Syllabus

Lecture - 1

21.1.2026

- Introduction
- Scan - Conversion

→ Mid

- 2D Tranformation
- 3D Transformation
- Clipping
- Polygon filling
- Curves

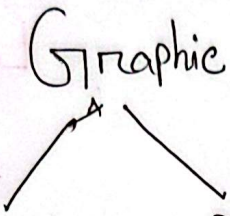
→ Final

Introduction:

Computer Graphics কি ?

↓

raw data → Process করে → Meaningful Info Generate করে।



Geometric object
- line
- Circle
- Ellipse

Requirements :

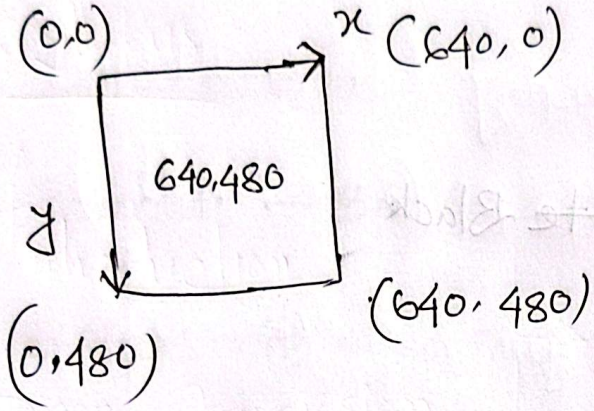
1. Input device
2. Output device
3. Display device
4. Interfacing device – Output কর interact করে Game -কর joystick

Graphics System.

২ প্রকার

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

② Passive → Output কর জন্য user কর control থাকে না। (Movie)



Resolution: Maximum number of pixels

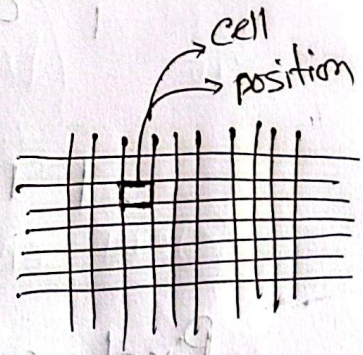
640X480

1024X720 ✓ better cause pixel beshi-

= একটা ইমেজে pixel সংখ্যা। একেকটি cell বলে।

যার pixel বেশী সেটি বেটার।

- ঘনত্ব 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 রকম color পাওয়া যাবে।

- 8 bit — 256 colour পাওয়া যাবে
- 24 bit —

R G B

8 ↓ 8 ↓ 8 ↓

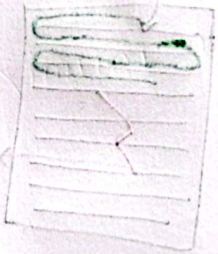
256 X 256 X 256 = 16,777,216 = 16.7 Million

Lecture-2
26.1.26

Version — 17.12

Gimet version — 3.7.6

Codeblocks with mingw 17.12

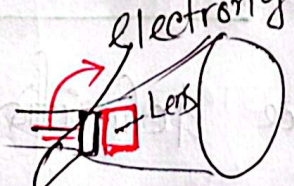


Introduction

2nd slide:

Display device ↓

CRT Monitor (Cathode Ray tube)



→ Cathode Ray ছিটে কোর তার কোর screen এ পৌঁছে যায়,

→ Voltage Regulation

→ এখানে electron গতিও ব্যবস্থাপনা করা হয়,

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

CRT Monitor Mechanism

① Raster Scan Display (JPG, JPEG, gif)

② Vector " " (.doc, .pdf)

Frame buffer ← electron gun visit করে

① Raster:

Row wised devided থাকলেও হবে না।

ফাইল Supported অনেকটা বড় হয়

Row wised visit করে যেখানে on হবে সেখানে off হয় (Left to Right)

(কোথায় on of হয় সেটা frame buffer থেকে হবে)

" ∴ frame buffer: Picture definition storage

↓

user through command

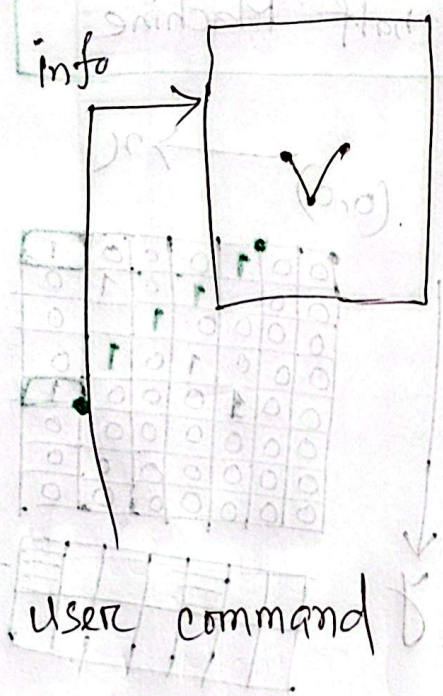
এই ফাইল continuously চলার পর screen দেখা যায়।

