

A cluster of light blue circles of various sizes, resembling bubbles or a molecular structure, located in the upper right corner of the slide.


Computer Graphics

&

Virtual Reality

Vaishali K.

Marks Distribution



• Theory	– 80 Marks
• Practical	– 25 Marks
• Internal Assessment	– 20 Marks
• Term Work	– 25 Marks
<hr/>	
• Total	-- 150 Marks

Syllabus

- Introduction to Computer Graphics & Output primitives
- Area Filling & Two Dimensional Transformation
- Two Dimensional Viewing
- Three Dimensional Transformation, Viewing and Projection

Vaishali K.

Syllabus

- Introduction to Animation
- Introduction to Virtual Reality
- Modeling
- Introduction to VR Programming



Text Books

- R. K. Maurya “ Computer Graphics with Virtual Reality” , Wiley India.
- Hearn & Baker “ Computer Graphic”
- Harrington “Computer Graphic”
- Vince “ Virtual Reality System”
- Burde & Coiffet “ virtual Reality Technology”

Vaishali K.

CHAPTER 1

Introduction to Computer Graphics & Output primitives



Vaishali K.

Introduction

- Computer graphic is a field of computer science that is concern with digitally synthesizing & manipulating visual contents.
- A science and technology of creating , storing displaying and manipulating image and object

Image & Objects

- Objects are real entities defined in three dimensional (3D) world coordinates
-
- Image is 2 dimensional (2D) representation of 3D object.

Applications of Computer Graphics



- Digital Art
- Special Effect
- Visual Effect
- Video Games
- Computer aided design
- Medical Imaging

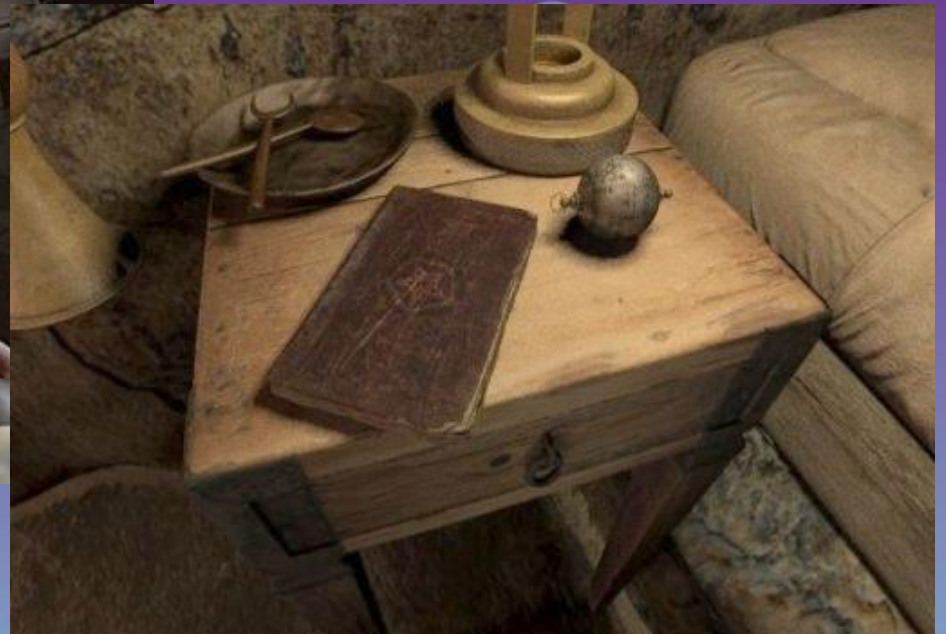
Vaishali K.

Computer Graphics is about animation (films)



Major driving force now

Games are very important in Computer Graphics



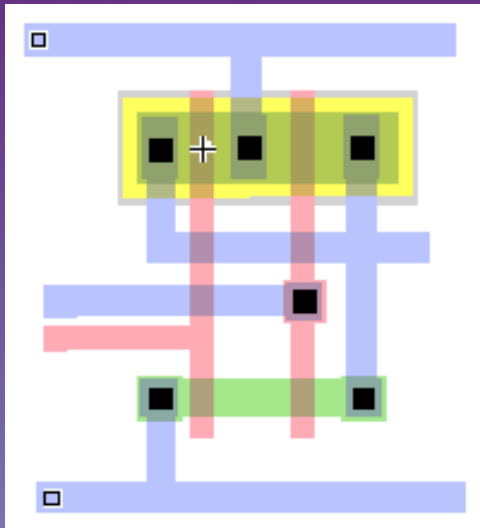
Medical Imaging is another driving force



Much financial support

Promotes linking of graphics with video, scans, etc.

Computer Aided Design too



Scientific Visualisation

To view below and
above our visual range

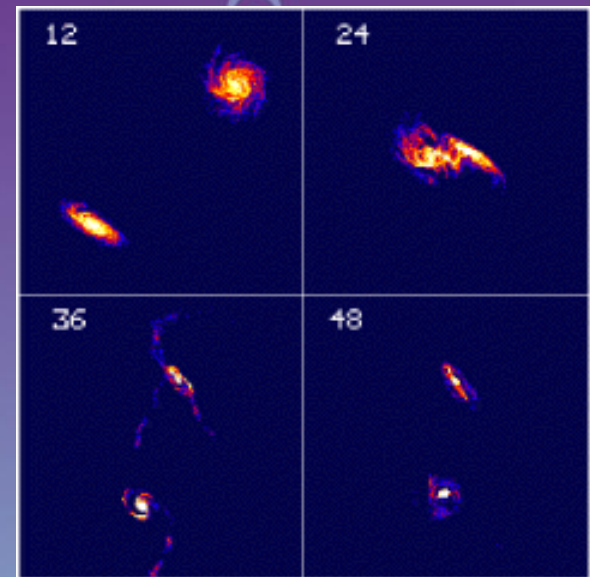
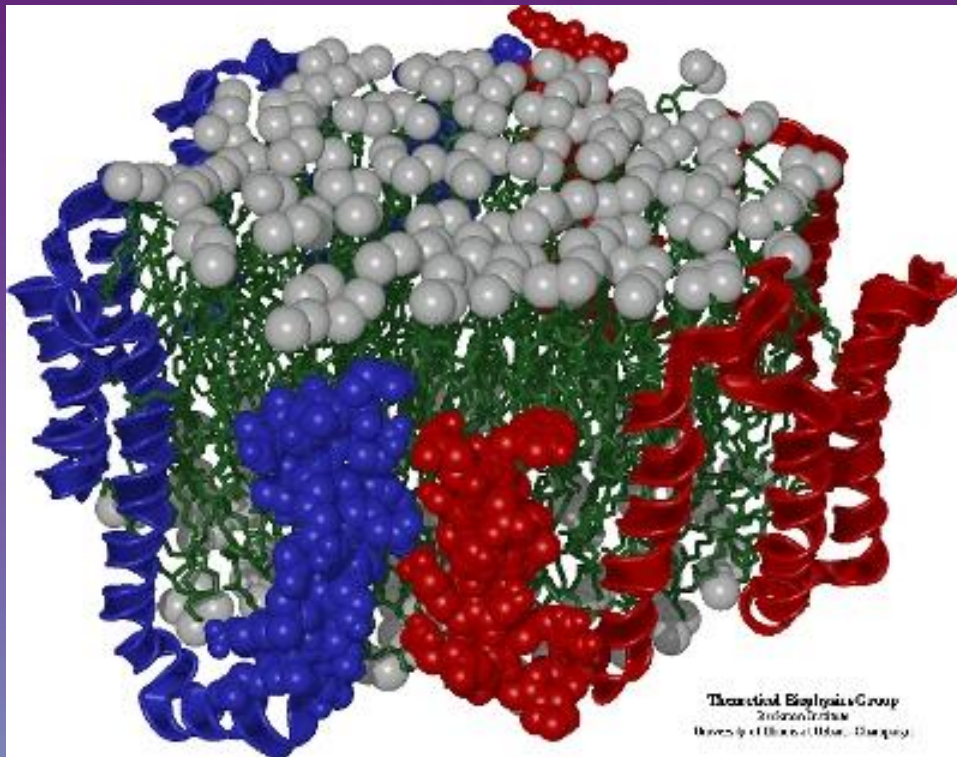
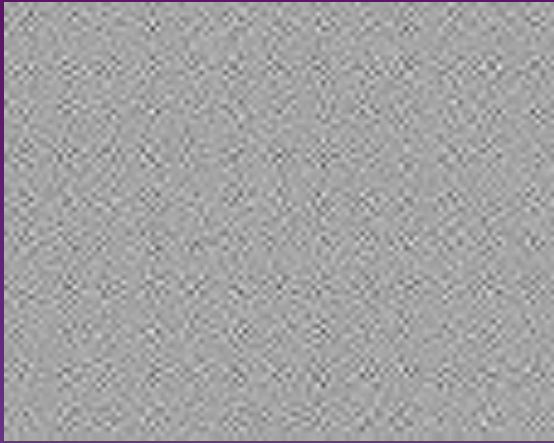


Image Representation

- Picture is a collection of picture elements called ***pixels***
- A Pixel is smallest addressable screen element.
- The process of determining the appropriate pixel for representing picture or object is called ***Rasterization***
- Process of representing continuous pictures as a graphical object is called ***Scan Conversion***.



