COMPUTER GRAPHICS
UNIT-I

R.MANIMEGALAI

DEPARTMENT OF COMPUTER SCIENCE

PERIYAR GOVT ARTS COLLEGE

CUDDALORE.

Objectives: To equip students to basics of computer drawing and prepare them for computer modelling of objects

UNIT – I : OVERVIEW OF GRAPHICS SYSTEMS AND OUTPUT PRIMITIVES

Video Display Devices- Raster Scan System- Random Scan Systems- Hard Copy Deices- Graphic Software- Line Drawing Algorithms: DDA- Bresenham's Line -Circle Generating Algorithms.

UNIT – II : ATTRIBUTES AND TWO DIMESIONAL TRANSFORMATIONS

Line Attributes- Curve Attributes-Color And Gray Scale Level- Area Fill Attributes- Character Attributes- Inquiry Functions- Basic Transformations - Composite Transformation – Other transformation

UNIT – III : TWO DIMENSIONAL VIEWING AND CLIPPING

The Viewing Pipeline- Window To Viewport Transformation –Clipping Operations- Point Clipping- Line Clipping: Cohen Sutherland- Liang Barsky-Sutherland Hodgeman

Polygon Clipping- Text Clipping- Exterior Clipping- Logical Classification Of Input Devices- Interactive Picture Construction

UNIT – IV : THREE DIMENSION TRANSFORMATION, VIEWING AND CLIPPING

Translation-Rotation-Scaling-Viewing Pipeline- Viewing Coordinates- Projections -View Volumes and General Projection Transformation- Clipping –

UNIT – V : VISIBLE SURFACE DETECTION METHODS

Classification of Visible Surface Detection Algorithms - Back Face Detection - Depth Buffer Method - A Buffer Method - Scan Line Method - Depth Sorting Method- BSP Tree Method -Area Sub Division Method - Octree Methods - Ray Casting Method

TEXT BOOK:

Computer Graphics(C version) , Donald Hearn and M.Pauline Baker, Pearson- 2nd Edit. 2012.

REFERENCE BOOKS:

1. Interactive Computer Graphics–A top down approach using Open GL, Edward Angel , Pearson, 5th Edition.

2. Computer Graphics, Peter Shirley, Steve Marschner, Cengage Learning, Indian

Video Display Devices

Primary output Device – <u>Video Monitor</u>

Video monitors are based on standard Cathode ray tube (CRT) design.

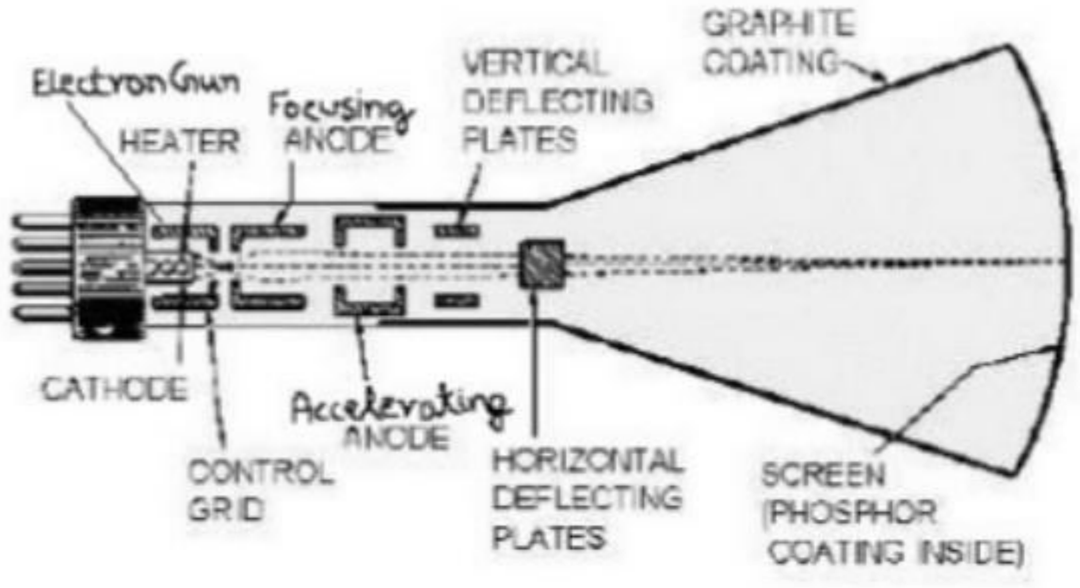
<u>Refresh Cathode Ray Tubes:-</u>

<u>Working</u>

- A beam of electrons (cathode rays), emitted by an electron gun, passed through focusing & deflecting system that direct the beam towards specified positions on the phosphor coated screen.

- The phosphor emits a small spot of light at each position contacted by the electron screen.
- For maintaining the screen picture glowing redraw the picture repeatedly by quickly directing the electron beam back over the same points. This type of display is called a **<u>refresh CRT.</u>**

Basic Design of a CRT and Operation of an electon gun with an accelerating anode.


Electron Gun
HEATER
Focusing ANODE
VERTICAL DEFLECTING PLATES
GRAPHITE COATING
CATHODE
CONTROL GRID
Accelerating ANODE
HORIZONTAL DEFLECTING PLATES
SCREEN (PHOSPHOR COATING INSIDE)

Primary Components of electron gun in CRT are:-

1.) Heated metal cathode: Heating Filament causes –vely charged electrons to "boiled off" and accelerate towards phosphor coated screen by a high +ve voltage.

2.) Accelerating anode: The accelerating voltage can be generated with a +vely charged metal coating on the inside of CRT envelop near the phosphor screen, or we can say Accelerating anode can be used.

3.) Control grid: It is a metal cylinder fits over cathode and used to control the intensity of electron beam by setting the voltage level.

4.) Focusing System: need to force the electron beam to converge into a small spot. Otherwise, electron would repel each other and beam will spread out.

5.) Deflection Coils: Deflection can be controlled either with electric field or magnetic field.

How the spot of light produced?

- The Kinetic energy of electron is absorbed by phosphor.
- Part of beam energy is converted by friction into heat energy and remainder causes phosphor atom to move up to higher quantum energy levels.
- After short time, " excited" phosphour atom come back to stable ground state, giving up their extra energy as small quantum of light energy.

Basic properties of phosphors:

- Phosphors Persistence: The time it takes the emitted light from the screen to decay to one-tenth of its original intensity.

  Low persitance require high refresh rate for eg animations.

  High persistance is used for complex, static pictures, monitors in the range from 10 to 60 microseconds.

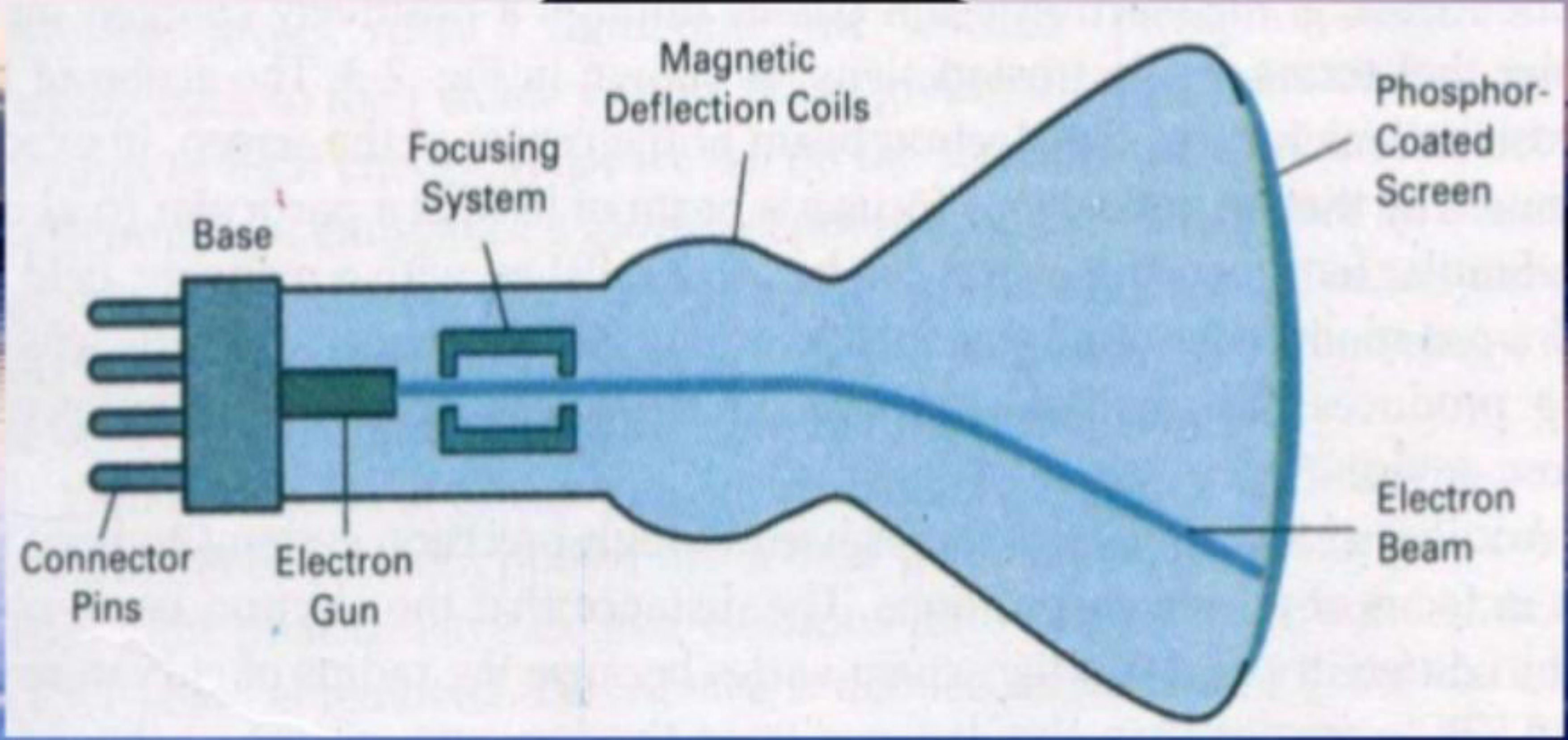
- Resolution: The max no. of pts that can be displayed without overlap on a CRT is referred to as resolution. **OR**

  No. of pts per centimeter that can be plotted horizontally and vertically.
- Intensity Distribution: of a spot on the screen is greatest at the center of the spot, and decrease with a gaussian distribution out to the edges of the spot.

- Aspect Ratio: It is the ratio of vertical pts to horizontal pts necessary to produce equal length lines in both directions on the screen.

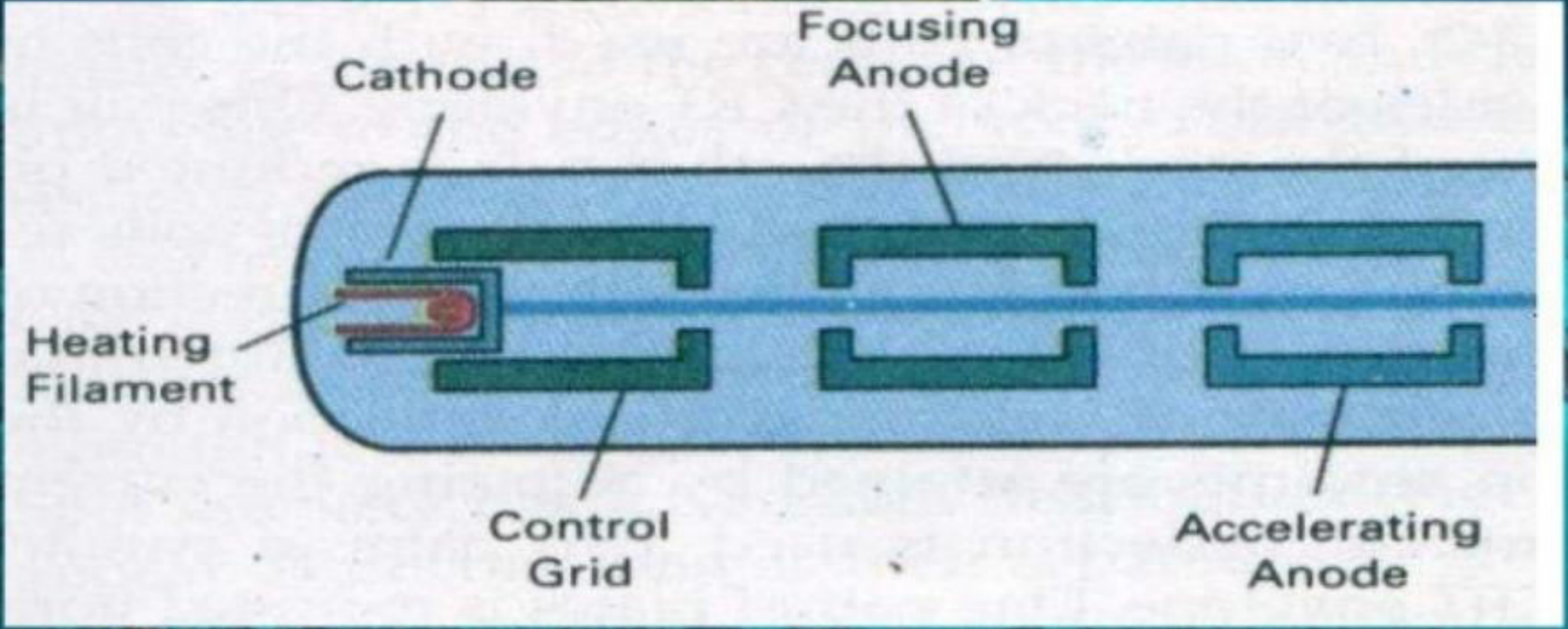
  Aspect ratio ¾ means that a vertical line plotted with 3 pts has the same length as a horizontal line plotted with 4 pts.

