CS4533 Lecture 2
Slides/Notes

Scan Conversion for 2D Line Segments; HW1 Discussion; Introduction to OpenGL (Notes, Ch 1)

By Prof. Yi-Jen Chiang

CSE Dept., Tandon School of Engineering

New York University

1

✗ Scan Conversion: Convert geometric entities to pixel assignment.
(Rasterization)

Scan Conversion for Line Segments

DDA Algorithm (Digital Differential Analyzer)

Given: line segment endpoints (X_1, Y_1) (X_2, Y_2)
They are in frame buffer grid positions ⟹ X_1, Y_1, X_2, Y_2 are all integers.
(pixel centers)

A line: differential eq:
$$\frac{dy}{dX} = m \longrightarrow slope.$$
Solve it numerically
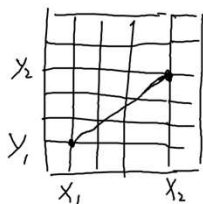
slope: $m = \dfrac{Y_2 - Y_1}{X_2 - X_1} = \dfrac{\triangle Y}{\triangle X}$ $\triangle Y = Y_2 - Y_1, \triangle X = X_2 - X_1$

Assume: $m \in [0, 1]$

2

<u>Alg</u> : move from X_1 to X_2. $\begin{bmatrix} \text{increase } X \text{ by } 1 \\ s \quad y \text{ by } dy = m \cdot dX \\ = m. \end{bmatrix}$ repeat.

Each X is an integer.

  y is <u>NOT</u> ∵ (∵ m is a floating point number)

  ⟹ round y (choose the best y that's integer)

Other slopes : by symmetry

$m > 1$ *
$m \in (0,1]$ ✓
$m \in (-1, 0)$ ✓
$m < -1$ *
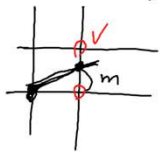
<u>Reason for $m \in (0,1]$</u> :

For $m > 1$:
$\begin{bmatrix} \text{increase } y \text{ by } 1 \\ \text{choose the best } x \text{ that's integer} \end{bmatrix}$

$m > 1$

$m \in (0,1]$.
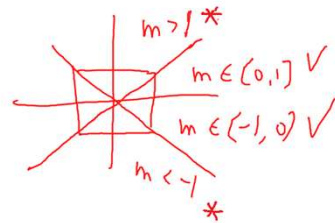
3

<u>Bresenham's Alg.</u> : <u>Goal</u> : Avoid all floating-pt computations

  <u>Assume</u> : slope $m \in (0,1]$ ↘ $\begin{bmatrix} \text{increase } X \text{ by } 1 \\ \text{choose the best } y \end{bmatrix}$ in each iteration.

<u>Input</u> : 2 endpts $(X_1, Y_1)(X_2, Y_2)$

  <u>Def</u> : $\Delta y = Y_2 - Y_1 > 0$. $\Delta X = X_2 - X_1 > 0$. (Integers)

(I)
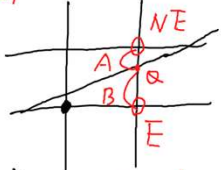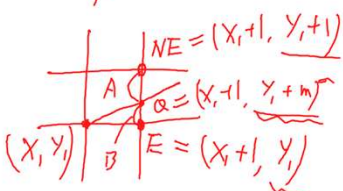

(From picture) → $A < B$. choose NE

$A - B < 0$ "

$\Delta X (A-B) < 0$ "

$\underbrace{}_{\le D}$ (Note: $\Delta X > 0$)

<u>Define</u> : $D = \Delta X (A - B)$

$D < 0$ choose NE

$D \ge 0$: E .

(II) Compute D_{start} (first D value)

$NE = (X_1 + 1, Y_1 + 1)$

A $Q = (X_1 + 1, Y_1 + m)$

(X_1, Y_1) B $E = (X_1 + 1, Y_1)$

$A = 1 - m$
$B = m$.
$m = \dfrac{\Delta y}{\Delta X}$

$D_{start} = \Delta X (A - B) = \Delta X (1 - m - m) = \Delta X (1 - 2m)$

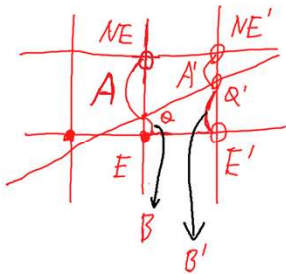$= \Delta X \left(1 - 2 \cdot \dfrac{\Delta y}{\Delta X}\right) = \Delta X - 2 \Delta y$.

$\left(\begin{array}{l} \Delta X, \Delta y : \text{integers} \\ \Rightarrow D_{start} : \text{integer} \end{array}\right)$

4

(III) Incremental Computation : Compute D_{new} from D_{old}.