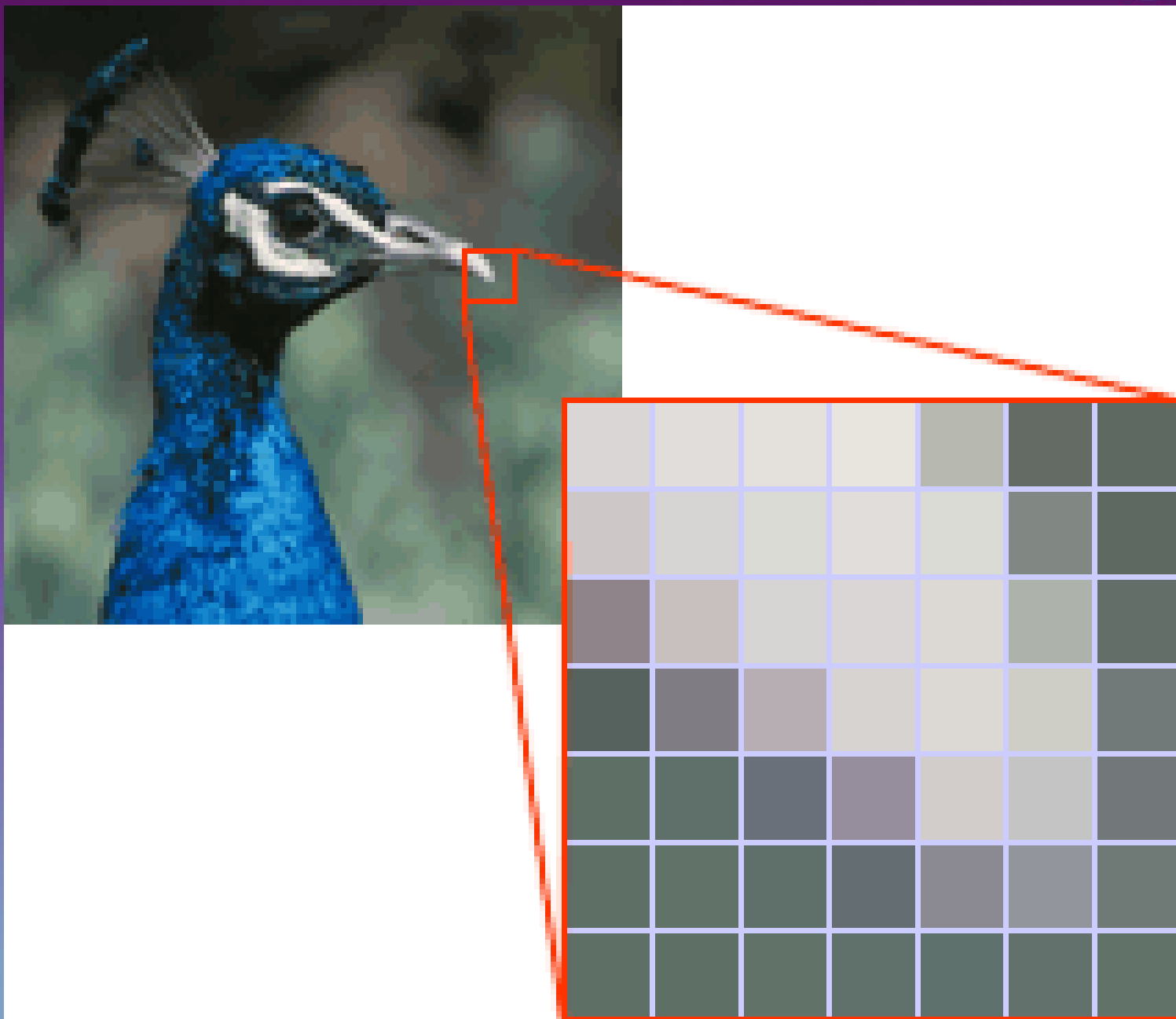
Vaishali K.

Bitmap vs vector based Graphics

- Raster Graphic or Bitmap Image – composed of pixel
- Vector Graphics – composed of path

Bitmap vs vector based Graphics

- Bitmap graphics are composed of pixels, each of which contains specific color information. A pixel is minutely small; a single image may be composed of hundreds of thousands of individual pixels. Much like cells revealed from a piece of tissue when seen under a microscope, these pixels are only clearly and individually visible when the image is magnified



Bitmap vs vector based Graphics

- Bitmap images are created and edited in Adobe Photoshop
- Bitmap images are mapped to an array of pixel
- The size of an image is based on the image resolution
- Images are not easily scalable
- It contains complex color variation.

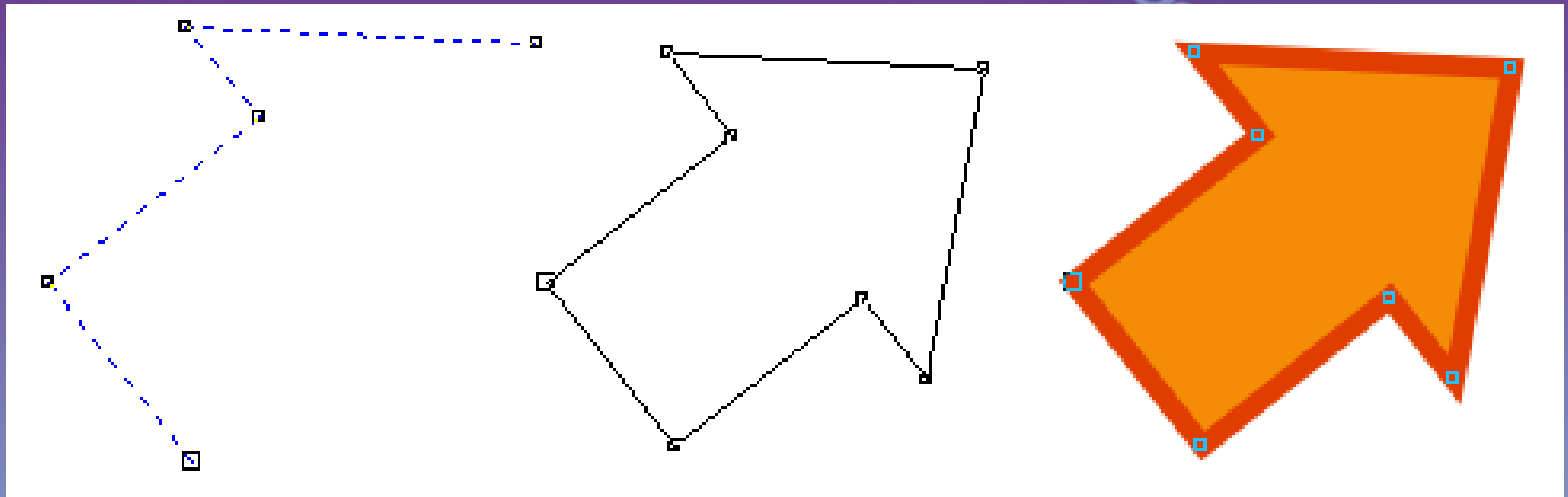
Vector Graphics


- Vector graphics are mathematical creations . Vector graphics consist of points, lines, and curves which, when combined, can form complex objects. These objects can be filled with solid colors, gradients, and even patterns.

-



- These objects can be filled with solid colors, gradients, and even patterns.



- 
- Vector based images are created and edited CoralDraw
 - These images have smooth edges and create curves or shapes.
 - Vector based images are good for precise illustration
 - Images are easily scalable.

