Scan-Conversion

Solved Problems: 01

The endpoints of a given line are (0, 0) and (6, 18). Compute each value of y as x steps from 0 to 6 and plot the results.

SOLUTION

An equation for the line was not given. Therefore, the equation of the line must be found. The equation of the line ($y = mx + b$) is found as follows. First the slope is found:

$$m = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{18 - 0}{6 - 0} = \frac{18}{6} = 3$$

Next, the y intercept b is found by plugging y_1 and x_1 into the equation $y = 3x + b$: $0 = 3(0) + b$. Therefore, $b = 0$, so the equation for the line is $y = 3x$ (see Fig. 3-32).

Solved Problems: Problem 02

What steps are required to plot a line whose slope is between 0° and 45° using the slope–intercept equation?

SOLUTION

1. Compute dx: $dx = x_2 - x_1$.
2. Compute dy: $dy = y_2 - y_1$.
3. Compute m: $m = dy/dx$.
4. Compute b: $b = y_1 - m \times x_1$.
5. Set (x, y) equal to the lower left-hand endpoint and set x_{end} equal to the largest value of x. If $dx < 0$, then $x = x_2$, $y = y_2$, and $x_{end} = x_1$. If $dx > 0$, then $x = x_1$, $y = y_1$, and $x_{end} = x_2$.
6. Test to determine whether the entire line has been drawn. If $x > x_{end}$, stop.
7. Plot a point at the current (x, y) coordinates.
8. Increment x: $x = x + 1$.
9. Compute the next value of y from the equation $y = mx + b$.
10. Go to step 6.

Solved Problems: Problem 03

Use pseudo-code to describe the steps that are required to plot a line whose slope is between 45° and −45° (i.e., $|m| > 1$) using the slope–intercept equation.

SOLUTION

Presume $y_1 < y_2$ for the two endpoints (x_1, y_1) and (x_2, y_2):

```
int x = x1, y = y1;
float xf, m = (y2 − y1)/(x2 − x1), b = y1 − mx1;
setPixel(x, y);
while (y < y2) {
    y++;
    xf = (y − b)/m;
    x = Floor(xf + 0.5);
    setPixel(x, y);
}
```

Problem 04:

Use pseudo-code to describe the DDA algorithm for scan-converting a line whose slope is between −45° and 45° (i.e., $|m| < 1$).

SOLUTION

Presume $x_1 < x_2$ for the two endpoints (x_1, y_1) and (x_2, y_2):

```
int x = x1, y;
float yf = y1, m = (y2 − y1)/(x2 − x1);
while (x <= x2) {
    y = Floor(yf + 0.5);
    setPixel(x, y);
    x++;
    yf = yf + m;
}
```

Solved Problems: Problem 05

Use pseudo-code to describe the DDA algorithm for scan-converting a line whose slope is between 45° and −45° (i.e., $|m| > 1$).

SOLUTION

Presume $y_1 < y_2$ for the two endpoints (x_1, y_1) and (x_2, y_2):

```
int x, y = y1;
float xf = x1, minv = (x2 - x1)/(y2 - y1);
while (y <= y2) {
   x = Floor(xf + 0.5);
   setPixel(x, y);
   xf = xf + minv;
   y++;
}
```

Solved Problems: Problem 06

What steps are required to plot a line whose slope is between 0° and 45° using Bresenham's method?

SOLUTION

1. Compute the initial values:

$$dx = x_2 - x_1 \qquad Inc_2 = 2(dy - dx)$$
$$dy = y_2 - y_1 \qquad d = Inc_1 - dx$$
$$Inc_1 = 2dy$$

2. Set (x, y) equal to the lower left-hand endpoint and x_{end} equal to the largest value of x. If $dx < 0$, then $x = x_2, y = y_2, x_{end} = x_1$. If $dx > 0$, then $x = x_1, y = y_1, x_{end} = x_2$.
3. Plot a point at the current (x, y) coordinates.
4. Test to see whether the entire line has been drawn. If $x = x_{end}$, stop.
5. Compute the location of the next pixel. If $d < 0$, then $d = d + Inc_1$. If $d \geq 0$, then $d = d + Inc_2$, and then $y = y + 1$.
6. Increment x: $x = x + 1$.
7. Plot a point at the current (x, y) coordinates.
8. Go to step 4.

Solved Problems: Problem 07

Indicate which raster locations would be chosen by Bresenham's algorithm when scan-converting a line from pixel coordinate (1, 1) to pixel coordinate (8, 5).

