Syllabus

Lecture - 1

21.1.2026

Introduction
Scan - Conversion
} → Mid

2D Tranformation
3D Transformation
Clipping
Polygon filling
Curves
} → Final

Introduction :

Computer Graphics কি ?

↓

raw data $\xrightarrow{\text{Process করে}}$ Meaningful Info Generate করে।

Graphic

Graph — Pics

Graph
↓
Geometric object
- → Line
- → Circle
- → Ellipse

Requirements :

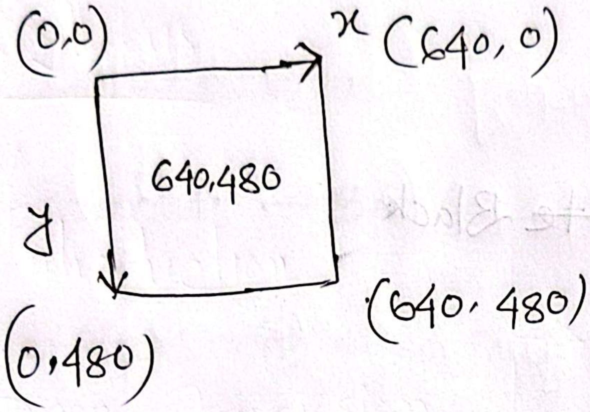
1. Input device
2. Output device
3. Display device
4. Interfacing device – output এর interact করে
   Game -এর Joystick

Graphics System.

২ প্রকার

① Active → যে graphics system output কে user control করতে পারে তাকে Active G.S. বলে (Game)

② Passive → Output এর উপর user এর control থাকে না। (Movie ?)



Resolution : প্রতিটি Maximum number of pixels

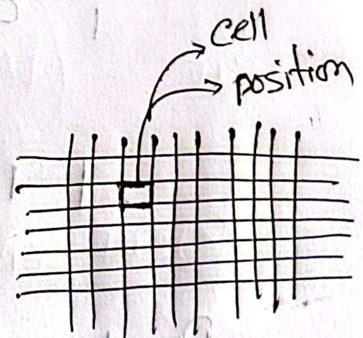
640X480

1024X720 w better cause Pixel beshi-

= গুনফল ই Pixel সংখ্যা। একেকটি cell বলে।

যার Pixel বেশী সেটিই বেটার।

- এ দাম ও বেশী হবে।



Pixel Representation :

দুইটা দিক দেখতে হয়

→ Position (x, y)

→ Color

→ RGB
→ CMYK } দুই Method

RGB → Display - এ use হয়

CMYK → Printer - এ use হয়

বাইনারিতে Convert হয়ে যাবে।

R G B

0 0 0 → ~~White~~ Black

1 0 0 → Red

0 1 0 → Green

0 0 1 → blue

1 1 0 →

0 1 1 →

1 0 1 →

1 1 1 → ~~Black~~ White

Pixel value - 3 bit করে use করলে - 8 টা color পাওয়া যাবে।

- 8 bit - 256 colour পাওয়া যাবে

= 24 bit ←

R G B

8↓ 8↓ 8↓

$256 \times 256 \times 256 = 16,777,216 = 16.7$ Million

Lecture-2
26.1.26

Version — 17.12

Glut version — 3.7.6

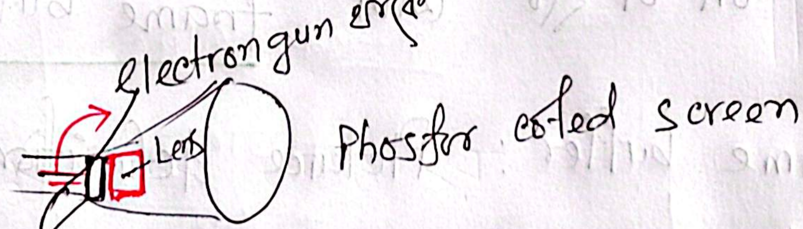
Codeblocks with mingw 17.12

Introduction

2nd slide :

Display device ↓

CRT Monitor (Cathode Ray tube)