Graphics Definitions

- Point
 - a location in space, 2D
 - sometimes denotes one pixel
- Line
 - straight path connecting two points
 - infinitesimal width, consistent density
 - beginning and end on points



Graphics Definitions



- Vertex
 - point in 3D
- Edge
 - line in 3D connecting two vertices
- Polygon/Face
 - arbitrary shape formed by connected vertices
 - fundamental unit of 3D computer graphics

Basic Graphic Pipeline

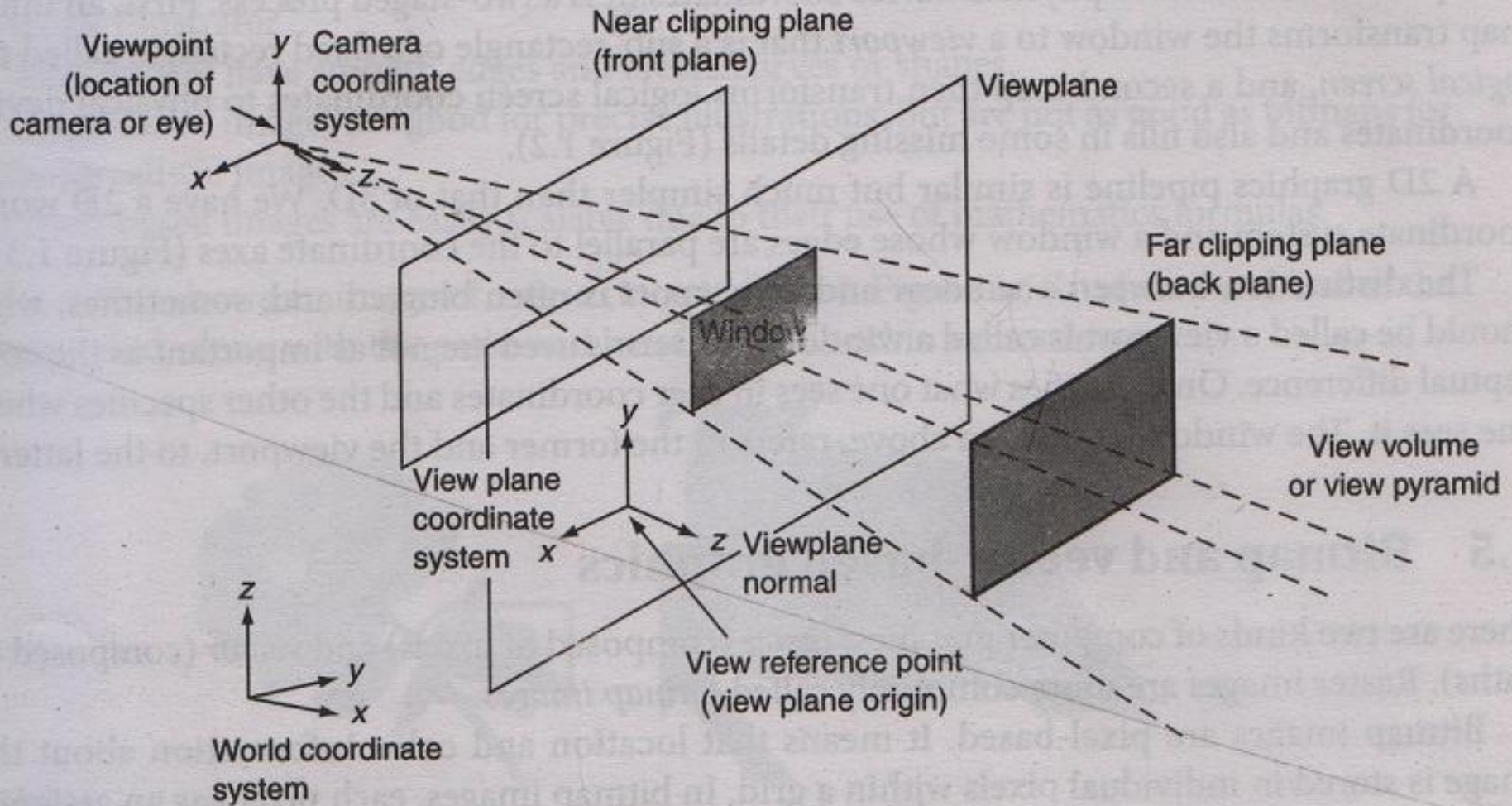


Figure 1.1 A 3D graphics pipeline.

Graphic Pipeline

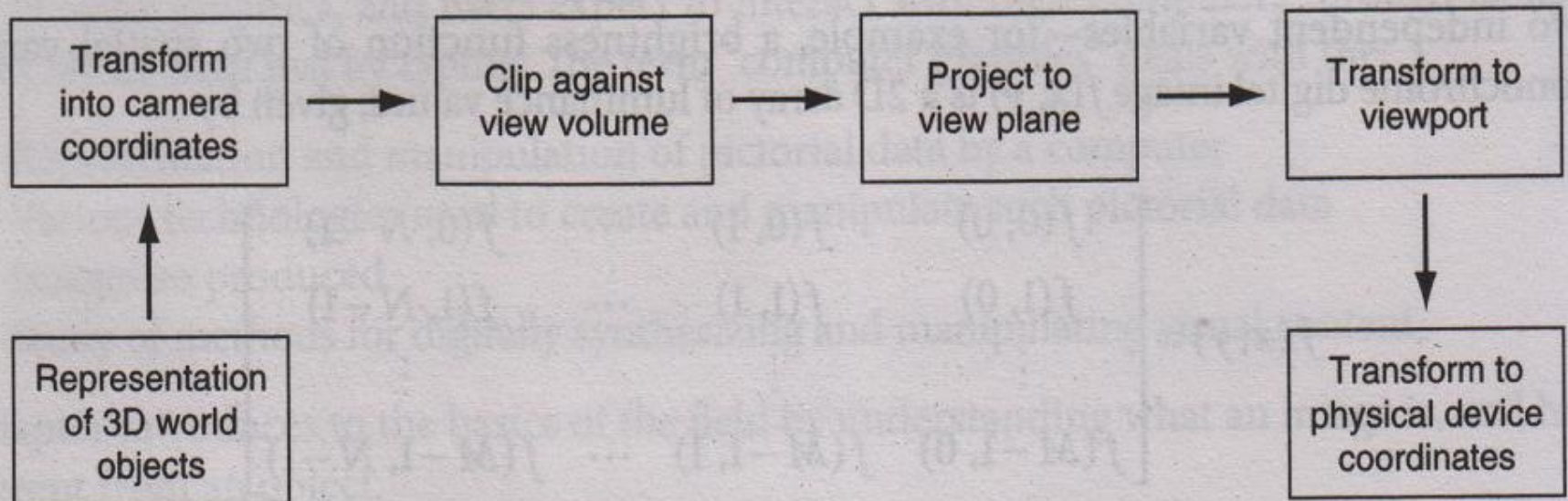


Figure 1.2 Transformation sequence in viewing pipeline.

Graphic Pipeline

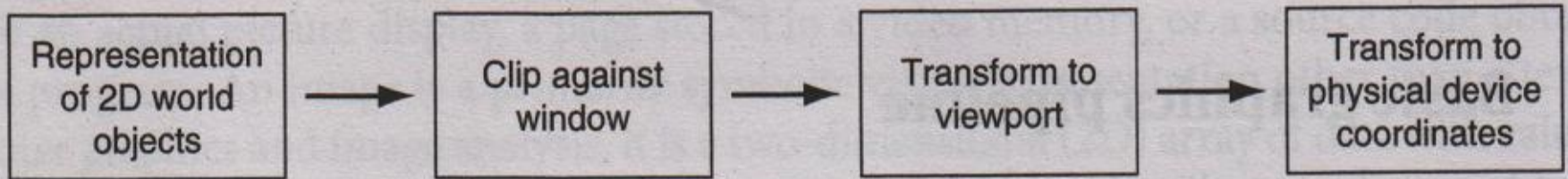


Figure 1.3 Transformation of a 2D coordinate system to physical device coordinates.