SOLUTION

First, the starting values must be found. In this case

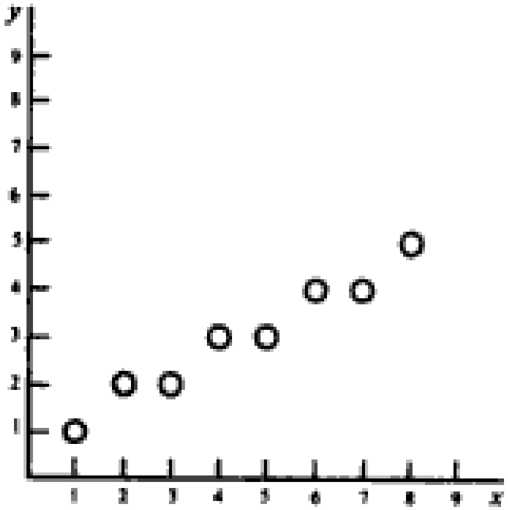
$$dx = x_2 - x_1 = 8 - 1 = 7 \qquad dy = y_2 - y_1 = 5 - 1 = 4$$

Therefore:

$$Inc_1 = 2dy = 2 \times 4 = 8$$
$$Inc_2 = 2(dy - dx) = 2 \times (4 - 7) = -6$$
$$d = Inc_1 - dx = 8 - 7 = 1$$

The following table indicates the values computed by the algorithm (see also Fig. 3-33).

d	x	y
1	1	1
$1 + Inc_2 = -5$	2	2
$-5 + Inc_1 = 3$	3	2
$3 + Inc_2 = -3$	4	3
$-3 + Inc_1 = 5$	5	3
$5 + Inc_2 = -1$	6	4
$-1 + Inc_1 = 7$	7	4
$7 + Inc_2 = 1$	8	5



Solved Problems: Problem 08

Modify the description of Bresenham's line algorithm in the text to set all pixels from inside the loop structure.

SOLUTION 1

```
int x = x'1, y = y'1;
int dx = x'2 − x'1, dy = y'2 − y'1, dT = 2(dy − dx), dS = 2dy;
int d = 2dy − dx;
while (x <= x'2) {
  setPixel(x, y);
  x++;
  if (d < 0)
    d = d + dS;

  else {
    y++;
    d = d + dT;
  }
}
```

SOLUTION 2

```
int x = x'1 − 1, y = y'1;
int dx = x'2 − x'1, dy = y'2 − y'1, dT = 2(dy − dx), dS = 2dy;
int d = −dx;
while (x < x'2) {
  x++;
  if (d < 0)
    d = d + dS;
  else {
    y++;
    d = d + dT;
  }
  setPixel(x, y);
}
```

Solved Problems: Problem 09

What steps are required to generate a circle using the polynomial method?

SOLUTION

1. Set the initial variables: r = circle radius; (h, k) = coordinates of the circle center; $x = 0$; i = step size; $x_{\text{end}} = r/\sqrt{2}$.
2. Test to determine whether the entire circle has been scan-converted. If $x > x_{\text{end}}$, stop.
3. Compute the value of the y coordinate, where $y = \sqrt{r^2 - x^2}$.
4. Plot the eight points, found by symmetry with respect to the center (h, k), at the current (x, y) coordinates:

$$
\begin{array}{ll}
\text{Plot}(x + h, y + k) & \text{Plot}(-x + h, -y + k) \\
\text{Plot}(y + h, x + k) & \text{Plot}(-y + h, -x + k) \\
\text{Plot}(-y + h, x + k) & \text{Plot}(y + h, -x + k) \\
\text{Plot}(-x + h, y + k) & \text{Plot}(x + h, -y + k)
\end{array}
$$

5. Increment x: $x = x + i$.
6. Go to step 2.

Solved Problems: Problem 10

What steps are required to scan-convert a circle using the trigonometric method?

SOLUTION

1. Set the initial variables: r = circle radius; (h, k) = coordinates of the circle center; i = step size; $\theta_{end} = \pi/4$ radians = 45°; $\theta = 0$.
2. Test to determine whether the entire circle has been scan-converted. If $\theta > \theta_{end}$, stop.
3. Compute the value of the x and y coordinates:

$$x = r\cos(\theta) \qquad y = r\sin(\theta)$$

4. Plot the eight points, found by symmetry with respect to the center (h, k), at the current (x, y) coordinates:

$$\begin{array}{ll} \text{Plot}(x+h, y+k) & \text{Plot}(-x+h, -y+k) \\ \text{Plot}(y+h, x+k) & \text{Plot}(-y+h, -x+k) \\ \text{Plot}(-y+h, x+k) & \text{Plot}(y+h, -x+k) \\ \text{Plot}(-x+h, y+k) & \text{Plot}(x+h, -y+k) \end{array}$$

5. Increment θ: $\theta = \theta + i$.
6. Go to step 2.

Solved Problems: Problem 11

What steps are required to scan-convert a circle using Bresenham's algorithm?