Focusing System



Accelerating Anode

The accelerating anode is maintained at sufficient high relative potential to accelerate the beam to necessary velocity .



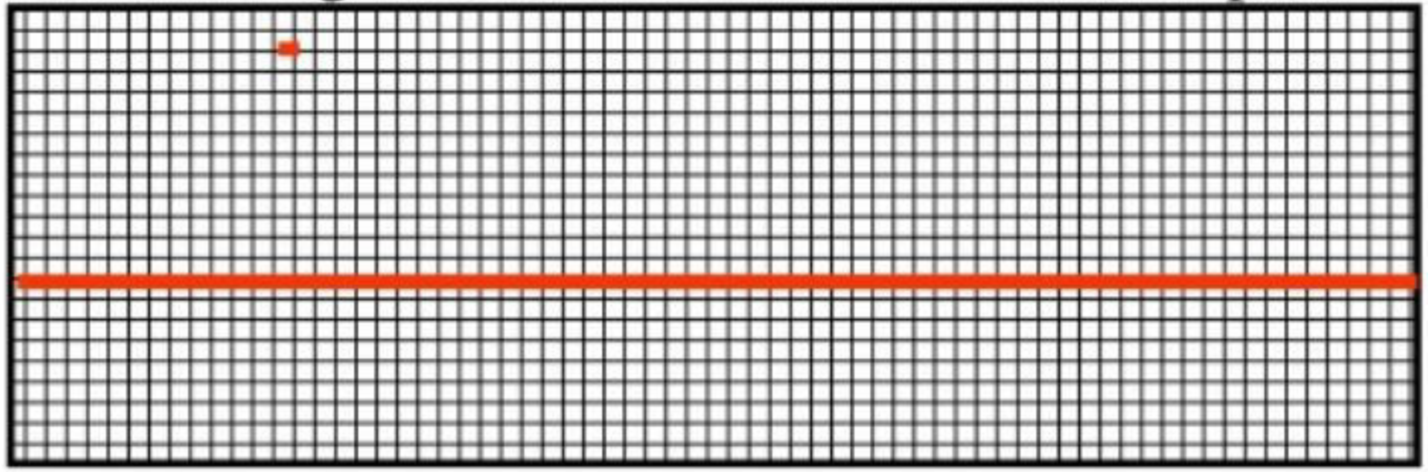
RANDOM SCAN DISPLAYS AND RASTER SCAN DISPLAYS

R.MANIMEGALAI
Guest Lecturer
Periyar Govt Arts College
Cuddalore

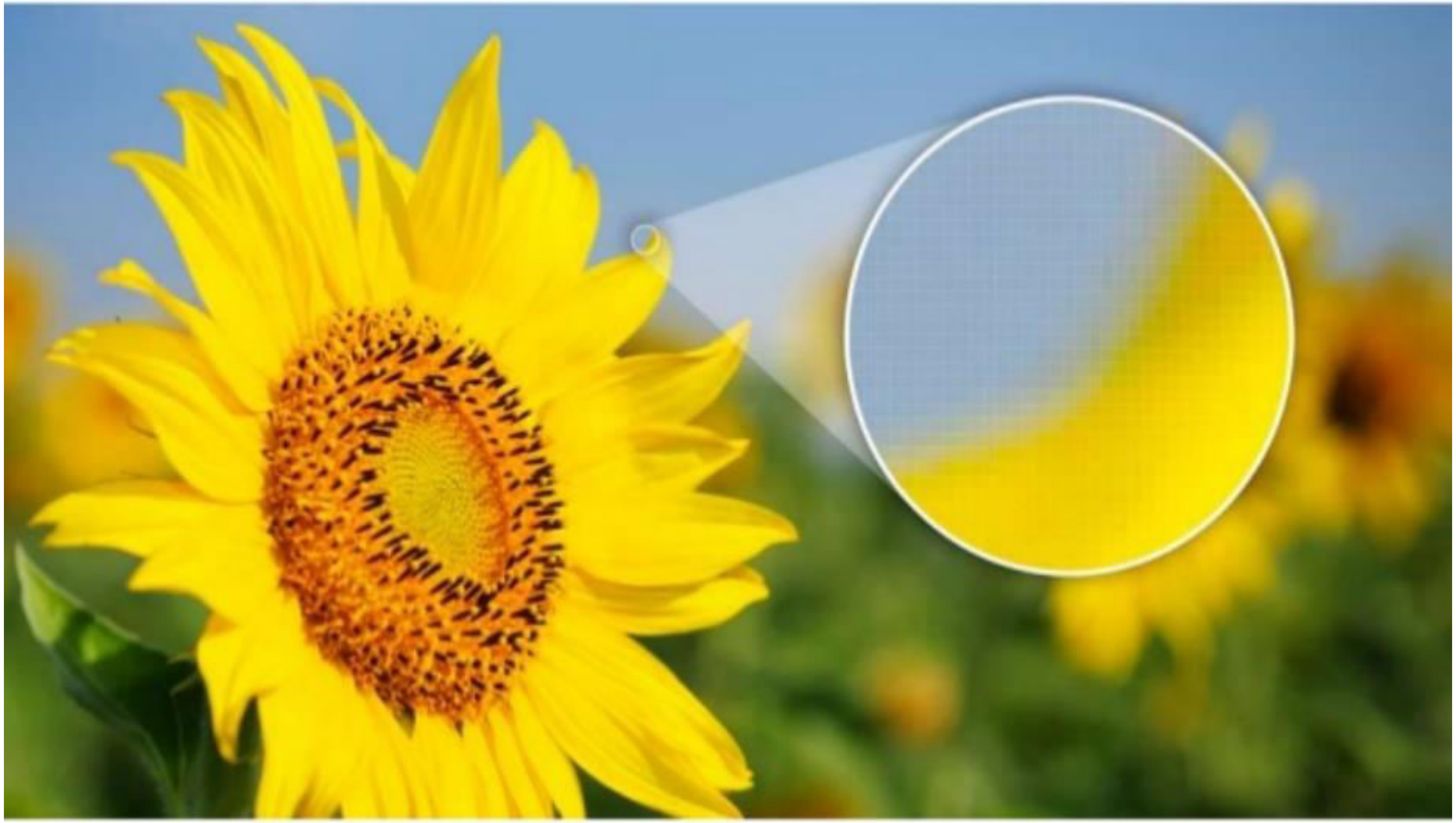
RASTER SCAN DISPLAY

- **Raster:** A rectangular array of points or dot.
- An image is subdivided into a sequence of (usually horizontal) strips known as "scan lines" which can be further divided into discrete pixels for processing in a computer system.