Display Devices

- Display Devices:
- Cathode Ray Tubes
- Raster Scan display Random Scan Display

Difference between raster Scan Display & Random Scan Display

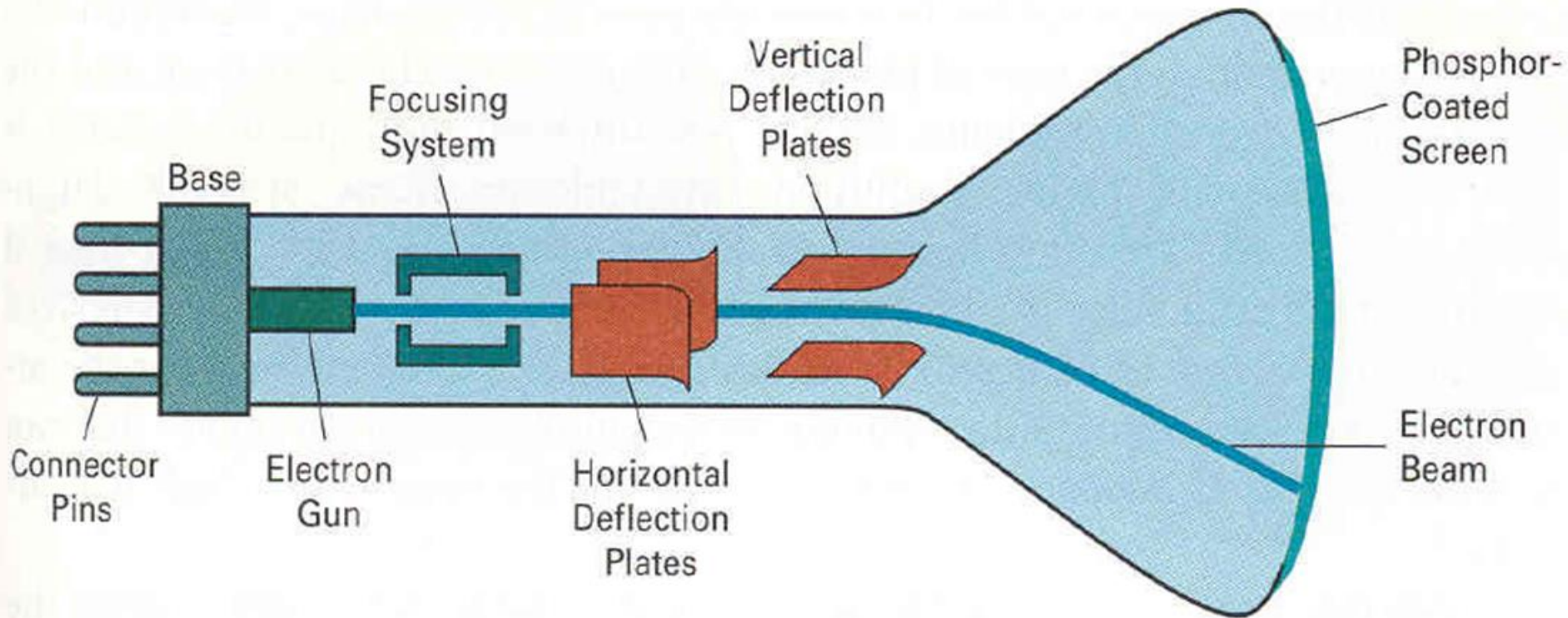
Input output Devices

- Output Devices
 - Volatile Displays
 - Static Flat panel Display
- Input Devices:
 - Touch Screen
 - Light Pen
 - Graphics Tablet



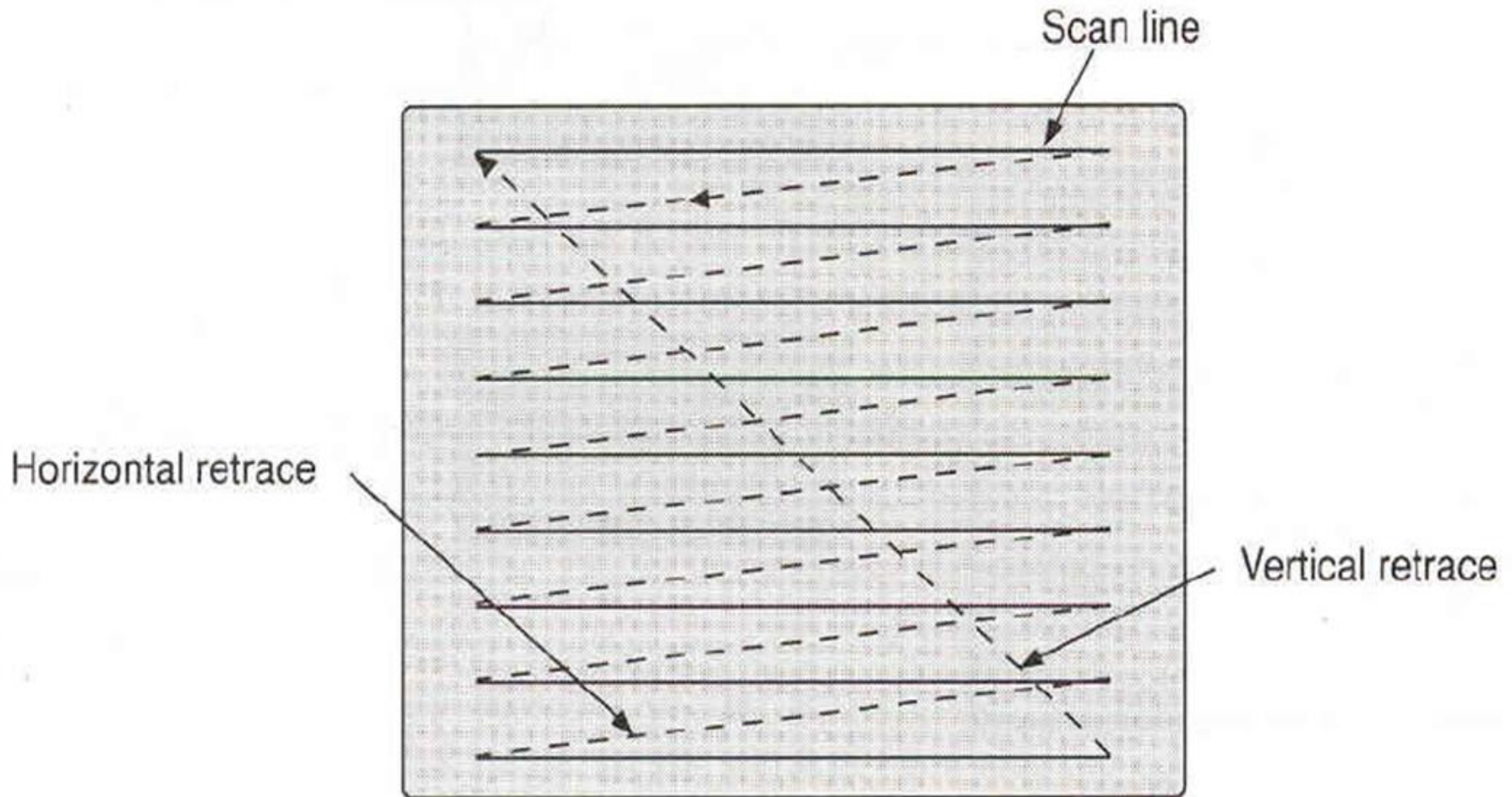
Display Device

Cathode Ray Tube (CRT)



Vaishali K.