SOLUTION

1. Set the initial values of the variables: (h, k) = coordinates of circle center; $x = 0$; y = circle radius r; and $d = 3 - 2r$.
2. Test to determine whether the entire circle has been scan-converted. If $x > y$, stop.
3. Plot the eight points, found by symmetry with respect to the center (h, k), at the current (x, y) coordinates:

$\text{Plot}(x + h, y + k)$	$\text{Plot}(-x + h, -y + k)$
$\text{Plot}(y + h, x + k)$	$\text{Plot}(-y + h, -x + k)$
$\text{Plot}(-y + h, x + k)$	$\text{Plot}(y + h, -x + k)$
$\text{Plot}(-x + h, y + k)$	$\text{Plot}(x + h, -y + k)$

4. Compute the location of the next pixel. If $d < 0$, then $d = d + 4x + 6$ and $x = x + 1$. If $d \geq 0$, then $d = d + 4(x - y) + 10$, $x = x + 1$, and $y = y - 1$.
5. Go to step 2.

Solved Problems: Problem 12

In the derivation of Bresenham's circle algorithm we have used a decision variable $d_i = D(\mathrm{T}) + D(\mathrm{S})$ to help choose between pixels S and T. However, function D as defined in the text is not a true measure of the distance from the center of a pixel to the true circle. Show that when $d_i = 0$ the two pixels S and T are not really equally far away from the true circle.

SOLUTION

Let $d\mathrm{S}$ be the actual distance from S to the true circle and $d\mathrm{T}$ be the actual distance from T to the true circle (see Fig. 3-35). Also substitute x for $x_i + 1$ and y for y_i in the formula for d_i to make the following proof easier to read:

$$d_i = 2x^2 + y^2 + (y-1)^2 - 2r^2 = 0$$

Since $(r + d\mathrm{T})^2 = x^2 + y^2$ and $(r - d\mathrm{S})^2 = x^2 + (y-1)^2$ we have

$$2r\,d\mathrm{T} + d\mathrm{T}^2 = x^2 + y^2 - r^2 \qquad \text{and} \qquad -2r\,d\mathrm{S} + d\mathrm{S}^2 = x^2 + (y-1)^2 - r^2.$$

Hence

$$2r\,d\mathrm{T} + d\mathrm{T}^2 - 2r\,d\mathrm{S} + d\mathrm{S}^2 = 0$$

$$d\mathrm{T}(2r + d\mathrm{T}) = d\mathrm{S}(2r - d\mathrm{S})$$

Since $d\mathrm{T}/d\mathrm{S} = (2r - d\mathrm{S})/(2r + d\mathrm{T}) < 1$, we have $d\mathrm{T} < d\mathrm{S}$. This means that, when $d_i = 0$, pixel T is actually closer to the true circle than pixel S.

Solved Problems: Problem 13

Write a description of the midpoint circle algorithm in which decision parameter p is updated using x_{i+1} and y_{i+1} instead of x_i and y_i.

SOLUTION

```
int x = 0, y = r, p = 1 - r;
while (x <= y) {
    setPixel(x, y);
    x++;
    if (p < 0)
        p = p + 2x + 1;
    else {
        y--;
        p = p + 2(x - y) + 1;
    }
}
```

Solved Problems: Problem 14

What steps are required to generate an ellipse using the polynomial method?

SOLUTION

1. Set the initial variables: a = length of major axis; b = length of minor axis; (h, k) = coordinates of ellipse center; $x = 0$; i = step size; $x_{\text{end}} = a$.
2. Test to determine whether the entire ellipse has been scan-converted. If $x > x_{\text{end}}$, stop.
3. Compute the value of the y coordinate:

$$y = b\sqrt{1 - \frac{x^2}{a^2}}$$

4. Plot the four points, found by symmetry, at the current (x, y) coordinates:

$$\text{Plot}(x + h, y + k) \qquad \text{Plot}(-x + h, -y + k)$$
$$\text{Plot}(-x + h, y + k) \qquad \text{Plot}(x + h, -y + k)$$

5. Increment x: $x = x + i$.
6. Go to step 2.

Solved Problems: Problem 15

What steps are required to scan-convert an ellipse using the trigonometric method?