A raster image is a collection of dots called pixels



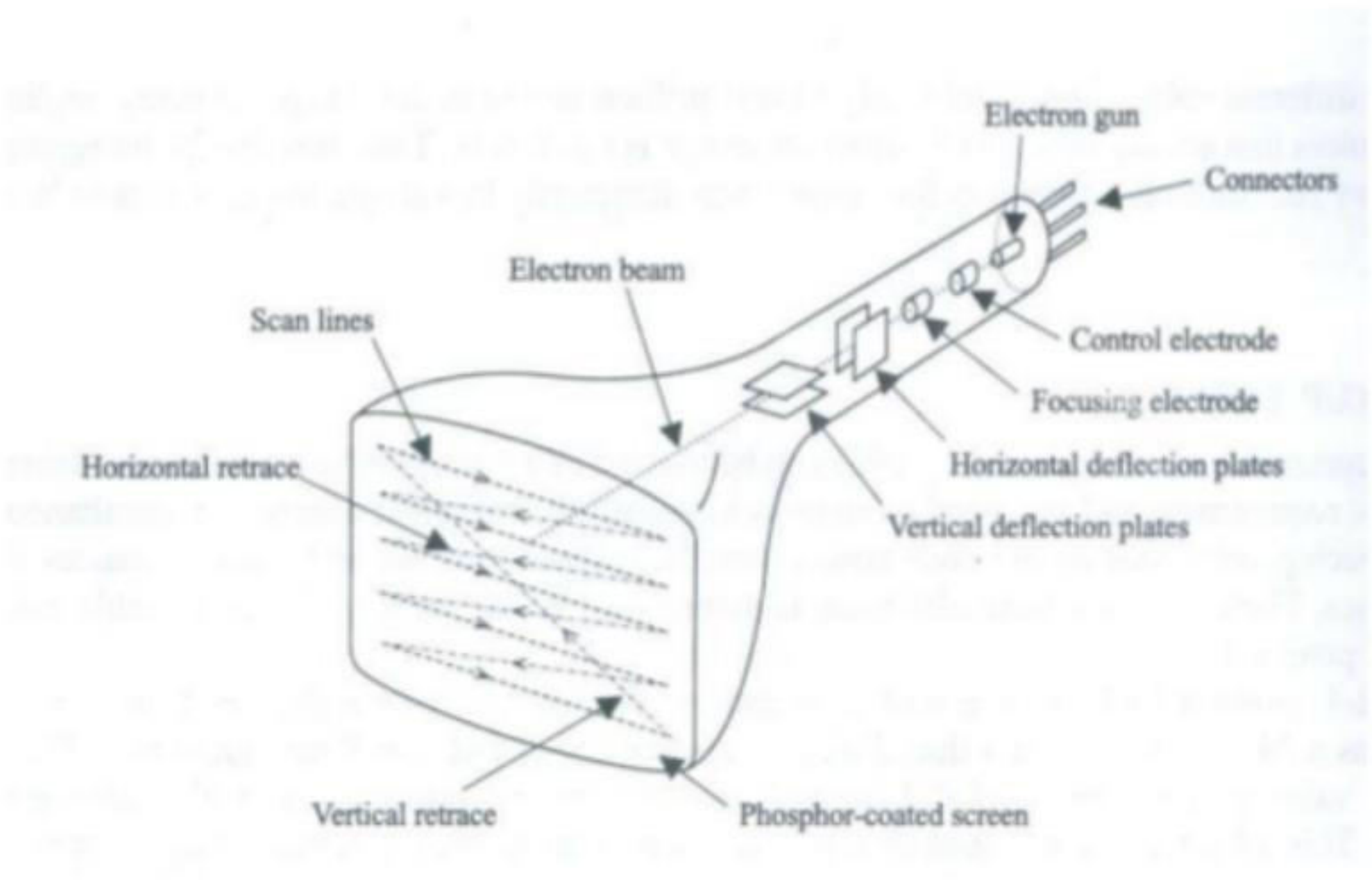
RASTER IMAGE



WORKING

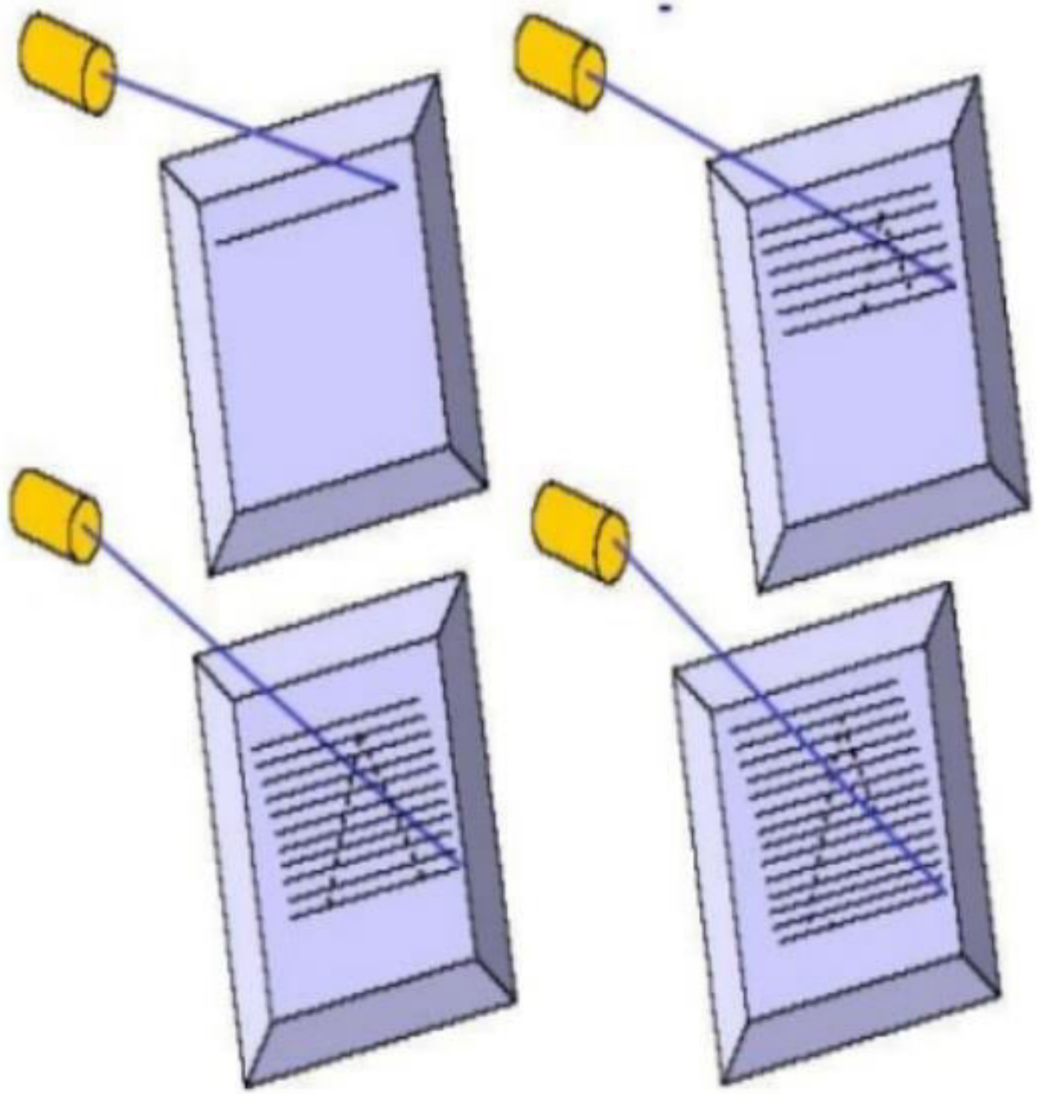
- In a raster scan system, the electron beam is swept across the screen, one row at a time from top to bottom.
- As the electron beam moves across each row, the beam intensity is turned on and off to create a pattern of illuminated spots.
- The return to the left of the screen, after refreshing each scan line is called **Horizontal retrace**.
- At the end of each frame the electron beam returns to the top left corner of the screen to begin the next frame is called **Vertical retrace**:

Raster Scan Display



WORKING

- Picture definition is stored in a memory area called the **refresh buffer** or **frame buffer**.
- **Refresh buffer** or **frame buffer** is memory area that holds the set of intensity values for all the screen points.
- Stored intensity values then retrieved from refresh buffer and "**painted**" on the screen one row (**scan line**) at a time.



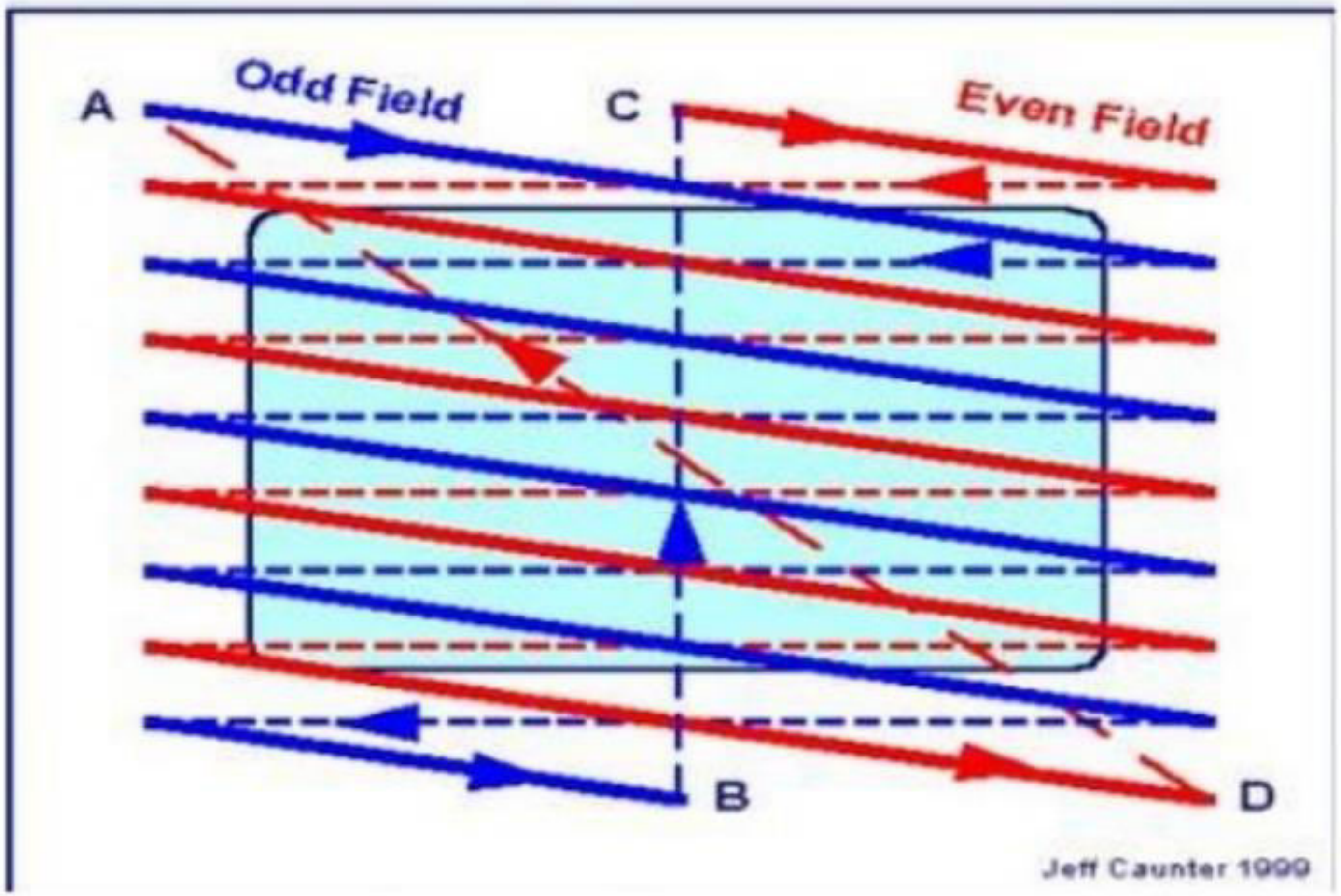
Object as set of discrete points across each scan line

- The quality of a raster image is determined by the total number pixels (**resolution**), and the amount of information in each pixel (**color depth**)
- A black-and-white system: each screen point is either on or off, so only **one bit** per pixel is needed to control the intensity of screen positions. Such type of frame buffer is called Bit map
- High quality raster graphics system have **24** bits per pixel in the frame buffer (a **full color** system or a **true color** system)
- Refreshing on raster scan displays is carried out at the rate 60 to 80 frame per second.

INTERLACING

- On some raster systems (TV), each frame is displays in two passes using an interlaced refresh procedure.
- Interlacing is primarily used for slower refresh rates.
- An effective technique to avoid **Flicker**.(Flicker occurs on CRTs when they are driven at a low refresh rate, allowing the brightness to drop for time intervals sufficiently long to be noticed by a human eye)

INTERLACING



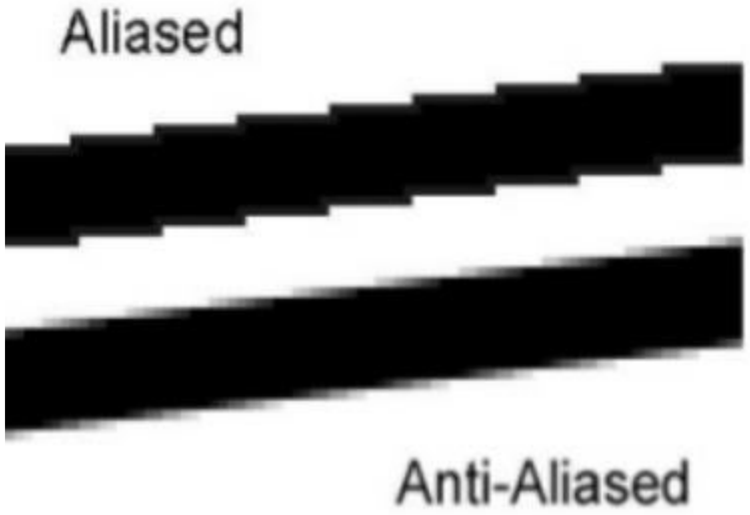
APPLICATIONS

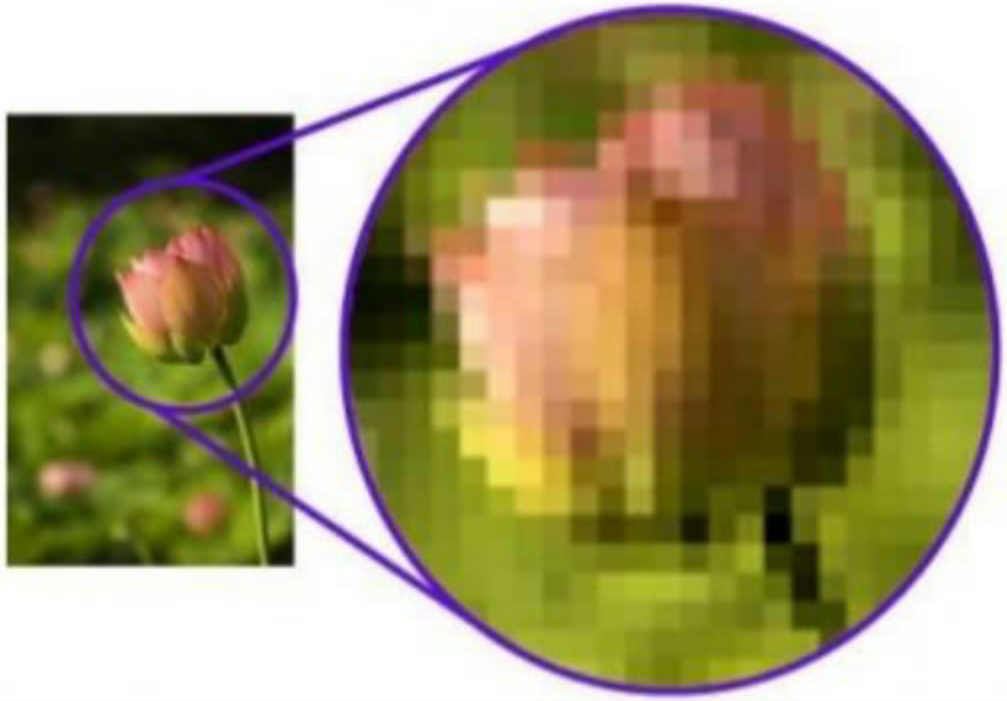
- Suited for realistic display of screens
- Home television computer printers create their images basically by raster scanning. Laser printers use a spinning polygonal mirror (or an optical equivalent) to scan across the photosensitive drum, and paper movement provides the other scan axis