Pixel ধরে ধরে কাজ করে।

Extra hardware দরকার।

Binary তে convert হয়ে

Vector Scan Display



- সরাসরি user command থেকে information নেই।
- Curve চিত্র show করে না
- complex না

ফাইল Supported সেইটা ছোট হয়।

- Frame buffer কে বলে 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

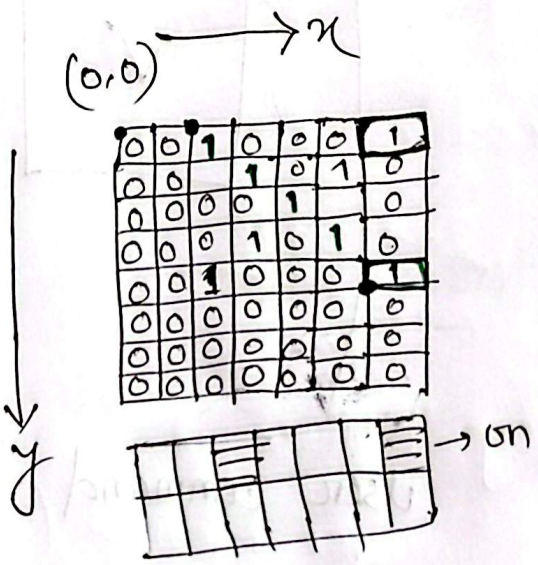
User Command

Move (2,0)

Line (4,4)

Move (-4,0)

Line (4,-4)



x-এর value
+ ডানে
- বামে

y-এর value
+ নিচে
- উপরে

Line → starting position to
↓
x,y position

Move → Position
↓
set 1

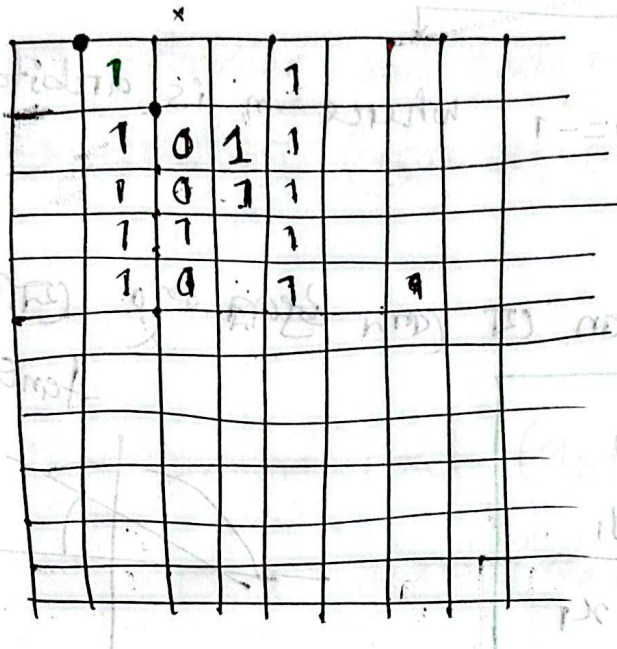
move (1,0)

Line (0,4)

Move (3,0)

Line (0,4)

Line (-3,4)



Lecture-3

2.2.26

Scan Conversion :

Line

circle

ellipse

Line:

→ Horizontal (x - axis)

→ vertical (y axis)