Raster Scan Mechanism



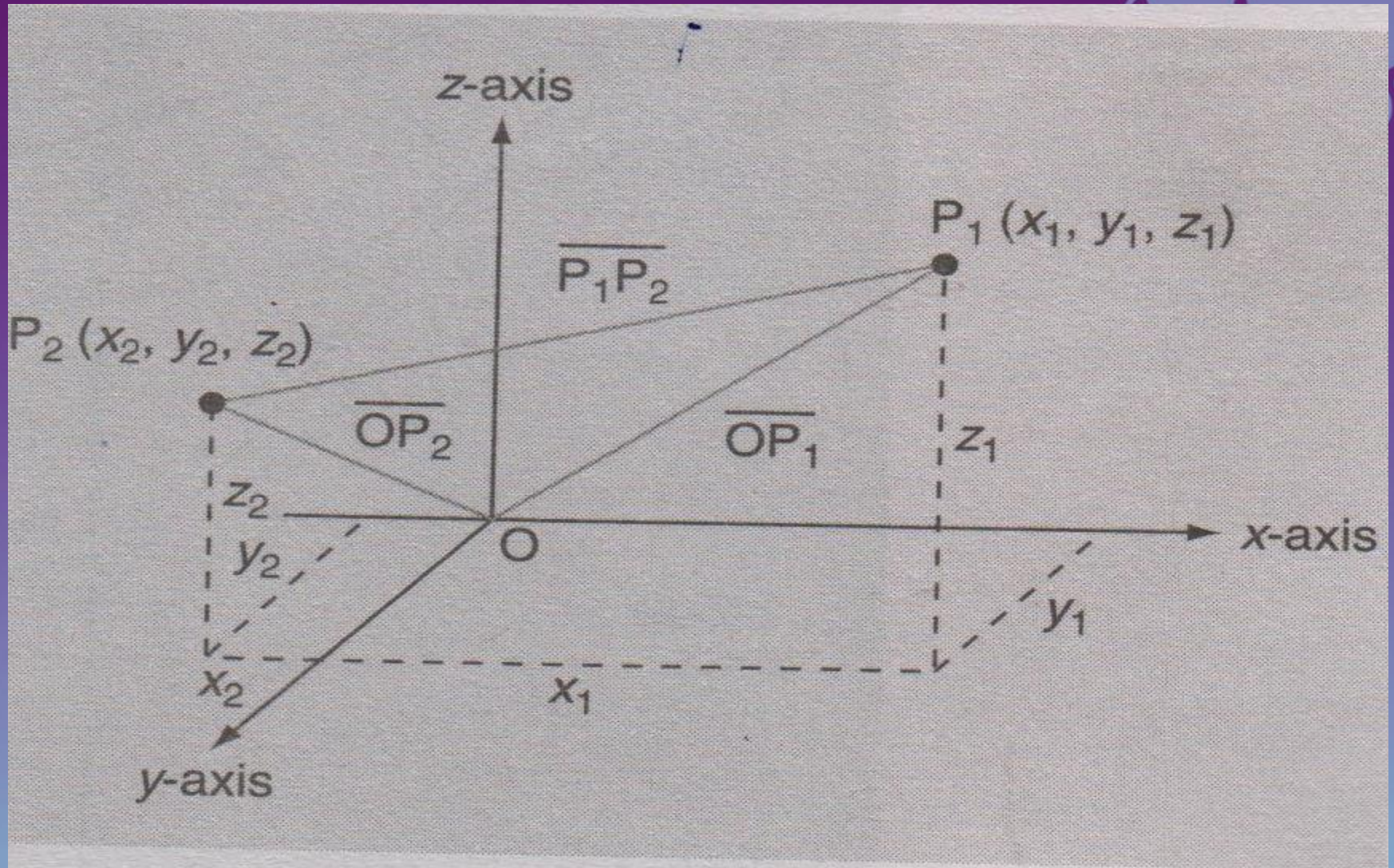
Vaishali K.

Difference

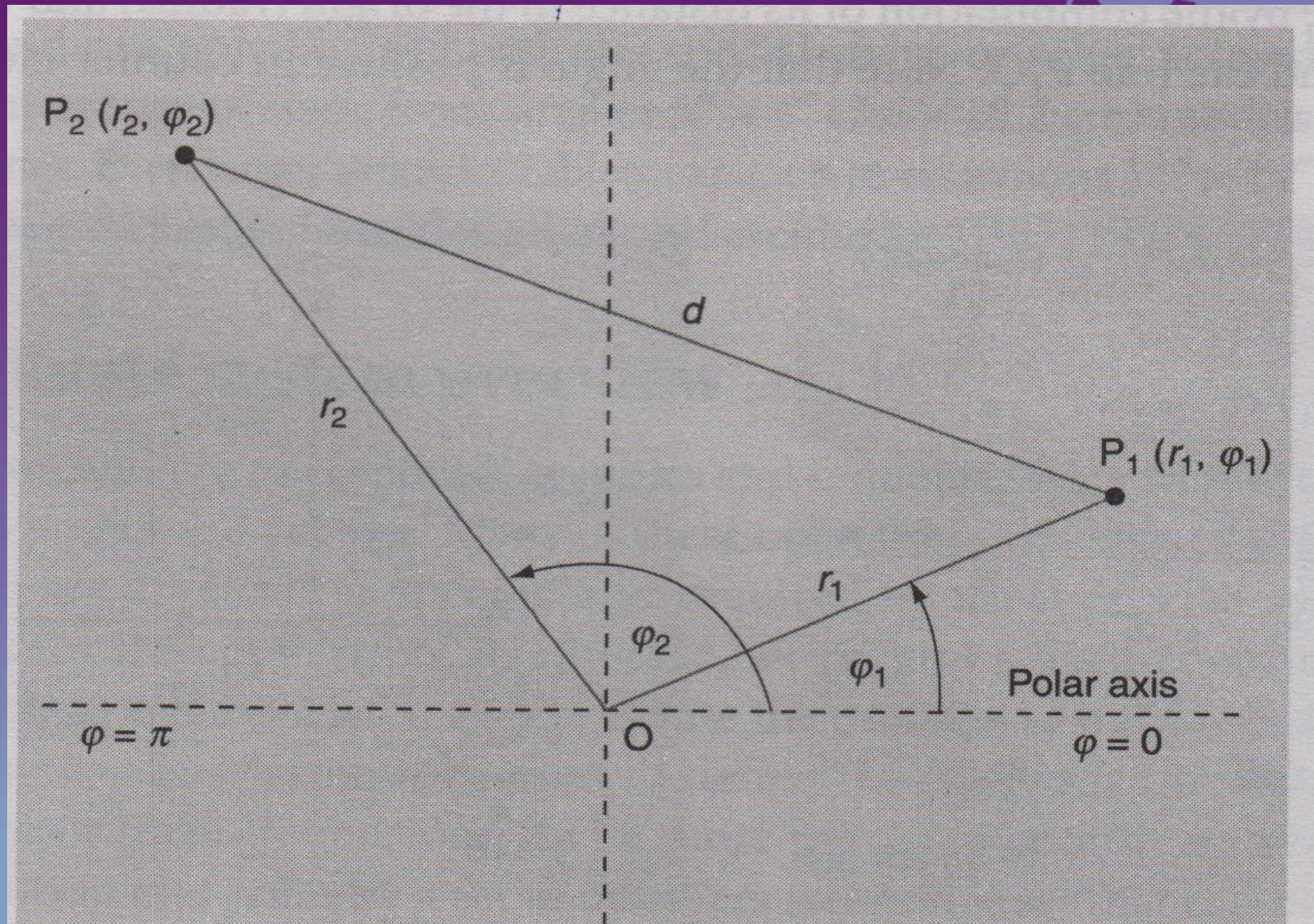


Raster Scan Display	Random Scan Display
It draws the image by scanning one row at a time	It draws the image by directing the electron beam directly to the part of the screen
They generally have resolution limited to pixel size	They have higher resolution than raster scan display
Lines are Jiggered and curves are less smooth	Line plots are straight and curves are smooth
They are more suited to geometric area drawing applications eg. monitors, TV	They are more suited to line drawing application Eg. CRO and pen plotter

