

Scan Conversion

נניח שאנחנו רוצים לצייר על המסך קו מ (x_0, y_0) ל (x_1, y_1) - איך אנחנו מציירים אותו על המסך? Scan Conversion זה המרה של הקו לייצוג שלו ברaster display. איך מייצגים קו? יש כמה דרכים:

- משוואה מרומזת: $f(x, y, \dots) = 0$. במקרה של קו ישר - $\alpha x + \beta y + \gamma = 0$.
 - נוסחה מפורשת: $y = f(x)$. במקרה של קו ישר - $y = mx + B$ (כאשר $m = \frac{y_1 - y_0}{x_1 - x_0}$)
 - הצגה פרמטרית: $x = f_x(t)$ $y = f_y(t)$ במקרה של קו ישר $P = P_0 + (P_1 - P_0)t$ עבור $0 \leq t \leq 1$.
- במקרה של קו ישר, אם נניח משתמשים בנוסחה המפורשת $y = mx + B$ בהנחה $|m| \leq 1$ (אם זה הפוך נהפוך בין x ל y):

```
for x = x0 to x1
  y = mx + b
  PlotPixel(x, round(y))
end
```

זה עובד - אבל הרבה עבודה חישובית. כדי לייעל את זה אפשר לעבוד בצורה אינקרמנטלית:

$$y_{i+1} = mx_{i+1} + B = m(x_i + \Delta x) + B = y_i + m\Delta x$$

ואם $\Delta x = 1$ אז $y_{i+1} = y_i + m$. כלומר בכל איטרציה של הלולאה מקדמים את x ב 1 ואת y ב m :

```
Line(x0, y0, x1, y1)
begin
  float dx, dy, x, y, slope
  dx := x1 - x0
  dy := y1 - y0
  slope :=  $\frac{dx}{dy}$ 
  y := y0
  for x := x0 to x1 do
    begin
      PlotPixel(x, Round(y))
      y := y + slope
    end
  end
end
```

מקרים מיוחדים:

- $m = \pm 1$ - קו אלכסוני
- $m = 0$ - קו אופקי
- $m = \infty$ - קו אנכי

אם $x_0 > x_1$ עבור $|m| \leq 1$ או $y_0 > y_1$ עבור $|m| \geq 1$ - נחליף בין (x_0, y_0) ל (x_1, y_1) .

- בעיות:
- מוסיפים כל הזמן מספרים ל float - יש כאן שגיאה מצטברת
 - פעולות על float הן יקרות
 - פעולות עיגול הן יקרות

כדי לפתור את הבעיות האלה נרצה שיטה אחרת:

Midpoint (~Bresenham) Line Drawing

- בהנחה ש: $x_0 < x_1$
- $y_0 < y_1$
- $0 < \text{slope} < 1$

עבור (x_p, y_p) , הפיקסל הבא יכול להיות או $E = (x_p + 1, y_p)$ או $NE = (x_p + 1, y_p + 1)$. איך נדע איזה? לפי $M = (x_p + 1, y_p + \frac{1}{2})$ כאשר $\text{sign}(M - Q)$ משוואת הישר המפורשת היא $y = \frac{\Delta y}{\Delta x}x + B$. נעבור לצורה המרומזת $f(x, y) = ax + by + c = 0$. אם נציב:

$$f(x, y) = \Delta y \cdot x - \Delta x \cdot y + B \cdot \Delta x = 0$$

ההחלטה לפי:

$$d = f(M) = f\left(x_p + 1, y_p + \frac{1}{2}\right) = \Delta y(x_p + 1) - \Delta x\left(y_p + \frac{1}{2}\right) + B\Delta x$$