→ Diagonal

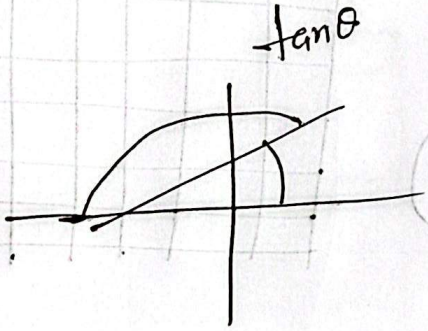
Diagonal

where $m=1$ | where $m=-1$ | where m is arbitary

ধনাত্মক দিকের tan যে কোন ঋণাত্মক কৃত সেটাই হলো m

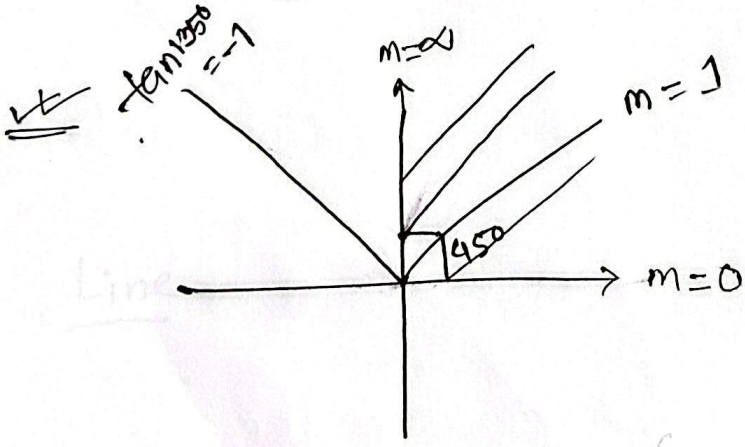
$$m = \tan\theta$$

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$



$y = mx + b$ — ছেদাংশ y অক্ষের

$y = mx$ where $b=0$



$m \rightarrow$ is arbitary

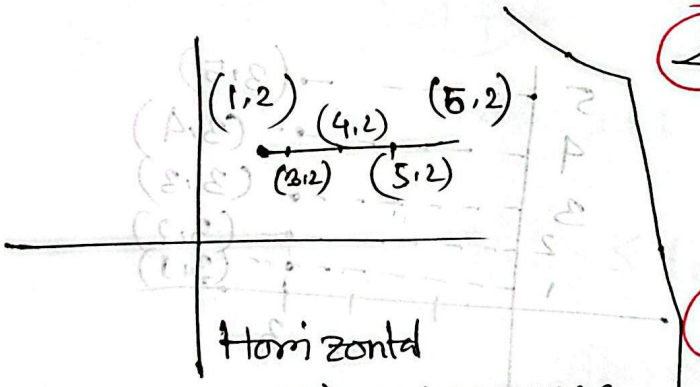
↳ m যে কোনো মান

Horizontal Line

Line এর জন্য 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 $(x_{s_0} \leftrightarrow x_e)$

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

4. Loop : Setpixel (x, y) with colors → Graphics এর জন্য
   $x = x + 1$
   if $(x < x_e)$ go to loop
   otherwise exit

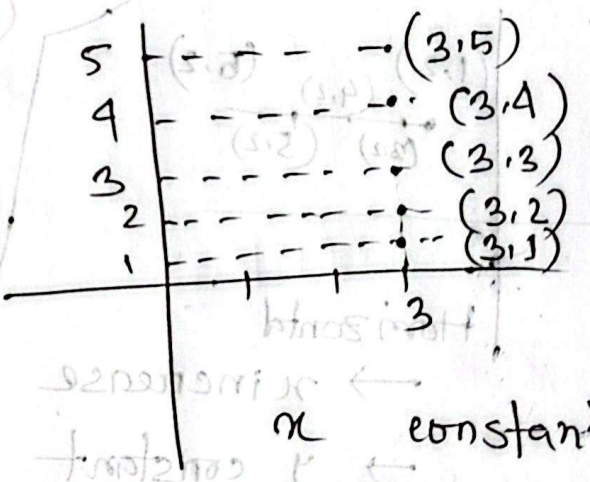
Float value নিতে হবে,

Glut এর ক্ষেত্রে এইটা যাবে.
তারপর যে language এ করবে তার header

Lecture 4

3.02.2026

Vertical Line:



Algorithm:

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

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

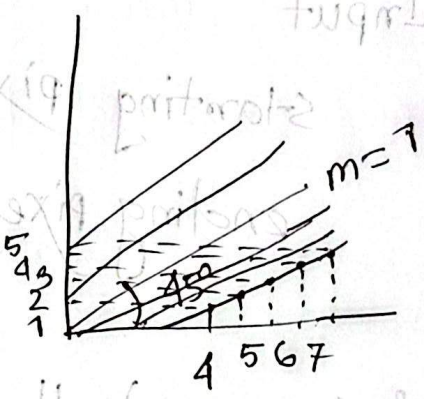
3. Initialization $x = x_s$, $y = y_s$

4. Loop: Set pixel (x, y) with color
   $y = y + 1$
   if $(y \leq y_e)$ go to Loop
   otherwise exit

Diagonal Line where m = 1

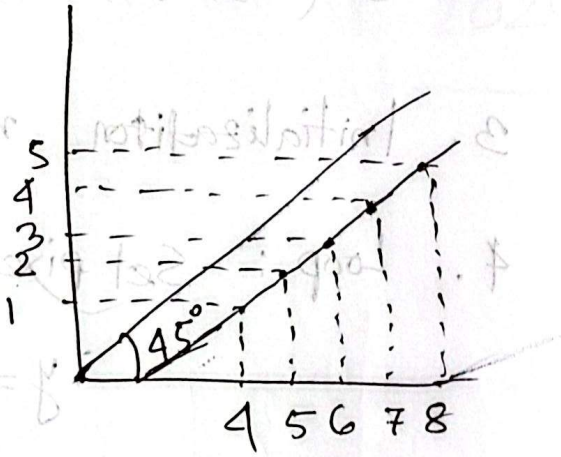
Algorithm:

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

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

✓ x এক বাড়লে y ও এক বাড়তেছে

✓ condition always independent variable.

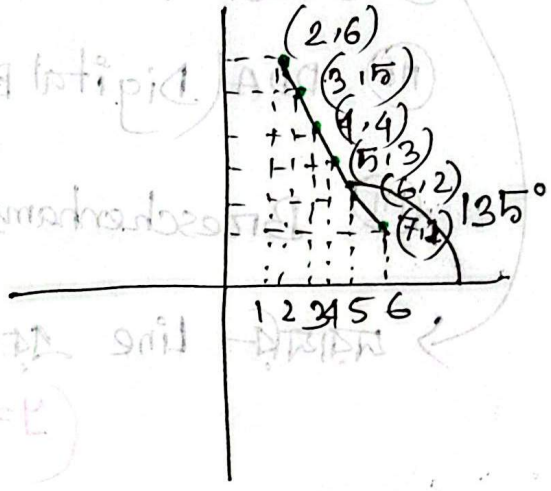
Diagonal Line where $m = -1$

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 $x = x_s$, $y = y_s$

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

$$\Delta y = \Delta x$$

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

   $x = x + 1$

   $y = y - 1$

   if $(x < x_e)$ go to loop

   otherwise exit

Diagonal Line where m is arbitary ---

৩ টি types

① Direct Line Drawing Algorithm

①ⓑ DDA (Digital Differential Analyzer) Algorithm

①ⓒ Breschenhams Algorithm

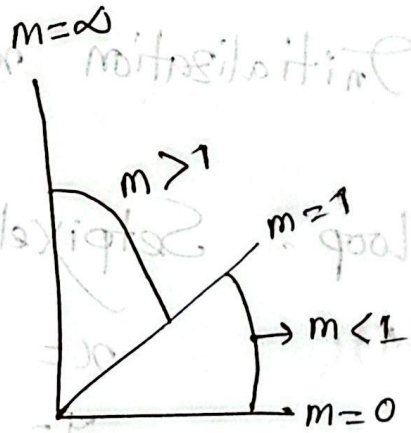
→ সরাসরি Line এর equation

$(y = mx + c)$

① Direct Line drawing Algo

Algorithm

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



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

y এর value পরিবর্তন এর জন্য x independent

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

m++ (increm
y calculation
এলগো ডিং Al

$y = mx + b$

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

5. $b = y_s - m x_s$ [b এর মান]

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

$x = x + 1$

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

if $(x < x_e)$ go to loop

otherwise exit

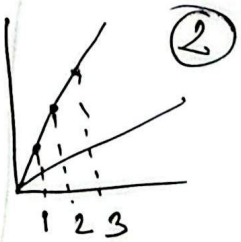
$(x_e - x_s) + 1$ পর্যন্ত loop চলবে

যখন ৫০ থেকে ২০০ এমন বড়ো সমস্য লাগবে তখনই time high

এই Algorithm efficient না

Disadvantage:

1. Computation-time high
2. $m > 1$ -এর জন্য আলো মতো কাজ করে না তার কারণ(y)গ্যাপ বেশী হয়।



তারপর DDA Algorithm শুরু হয়েছে

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

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

Answer

y এর Calculation দেখাতে হবে। নয়তো মার্ক কাটা

Lecture - 2

17.2.22

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

Ex-1

Starting $(2, 2)$

Ending $(5, 9)$

$$m = \frac{9-2}{5-2} = 0.8 < 1$$

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)

Lecture - 6

17.2.26

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

Math

Starting (2, 2)

Ending (7, 6)