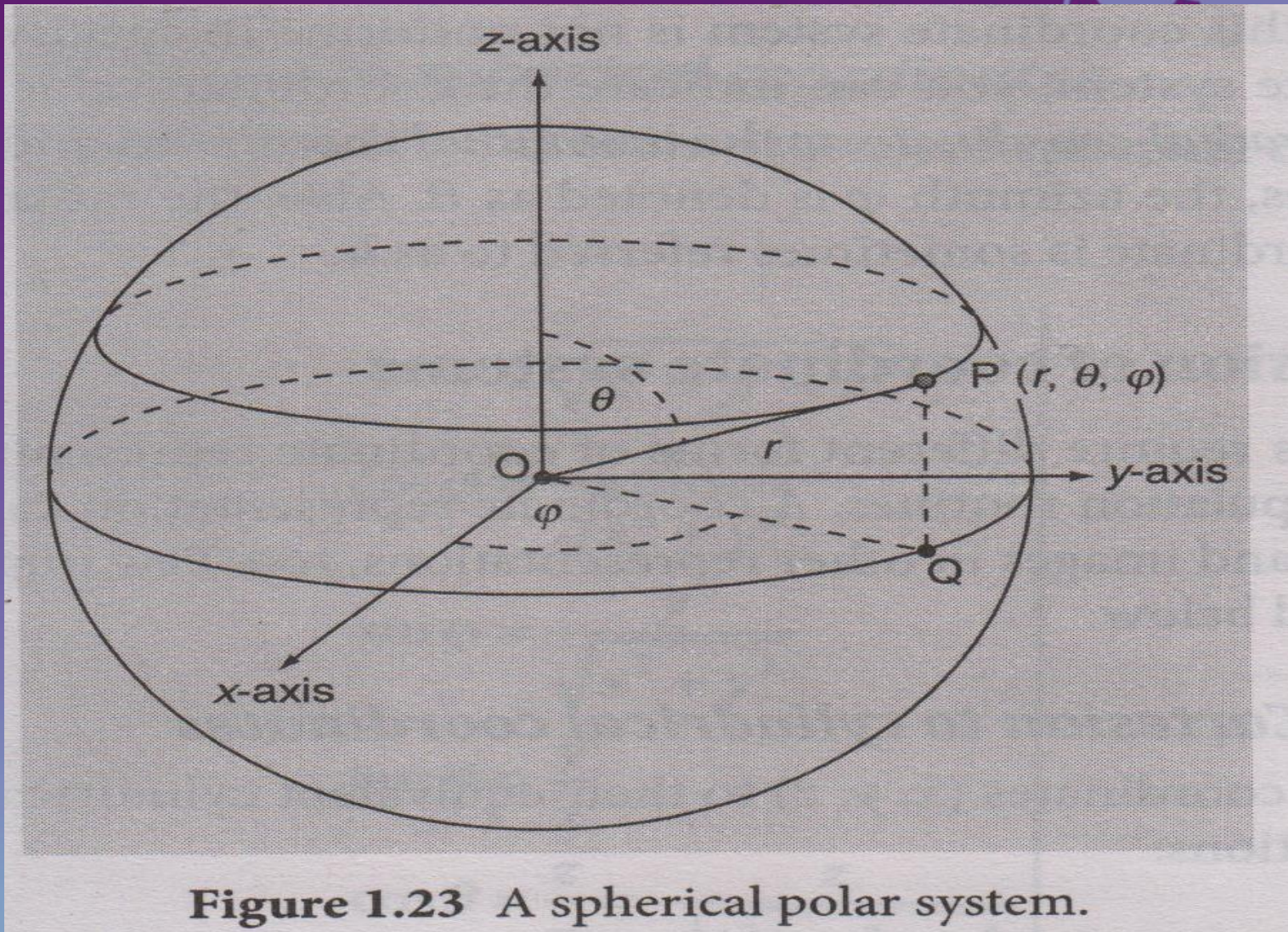
Cartesian coordinate system



Polar coordinate system



Spherical polar system



Cylindrical polar system

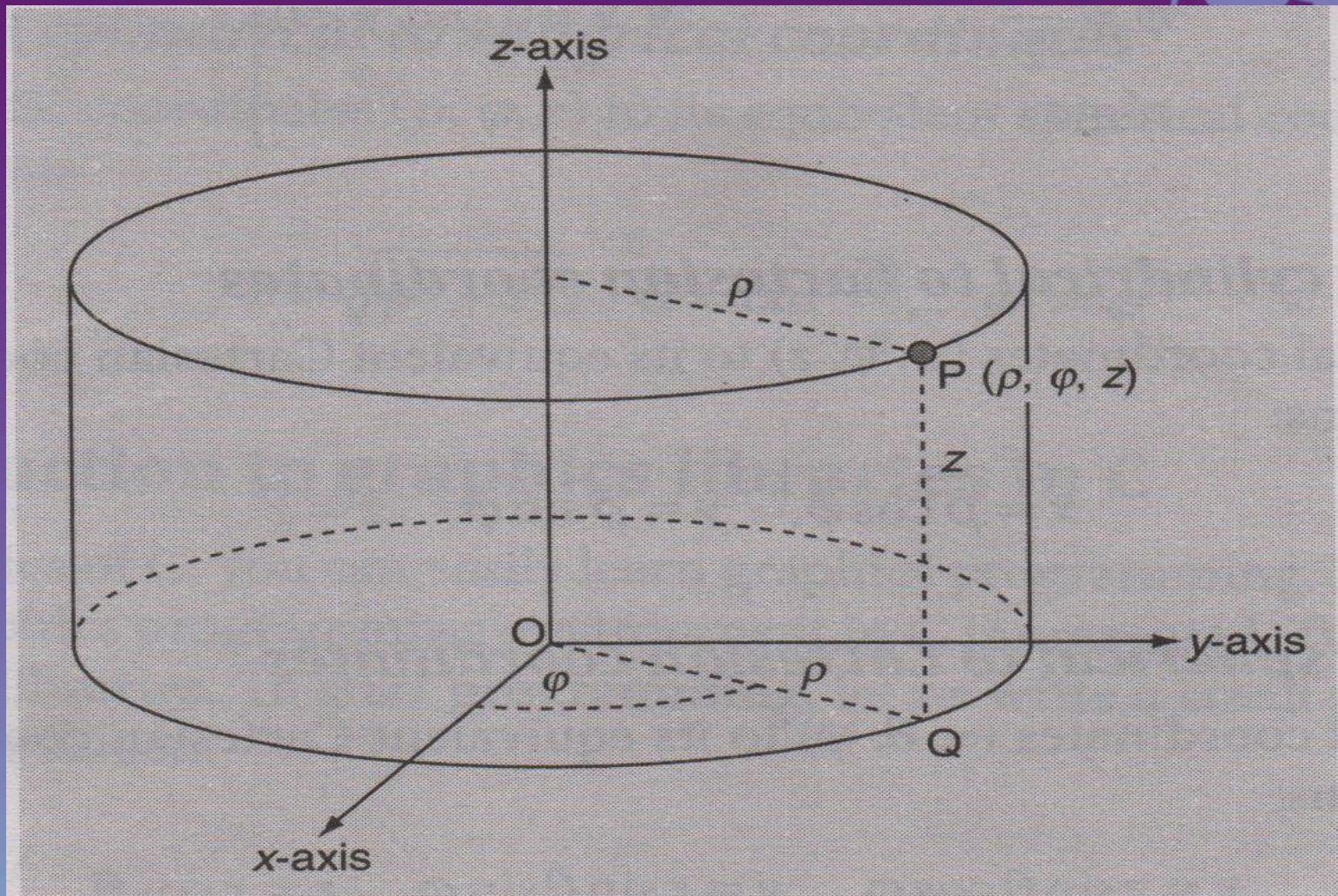


Figure 1.24 A cylindrical polar system.

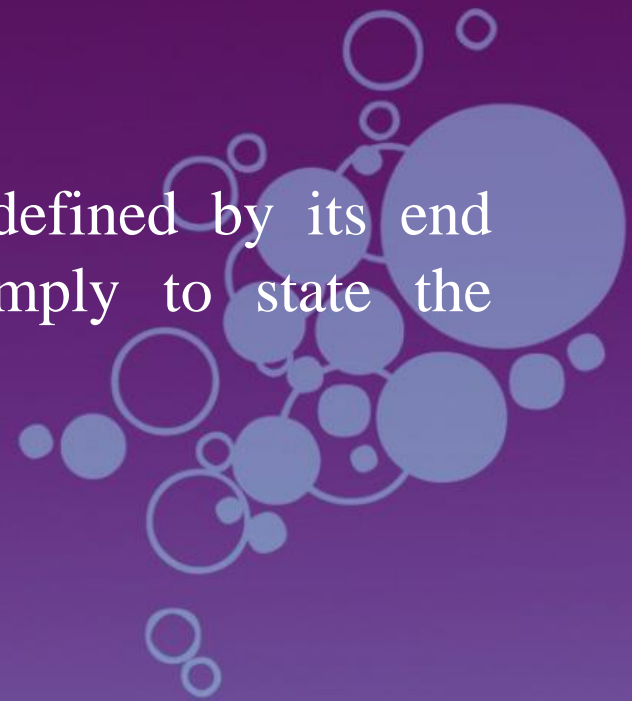
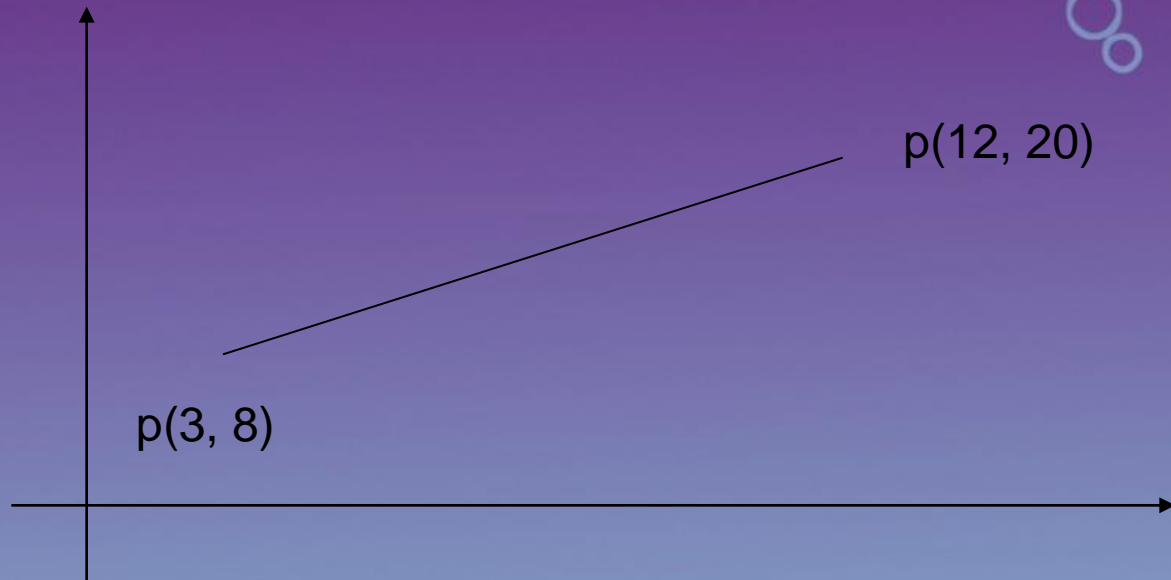
Scan conversion Algorithms