אם $d > 0$ נלך NE, אם $d \leq 0$ נלך E. אבל אנחנו לא רוצים לעשות את החישוב היקר הזה כל פעם - אנחנו רוצים גם כאן שיטה אינקרמנטלית! נשים לב - מה קורה עבור $x_p + 2$? אם עבור $x_p + 1$ יצא E, אז:

$$M_{\text{new}} = \left(x_p + 2, y_p + \frac{1}{2}\right)$$

$$d_{\text{new}} = f\left(x_p + 2, y_p + \frac{1}{2}\right) = \Delta y(x_p + 2) - \Delta x\left(y_p + \frac{1}{2}\right) + B \cdot \Delta x$$

$$d_{\text{old}} = f\left(x_p + 1, y_p + \frac{1}{2}\right) = \Delta y(x_p + 1) - \Delta x\left(y_p + \frac{1}{2}\right) + B \cdot \Delta x$$

$$d_{\text{new}} = -d_{\text{old}} + \Delta y$$

$$\boxed{d_{\text{new}} = d_{\text{old}} + \Delta E}$$

ואם עבור $x_p + 1$ יצא NE:

$$M_{\text{new}} = \left(x_p + 2, y_p + \frac{3}{2}\right)$$

$$d_{\text{new}} = f\left(x_p + 2, y_p + \frac{3}{2}\right) = \Delta y(x_p + 2) - \Delta x\left(y_p + \frac{3}{2}\right) + B \cdot \Delta x$$

$$d_{\text{old}} = f\left(x_p + 1, y_p + \frac{3}{2}\right) = \Delta y(x_p + 1) - \Delta x\left(y_p + \frac{3}{2}\right) + B \cdot \Delta x$$

$$d_{\text{new}} = -d_{\text{old}} + \Delta y$$

$$\boxed{d_{\text{new}} = d_{\text{old}} + \Delta_{NE}}$$

ויש לנו כאן רק חיבור שלמים!

האלגוריתם

הנקודה הראשונה (x_0, y_0) - נקודת האמצע הראשונה $(x_0 + 1, y_0 + \frac{1}{2})$ -

$$d_{\text{start}} = f\left(x_0 + 1, y_0 + \frac{1}{2}\right) = \Delta y (x_0 + 1) - \Delta x \left(y_0 + \frac{1}{2}\right) + B \cdot \Delta x = \dots$$

$$\dots = \Delta y \cdot x_0 - \Delta x \cdot y_0 + B \cdot \Delta x + \Delta y - \Delta x \frac{b}{2} = \dots$$

$$\dots = f(x_0, y_0) + \Delta y - \frac{\Delta x}{2} = \Delta y - \frac{\Delta x}{2}$$

כדי להימנע משברים, נכפיל ב-2. כלומר:

$$f(x, y) = 2(\Delta y \cdot x - \Delta x \cdot y + B \cdot \Delta x) = 0$$

$$d_{\text{start}} = 2\Delta y - \Delta x$$

$$\Delta_E = 2\Delta y \quad \Delta_{NE} = 2(\Delta y - \Delta x)$$

ציור מעגלים

המשוואה המרומזת של מעגל (שמרכזו, לשם הפשטות, בראשית הצירים) היא $x^2 + y^2 - R^2 = 0$. הנוסחה המפורשת היא $y = \pm\sqrt{R^2 - x^2}$. ובהצגה פרמטרית $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} R \cos(t) \\ R \sin(t) \end{bmatrix}$ בשביל לצייר מעגל בצורה הכי פשוטה:

```
for x = -R to R
  y = sqrt(R^2 - x^2)
  PlotPixel(x, round(y))
  PlotPixel(x, -round(y))
end
```

כמובן - מדובר באלגוריתם מאוד יקר, שצריך כל פעם להוציא שורש. שיפור אחד הוא להשתמש בסימטריה, לחשב שמינית מעגל ולצייר לכל נקודה 8 פעמים (עם היפוך $x \leftrightarrow y$ ועם היפוכי סימנים). אבל זה עדיין יקר, ונרצה גם כאן להשתמש בשיטת MidPoint. אם מתחילים מ $(x_0, y_0) = (0, R)$ - כלומר השמינית הראשונה ימינה מהנקודה העליונה של המעגל - אז אפשר ללכת או E או SE ושוב נשתמש בסימן של $d(x, y)$ כדי להחליט, כאשר:

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

כלומר: • נתחיל עם $(x_0 = 0, y_0 = R)$ ו

$$d_{\text{start}} = d\left(x_0 + 1, y_0 - \frac{1}{2}\right) = d\left(1, R - \frac{1}{2}\right) = \frac{5}{4} - R$$

