CS6533/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

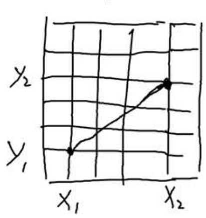
DDA Algorithm (Digital Differential Analyzer)

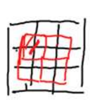
A line: differential eq.: $\frac{dy}{dx} = m \longrightarrow$ slope.

Solve it numerically

Given: line segment endpoints (x_1, y_1) (x_2, y_2)

They are in frame buffer grid positions $\Rightarrow$ x_1, y_1, x_2, y_2 are all integers.
(pixel centers)





Slope: $m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{\Delta y}{\Delta x}$, $\Delta y = y_2 - y_1$, $\Delta x = x_2 - x_1$

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

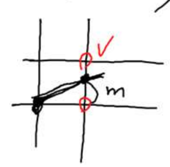
2

Alg: move from X_1 to X_2. $\left[\begin{array}{l} \text{increase } x \text{ by } 1 \\ \text{" } y \text{ by } dy = m \cdot dX \\ \qquad\quad = m. \end{array}\right]$ repeat.

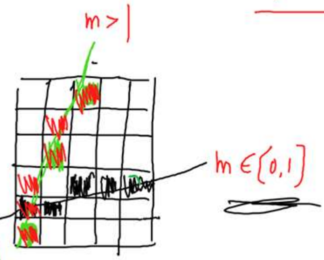
Each x is an integer.

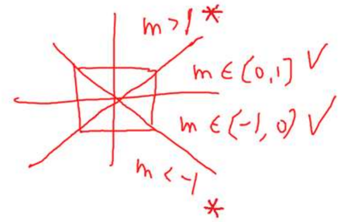
y is NOT " ($\because$ m is a floating point number)

$\Rightarrow$ round y (choose the best y that's integer)



Other slopes: by symmetry





Reason for $m \in [0,1]$:

For $m > 1$:
[increase y by 1
[choose the best x that's integer.

3

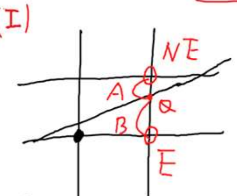
Bresenham's Alg.: Goal: Avoid all floating-pt computations

Assume: slope $m \in [0,1]$ $\rightarrow$ $\left[\begin{array}{l} \text{increase } X \text{ by } 1 \\ \text{choose the best } y \end{array}\right]$ in each iteration.

Input: 2 endpts $(X_1, Y_1)(X_2, Y_2)$

Def: $\Delta Y = Y_2 - Y_1 > 0$. $\Delta X = X_2 - X_1 \geq 0$. (Integers)

(I)



(From picture) $\rightarrow$ $A < B$. choose NE

$A - B < 0$ "

$\underbrace{\Delta X (A-B)}_{= D} < 0$ "

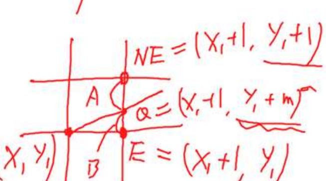
(Note: $\Delta X > 0$)

Define: $D = \Delta X (A - B)$

$D < 0$ choose NE

$D \geq 0$ " E.

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



$A = 1 - m$

$B = m$.

$\boxed{m = \frac{\Delta Y}{\Delta X}}$

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

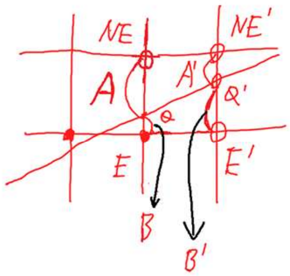
$= \Delta X \left(1 - 2 \cdot \frac{\Delta Y}{\Delta X}\right) = \Delta X - 2 \Delta Y.$

($\Delta X, \Delta Y$: integers $\Rightarrow$ D_{start}: integer)

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 :

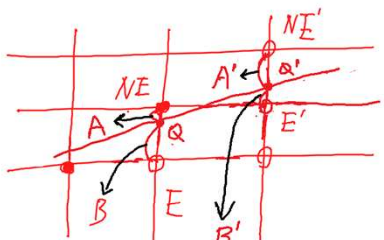
$$(A' - A) - (B' - B) = -2m$$

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

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

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

(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

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

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

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

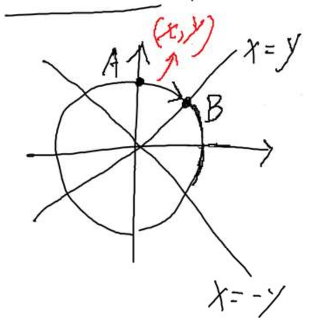
Summary:

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

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

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

HW1 Part (a)



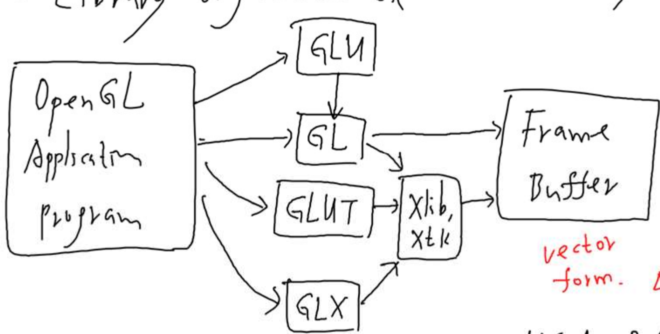
The points in the 8 symmetric regions.

$\boxed{(x, y)}$	(y, x)
$(-x, y)$	$(y, -x)$
$(x, -y)$	$(-y, x)$
$(-x, -y)$	$(-y, -x)$

7

Introduction to OpenGL

* Library organization (for X window system)



GLU: graphics utility library

GL: OpenGL Library (indept. of window system)

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

GLEW

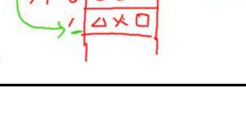
vector form.

* OpenGL Command Syntax: gl Vertex 3 f [v] (   )

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

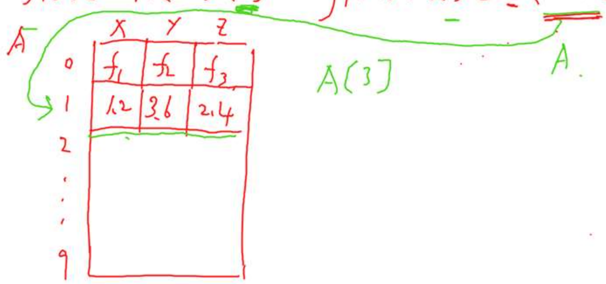
Data Type
i : integer
f : float ..

eg. glVertex3f (1.2, 3.6, 2.4)
glVertex3fv (ptr)

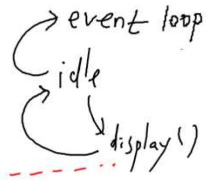


8

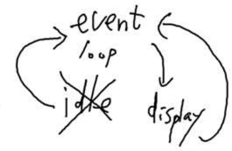
float A[10][3]   glVertex3fv(A[1])




```
idle( )
{
    ⋮
    display( );
}
```




```
idle( )
{
    ⋮
    glutPostRedisplay( );
}
```



* Discussed the handout sample code "example.cpp".

9