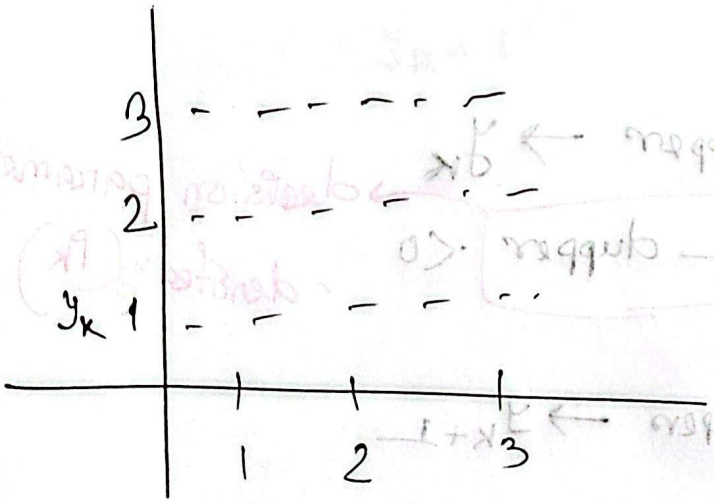
$$m = \frac{6-2}{7-2} = 0.8 < 1$$

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			

Brescenham's Algorithm:



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

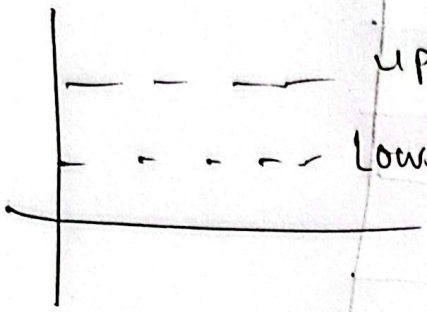
$y_k = y_k / y_{k+1}$

ceiling value এর জন্য 1 বেশী হবে

floor এর জন্য পরিবর্তন হবে না

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

$y_{k+1} = y_k / y_{k+1}$ [আগেরটা থাকলে তখন 1 নেওয়া হবে



এই দুইটার difference অনুযায়ী change হয়।

$$d_{lower} = y - y_k$$
$$d_{upper} = (y_{k+1}) - y$$

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

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

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

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

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

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

$$P_k < 0$$

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

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

$$P_k \geq 0$$

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

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

Lecture – 7

18.2.26

$d_{lower} = y - y_k$

$d_{upper} = (y_{k-1}) - y$

$y = m(x_k + 1) + b$

x_k এর update value

এবং y এর Actual value

$= m \cdot (x_k + 1) + b - y_k$

$= y_{k+1} - y = y_k + 1 - m(x_k + 1) - b$

$d_{lower} - d_{upper}$

$= 2m(m_k + 1) - 2y_k + 2b - 1 \quad \left[\because m = \frac{\Delta y}{\Delta x}\right]$

$= 2\frac{\Delta y}{\Delta x}(x_k + 1) - 2y_k + 2b - 1$

$= \frac{1}{\Delta x}(2\Delta y x_k + 2\Delta y - 2\Delta x y_k + 2\Delta x b - \Delta x)$

$\Delta x(d_{lower} - d_{upper}) = 2\Delta y x_k - 2\Delta x y_k + (2\Delta y + 2\Delta x b - \Delta x)$

$= 2\Delta y x_k - 2\Delta x y_k + c$

$\therefore P_k = 2\Delta y x_k - 2\Delta x y_k + c$

Decision parameter fixed কোনো value না বার বার চেঞ্জ হয়।

পরবর্তী decision parameter

$P_{k+1} = 2\Delta y x_{k+1} - 2\Delta x y_{k+1} + c$

$\rightarrow P_{k+1} - P_k = 2\Delta y (x_{k+1} - x_k) - 2\Delta x (y_{k+1} - y_k)$ (difference 1)

$P_{k+1} = P_k + 2\Delta y - 2\Delta x (y_{k+1} - y_k)$

if $P_k < 0$

$y_{k+1} = y_k$

$= P_k + 2\Delta y - 2\Delta x (y_k - y_k)$

$P_{k+1} = P_k + 2\Delta y$

x, y এবং decision parameter দুই জেতা লাগবে

$P_k \geq 0 \rightarrow P_{k+1} = P_k + 2\Delta y - 2\Delta x$

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

দুই ক্ষেত্রে আলাদা আলাদা

Initial decision Parameter

$$P_0 = 2\Delta y - \Delta x$$

Algorithm :

1. Input → (x_s, y_s), (x_e, y_e)
2. $m = (y_e - y_s) / (x_e - x_s)$
3. Calculate $\Delta x, \Delta y, 2\Delta x, 2\Delta y$
4. If $(m < 1)$ go to step 5
   otherwise step 8
5. If $(x_s > x_e)$, Swap
6. Initial D.P, $P_0 = 2\Delta y - \Delta x$
7. Loop : $(k = 0 - \Delta x)$ [Δx times Repeat
   Setpixel (x_k, y_k) with color

if($P_K < 0$)