→ Cathode Ray হিট করে তার ফলে screen এ দেখা যায়।

→ Voltage Regulation

→ এখানে electron গতিতে অবস্থান স্থির হয়।

→ Refreshing হয় (নির্দিষ্ট time পর Repeat হয়) যার ফলে আগের image হারায় না।

CRT Monitor Mechanism

(i) Raster Scan Display (JPG, JPEG, .gif)

(ii) Vector " " (.doc, .pdf)

Frame buffer ← electron gun visit করে

(i) Raster :

ফাইল Supported

অনেক'টা বড়

হয়

Row wised devided

সরলরেখা হবে না।

off

Row wised visit করে যেখানে on হবে, সেখানে

হত (Left to Right)

কোথায় on off হত সেটা frame buffer থেকে থাকে

" এ frame buffer : Picture definition storage

↓

user through command

এই টা continuously চলার পর Screen

দেখা যায়।

✓ Pixel ধরে ধরে কাজ করে।

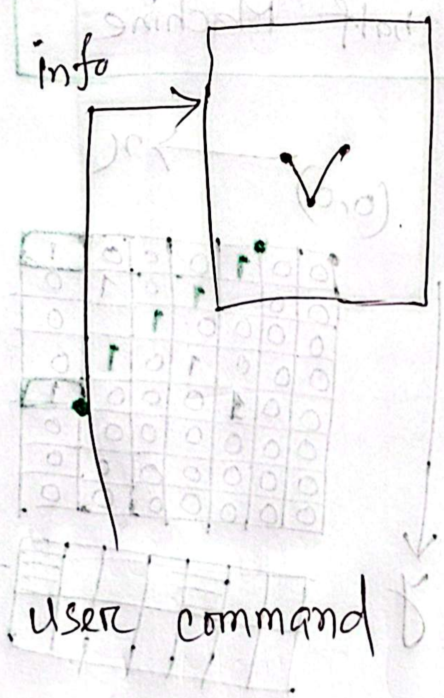
✓ Extra hardware দরকার

Binary তে convert হয়

user com. ...

Vector Scan Display

- সরাসরি user command থেকে information নেই।



- Curve টাইপ show করেনা
- complex না

ফাইল Supported সেইটা ছোট হয়।

Frame buffer কি এবং কিভাবে picture definition storage হয়-

Difference between vector vs Raster

User command কিভাবে frame buffer এ যাবে?

Move, Line
(2,0)

: Assembly language
half ~~bu~~ human
half Machine

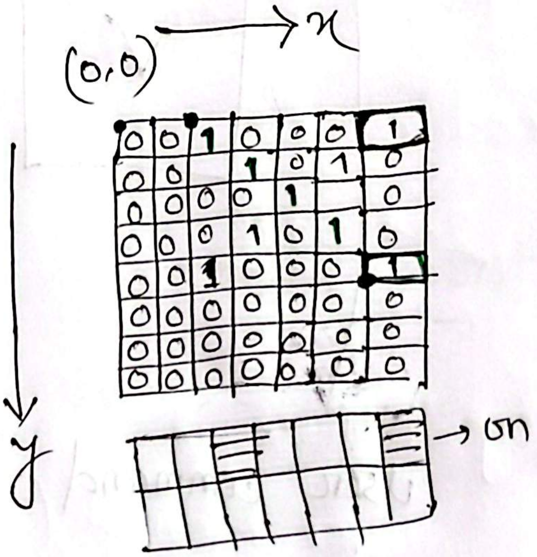
User command

Move (2,0)

Line (4,4)

Move (-4,0)

Line (4, -4)