- Objects are real entities defined in three dimensional (3D) world coordinates
- Image is 2 dimensional (2D) representation of 3D object.
- A science and technology of creating , storing displaying and manipulating image and object

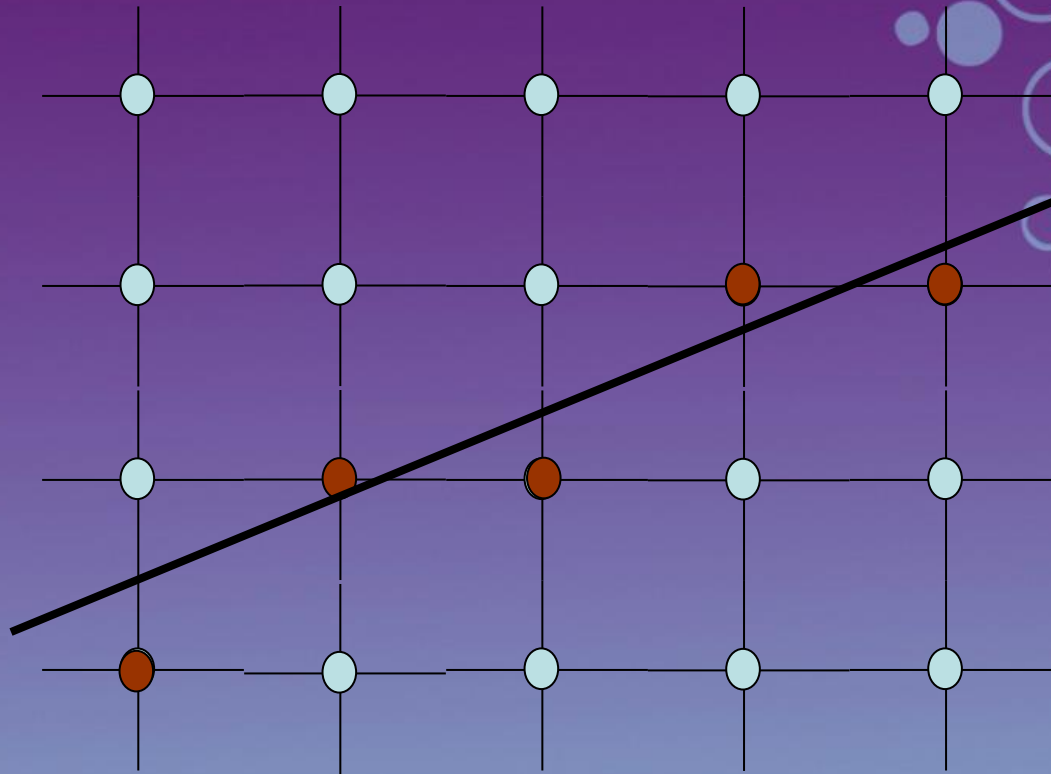
➤ Line Drawing

- A (straight) line can be mathematically defined by its end points. To describe a line we need simply to state the coordinates of the two ends.

e.g. - $(3, 8), (12, 20)$

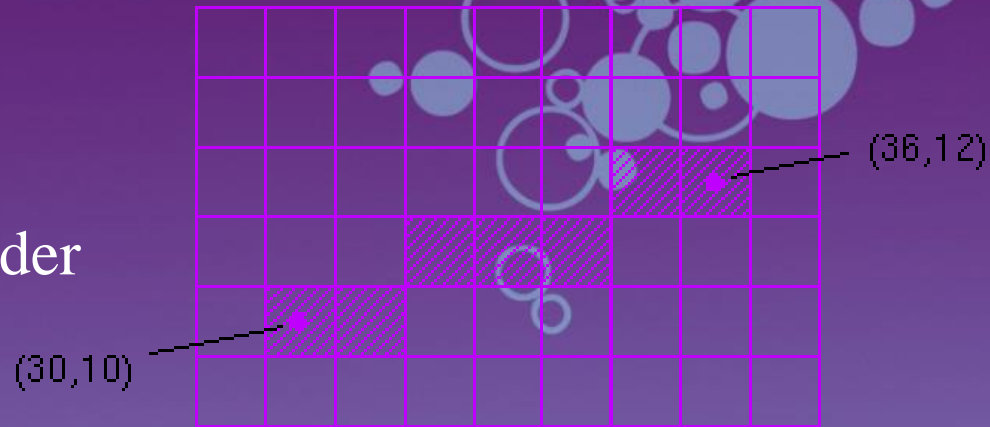


- Drawing lines on a raster grid implicitly involves approximation. The general process is called *rasterization* or *scan-conversion*.



- Lines should ideally have following properties

- straight
- pass through endpoints
- smooth
- independent of endpoint order
- uniform brightness
- brightness independent of slope
- efficient



➤ Line Drawing Algorithms

- Digital Differential Analyzer (DDA) Algorithm
- Bresenham's Line drawing algorithm



➤ Digital Differential Analyzer (DDA) Algorithm

- Takes an incremental approach in order to speed up scan conversion
- DDA is a scan conversion line algo based on calculating either Δx or Δy
- DDA is an algo for calculating pixel positions along a line. This is done by taking unit step with one of the coordinates & calculating corresponding values for other coordinates.
- (x_i, y_i) denotes current pixel coordinates, (x_{i+1}, y_{i+1}) denotes next coordinate pixels

Algorithm

1. Read both end-points as (x_1, y_1) & (x_2, y_2)

2. Calculate Δx & Δy as

$$\Delta x = |x_2 - x_1|$$

$$\Delta y = |y_2 - y_1|$$

3. If $(\Delta x \geq \Delta y)$ then

$$\text{length} = \Delta x$$

Else

$$\text{length} = \Delta y$$

End If

4.



