Group - C: Computer Graphies

Introduction: Co-ordinate System, Information Handling Software, Graphics Software, Area of Application,Translation, Rotation, Scaling, Matrix Representation. Homogeneous Co-ordinate System. Composite Transformation, Inverse Transformation, Computer Art, Animation, Morphing, Projection & Clipping, 2D & 3D Transformations. Lines, Curves and their presentations.

COMPUTER GRAPHICS

INTRODUCTION
PART-A

1.Define Computer Graphics.

Computer Graphics remains one of the most existing and rapidly growing computer fields. Computer graphics may be defined as the pictorial representation or graphical representation of objects in a computer

2.Write short notes on video controller.

Video controller is used to control the operation of the display device.A fixed area of the system is reserved for the frame buffer and the video controller is given direct access to the frame buffer memory Here, the frame buffer can be anywhere in the system memory, and the video controller accesses the frame buffer to refresh the screen. In addition to the video controller, more sophisticated raster systems employ other processors as coprocessor sand accelerators to implement various graphics operations.

3.Write notes on Graphics controller?

An application program is input and stored in the system memory along with a graphics package. Graphics commands in the application program are translated by the graphics package into a display file stored in the system memory. This display file is then accessed by the display processor to refresh the screen. The display processor cycles through each command in the display file program once during every refresh cycle. Sometimes the display processor in a random-scan system is referred to as a display processing unit or a graphics controller

4.List out a few attributes of output primitives?

Attributes are the properties of the output primitives; that is, an attribute describes how a particular primitive is to be displayed. They include intensity and color specifications, line styles, text styles, and area-filling patterns. Functions within this category can be usedto set attributes for an individual primitive class or for groups of output primitives.

5.What is vertical retrace of the electron beam?

In raster scan display at the end of one frame the electron beam returns to the left top corner of the screen to start the next frame is called vertical retrace of the electron beam.

6. Define persistence, resolution and aspect ratio.

Persistence is defined as the time it takes the emitted light from the screen to decay to one tenth of its original intensity.The maximum number of points that can be displayed without overlap on a CRT is referred to as the resolution.Aspect ratio is the ratio of the vertical points to horizontal points necessary to produce equal length lines in both directions on the screen.

7.What is the major difference between symmetrical DDA and simple DDA

Simple DDA	Symmetric DDA
In simple DDA,is transformed to $m=e\Delta x/e\Delta y$ where e, call it the increment factor, is a positive real number	In symmetric DDA, *e* is chosen such that *though* both the co-ordinates of the resultant points has tobe rounded off, it can be done so very efficiently,thus quickly.
*e*is chosen assuch that one of the coordinate is integral and only the other coordinate has to be rounded. i.e. P(i+1) = P(i)+())There one coordinate is being incremented by 1 and the other by $e*\Delta y$	*e* is chosen as 1/2^n where 2^(n-1) <= $\max(\lvert\Delta x\rvert,\lvert\Delta y\rvert)$ < 2^n. In other words the length of the line is taken to be 2^n aligned. The increments for the two coordinates are $e*\Delta x$ and $e*\Delta y$.

8. What is horizontal and vertical retrace?

The return to the left of the screen after refreshing each scan line is called as the horizontal retrace.Vertical retrace: At the end of each frame the electron beam returns to the top left corner of the screen to the beginning the next frame.

9. What is interlaced refresh?

Each frame is refreshed using two passes. In the first pass, the beam sweeps across every other scan linefrom top to bottom. Then after the vertical retrace, the beam traces out the remaining scan lines.

10. What is a raster scan system?

In a raster scan system the electron beam is swept across the screen, one row at a time 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. Picture information is stored in a memory area called refresh buffer or frame buffer. Most suited for scenes with subtle shading and color patterns.

11. What is a random scan system?

In random scan display unit, a CRT has the electron beam directed only to the parts of the screen where a picture is to be drawn. This display is also called as vector displays. Picture definition is stored as asset of line drawing commands in a memory referred to the refresh display file or display list or display program.

12. Write down the attributes of characters.

The appearance of displayed characters is controlled by attributes such as font, size, color and orientation. Attributes can be set both for entire character strings (text) and for individual characters defined as marker symbols. The choice of font gives a particular design style. Characters can also be displayed as underlined, in boldface, in italics and in outline or shadow styles.

13. What is scan conversion and what is a cell array?

Digitizing a picture definition given in an application program into a set of pixel intensity values for storage in the frame buffer by the display processor is called scan conversion.The cell array is a primitive that allow users to display an arbitrary shape defined as a two dimensional grid pattern.

14. Write down any two line attributes. (AU NOV/DEC 2011)

The basic attributes of a straight line segment are its:

- Type: solid, dashed and dotted lines.
- Width: the thickness of the line is specified.
- Color: a color index is included to provide color or intensity properties.

15. What are line caps?

The shape of the line ends are adjusted to give a better appearance by adding line caps.

Butt cap : obtained by adjusting the end positions of the component parallel lines so that the thickline is displayed with square ends that is perpendicular to the line path.

Round cap: obtained by adding a filled semicircle to each butt cap.

Projecting square cap: extend the line and add butt caps that are positioned one-half of the linewidth beyond the specified endpoints.