➢ Common raster image formats include BMP (Windows Bitmap), JPEG (Joint Photographics Expert Group), GIF (Graphics Interchange Format) , PNG (Portable Network Graphic), PSD (Adobe PhotoShop)

DISADVANTAGE

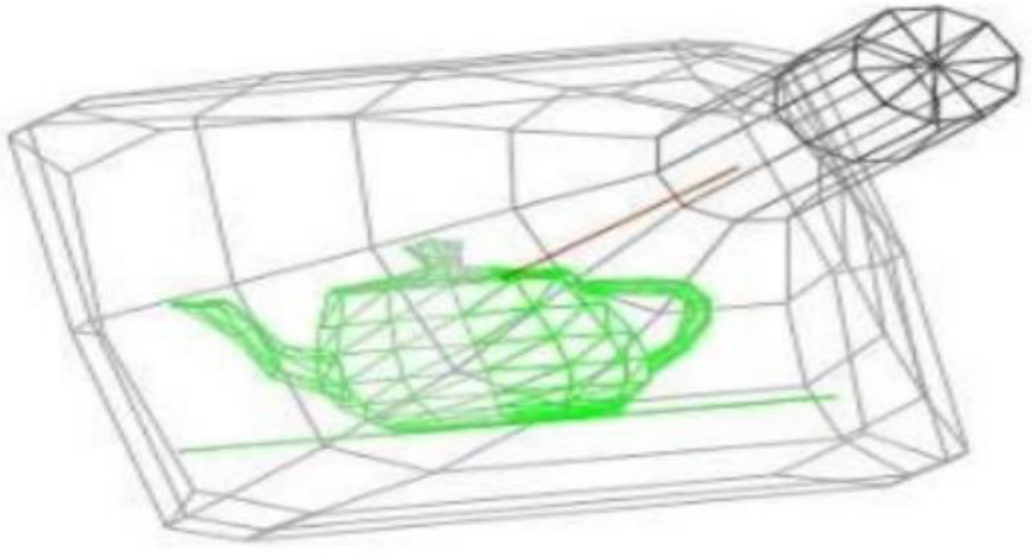
- To increase size of a raster image the pixels defining the image are be increased in either number or size Spreading the pixels over a larger area causes the image to lose detail and clarity.
- Produces jagged lines that are plotted as discrete points



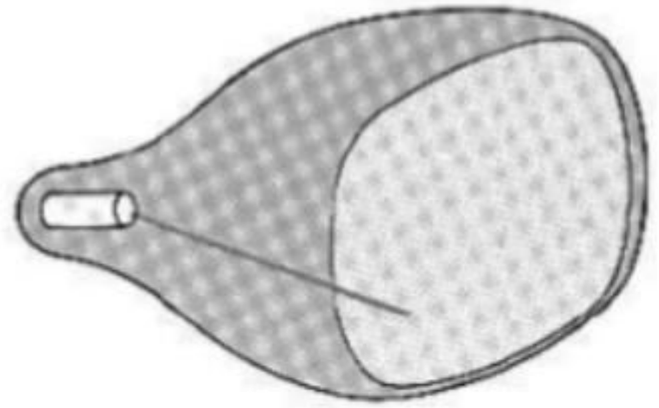


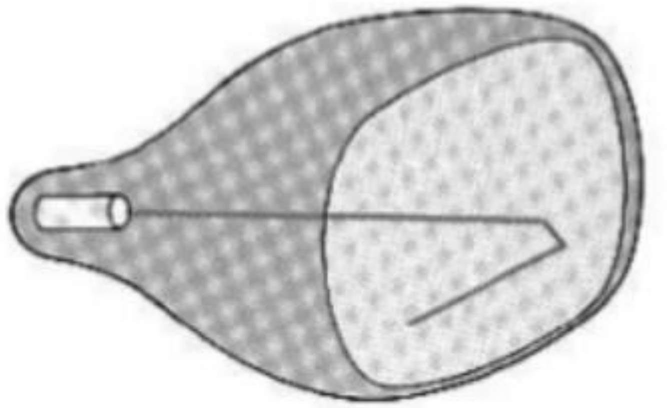
RANDOM SCAN DISPLAY

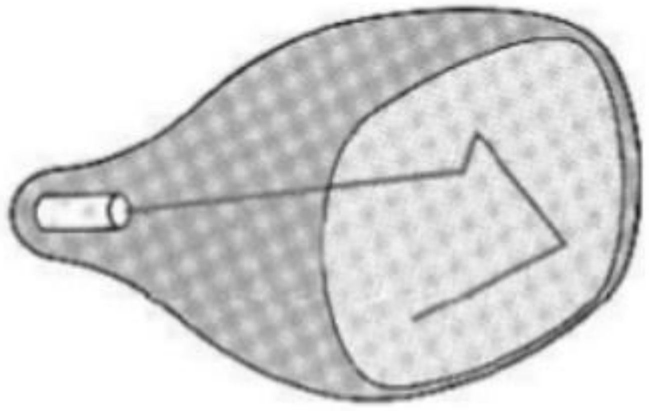
- Random scan display is the use of geometrical primitives such as points, lines, curves, and polygons, which are all based upon **mathematical**

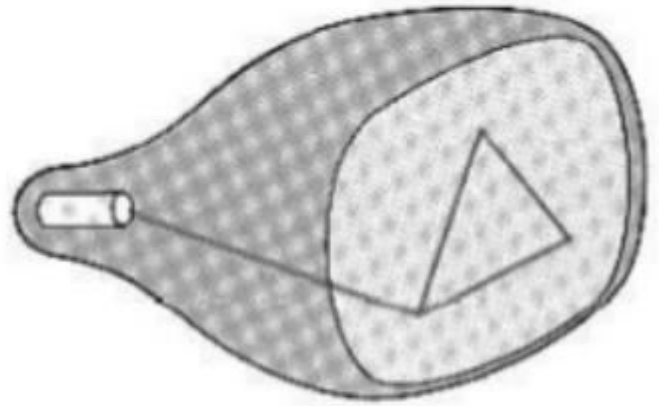


RASTER SCAN DISPLAY


(a)

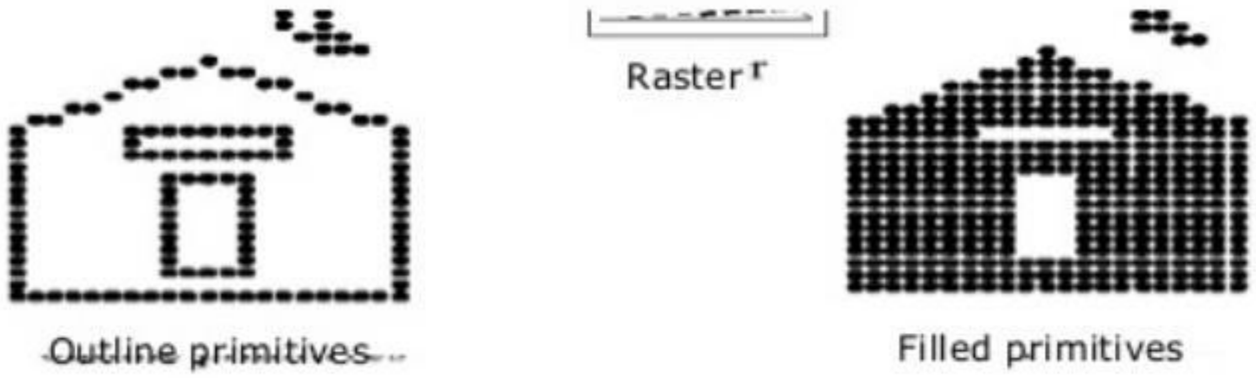

(b)


(c)

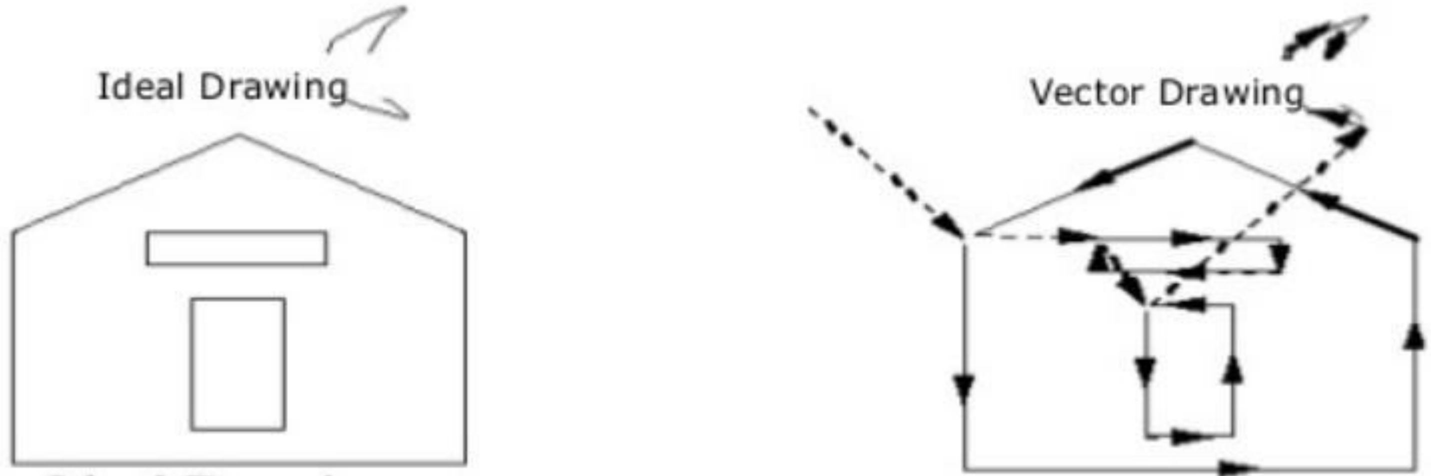

(d)

- **Refresh rate** depends on the number of lines to be displayed.
- Picture definition is now stored as a line-drawing commands an area of memory referred to as **refresh display file** (**display list**).
- To display a picture, the system cycle through the **set of commands** in the display file, drawing each component line in turn.
- Random scan displays are designed to draw all the component lines of a picture 30 to 60 times each second

➢A Raster system produces jagged lines that are plotted as discrete points sets.



➢Vector displays product smooth line drawing



Ideal Drawing

Vector Drawing

- Random scan displays are designed for **line-drawing applications** and can not display realistic shaded scenes

Advantages

- Random scan displays have higher resolution than raster systems.
- Vector displays product smooth line drawing.
- This minimal amount of information translates to a much smaller file size. (file size compared to large raster images)
- On zooming in, and it remains smooth
- The parameters of obje.cts are stored and can be later modified.