• אם $d < 0$ נזוז E:

$$\Delta_E = d\left(x_0 + 2, y_0 - \frac{1}{2}\right) - d\left(x_0 + 1, y_0 - \frac{1}{2}\right) = 2x_0 + 3$$

• אם $d > 0$ נזוז SE:

$$\Delta_{SE} = d\left(x_0 + 2, y_0 - \frac{3}{2}\right) - d\left(x_0 + 1, y_0 - \frac{1}{2}\right) = 2(x_0 - y_0) + 5$$

- נשים לב ש Δ_E ו Δ_{SE} כבר לא קבועים!
- בגלל d גדל בערכים שלמים, ובגלל שאיכפת לנו רק מהסימן שלו, אפשר להשתמש ב $d_{\text{start}} = 1 - R$ ולקבל אלגוריתם שעובד עם שלמים בלי להשפיע על התוצאה.

מילוי פוליגונים

בהינתן פוליגון, אנחנו רוצים למלא אותו בצבע מסוים.

הנחות: • הפוליגון פשוט:

- אין הצטלבויות

- אין בו חורים

איך צובעים אותו? יש שתי שיטות:

Flood Fill

נבחר נקודה בתוך הפוליגון, ונצבע אותה. נלך רקורסיבית לארבעת השכנים שלה ונבדוק כל אחד אם צריך לצבוע אותו (כלומר אם הוא לא על הגבול ולא צבוע). אם כן - נצבע אותו, וניכנס רקורסיבית גם לשכנים שלו. השיטה הזאת מאוד מאוד יקרה - יש כאן רקורסיה עם הסתעפות 4 (אי אפשר להפוך אותה לרקורסיית קריאת זנב) וצריך לבדוק לכל נקודה ונקודה אם היא בתוך הפוליגון.

Scan Conversion

תזכורת: ניתן לבדוק אם נקודה נמצאת בתוך פוליגון באמצעות ספירת נקודות החיתוך של הפוליגון עם קרן כלשהי מהנקודה. אם הוא אי-זוגי - הנקודה בתוך הפוליגון.

```
for  $j := 0$  to ScreenYMax do  
   $I :=$ points of intersection on edges from  $P$  with line  $y = j$   
  Sort  $I$  in increasing  $x$  order and fill with color  $c$  alternating segments  
end
```

בעיה - מה עושים כשהקרן נחתכת עם שני קווים? מתייחסים לזה כאחד או כ2? המצב הזה יכול להיות או כניסה/יציאה או כניסה+יציאה. הפתרון הוא להחליט שכל edge הוא קטע פתוח-סגור כאשר לוקחים את ה y המינימלי שלו ולא את ה y המקסימלי שלו.

אפשר לשפר את האלגוריתם. לא צריך כל פעם למצוא את החיתוך עם כל הקשתות - אפשר להחזיק את כל הקשתות ברשימה מקושרת (A) ממוינת לפי y_{\min} של הקשת, וכשסורקים פס רוחב - לא צריך להתייחס לקשתות שה y_{\min} שלהן גדול מדי (כלומר מתחילים מאמצע הרשימה). וכשעוברים y_{\max} של קשת מוציאים אותה מהרשימה (מוציאים ישר כשמגיעים ל y_{\max} - כי אנחנו לא רוצים אותו)

```
ScanConvert( $P, c$ )  
Sort all edges  $E = \{E_j\}$  in increasing  $\text{MinY}(E_j)$  order  
 $A := \emptyset$   
for  $k := 0$  to ScreenYMax do  
  for each  $E_j \in E$   
    if  $\text{MinY}(E_j) \leq k$   
       $A := A \cup \{E_j\}$   
       $E := E - E_j$   
    for each  $E_j \in A$   
      if  $\text{MaxY}(E_j) \leq k$   
         $A := A - E_j$   
   $I :=$ Points of intersection of members from  $A$  with line  $y = k$   
  Sort  $I$  in increasing  $x$  order and fill with color  $c$  alternating segments  
end
```