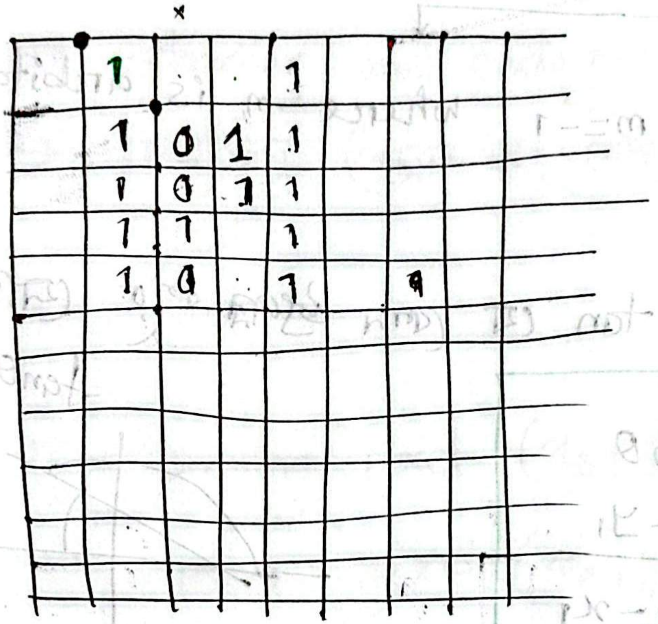


x-এর value	y-এর value
+ ডানে	+ নিচে
− বামে	− উপরে

Line → starting position to
↓
x,y position

Move → Position
↓
set 1

move (1,0)
Line (0, 4)
Move (3,0)
Line (0,4)
Line (-3,4)



Lecture-3
2.2.24

Scan Conversion :

Line
circle
ellipse

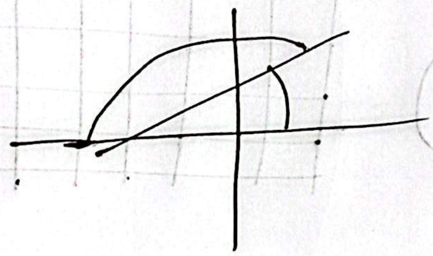
Line:
→ Horizontal (x - অক্ষ)
→ vertical (y অক্ষ)
→ Diagonal

Diagonal

- where m = 1
- where m = -1
- where m is arbitrary

ধনাত্মক দিকের tan যে কোন উৎপন্ন করে সেটিই হল m

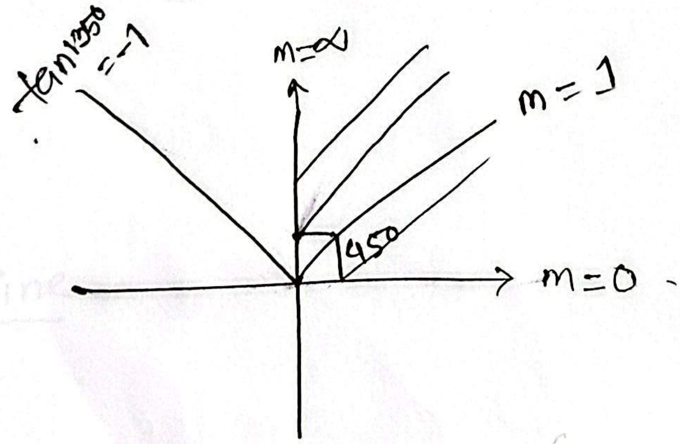
tanθ



$$\boxed{m = \tan\theta \qquad m = \frac{y_2 - y_1}{x_2 - x_1}}$$

$y = \cancel{ax}\ mx + b$ ছেদাংশ y অক্ষের

$y = mx$ where $b = 0$



m → is arbitrary

└→ m এর যেকোনো মান

Horizontal Line

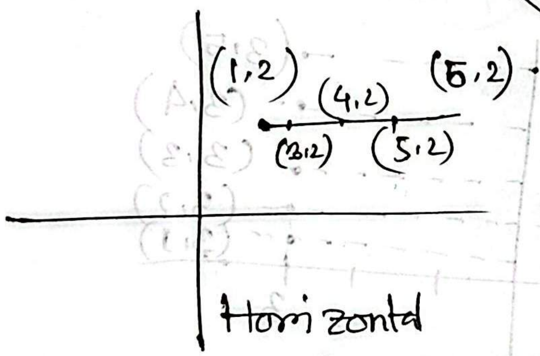
Line এর ক্ষেত্রে starting and ending pixel

end point → প্রান্তবিন্দু (শুরু এবং শেষ)