(1) If E is chosen $(D_{old} \geq 0)$

$\underline{Y(Q') - Y(Q) = m}$

$A' - A = -m \quad —①$
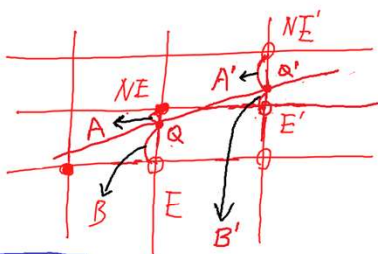
$B' - B = m \quad —②$

①-②:

$(A' - A) - (B' - B) = -2m$

$(A' - B') = (A - B) - 2m$ ← $\boxed{* \; \Delta X \text{ on both sides}}$

$\underbrace{\Delta X (A' - B')}_{D_{new}} = \underbrace{\Delta X (A - B)}_{D_{old}} \underbrace{- 2m \cdot \Delta X}_{-2 \cdot \frac{\Delta Y}{\Delta X} \cdot \Delta X}$

$\boxed{D_{new} = D_{old} - 2 \Delta Y \quad \text{if } D_{old} \geq 0}$

---

(2) If NE is chosen $(D_{old} < 0)$

$Y(Q') - Y(Q) = m.$

$A' - A = 1 - m \quad —①$

$B' - B = m - 1 \quad —②$

①-②

$(A' - A) - (B' - B) = 2(1 - m)$

$(A' - B') = (A - B) + 2(1 - m)$ ← $\boxed{* \; \Delta X \text{ on both sides}}$

$\underbrace{\Delta X (A' - B')}_{D_{new}} = \underbrace{\Delta X (A - B)}_{D_{old}} + \underbrace{2 \cdot \Delta X (1 - \frac{\Delta Y}{\Delta X})}_{= 2\Delta X - 2\Delta Y} \quad \left(m = \frac{\Delta Y}{\Delta X} \right)$

$\boxed{D_{new} = D_{old} + 2 \Delta X - 2 \Delta Y \quad \text{if } D_{old} < 0.}$
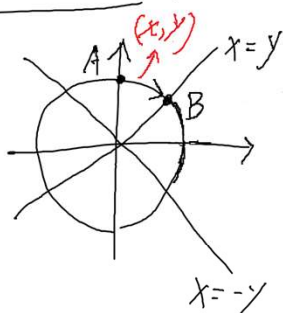
Summary:

$D_{start} = \Delta X - 2 \Delta Y$

$D_{new} = D_{old} + \begin{cases} -2\Delta Y & \text{if } D_{old} \geq 0 \\ 2(\Delta X - \Delta Y) & \text{else.} \end{cases}$

HW1 Part (a)
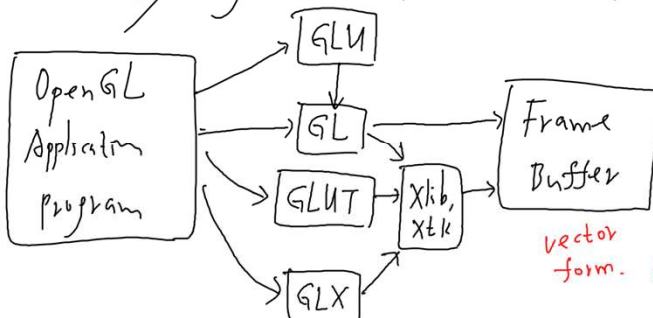

The points in the 8 symmetric regions.

(x, y) (y, x)

$(-x, y)$ $(y, -x)$

$(x, -y)$ $(-y, x)$

$(-x, -y)$ $(-y, -x)$

Introduction to OpenGL

* Library organization (for X window system)

GLU: graphics utility library.

GL: Open GL Library (indept. of window system)

GLUT: graphics utility toolkit (interface with windows system X window, MS windows)

eg. glVertex 3f (1.2, 3.6, 2.4)
glVertex 3f v (ptr)

* OpenGL Command Syntax:

glVertex 3-f [v] (   )

components
2: (x, y)   4: (x, y, z, w)
3: (x, y, z)

vector form.

Data Type
i: integer
f: float ..

A 0 [0 0 0] A(1)
1 [△ × □]

float A[10][3]   glVertex3fv(A[1])

A
| X   Y   Z
| 0   f₁   f₂   f₃
| 1   1.2   3.6   2.4   A[3]   A
| 2
| ⋮
| 9

[idle ( )   → event loop
[⋮   idle
[display ( )   ↓
}   display ( )
                    ← ── ── ──

[idle ( )   event loop ←
[⋮   ↓   ↑
[glutPostRedisplay ( );   idle   display
}
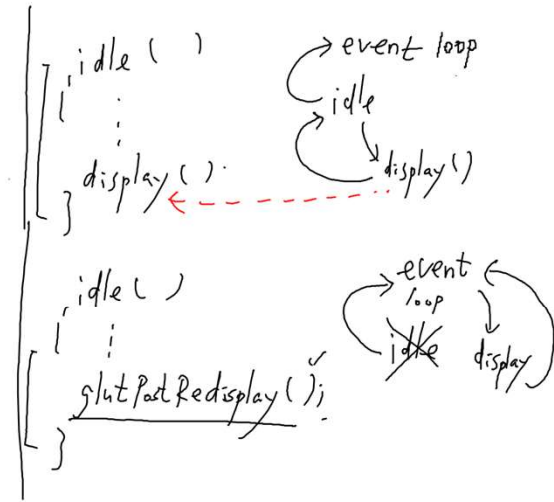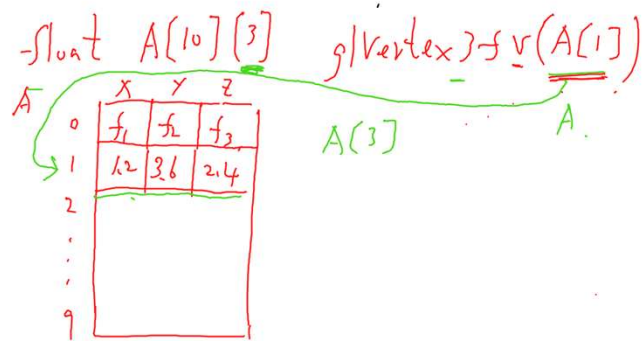
* Discussed the handout sample code "example.cpp".