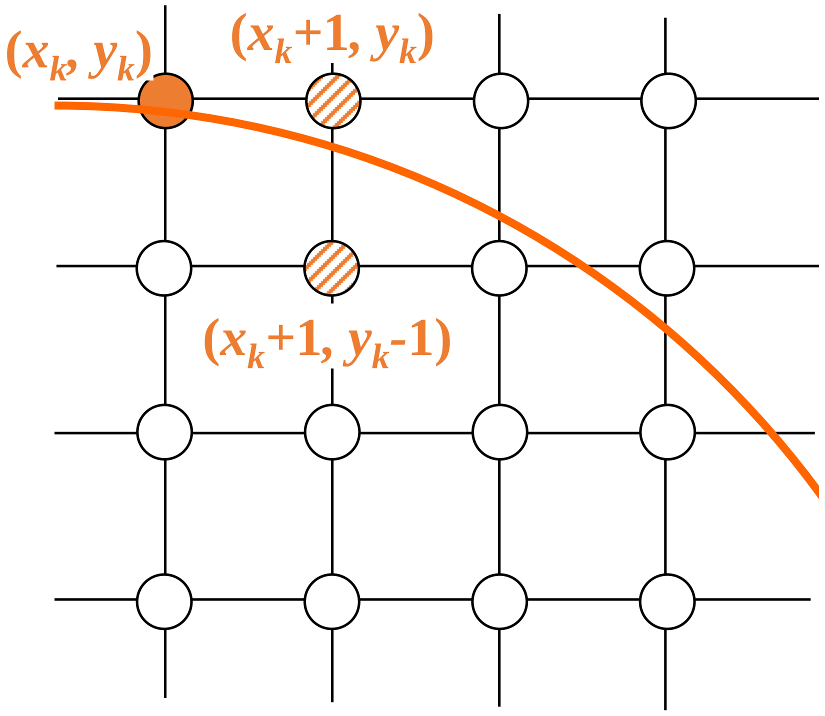
Midpoint Circle Drawing Algorithm

- Similarly to the case with lines, there is an incremental algorithm for drawing circles – the midpoint circle algorithm
- In the midpoint 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



Midpoint Circle Drawing Algorithm

- 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 Drawing Algorithm

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

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

- Our decision variable can be defined as:

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

- 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

Bresenham Circle Drawing Algorithm

- To ensure things are as efficient as possible we can do all of our calculations incrementally

$$p_{k+1} = f_{circ}\left(x_{k+1}+1, y_{k+1}-\tfrac{1}{2}\right)$$

$$= [(x_k+1)+1]^2 + \left(y_{k+1}-\tfrac{1}{2}\right)^2 - r^2$$

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

- where y_{k+1} is either y_k or y_k-1 depending on the sign of p_k

Bresenham Circle Drawing Algorithm

- The first decision variable is given as:

$$
\begin{aligned}
p_0 &= f_{circ}(1, r - \tfrac{1}{2}) \\
&= 1 + (r - \tfrac{1}{2})^2 - r^2 \\
&= \tfrac{5}{4} - r
\end{aligned}
$$

- Then if $p_k < 0$ then the next decision variable is given as: $p_{k+1} = p_k + 2x_{k+1} + 1$
- If $p_k > 0$ then the decision variable is: $p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$

Bresenham Circle Drawing Algorithm

- Input radius r and circle centre (x_c, y_c), then set the coordinates for the first point on the circumference of a circle centred on the origin as:

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

- Calculate the initial value of the decision parameter as:

$$p_0 = \frac{5}{4} - r$$

- if r is an integer, then p_0 can be rounded to $1 - r$.
- Perform the test, starting with $k = 0$ at each position x_k, perform the following test.
  - (i) If $p_k < 0$, the next point along the circle centred on (0, 0) is (x_k+1, y_k) and:

$$p_{k+1} = p_k + 2x_{k+1} + 1$$

Bresenham Circle Drawing Algorithm

- (ii) If $p_k > 0$ then the next point along the circle is (x_k+1, y_k-1) and:

$$p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$$

where $2x_{k+1} = 2x_k + 2$ and $2y_{k+1} = 2y_k - 2$

- Identify the symmetry points in the other seven octants
- Move (x, y) according to:

$$x = x + x_c \qquad y = y + y_c$$

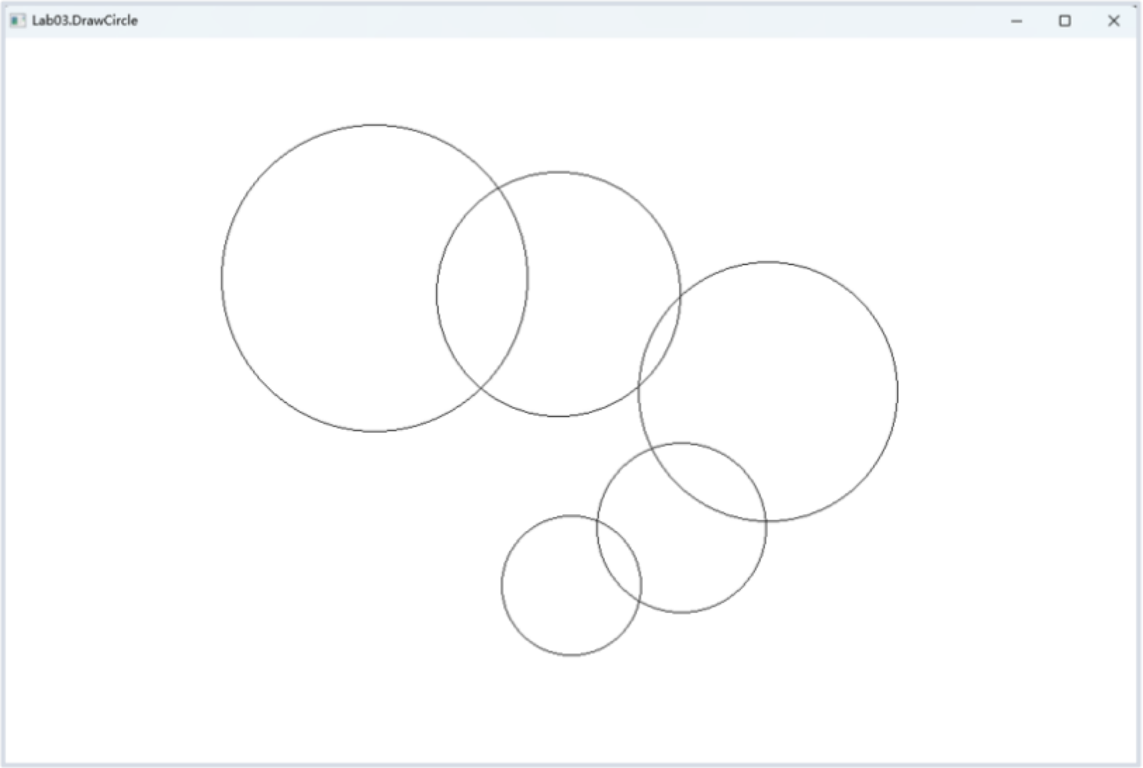
- Repeat steps 3 to 5 until $x >= y$

Circle Drawing Algorithm

- The key insights in the circle algorithm are:
  - Eight-way symmetry can hugely reduce the work in drawing a circle
  - Moving in unit steps along the x axis at each point along the circle's edge we need to choose between two possible y coordinates

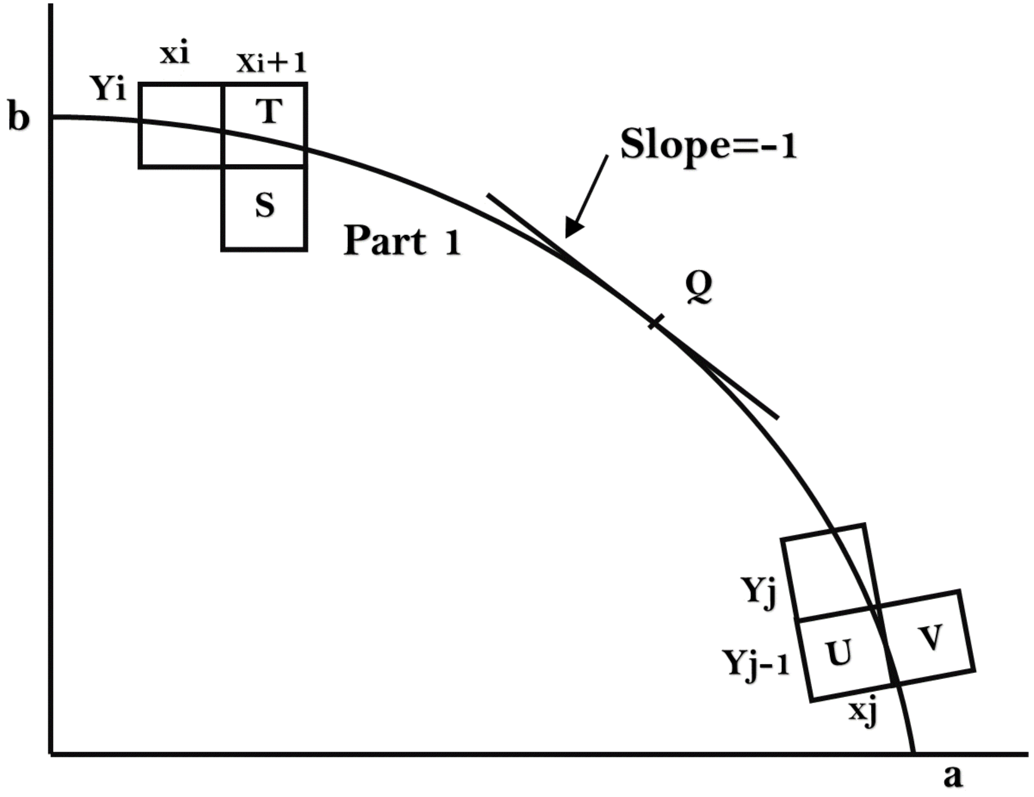
Assignment: Circle Drawing Algorithm

- 实验编号：3
- 实验名称：圆绘制算法
- 实验内容
  - Bresenham圆绘制算法



Extra Credit

- Could you draw ellipse using Midpoint/Bresenham algorithm?



Reference

- https://en.wikipedia.org/wiki/Midpoint_circle_algorithm
- https://www.geeksforgeeks.org/midpoint-ellipse-drawing-algorithm/
- http://members.chello.at/~easyfilter/bresenham.html