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

$y_{k+1} = y_k$

$P_{k+1} = P_k + 2\Delta y$

else

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

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

$P_{k+1} = P_k + 2\Delta y - 2\Delta x$

8. if ($y_s > y_e$) swap

9. Initial D.P. $P_0 = 2\Delta x - \Delta y$

10. Loop : ($k = 0 - \Delta y$) (Δy times)

Setpixel (x_k, y_k) with color

if($P_k < 0$)

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

$x_{k+1} = x_k$

$P_{k+1} = P_k + 2\Delta x$

else

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

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

$P_{k+1} = P_k + 2\Delta x - 2\Delta$

Start (20, 10) End (30, 18)

$m = \frac{18 - 10}{30 - 20}$

$= \frac{8}{10}$

$m = 0.8$

$\Delta x = 10$

$\Delta y = 8$

$P_0 = 16 - 10$

$= 6$

$2\Delta x = 20$

$2\Delta y = 16$

k	P_k	x	y	x, y	
0	6	20	10	(20, 10)	$P_1 = P_0 + 2\Delta y - 2\Delta x$
1	2	21	11	(21, 11)	$= 2$
2	-2	22	12	(22, 12)	
3	14	23	12	(23, 12)	$P_2 = P_1 + 2\Delta y - 2\Delta x$

Lecture-8

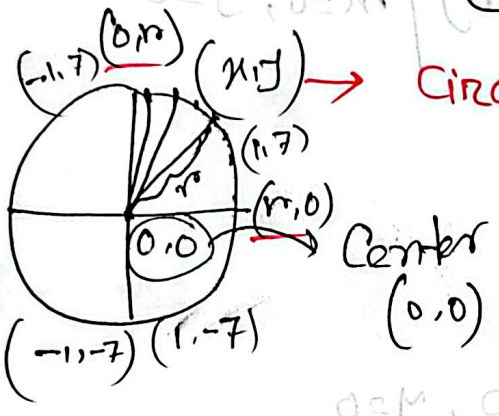
24.2.2026

Circle

General formula

$x^2 + y^2 = r^2$

↳ radius



→ Circle এর উপর যেসব value থাকবে pixel value

given value/input ⟶ radius (r)

r এর value সবখানে সমান (distance same)

শুধু Co-ordinate same করি (যার distance same হবে

Circle -এর জন্য দুইটা Algo -

① Direct circle drawing

② Mid-point circle drawing

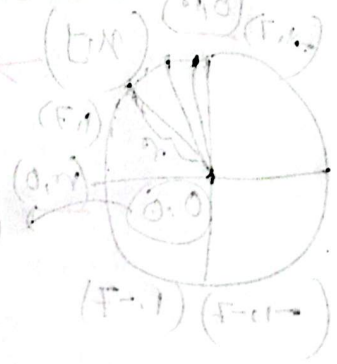
$$x^2 + y^2 = r^2$$

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

→ Direct circle drawing Algo

~~1. Initial Pixel (0,r)~~

Initial Pixel $(0,r)$ / $x=0, y=0$



Algo.

1. Input → r
2. Initialization $x=0, y=0$
3. Loop : Setpixel (x,y) with colour

" (x,y) "

" $(-x,-y)$ "

" $(x,-y)$ "

$x = x+1$

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

if $(x \le r)$ go to loop;

otherwise exit

Flood fill

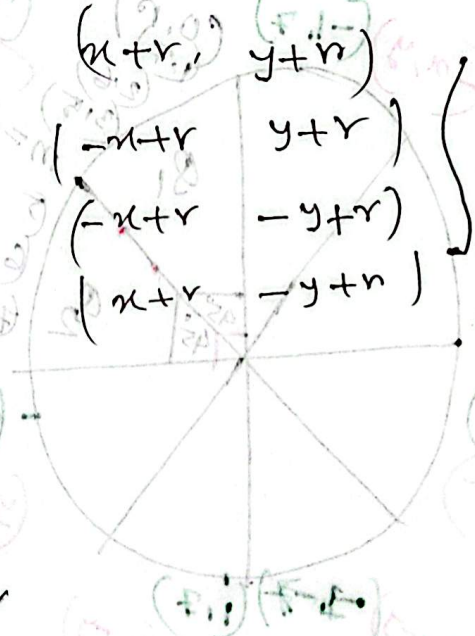
① r এর value (0-1)

② $x = x + 1$ এখানে (0.1 করে বাড়াতে হবে)

