Cabin H26

LECTURE NO 1 8-02-18

Books

Project

Android app

user panel admin panel

login

password

camera open - scan bar code - add attendance

- How to build android app
- Barcode detection
-

Pillars of Computer Graphics
- Imaging
- modeling
- Rendering
- Animation

open CV

aamesh.mahmood@ciitlahore.edu.pk

pdf drive.net

Book

- Fundamental of compute graphics 2nd edition
- computer graphics with open GL ← Reference Book

Pipeline / Pillars.

Imaging:-
working in 2D model.

Modeling:-
working in 3D Model.

Rendering:- of on
Projection, 3D model to 2D screen.

Animation

LECTURE No 2

14-02-18

Pillars of Computer Graphics:-

- ✓ Imaging:-
  Representing 2D images.
- ✓ Modeling:-
  Representing 3D object.
- ✓ Rendering:-
  constructing 2D image from 3D model
- ✓ Animation:-
  Simulating changes over time.
- ✓ Application
  entertainment
  computer aided design (Modulation of anything contains specific properties)
  scientific visualization
  Training (Training via simulators)
  education
  E- commerce
  Computer

LECTURE NO.3.

15-02-18.

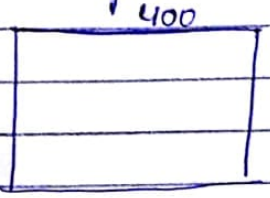
Raster Systems

- Basic unit of Raster Systems are pixels
- cubic - specific dimentions. (two dimentions x & y)
- examples:.

monitor- TV screen, mobile etc.

Resolutions :-

- Total number of pixels present on a Raster system is known as the resolution of raster system.



- Single image present on raster system → frames.

Greater the number of frames/sec higher the quality of picture.

How many frames are processed in Raster System
└ images present in Raster System.

- every pixels has definite integer values.

⇒ Algorithms

- Line Drawing Algorithms.
-
-

LINE DRAWING ALGORITHM :-

$y = mx + c$

↓ ↓

Slope(θ) y intercept

PROBLEM

1) we get float values as a result of which we have to use round off functions which increase the processing time.

2) We cannot get exact values in case of float
e.g 1.1, 1.4, 1.2 ≈ 1 etc.

3) Multiplication of m & x is not desirable. It slows down the processor.

Processing time of multiplication > addition

Value of m is also in float

S

Visual Studio 2010. (preferable)
open cv (how to integrate)
↳ youtube video

LECTURE NO 4

21-02-18:

DDA vimp 5 marks question in paper

Digital Differential Analyzer

$\Delta y = y_2 - y_1$

$\Delta x = x_2 - x_1$

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

int y = y1

if (m<1)

library

#include graphics.h

Draw_pixel (x1, y1, RGB_value)

0-255

x coordinate y coordinate

for (x1 = 0; x1 <= xf; x++) {

x1 = x1 x <= xf final value of x

y = y + m;

Draw_pixel (x1, (Round) y, RGB);

x, (Round) y, RGB

}

output

zigzag straight line.

1) value of m is float

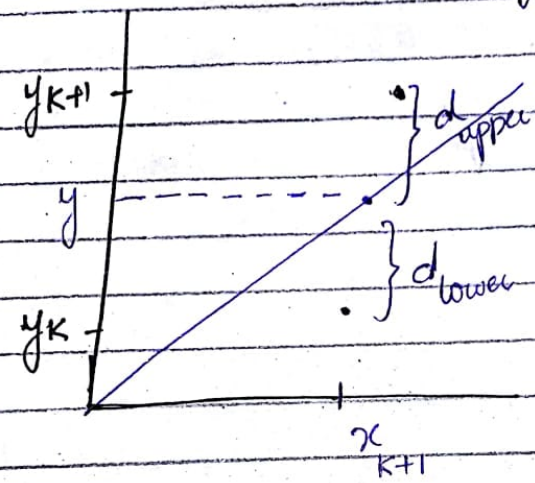
2) Result is in float.

3) Only sol is that it solves problem of mul

Q Why we have to face problems while drawing line using line drawing or why we use DDA or what is a problem.

BRESENHAM'S Midpoint Line Drawing Algo.

- The main problem is solving floating values
- Takes great processing time.
- Design algo in such a way that we don't have to solve any floating value.
- automatically select y_k or y_{k+1}, only condition is that there is no floating value.



$$y = mx + c$$

$$y = mx_k + c$$

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

$$d_{upper} = y_{k+1} - y$$

$$= y_{k+1} - m(x_k + 1) - c$$

$$d_{lower} = y - y_k$$

$$= m(x_k + 1) + c - y_k$$

$$d_{lower} - d_{upper} = m(x_k+1) + c - y_k - y_k - 1 + m(x_k+1) + c$$

$$= 2m(x_{k+1}) + 2c - 2y_k - 1$$ constat

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

$$\Delta x(d_{lower} - d_{upper}) = 2\Delta y(x_{k+1}) - 2\Delta x y_k + A\Delta x$$

P_k

will never gives floating value

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

decission parameter will determine whether y pixel will go up or down.

if $P_k < 0$

then y_k because the distance of $y_{upper} > y$

and $y < y_{lower}$ so (x_{k+1}, y_k)

if $P_k > 0$

(x_{k+1}, y_{k+1})

LECTURE NO 5

22-02-18.