4. $\Delta x = (x_2 - x_1) / \text{length}$
 $\Delta y = (y_2 - y_1) / \text{length}$

5. $x = x_1 + 0.5 * \text{sign}(\Delta x)$
 $y = y_1 + 0.5 * \text{sign}(\Delta y)$

6. $i = 1$
While ($i \leq \text{length}$)
{
Plot (Integer (x), Integer (y))
 $x = x + \Delta x$
 $y = y + \Delta y$
 $i = i + 1$

7. Stop



- #include<iostream.h>
- #include<conio.h>
- #include<math.h>
- #include<graphics.h>
- void main()
- {
- float x,y,x1,y1,x2,y2,dx,dy,len,steps;
- int i,gd,gm;
- clrscr();
- cout<<"enter the value of x1: ";
- cin>>x1;
- cout<<"enter the value of y1: ";
- cin>>y1;
- cout<<"enter the value of x2: ";
- cin>>x2;
- cout<<"enter the value of y2: ";
- cin>>y2;
- detectgraph(&gd,&gm);
- initgraph(&gd,&gm,"C:\\tc\\bgi");
- line(getmaxx()/2,0,getmaxx()/2,getmaxy());
- line(0,getmaxy()/2,getmaxx(),getmaxy()/2);
- if(x1==x2&& y1==y2)
- {
- putpixel(x+getmaxx()/2,getmaxy()/2-y,WHITE);
- }
- else
- {
- putpixel(x+getmaxx()/2,getmaxy()/2-y,WHITE);
- dx=abs(x2-x1);
- dy=abs(y2-y1);
- if(dx>=dy)



```
if(dx>=dy)
steps=dx;
else
steps=dy;

dx=(x2-x1)/steps;
dy=(y2-y1)/steps;
x=x1;
y=y1;
i=1;
while(i<=steps)
{
putpixel(x+getmaxx()/2,getmaxy()/2-y,WHITE);
x=x+dx;
y=y+dy;
i=i+1;
}

getch();
closegraph();
}
```



Advantages & Disadvantages

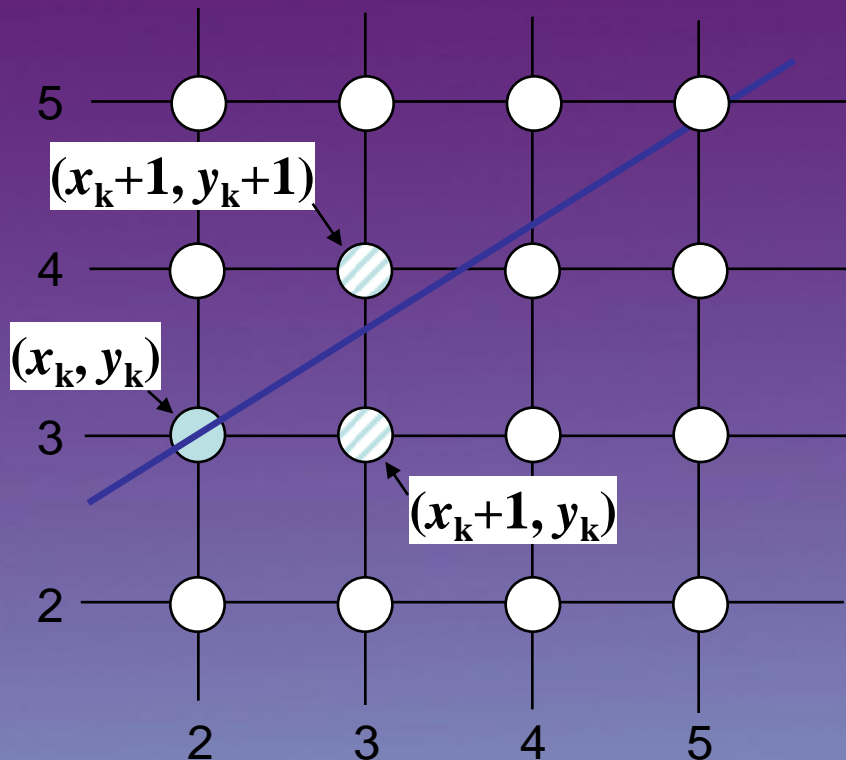
- Advantage:
 - It is the simplest algorithm and it does not require special skills for implementation.
 - It is faster method for calculating pixel positions than direct use of equation $y = mx + b$.
- Disadvantages:
 - The rounding operations and floating point arithmetic involved are time consuming



➤ Bresenham's Line Drawing Algorithm

- Bresenham's line drawing algorithm is another incremental scan conversion algorithm
- It uses only integer calculation hence it is more efficient
- Basic principle is to select optimal raster location to represent a straight line.
- To accomplish this the algo always changes either x or y by one unit depending on slope of line.
- The increment in other variable is determine by examining distance between actual line location & nearest pixel. This distance is called decision variable or error.

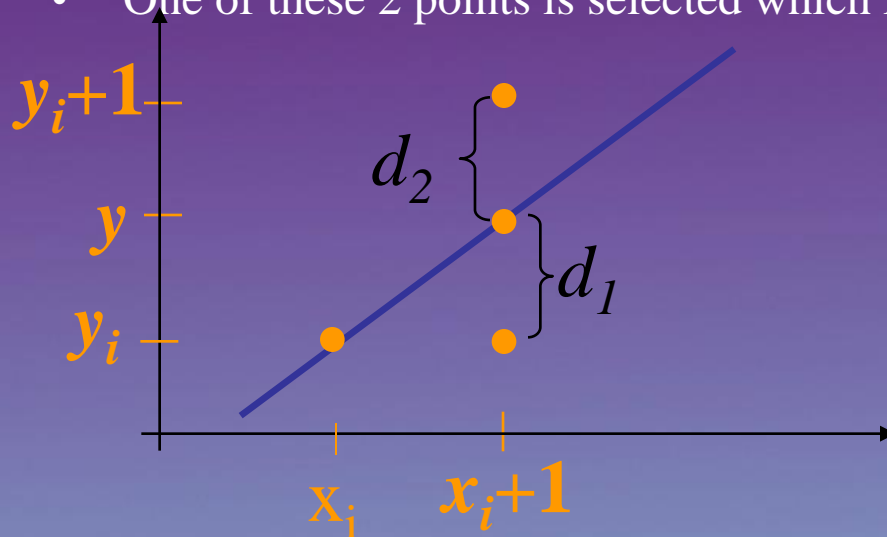
e.g. Move across the x axis in unit intervals and at each step choose between two different y coordinates



For example, from position $(2, 3)$ we have to choose between $(3, 3)$ and $(3, 4)$. We would like the point that is closer to the original line.

➤ Case I: $m \leq 1$, +ve slope

- Let us consider a line with +ve slope i.e. $m \leq 1$, therefore x is changed in unit steps.
- Let us assume that point (x_i, y_i) is drawn & we have to decide next point.
- We have 2 options , either (x_i+1, y_i) or (x_i+1, y_i+1) .
- One of these 2 points is selected which is nearer to actual position.
 - The y coordinate on mathematical line at pixel column position x_i+1 is defined as



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

Bresenham's Line Drawing Algorithm



1. Read both end points of a line as (x_1, y_1) (x_2, y_2)

2. Calculate Δx & Δy as

$$\Delta x = \text{abs}(x_2 - x_1)$$

$$\Delta y = \text{abs}(y_2 - y_1)$$

3. Initialize value of (x, y) as (x_1, y_1)

$$x = x_1, y = y_1$$

$$4 \text{ } e = \Delta y / \Delta x - (1/2)$$

5. $i = 1$

6. Plot the pixel

`putpixel(x, y)`

Vaishali K.

7. While ($e \geq 0$)

{

$y = y + 1$

$e = e - 1$

}

$x = x + 1$

$e = \Delta y / \Delta x + e$


8. $l = l + 1$

9. If ($i \leq \Delta x$) then goto step 6

10. Stop



1. Read the line end point (x_1, y_1) and (x_2, y_2) such that they are not equal.
2. $\Delta x = |x_2 - x_1|$ and $\Delta y = |y_2 - y_1|$
3. Initialize starting point
 $x = x_1$
 $y = y_1$
4. $s_1 = \text{Sign}(x_2 - x_1)$
 $s_2 = \text{Sign}(y_2 - y_1)$
 [Sign function returns - 1, 0, 1 depending on whether its argument is <0, = 0, > 0 respectively]
5. if $\Delta y > \Delta x$ then
 Exchange Δx and Δy
 Ex_change = 1
else
 Ex_change = 0
end if
 [Interchange Δx and Δy depending on the slope of the line and set Ex_change flag accordingly]
6. $e = 2 * \Delta y - \Delta x$
 [Initialize value of decision variable or error to compensate for nonzero intercept].
7. $i = 1$ [Initialize counter]
8. Plot (x, y)
9. while $(e \geq 0)$
 { if (Ex_change = 1) then
 $x = x + s_1$
 else



```
        y = y + s2
    end if
    e = e - 2 * Δx
}
10. if Ex_change = 1 then
    y = y + s2
else
    x = x + s1
end if
e = e + 2 * Δy
11. i = i + 1
12. if ( i ≤ Δx ) then go to step 8
13. Stop
```


THANK YOU



Vaishali K.