RASTER SCAN SYSTEM

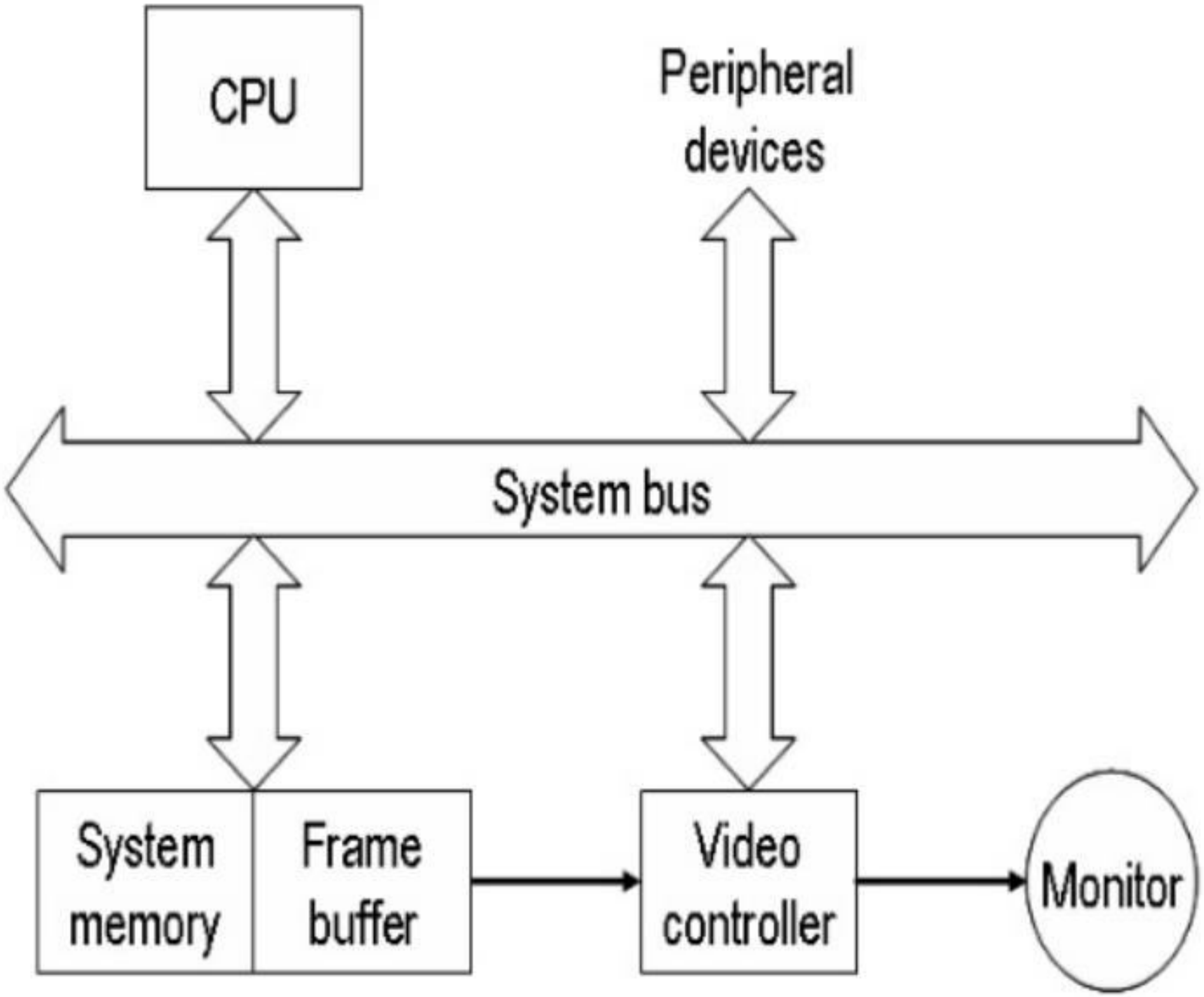
- In addition to the central processing unit (CPU), a special processor, called the **video controller** or **display controller**, is used to control the operation of the display device.

- A fixed area of the system memory is reserved for the frame buffer, and the video controller is given direct access to the frame buffer memory.

- Operation performed:

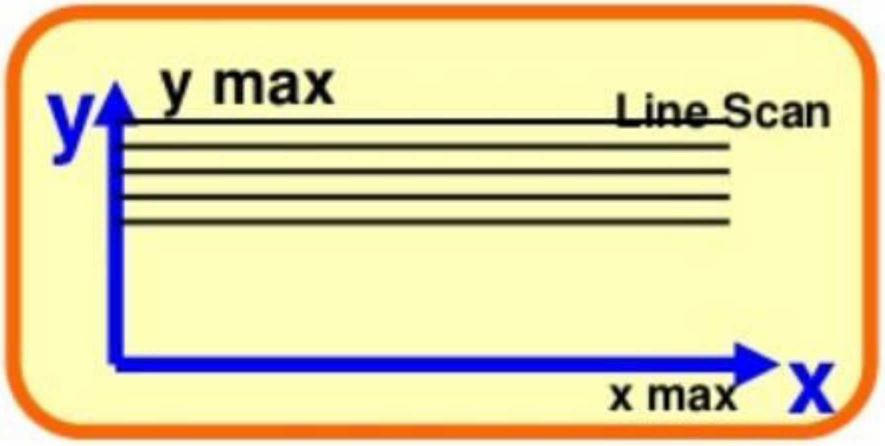
1. Refreshing operation

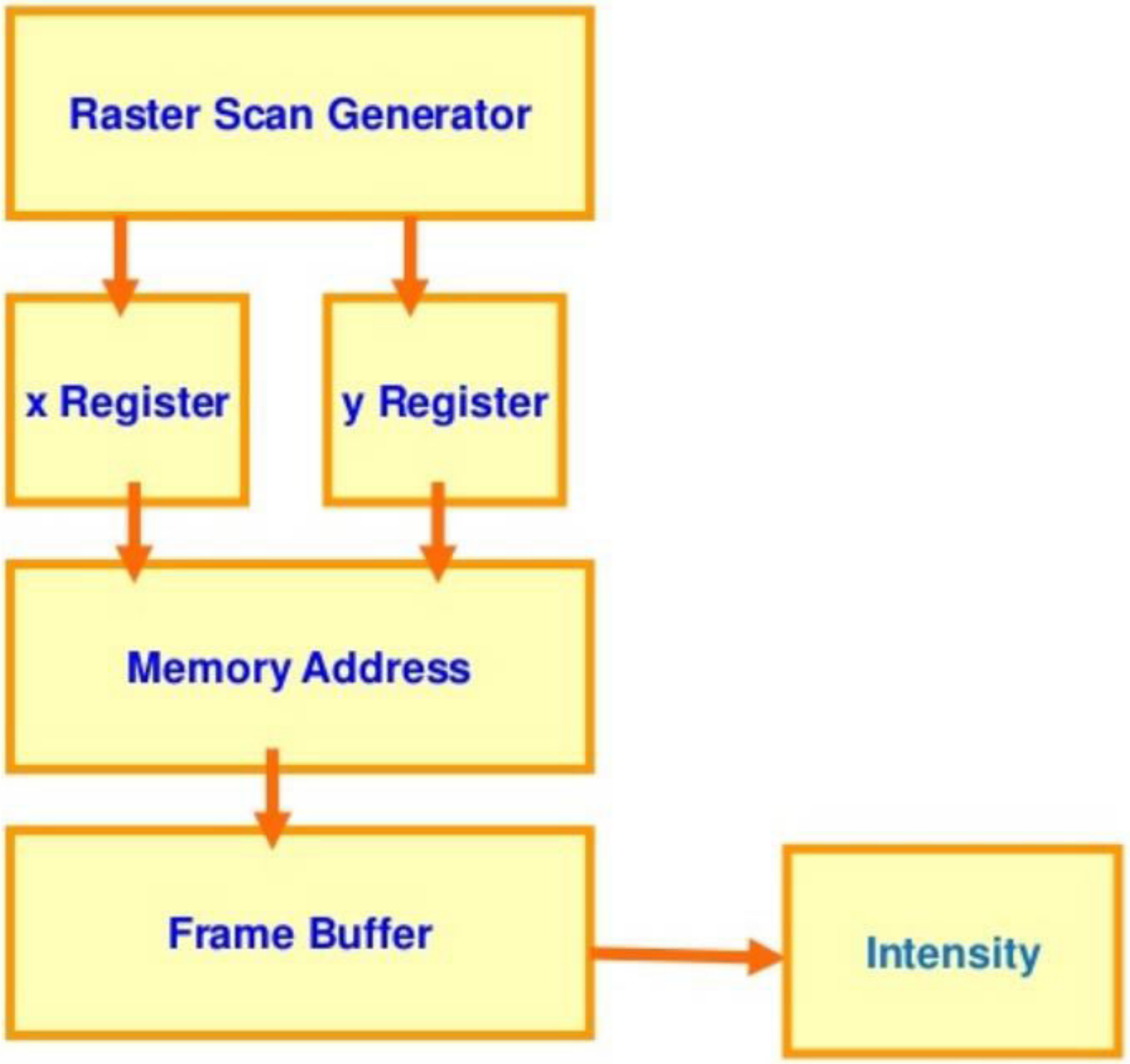
2. Transformation (Areas of the screen can be enlarged, reduces, or moved during the refresh cycles)


CPU
Peripheral devices
System bus
System memory
Frame buffer
Video controller
Monitor

The Basic refresh operation of the video controller.

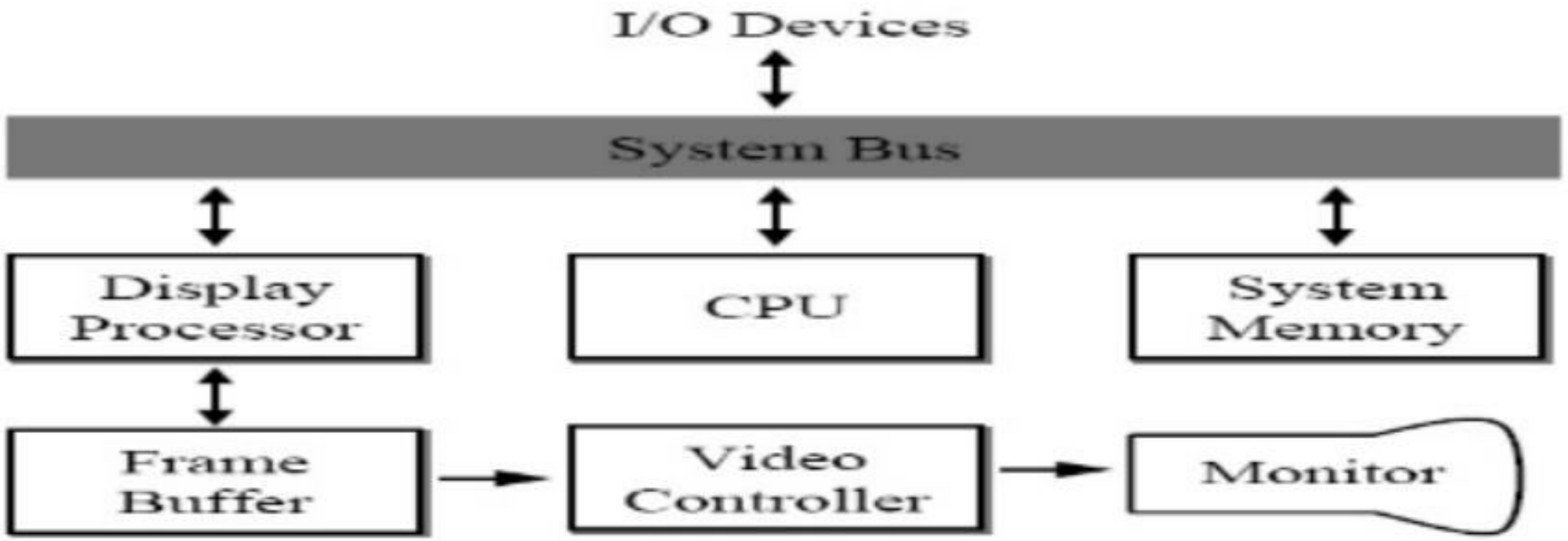
- **Frame buffer** location, and the corresponding screen positions, are referenced in Cartesian coordinates
- **Scan lines** are then labeled from y_{max} at the top of the screen to 0 at the bottom. Along each scan line, screen **pixel** positions are labeled from 0 to x_{max}
- Two registers are used to store the coordinates of the screen pixels.