SOLUTION

1. Set the initial variables: a = length of major axis; b = length of minor axis; (h, k) = coordinates of ellipse center; i = counter step size; $\theta_{end} = \pi/2$; $\theta = 0$.
2. Test to determine whether the entire ellipse has been scan-converted. If $\theta > \theta_{end}$, stop.
3. Compute the values of the x and y coordinates:

$$x = a\cos(\theta) \qquad y = b\sin(\theta)$$

4. Plot the four points, found by symmetry, at the current (x, y) coordinates:

$$\text{Plot}(x + h, y + k) \qquad \text{Plot}(-x + h, -y + k)$$
$$\text{Plot}(-x + h, y + k) \qquad \text{Plot}(x + h, -y + k)$$

5. Increment θ: $\theta = \theta + i$.
6. Go to step 2.

Solved Problems: Problem 16

What steps are required to scan-convert an arc using the trigonometric method?

SOLUTION

1. Set the initial variables: a = major axis; b = minor axis; (h, k) = coordinates of arc center; i = step size; θ = starting angle; θ_1 = ending angle.
2. Test to determine whether the entire arc has been scan-converted. If $\theta > \theta_1$, stop.
3. Compute the values of the x and y coordinates:

$$x = a\cos(\theta) + h \qquad y = a\sin(\theta) + k$$

4. Plot the points at the current (x, y) coordinates: Plot(x, y).
5. Increment θ: $\theta = \theta + i$.
6. Go to step 2.

(Note: for the arc of a circle $a = b$ = circle radius r.)

Solved Problems: Problem 17

What steps are required to generate an arc of a circle using the polynomial method?

SOLUTION

1. Set the initial variables: r = radius; (h, k) = coordinates of arc center; x = x coordinate of start of arc; x_1 = x coordinate of end of arc; i = counter step size.
2. Test to determine whether the entire arc has been scan-converted. If $x > x_1$, stop.
3. Compute the value of the y coordinate:

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

4. Plot at the current (x, y) coordinates:

$$\text{Plot}(x + h, y + k)$$

5. Increment x: $x = x + i$.
6. Go to step 2.

Solved Problems: Problem 18

Write a pseudo-code procedure to implement the boundary-fill algorithm in the text in its basic form, using the 4-connected definition for region pixels.

SOLUTION

```
BoundaryFill (int x, y, fill_color, boundary_color)
{
  int color;
  getPixel(x, y, color);
  if (color != boundary_color && color != fill_color) {
    setPixel(x, y, fill_color);
    BoundaryFill(x + 1, y, fill_color, boundary_color);
    BoundaryFill(x, y + 1, fill_color, boundary_color);
    BoundaryFill(x – 1, y, fill_color, boundary_color);
    BoundaryFill(x, y – 1, fill_color, boundary_color);
  }
}
```

Solved Problems: Problem 19

Write a pseudo-code procedure for generating the Koch curve K_n (after the one in the text for generating C_n).

SOLUTION

```
Koch-curve (float x, y, len, alpha; int n)
{
   if (n > 0) {
      len = len/3;
      Koch-curve(x, y, len, alpha, n - 1);
      x = x + len*cos(alpha);
      y = y + len*sin(alpha);
      Koch-curve(x, y, len, alpha - 60, n - 1);
      x = x + len*cos(alpha - 60);
      y = y + len*sin(alpha - 60);
      Koch-curve(x, y, len, alpha + 60, n - 1);
      x = x + len*cos(alpha + 60);
      y = y + len*sin(alpha + 60);
      Koch-curve(x, y, len, alpha, n - 1);
   } else
      line(x, y, x + len*cos(alpha), y + len*sin(alpha));
}
```

Solved Problems: Problem 20

Presume that the following statement produces a filled triangle with vertices at (x_1, y_1), (x_2, y_2), and (x_3, y_3):

triangle$(x_1, y_1, x_2, y_2, x_3, y_3)$

Write a pseudo-code procedure for generating the Sierpinski gasket S_n (after the procedure in the text for generating C_n).

SOLUTION

```
S-Gasket (float x1, y1, x2, y2, x3, y3; int n)
{
  float x12, y12, x13, y13, x23, y23;
  if (n > 0) {
    x12 = (x1 + x2)/2;
    y12 = (y1 + y2)/2;
    x13 = (x1 + x3)/2;
    y13 = (y1 + y3)/2;
    x23 = (x2 + x3)/2;
    y23 = (y2 + y3)/2;
    S-Gasket(x1, y1, x12, y12, x13, y13, n - 1);
    S-Gasket(x12, y12, x2, y2, x23, y23, n - 1);
    S-Gasket(x13, y13, x23, y23, x3, y3, n - 1);
  } else
    triangle(x1, y1, x2, y2, x3, y3);
}
```

Thanks