③ y এর value Round করা যাবে না।

✓ float use করলে শুধু positive coordinate থাকবে

✓ int use করলে



→ করতে হবে

Disadvantage

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

$\sqrt{r^2 - x \cdot x}$ → complex process

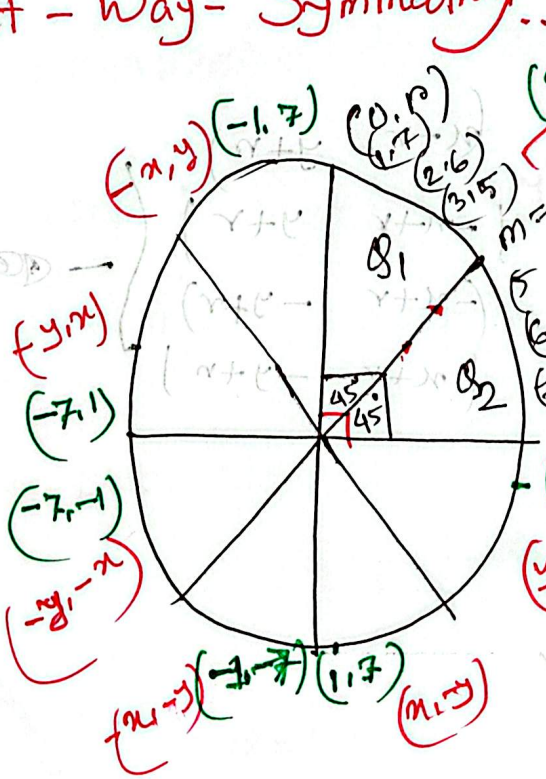
তাই Computational time বেশী

(11) ~~R~~ r -এর value কথানো ৩

Pixel - যখন x-axis এর কাছে থাকতে তখন pixel-Gap বেড়ে যাবে।

Loop -এতে Pixel GAP কেমন থাকে সেটি কমানোর জন্য কয়টি মেথড বলা হয় সেটি হলো

Eight - Way - Symmetry...



x -এর value 1 থেকে 1 (1,1) (x=y)

0 থেকে চলবে যতক্ষণ $x \leq y$ পর্যন্ত

$x > y$ হলে break

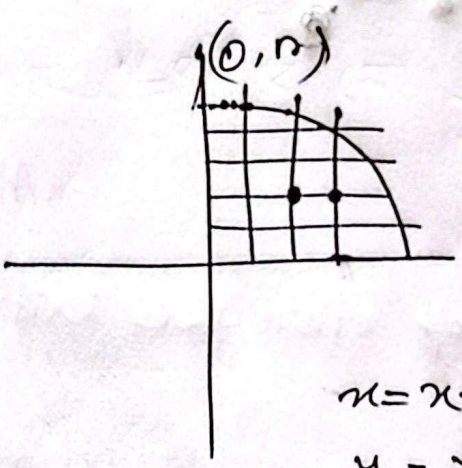
Lecture-9

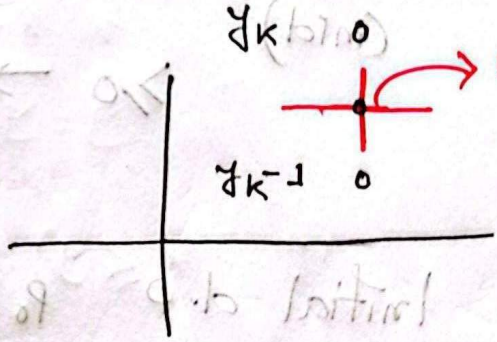
25.2.26

$x^2+y^2=r^2$

Peircle $= x+y^2-r^2$

$x^2+y-36 \begin{cases} <0 & \text{inside} \\ =0 & \text{on the} \\ >0 & \text{outside} \end{cases}$