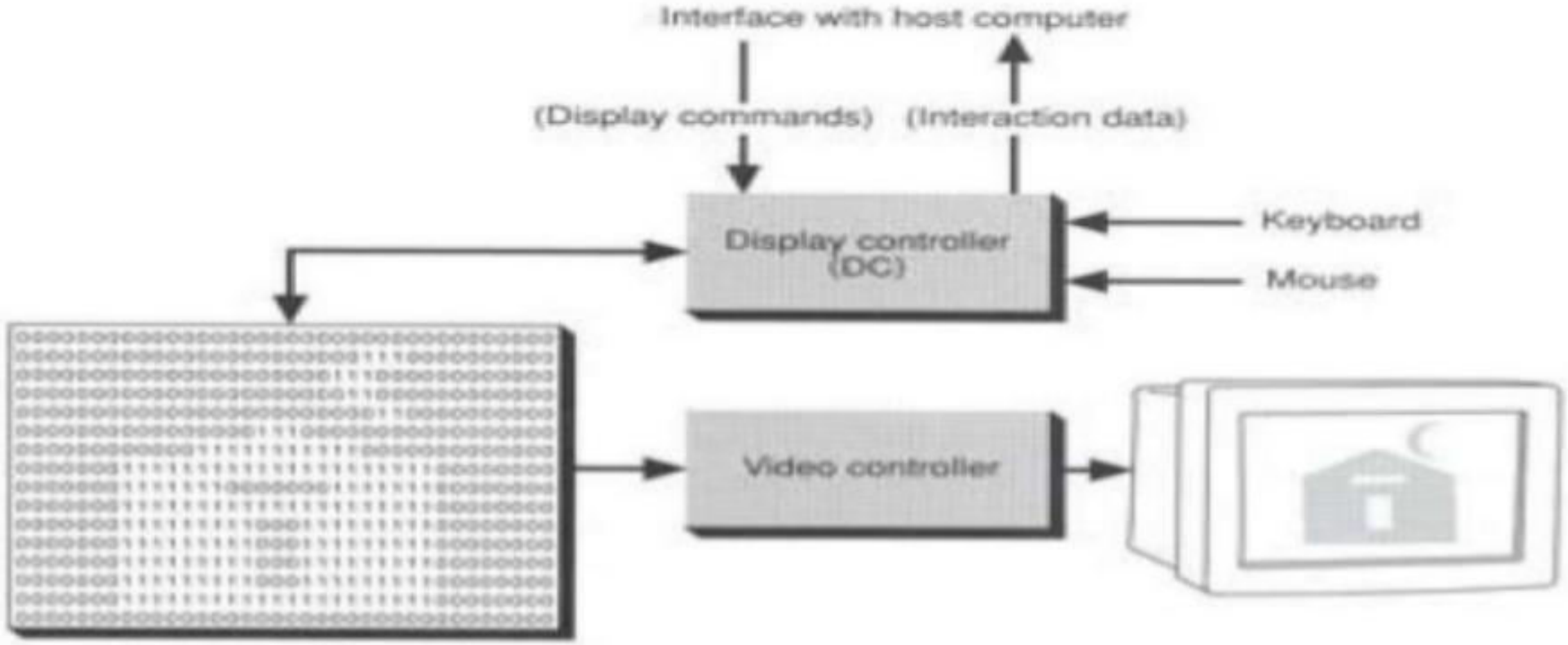

Raster Scan Generator
x Register
y Register
Memory Address
Frame Buffer
Intensity

DISPLAY PROCESSOR

- The purpose of the DP is to free the CPU from the graphics chores.
- A major task of the display processor is **Scan Conversion.**
- Scan Conversion: is digitizing a picture definition given in an application program into a set of pixel intensity values for storage in the frame buffer.



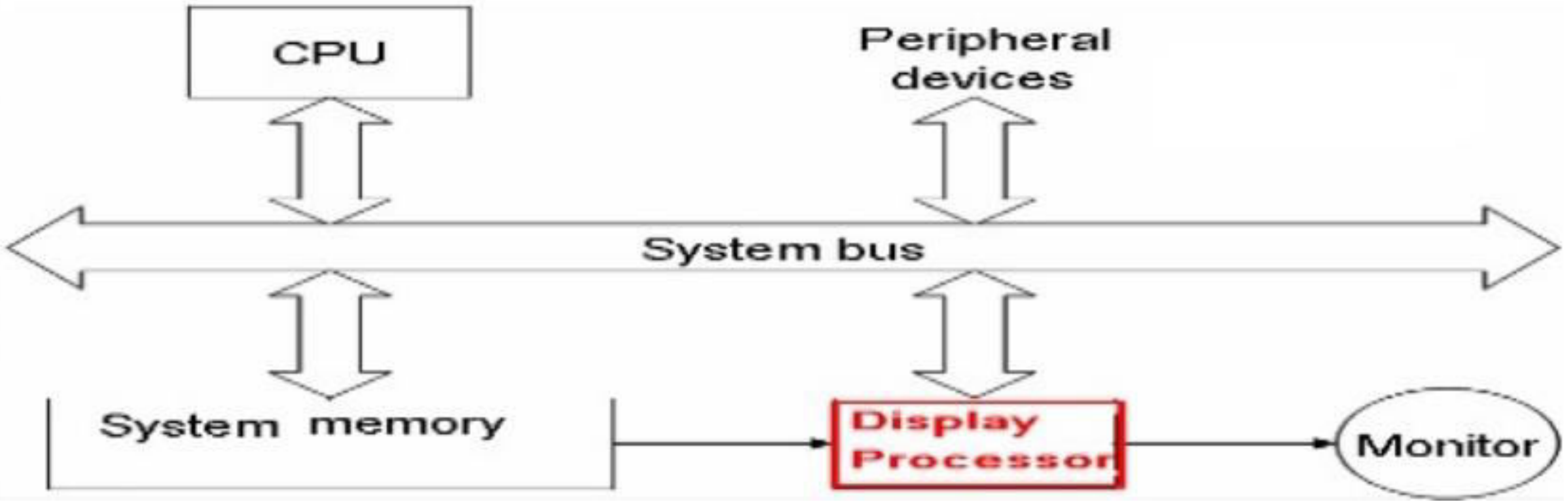
I/O Devices
System Bus
Display Processor
CPU
System Memory
Frame Buffer
Video Controller
Monitor



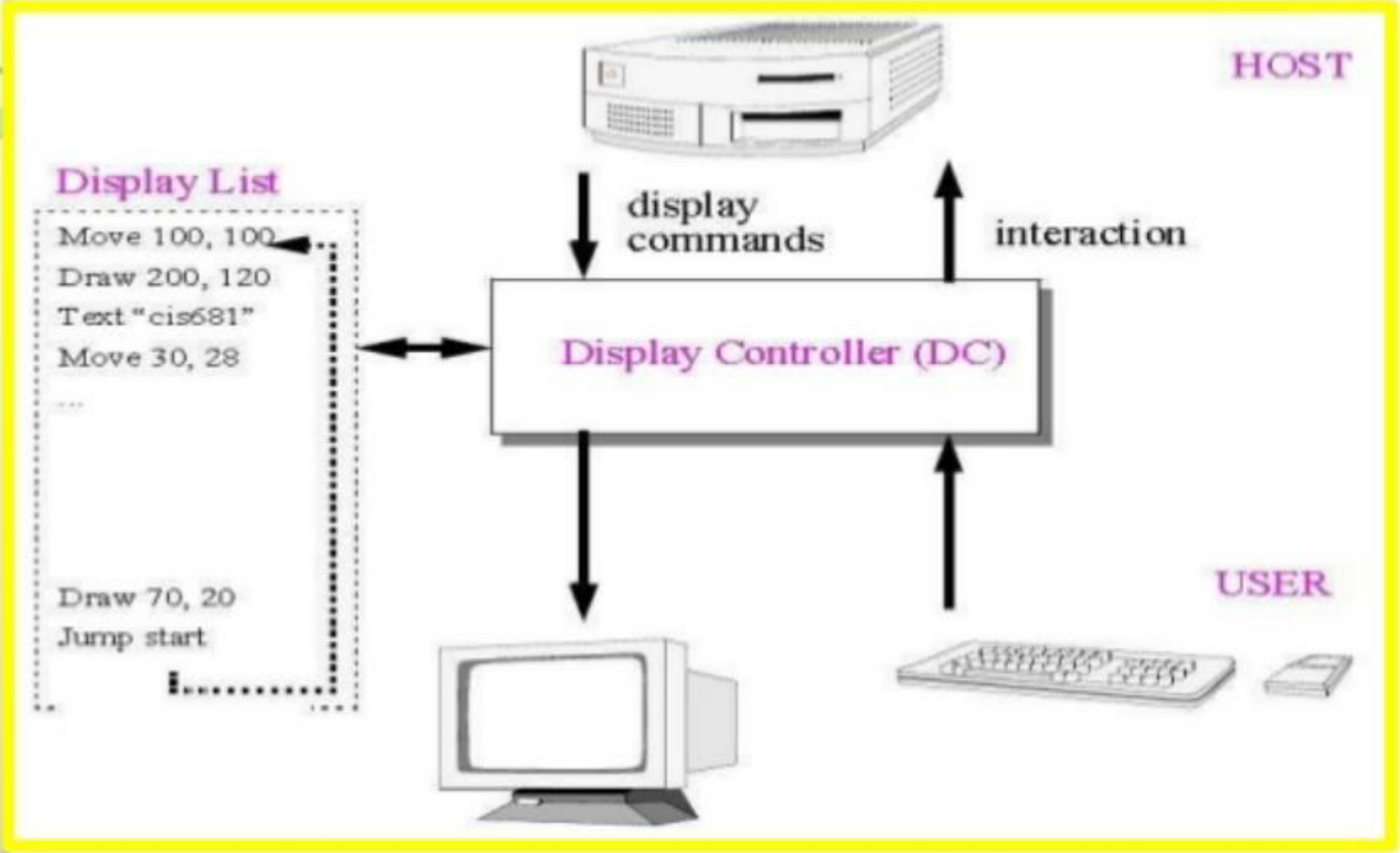
Interface with host computer
(Display commands)
(Interaction data)
Display controller (DC)
Keyboard
Mouse
Video controller

RASTER SCAN SYSTEM

- Graphic commands are translated by the graphics package into a display file stored in the system memory.
- This file is then accessed by the **display processor unit** (**DPU**)(graphic controller) to refresh the screen.



RASTER SCAN SYSTEM



Graphics Software

- A graphics software is an intermediary between an application program & the graphics hardware. The output primitives & interaction devices that a graphics package supports can range from rudimentary to extremely rich.

There are two general classifications for graphics software:

- **General Programming packages**: provides an extensive set of graphics functions that can be used in a high-level programming language, such as C or FORTRAN. Basic functions in a general package include those for generating picture components (straight line, circle, polygon etc),setting color and intensity values, & applying transformations.

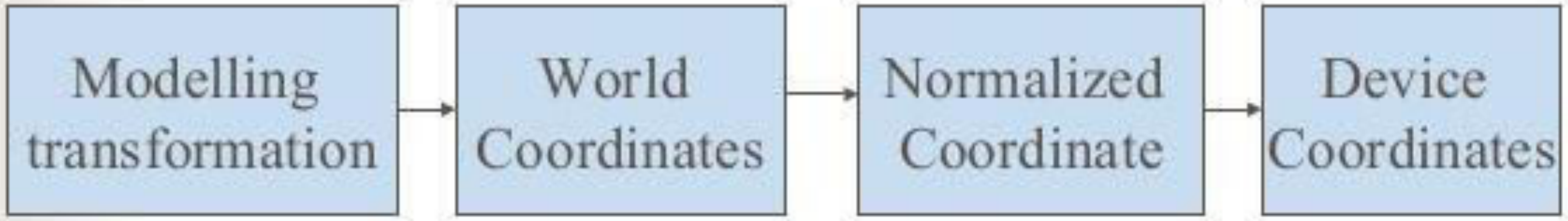
- **Special-purpose applications packages:**

Designed for nonprogrammers, so that users can generate displays without worrying about how graphics operations work. Example of such application packages are the artist's painting programs and various business,medical and CAD systems.

Coordinate Representation

General Graphics packages are designed to be used with Cartesian coordinates.

Several different Cartesian reference frames are used to construct & display a scene.



Graphics Functions

These packages provides users with a variety of functions for creating & manipulating pictures.

- Output primitives: basic building blocks.
- Attributes: properties of the output primitives.
- Geometric transformations: changing size, position & orientation.

- Modeling transformations: construct scene using object descriptions.
- Viewing transformations: are used to specify the view that is to be presented.
- Input Functions: used to control & process the data flow from the interactive devices such as mouse, tablet or joystick.
- Control operations: contains no. of housekeeping tasks such as clearing a display screen & initializing parameters.

Software Standards

A standard graphics package such as GKS(Graphical kernal system) & PHIGS (Programmers Hierarchical Interactive graphics system) implements a specification designated as standard by an official national or international standard bodies by ISO and ANSI(American National Standard Institute).

- The main purpose of such standards is to promote portability of application programs & of programmers.
- Non-official standards are also developed, promoted & licensed by individual companies or by consortia of companies eg Adobe's Post script & MIT's X window system are two industry standards.

- GKS originally designed as a 2-D graphics packages, a 3-D GKS extension was subsequently developed.
- PHIGS is a extension of GKS having increased capabilities for object modeling, color specification, surface rendering etc.
- Extension of PHIGS called PHIGS+ provide 3-D surface shading capabilities.