Algorithm

1. Input → starting pixel (x_s, y_s)
   ending pixel (x_e, y_e)

2. if $(x_s > x_e)$ then swap
   $(x_s \leftrightarrow x_e)$



Horizontal
→ x increase
→ y constant
$(y_s = y_e)$

3. Initialization $x = x_s$,
   $y = y_s$

4. Loop : $\boxed{\text{Setpixel } (x, y) \text{ with color}}$ → Graphics এর কাজ

   $x = x + 1$
   if $(x \leq x_e)$ go to to loop
   otherwise exit

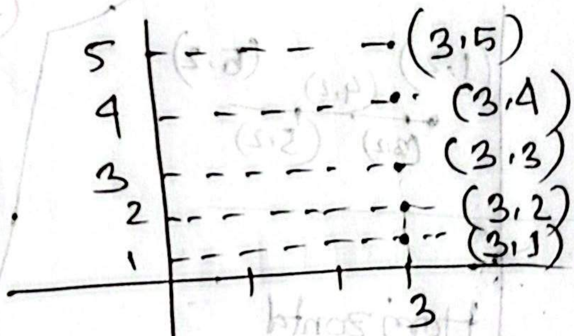
Float value নিতে হবে,

Glut এর জন্য হেডার ফাইল লাগে
তারপর যে language use করবো তার header

Lecture - 4

3. 02. 2026

Vertical Line:



x constant

$y + 1$

Algorithm :

1. Input

   starting pixel (x_s, y_s) and

   ending pixel (x_e, y_e)

2. If $(y_s > y_e)$, then swap $y_s \leftrightarrow y_e$

3. Initialization $x = x_s$, $y = y_s$

4. Loop : Set pixel (x, y) with color

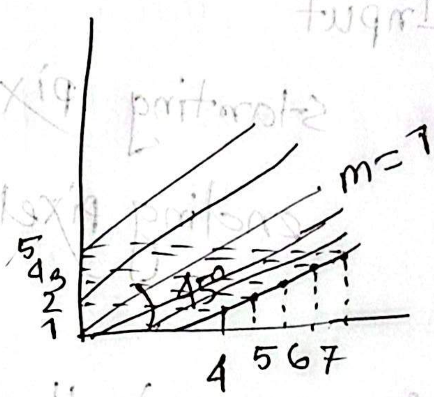
   $y = y + 1$

   if $(y \le y_e)$ go to Loop

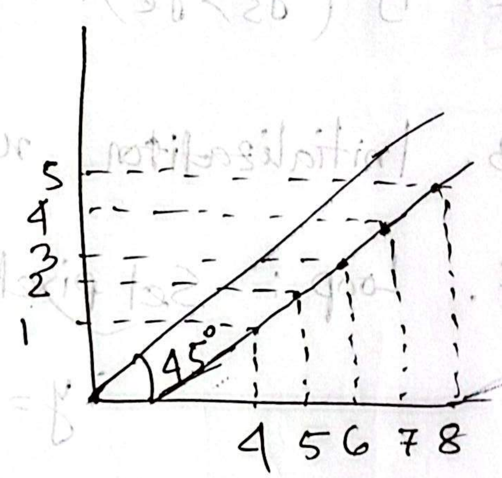
   otherwise exit

Diagonal Line where $m = 1$

Algorithm:



1. Input
   starting pixel (x_s, y_s)
   Ending pixel (x_e, y_e)



2. If $(x_s > x_e)$ then
   swap $x_s \leftrightarrow x_e$, $y_s \leftrightarrow y_e$

3. Initialization $x = x_s$, $y = y_s$

4. Loop: Setpixel (x, y) with color
   $x = x + 1$
   $y = y + 1$

   if $(x \leq x_e)$ go to loop

   otherwise exit

$(4,1)\ (5,2)\ (6,3)\ (7,4)$

✓ এক এক করে বাড়ছে

✓ x এবং y এর Difference একই

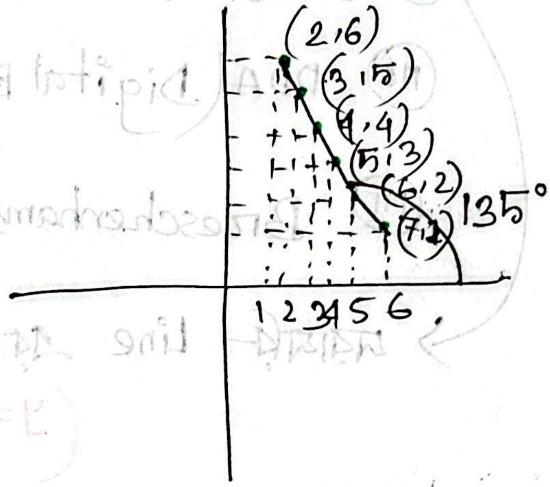
$$\boxed{m = \frac{\Delta y}{\Delta x}}$$

✓ x এক বাড়লে y ও এক বাড়তেছে

✓ condition alwey independent variable.