এটাই বেশি লাগে

$x = x+1$

$y = y / y-1$

$\rightarrow$ 1st $\rightarrow (x_{k+1}, y_k)$

2nd $\rightarrow (x_{k+1}, y_k-1)$

$$\text{Midpoint} = \frac{x_{k+1}+x_{k+1}}{2}, \frac{y_k+y_k-1}{2}$$

$$= x_{k+1}\ y_k - \frac{1}{2}$$

$P_{circle} = x^2 + y^2 - r^2$

$= (x_k + 1)^2 + (y_k - \frac{1}{2})^2 - r^2$

value > 0

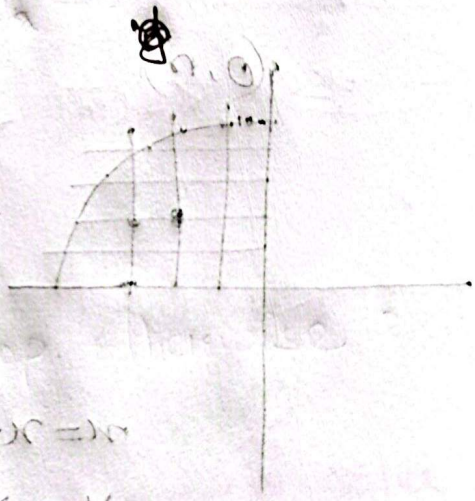
Middle point কে depend করে decision

হবে y_k নাকি y_{k-1} হবে

$P_{circle} < 0 \longrightarrow y_k$

(mid)

$\geq 0 \rightarrow y_{k-1}$



Initial d.P $p_0 = 1 - r$

$x_k = 0$

$y_k = r$

$(0+1)^2 + (r - \frac{1}{2})^2 - r^2$

$= \frac{3}{4} - r$

$= 1 - r$

$$P_K = (x_{K+1})^2 + \left(y_K - \frac{1}{2}\right)^2 - r^2$$

$$P_{K+1} = (x_{K+1})^2 + \left(y_{K+1} - \frac{1}{2}\right)^2 - r^2$$

$$P_{K+1} - P_K =$$

$$P_{K+1} \cdot$$

Algorithm:

1. Input r
2. Initialization $x=0$, $y=r$
3. $P_0 = 1-r$

$x=0$ to $x=y$

4. Loop: Setpixel $(x,y)\ (y,x),\ (y,-x)\ (x,-y)$

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

$x = x+1$

if $(P_K < 0)$

$y = y$

$P_{K+1} = P_K + 2x_K + 3$

if $(P_k < 0)$

$$P_{k+1} = P_k + 2x_k + 3$$

$$x = x+1$$

$$y = y$$

else

$$P_{k+1} = P_k + 2x_k - 2y_k + 5$$

$$x = x+1$$

$$y = y-1$$

if $(x \leq y)$ go to loop, otherwise exit

Math:

Given $r = 8$

Initial pixel $(0, 8)$, $x = 0$, $y = 8$

$P_0 = 1 - 8 = -7$

K	P_K	x_K, y_K	x_{K+1}, y_{K+1}	
0	-7	(0, 8)	(1, 8)	$P_1 = P_0 + 2x_K + 3$ $= -7 + 4$
1	-4	(1, 8)	(2, 8)	1
2	-1	(2, 8)	(3, 7)	-6
3	-6	(3, 7)	(4, 7)	
4	3	(4, 7)	(5, 6)	
5	2	(5, 6)	(6, 5)	

$x > y$

$x > y$

ACTIVE
Date :
Page :

Math $r = 8$

so that, initial pixel $x = 0$, $y = 8$

$\therefore P_0 = 1 - 8 = -7$

k	P_k	(x_k, y_k)	(x_{k+1}, y_{k+1})	$P_{k+1} = P_k + 2x_k + 3$
0	-7	(0, 8)	(1, 8)	$P_1 = P_0 + 2x_k + 3 = -7 + 2.0 + 3 = -4$
1	-4	(1, 8)	(2, 8)	$P_2 = -4 + 2*1 + 3 = 1$
2	1	(2, 8)	(3, 8)	$P_3 = 1 + 2*2 + 2*8 + 5 = -6$
3	-6	(3, 7)	(4, 7)	$P_4 = -6 + 2*3 + 3 = 3$
4	3	(4, 7)	(5, 6)	$P_5 = 3 + 2*4 - 2*7 + 5 = 2$
~~5~~	~~2~~	~~(5, 6)~~	~~(6, 5)~~	

Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8
(0,8)	(8,0)	(8,0)	(0,-8)	(0,-8)	(-8,0)	(-8,0)	(0,8)
(1,8)	(8,1)	(8,-1)	(1,-8)	(-1,-8)	(-8,-1)	(-8,1)	(-1,8)
(2,8)	(8,2)	(8,-2)	(2,-8)	(-2,-8)	(-8,-2)	(-8,2)	(-2,8)
(3,7)	(7,3)	(8,-3)	(3,-7)	(-3,-7)	(-7,-3)	(-7,3)	(-3,7)
(4,7)	(7,4)	(8,-4)	(4,-7)	(-4,-7)	(-7,-4)	(-7,4)	(-4,7)
(5,6)	(6,5)	(6,-5)	(5,-6)	(-5,-6)	(-6,-5)	(-6,5)	(-5,6)