- **GKS primitives:**

There are basic four primitives:

(a) Polyline: used to draw lines.

POLYLINE(n, X, Y)

n = length of an array

X & Y = array of x,y coordinate

(b) Polymarker: used to plot points.

POLYMARKER(n, X, Y)

n = number of data points

(c) Fill Area: also used to draw line but it always connects the first and last points in the array.

FILL AREA(n, X, Y)

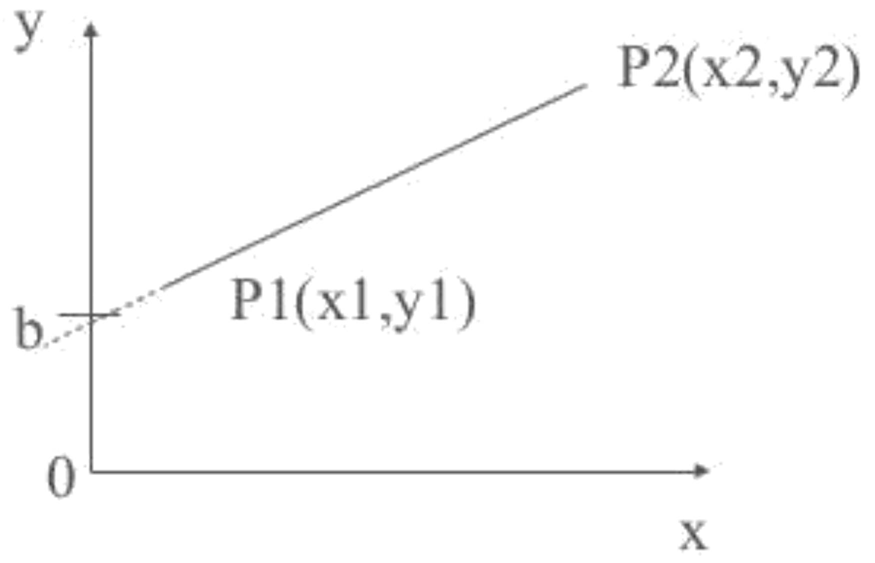
(d) Text: used to print the "string" or "text" starting at the given coordinates.

TEXT(x, y, "String")

- **GKS Segments:** The segment command is the GKS method of developing complex objects. Any group of valid GKS code can be clustered together into a segment through the use of the CREATE SEGMENT (n) command. A segment listing is terminated with CLOSE SEGMENT command. For eg.

```
CREATE SEGMENT (1)
 POLYLINE (5, X, Y)
  CLOSE SEGMENT
```

The two endpoints of a line segment are specified at positions **(x1,y1)** and **(x2,y2).**



We can determine the value for slope m & b intercept as

$$\mathbf{m = y2\text{-}y1/x2\text{-}x1} \qquad (2)$$

And, $$\mathbf{b = y1 - mx1} \qquad (3)$$

For a given x interval Δx along a line, we can compute the corresponding y interval Δy from

$$\mathbf{\Delta y = m\ \Delta x} \qquad (4)$$

Similarly, we can obtain x interval Δx by Δy:

$$\mathbf{\Delta x = \Delta y/m} \qquad (5)$$

If |**m**|<**1**, then for every integer value of x between and excluding x1 and x2, calculate the corresponding value of y using equation **$\Delta y = m\,\Delta x$** & scan convert (x,y).

If |**m**|>**1**, then for every integer value of y between and excluding y1 and y2, calculate the corresponding value of x using equation **$\Delta x = \Delta y/m$** & scan convert (x,y).

If |**m**|=**1**, **$\Delta x = \Delta y$.** In each case, a smooth line with slope m is generated between the specific endpoints.

Example 1 The endpoints of line are(0,0) & (6,18). Compute each value of y as x steps from 0 to 6 and plot the result.

Solution : Equation of line is y= mx +b

m = y2-y1/x2-x1= 18-0/6-0 = 3

Next the y intercept b is found by plugging y1& x1 into the equation y = 3x + b,

0 = 3(0) + b. Therefore, b=0, so the equation for the line is y= 3x.

While this approach is mathematically sound, it involves floating-point computation (multiplication & addition) in every step that uses the line equation since m & b are generally real numbers. The challenge is to find a way to achieve the same goal as quickly as possible.

DDA Algorithm

The digital differential analyzer (DDA) algorithm is an incremental scan-conversion method. Such an approach is characterized by performing calculations at each step using results from the preceding step. Suppose, at step i we have calculated(xi, yi) to be a point on the line.Since the next point (xi+1,yi+1) should satisfy $\Delta y/\Delta x = m$ where $\Delta y = yi+1 - yi$ & $\Delta x = xi+1 - xi$.

We have, $y_{i+1} = y_i + m\Delta x$

$$y_{i+1} = y_i + \Delta y \qquad (1)$$

Or $$x_{i+1} = x_i + \Delta y/m \qquad (2)$$

Algorithm:

(x1,y1) (x2,y2) are the end points and dx, dy are the float variables.

```
(i)  If abs(x2-x1) > abs(y2-y1) then
         length = abs(x2-x1)
     else
         length = abs(y2-y1)
     endif
```

```
(ii)    dx = (x2-x1)/length
        dy = (y2-y1)/length
(iii)   x = x1 + 0.5
        y = y1 + 0.5
(iv)    i = 0
(v)     Plot (trunc(x), trunc(y))
(vi)      x = x + dx
          y = y + dy
(vii)    i = i + 1
```

(viii) If i < length then go to step (v)

(ix) Stop

Example 2 Scan convert a line having end points (3,2) & (4,7) using DDA.

Solution: x2 - x1 = 4-3 = 1

y2 - y1 = 7-2 = 5

As, abs(x2-x1) < abs(y2-y1) then

length = y2-y1 = 5

dx = (x2-x1)/ length = 1/5 = .2

dy = (y2-y1)/ length = 5/5 = 1

x1	y1	x2	y2	L	dx	dy	i	x	y	Result	Plot
3	2	4	7	5	.2	1	0	3.5	2.5	3.5, 2.5	3,2
							1	3.7	3.5	3.7,3.5	3,3
							2	3.9	4.5	3.9,4.5	3,4
							3	4.1	5.5	4.1,5.5	4,5
							4	4.3	6.5	4.3,6.5	4,6
							5	4.5	7.5	4.5,7.5	4,7

Limitations of DDA:

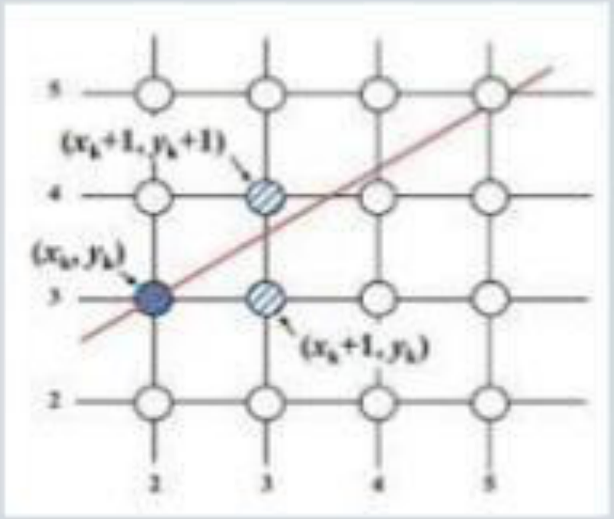
BRESENHAM'S LINE

DRAWING ALGORITHM

INDEX

INTRODUCTION

- Raster line-generating algorithm
- Developed by Bresenham
- Scan conversion takes place using only incremental integer calculations
- Accurate and efficient than DDA



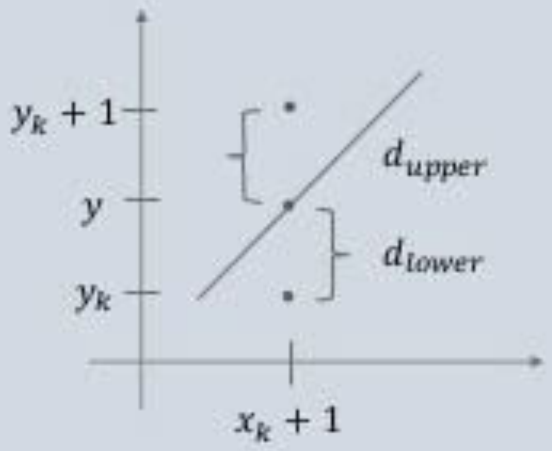
DERIVATION

- Starting from the left endpoint (x0, y0) of a given line, we step to each successive column (x position) and plot the pixel whose scan-line y value is closest to the line path.
- At sample positions $x_k + 1$ the vertical separations from the line are labelled d_{upper} and d_{lower}
- y coordinate on the line at $x_k + 1$ is,

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

- so,

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



DERIVATION

- It can be used to make decision about which pixel is closer to the line
- This decision is based on the difference between the two pixel positions,

$$d_{upper} - d_{lower} = 2m(x_k + 1) - 2y_k + 2b - 1$$

- By substituting $m = \Delta y / \Delta x$ and both are differences of end points,

$$\begin{aligned}\Delta x(d_{upper} - d_{lower}) &= \Delta x\left(2\left(\frac{\Delta y}{\Delta x}\right)(x_k + 1) - 2y_k + 2b - 1\right)\\ &= 2\Delta y.\, x_k - 2\Delta x.\, y_k + 2\Delta y + \Delta x(2b - 1)\\ &= 2\Delta y.\, x_k - 2\Delta x.\, y_k + C\end{aligned}$$

DERIVATION

- Now, a decision parameter P_k for the kth step along a line,

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

- The sign of P_k is same as that of $d_{upper} - d_{lower}$
- If P_k is –ve then we choose the lower pixel i.e. y_k only, otherwise we choose the upper pixel i.e. $y_k + 1$
- So, for $P_k + 1$ at step $k + 1$,

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

- Subtracting P_k,

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

DERIVATION

- x_{k+1} is same as $x_k + 1$ so,

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

- Here, $y_{k+1} - y_k$ is either 0 or 1 depending on the sign of P_k
- If $P_k < 0$, the next point to plot is $(x_k + 1, y_k)$ and new value of P is,

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

- If $P_k > 0$, the next point to plot is $(x_k + 1, y_k + 1)$ and new value of P is,

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

- The first decision parameter P_0 is evaluated at (x_0, y_0) is,

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

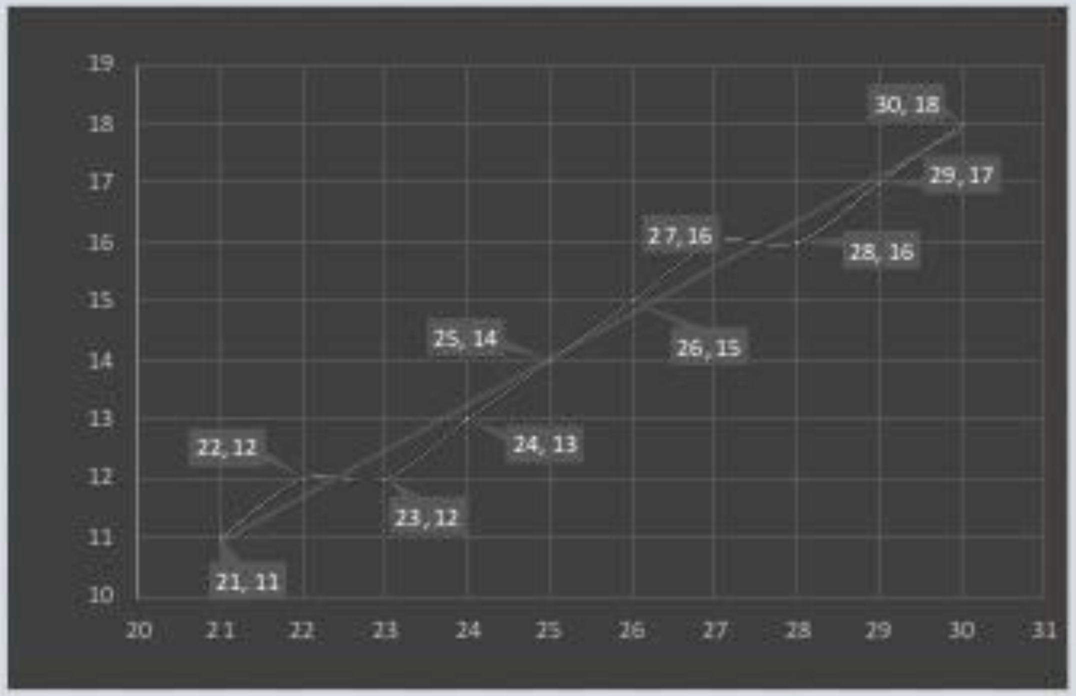
EXAMPLE

- End points (20,10) and (30,18)
  - Δx=x2-x1 =30-20 =10
  - Δy=y2-y1 =18-10 =8
  - m= $\Delta y/\Delta x$=8/10=0.8

k	P_k	(x_{k+1}, y_{k+1})
0	$6 > 0$	(21,11)
1	$2 > 0$	(22,12)
2	$-2 < 0$	(23,12)
3	$14 > 0$	(24,13)
4	$10 > 0$	(25,14)

k	P_k	(x_{k+1}, y_{k+1})
5	$6 > 0$	(26,15)
6	$2 > 0$	(27,16)
7	$-2 < 0$	(28,16)
8	$14 > 0$	(29,17)
9	$10 > 0$	(30,18)