Diagonal Line where $m = -1$

Algorithm:



1. Input →

   Starting pixel (x_s, y_s)

   Ending pixel (x_e, y_e)

2. If $(x_s > x_e)$ then swap $x_s \leftrightarrow x_e$,

   $y_s \leftrightarrow y_e$

x এর Value বাড়ছে

y এর Value কমছে

3. Initialization $x = x_s$, $y = y_s$

$\boxed{m = \frac{\Delta y}{\Delta x} = -1}$

$\boxed{\Delta y = \Delta x}$

4. Loop: Setpixel (x, y) with color

   $x = x + 1$

   $y = y - 1$

   if $(x \leq x_e)$ go to loop

   otherwise exit

Diagonal Line where m is arbitary ---

৩ টা Types

(i) Direct Line Drawing Algorithm

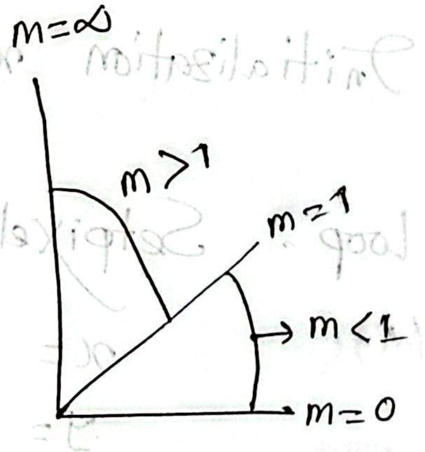
(ii) DDA (Digital Differential Analyzer) Algorithm

(iii) Brescheham's Algorithm

→ সরাসরি Line এর equation

$(y = mx + c)$

(i) Direct Line drawing Algo



Algorithm

1. Input
   → Starting pixel (x_s, y_s)
   Ending pixel (x_e, y_e)

y এর value পরিবর্তন এর জন্য x independent

2. If $(x_s > x_e)$, then swap
   $x_s \longleftrightarrow x_e$, $y_s \longleftrightarrow y_e$

x++ (increme
y calculation
লাগে তাই Al

3. Initialization $x = x_s$, $y = y_s$

$y = mx + b$

4. $m = \dfrac{(y_e - y_s)}{(x_e - x_s)}$ → [m এর মান বের করে নিতে হবে]

[b এর মান]

5. $b = y_s - mx_s$

6. Loop : Setpixel (x, y) with color

$x = x + 1$

$y = mx + b$, Round (y)

if $(x \leq x_e)$ go to loop

otherwise 'exit'

$(x_e - x_s) + 1$ পর্যন্ত loop চলবে

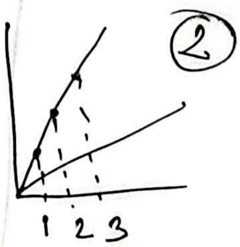
50 থেকে 200 গুন বেশী সময় লাগবে এজন্যই time high

এই Algorithm efficient না

Disadvantage :

1. Computation time high

2. $m > 1$ -এর জন্য ভালো মতো কাজ করে না তার কারন (y) গ্যাপ বেশী হয়।



তারপর DDA Algorithm এসেছে

Math:

Starting (1, 1)
Ending (4, 5)

4. $m = \frac{(5-1)}{(4-1)} = \frac{4}{3}$

5. $b = 1 - \frac{4}{3} \cdot 1$

$= -\frac{1}{3}$

6.