$= 2\Delta y(x_{k+1}) - 2\Delta x y_k + A\Delta x$

$P_k = 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + A\Delta x.$

$P_k = 2\Delta y x_k - 2\Delta x y_k + \underbrace{A\Delta x + 2\Delta y}_{\text{Constant } Z}$

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

$P_{k+1} - P_k = 2\Delta y x_{k+1} - 2\Delta x y_{k+1} + Z - 2\Delta y x_k + 2\Delta x y_k - Z$

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

$P_{k+1} = 2\Delta y - 2\Delta x(\underbrace{y_{k+1} - y_k}_{0/1.}) + P_k$

Note

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

e.g $5 - 4 = 1$

Case1 $P_k < 0$

x_{k+1}, y_k

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

- if $P_k < 0$ choose x_{k+1}, y_k $\therefore y_k - y_k = 0.$

Case2 $P_k > 0$

x_{k+1}, y_{k+1}

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

- if $P_k > 0$ choose y_{k+1} $\therefore y_k 1$

Assignment.

Convert these pseudocode into proper program and also show output.

Example

(20,10) (30,18)

if $m<1$

1) Take values of (x_0, y_0), (x_f, y_f)

2) Draw (x_0, y_0)

3) Calculate all constants

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

$= 16 - 10$

$= 6.$

4) if $P_K < 0$

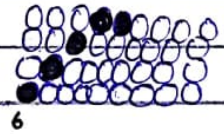
(x_{k+1}, y_k)

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

if $P_K > 0$

(x_{k+1}, y_{k+1})

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



Extra

if $m<>1$

- select P_K, don't draw it
- just replace value of x_{k+1} with y_k and y_k with x_{k+1}.

swap.

e.g (2,4) ⇔ (4,2)

5) Repeat ~~the~~ step 3 & 4 till $(x,y) = (x_f, y_f)$

Extra

if $x_f < x_0$ and $y_f < y_0$ then ~~ist~~ instead of use x_{k+1} and y_{k+1} we'll use x_{k-1} & y_{k-1}.

Bresenham's Circle Drawing Algorithm.

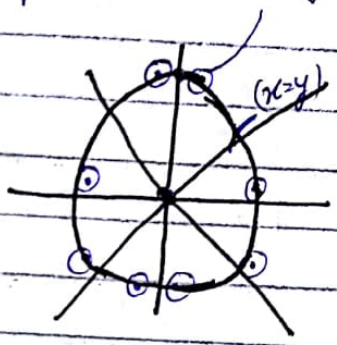
- Divide Raster system into 8 parts.

$(x,y) \approx (2,4)$

$x^2 + y^2 = R^2$

$y = R$

$x = 0$



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

Repeat till ~~xx~~ $x = y$ or $m\left(\frac{\Delta x}{\Delta y} = 1\right)$

Note

decision on the basis of (x_{k+1}, y_k) OR (x_{k+1}, y_{k-1})

LECTURE No 6

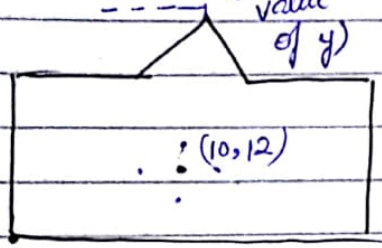
Polygon Filling Algorithm.

↳ Shape formed By the connection of three or more lines.

1) 4 connectivity

edges = lines

vertex :- connection of edges in a point
- connected point



Limitation

1) Different colors of edges and filling inside.

intensity value :- RGB value.

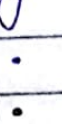
R G B

0 0 255 for Blue.

if uchar (x, y, 0) == 255 && (x, y, 1) == 0 && (x, y, 2) == 0

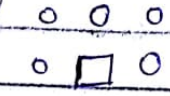
open cv
BGR

1) Give color to background (edges)
2) Give check whether if pixel color is background and foreground or not
if not then assign foreground color.



1) Select one pixel, make four connections
(x_k, y_{k+1}), (x_{k+1}, y_k), (x_{k-1}, y_k),
(x_{k+}, y_{k-1})

2) 8-connectivity
- Select foreground & background
- Make 8 connection of a pixel at any point.

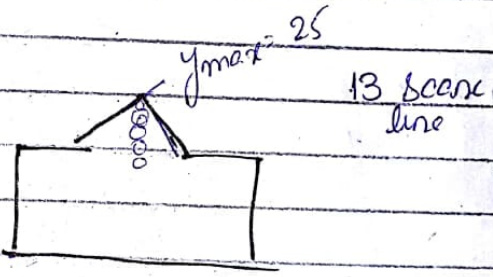


- check for foreground and background color.

if(...., 0==0 && ...., 1==0 && ... 2==0)
↑
Block.