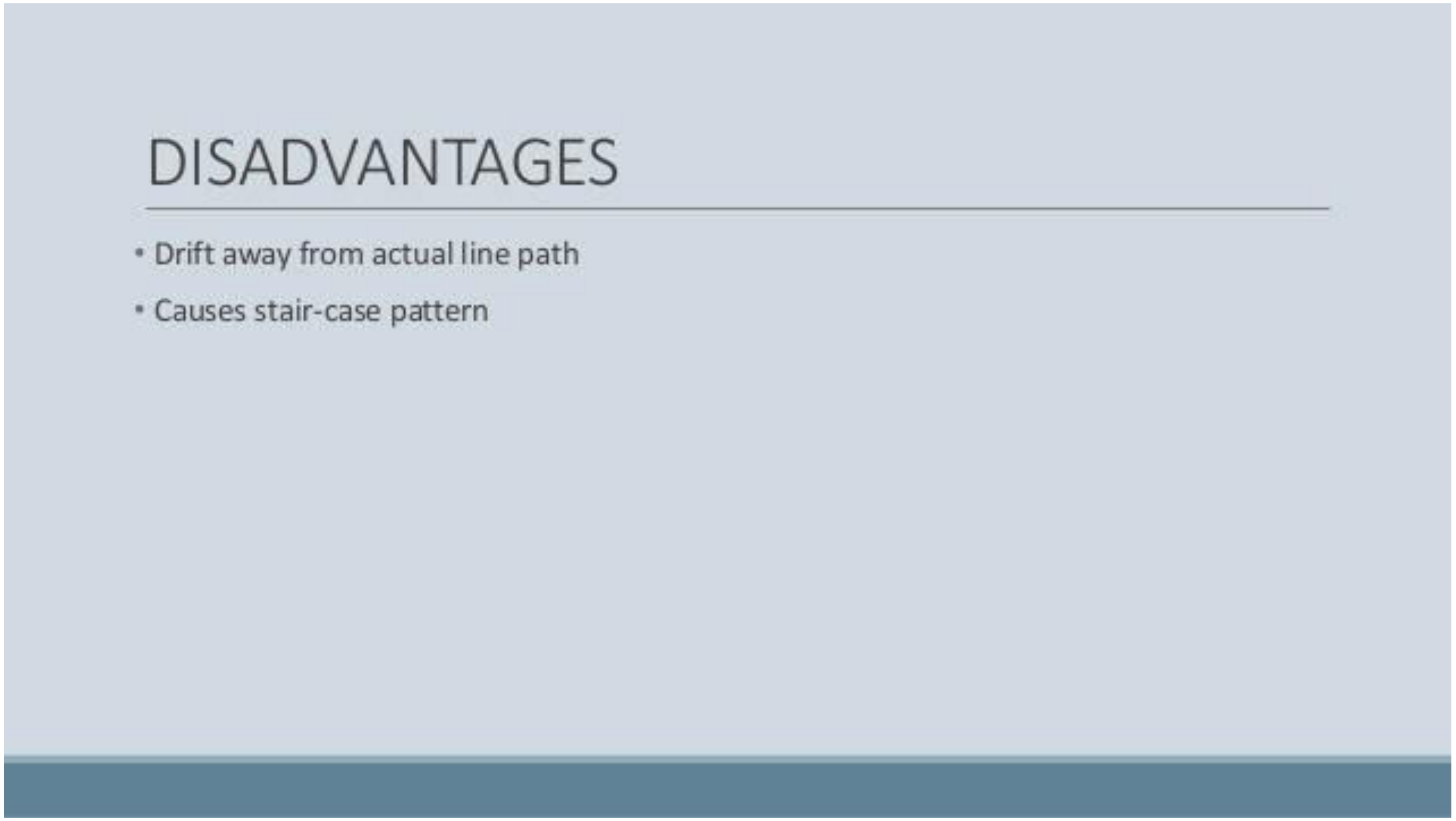
EXAMPLE



19
18
17
16
15
14
13
12
11
10
20
21
22
23
24
25
26
27
28
29
30
31
30, 18
29, 17
27, 16
28, 16
26, 15
25, 14
24, 13
22, 12
23, 12
21, 11

ADVANTAGES

- Uses fixed points
- Easy to calculate (only addition & subtraction)
- Fast execution compare to DDA
- More accurate and efficient



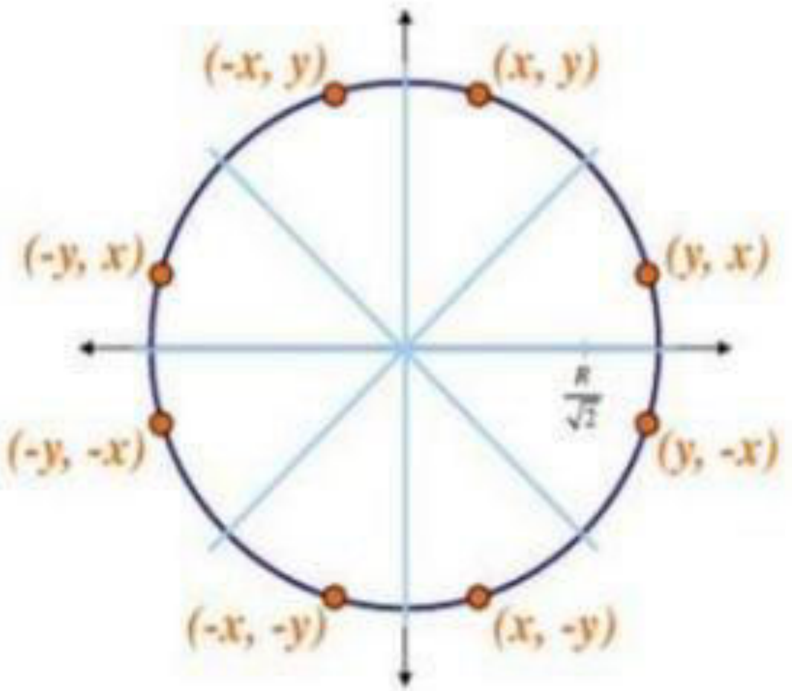
DISADVANTAGES

- Drift away from actual line path
- Causes stair-case pattern

Circle Generation Algorithm

- Circle Generation is a tricky task for computer because we have to choose pixels along a circular path.
- Selection of right pixel for illumination requires more computation time as compare to Line drawing algorithm.
- Selection of appropriate pixels is based on decision parameter which is same as line drawing algorithm but computation is different.
- Two algorithm for Circle generation are:
- Breshenham's Circle Generation Algorithm
- Mid Point Circle generation Algorithm.

8- point symmetry in circles



Breshenham’s Circle Generation Algorithm

1. Set x=0; y=10 here y contains value for radius of circle
2. **p=3-(2*r)** calculate first decision parameter which calculate which pixel will glow on the circumference of the circle.
3. Repeat setp 3 until (i<y), set always set i=x initially
4. If p<0

   calculate x=x+1, Y=Y and next decision parameter by **p=p+4*x+6**;

Otherwise

Calculate y=y-1 and x=x+1;

Calculate next decision parameter by **p=p+4*x-4*y+10**;

calculte points for all octants of circle by using and paint corresponding points with color

```
    putpixel(xc+x,yc-y,4);      putpixel(xc-x,yc-y,4);
    putpixel(xc+x,yc+y,4);      putpixel(xc-x,yc+y,4);
    putpixel(xc+y,yc-x,4);      putpixel(xc-y,yc-x,4);
    putpixel(xc+y,yc+x,4);      putpixel(xc-y,yc+x,4);
```

5. Stop algorithm when (i>y) or (i==y)

Code section from circle Program

```
•   x=0;y=100;
•   p=3-(2*r);
•   for(int i=0;i<y;i++)
•   {
•   if(p<0)
•           { x=x+1; p=p+4*x+6;  }
•   else
          { y=y-1;  x=x+1; p=p+4*x-4*y+10;  }
•   putpixel(xc+x,yc-y,4);      putpixel(xc-x,yc-y,4);
•   putpixel(xc+x,yc+y,4);      putpixel(xc-x,yc+y,4);
•   putpixel(xc+y,yc-x,4);      putpixel(xc-y,yc-x,4);
•   putpixel(xc+y,yc+x,4);      putpixel(xc-y,yc+x,4);
•   delay(40);
•   }
```

Numerical based on Breshenham's Circle generation algorithm

Generate Circle when radius is 10 using breshenham's circle generation algorithm
Let Circle radius is 10 X=0 and y=r or y=10 P=3-2*10; p=-17

(x,y)	(y,x)	(-x,y)	(-y,x)	(-x,-y)	(-y,-x)	(x,-y)	(y,-x)	(p<0) P=p+4x+6; x=x+1 (p>=0) P=p+4(x-y)+10 x=x+1, y=y-1
(0,10)	(10,0)	(-0,10)	(-10,0)	(-0,-10)	(-10,-0)	(0,-10)	(10,-0)	P=-17
(1,10)	(10,1)	(-1,10)	(-10,1)	(-1,-10)	(-10,-1)	(1,-10)	(10,-1)	P= -7
(2,10)	(10,2)	(-2,10)	(-10,2)	(-2,-10)	(-10,-2)	(2,-10)	(10,-2)	P=7
(3,9)	(9,3)	(-3,9)	(-9,3)	(-3,-9)	(-9,-3)	(3,-9)	(9,-3)	P=-7
(4,9)	(9,4)	(-4,9)	(-9,4)	(-4,-9)	(-9,-4)	(4,-9)	(9,-4)	P=15
(5,8)	(8,5)	(-5,8)	(-8,5)	(-5,-8)	(-8,-5)	(5,-8)	(8,-5)	P=13
(6,7)	(7,6)	(-6,7)	(-7,6)	(-6,-7)	(-7,-6)	(6,-7)	(7-6)	P=19
(7,6)	(6,7)	(-7,6)	(-6,7)	(-7,-6)	(-6,-7)	(7,-6)	(6-7)	don't Draw These Points Because (x>y). Stop Algorithm when X>=Y.

Question

- **Generate Circle using breshenham's Circle generation algorithm when radius is 5.**

(x,y)	(y,x)	(-x,y)	(-y,x)	(-x,-y)	(-y,-x)	(x,-y)	(y,-x)	(p<0) P=p+4x+6; x=x+1 (p>=0) P=p+4(x-y)+10 x=x+1, y=y-1
(0,5)	(5,0)	(-0,5)	(-5,0)	(-0,-5)	(-5-0)	(0,-5)	(5,-0)	P=--7
(1,5)	(5,1)	(-1,5)	(-5,1)	(-1,-5)	(-5,-1)	(1,-5)	(5,-1)	P= 3
(2,4)	(4,2)	(-2,4)	(-4,2)	(-2,-4)	(-4,-2)	(2,-4)	(4,-2)	P=5
(3,3)	(3,3)	(-3,3)	(-3,3)	(-3,-3)	(-3,-3)	(3,-3)	(3,-3)	Algorithm will stop when x==y or x>y (don't calculate next points) Draw all the calculated points