Answer

x	$y = mx+b$	Round(y)	(x, y)
1	1	1	(1, 1)
2	$\frac{4}{3} \cdot 2 - \frac{1}{3} = \frac{7}{3} = 2.33$	2	(2, 2)
3	$\frac{4}{3} \cdot 3 - \frac{1}{3}$ $\frac{11}{3} = 3.66$	4	(3, 4)
4	$\frac{4}{3} \cdot 4 - \frac{1}{3}$ $\frac{15}{3}$	5	4, 5

y এর Calculation দেখাতে হবে। নাহলে মার্ক কাটবে

Lecture - 6

17.2.26

$m < 1$	$m > 1$
x independent	x dependent
$x = x+1$	
y dependent	y in
$y = y + m$	$y = y+1$
	$x = x + \frac{1}{m}$

Math

Starting (2, 2)

Ending (7, 6)

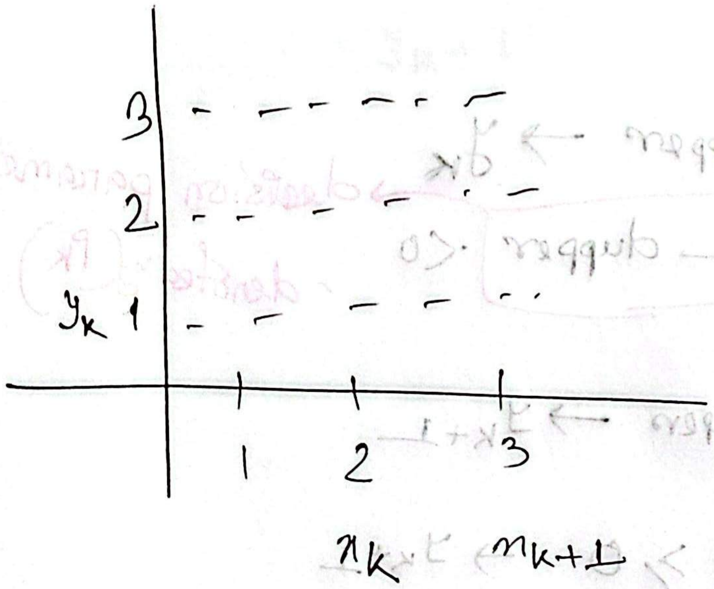
$$m = \frac{6-2}{7-2} = 0.8 < 1$$

x	y = y+m	Round(y)	(x,y)
2	2	2	(2,2)
3	2+0.8=2.8	3	(3,3)
4	2.8+0.8=3.6	4	(4,4)
5	3.6+0.8=4.4	4	(5,4)
6	4.4+0.8=5.2	5	(6,5)
7	6	6	(7,6)

If $m > 1$

y	$x = x + \frac{1}{m}$	Round(x)	(x, y)
5			

Bresenham's Algorithm :



$$x_{k+1} = x_k + 1$$

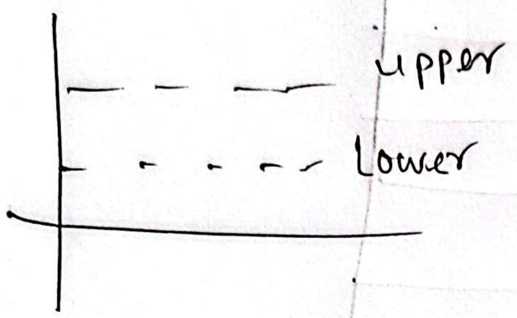
$$y_k = y_k / y_{k+1}$$

ceiling Value এর জন্য 1 বেশী হচ্ছে

floor এর জন্য পরিবর্তন হচ্ছে না

$x_{k+1} = x_{k+1}$

$y_{k+1} = y_k / y_{k+1}$ [আগেরটা থাকবে অথবা 1 যোগ হবে



→ এই দুইটার difference অনুযায়ী change হয়।

$$\boxed{\begin{aligned} d_{lower} &= y - y_k \\ d_{upper} &= (y_{k+1}) - y \end{aligned}}$$

$d_{lower} < d_{upper} \to y_k$

$\boxed{d_{lower} - d_{upper}} < 0$ → decision parameter - denoted by (P_k)

$d_{lower} \geq d_{upper} \to y_{k+1}$

$d_{lower} - d_{upper} \geq 0 \to y_{k+1}$

$x_{k+1} = x_k + 1$

$y_{k+1} = y_k / y_{k+1}$

$P_k < 0$

$x_{k+1} = x_k + 1$

$y_{k+1} = y_k$

$P_k \geq 0$

$x_{k+1} = x_k + 1$

$y_{k+1} = y_k + 1$

CamScanner