3) Scanline Filling.

1) Take size of polygon as input



xmin, ymin, xmax, ymax

2) Calculate no of scanline passing through this polygon. — L
travelling of pixel from left and right

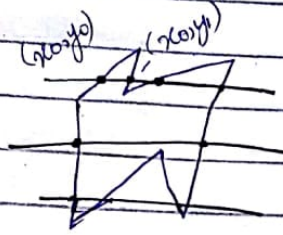
scane line = $y_{max} - y_{min}$

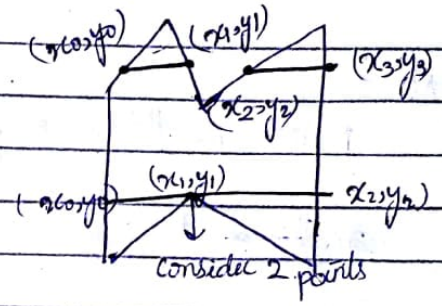
Retracing → Remove dump value

Reducing - vertical (edge to edge & jump to top right)

- horizontal. (from left to right & jump to left)

3) if scaneline passes through the edges this will always gives even no of intersection points





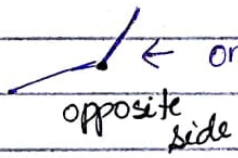
if scaneline passes through the vertex then the no of intersection point depends upon the position of a vertex.

either even or odd

Note

We want even no of intersection point

4) While forming the vertex both lines are at same side then we consider only one point of intersection



on the other hand if lines are not at same side (one increasing other decreasing) then consider two intersection point

LECTURE NO 6

7-03-18.

Resolution

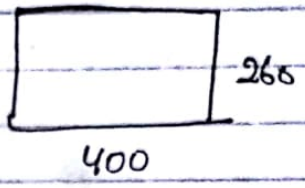
Total number of pixel passed on a raster system

Aspect Ratio

$$= \frac{\text{height}}{\text{width}} = \frac{H}{W} = \frac{\text{Rows}}{\text{columns}}$$

Note

Never in floating point



$$\frac{260}{400} = 13:20.$$

Frame Buffer:-

temporarily Memory.

image = Frames

e.g

30 fps ⇒ 30 images in one sec.

- images/frames are stored in ~~xxx~~ frame buffer.
- temporarily memory which contains the intensity value of pixel of next frame
- which is in KB, MB etc.

Intensity value:-

RGB value of pixel stored in binary value in 8 bits

- greater/higher the fps real the animation would be

Animation:- 29.7 fps ⇒ Real Animation
higher than 29.7 ⇒ there is no difference.

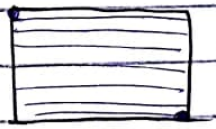
4) Scan Conversion.
Process in which we convert intensity value of pixel in binary and stored them into frame buffer. for futhur processing.

5) Retracing:- /Refresh:
Process in which remove value of old pixel and then add new pixel from frame buffer.

- Movement of scanline. from left
- Remove old pixel.
- Add new pixel.
put

5.1 Horizontal Retracing.
- Scan line from left to Right
- Again come to left Position.

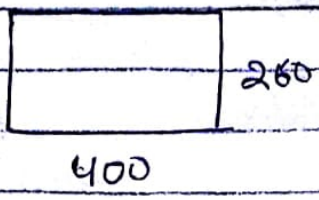
5.2 Vertical Retracing:-
- left top – Right bottom
- Again come to left top
- For whole single frame
- completion of frame.



Note
Random System.
- exact coordinate is noted.
- No movement of scanline.

Numericals

26 f/s



- value change = $400 \times 260 \times 26$

- frame in one sec = 26 $\Rightarrow$ frames in nano = 26×10^{-9}

one frame in sec = $\frac{1}{26}$

nano sec = $\frac{1}{26} \times 10^{-9}$.

Resolution = 400×260

25 f/s

intensity value = 4 bit.

Memory Required for frame buffer in sec = $\frac{4 \times 400 \times 260 \times 25}{8}$

$= 52000 \times 25$

$= 52 \times 10^3$ bytes

$= 52$ KB

$= 1300$ KB.

Note

when values like value of intensity level is not given then assume it.

25 fps ⇒ 4 bit.

Time Required to Refresh one Row:-

$$\frac{1}{260 \times 25} = 1.5 \times 10^{-4}.$$

Q How much time is required in scaling one Row.

⇒ Refresh 4 columns.

Total:- $400 = \frac{4}{400}$ (gives four columns)

$$\frac{4}{400 \times 25} = \frac{0.0004 \times 10^6}{10^6}$$

$$= 400 \times 10^{-6}$$

Note

There is no columnwise (vertical) refreshing.