16. What are different methods of smoothly joining two line segments?

Miter joins: Accomplished by extending the outer boundaries of each of the two lines until theymeet.

Round join: produced by capping the connection between the two segments with a circularboundary whose diameter is equal to the line width.

Bevel join: generated by displaying the line segments with butt caps and filling in the triangulargap where the segments meet.

17. Write down the attributes of characters.

The appearance of displayed characters is controlled by attributes such as font, size, color and orientation. Attributes can be set both for entire character strings (text) and for individual characters defined as marker symbols. The choice of font gives a particular design style. Characters can also be displayed as underlined, in boldface, in italics and in outline or shadow styles.

18. Briefly explain about the unbundled and bundled attributes.

Unbundled attributes: how exactly the primitive is to be displayed is determined by its attribute setting.These attributes are meant to be used with an output device capable of displaying primitives the wayspecified.

Bundled attributes: when several kinds of output devices are available at a graphics installation, it is convenient for a user to be able to say how attributes are to be interpreted on different o/p devices. This is accomplished by setting up tables for each output device that lists sets of attribute value that are to be used on that device to display each primitive type. A particular set of attribute values for a primitive on each o/p device is then chosen by specifying the appropriate table index. Attributes specified in this manner is called as bundled attribute.

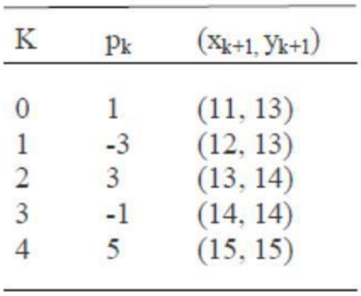
19. Digitize a line from (10,12) to (15,15) on a raster screen using Bresenham's straight linealgorithm.

The line has a slope of 0.6 with$\Delta x=5$, $\Delta y=3$

The initial decision parameter has the value$P0=2\Delta y-\Delta x=6-5=1$

And the increments for calculating successive decision parameters are$2\Delta y-2\Delta x=6-10=-4$

We plot the point (x0.y0) =(10,12), and determine successive pixel positions along theline path from the decision parameter as



K	p_k	(x_{k+1}, y_{k+1})
0	1	(11, 13)
1	-3	(12, 13)
2	3	(13, 14)
3	-1	(14, 14)
4	5	(15, 15)

20. Define pixel.

Pixel is a shortened form of picture element. Each screen point is referred to as pixel or pel

21. Define aliasing.

Displayed primitives generated by the raster algorithms have a jagged, stair step appearance because the sampling process digitizes coordinate points on an object to discrete integer pixel positions. This distortion of information due to low frequency sampling is called aliasing

22. What is antialiasing?

Appearance of displayed raster lines by applying antialiasing methods that compensate for the undersampling process.

Nyquist sampling frequency: to avoid losing information, the sampling frequency to at least twice that ofthehighest frequency occurring in the object. Fs=2*fmax.

23. What is antialiasing by super sampling or post filtering?

This is a technique of sampling object characteristics at a high resolution and displaying results at a lower resolution.

24. What is antialiasing by area sampling or prefiltering?

An alternative to super sampling is to determine pixel intensity by calculating areas of overlap of eachpixel with the objects to be displayed .antialiasing by computing overlaps areas is referred to as areasampling or prefiltering.

25. What is antialiasing by pixel phasing?

Raster objects can be antialiased by shifting the display location of pixel areas. This is applied by ―micropositioning‖ the electron beam in relation to object geometry.

Part B

1. Explain the Bresenham‘s line drawing algorithm with example.

BRESENHAM’S LINE ALGORITHM

1. Input the two line endpoints and store the left end point in (x0,y0)
2. load (x0,y0) into frame buffer, ie. Plot the first point.
3. Calculate the constants Δx, Δy, 2Δy and obtain the starting value for the decision parameter as P0 = 2Δy-Δx
4. At each xkalong the line, starting at k=0 perform the following test

If Pk< 0, the next point to plot is(xk+1,yk) and
Pk+1 = Pk+ 2Δy

Otherwise, the next point to plot is (xk+1,yk+1) and
Pk+1 = Pk+ 2Δy - 2Δx 5. Perform step4 Δx times.

Implementation of Bresenham Line drawing Algorithm voidlineBres (int xa,intya,intxb, int yb)

```
 {
int dx = abs( xa – xb) , dy = abs (ya - yb);
int p = 2 * dy – dx;
inttwoDy = 2 * dy, twoDyDx = 2 *(dy - dx);
int x , y, xEnd; /* Determine which point to use as start, which as end *
/ if (xa> x b )
{
x = xb;
y = yb;
xEnd = xa;
}
else
{
x = xa;
y = ya;
xEnd = xb;
}
setPixel(x,y);
while(x<xEnd)
{
x++;
if (p<0) p+=twoDy;
else
{
y++;
p+=twoDyDx;
}
setPixel(x,y);
}
}
```

Example : Consider the line with endpoints (20,10) to (30,18)

The line has the slope m= (18-10)/(30-20)=8/10=0.8

∆x = 10∆y=8

The initial decision parameter has the value p0 = 2∆y- ∆x = 6
and the increments for calculating successive decision parameters are

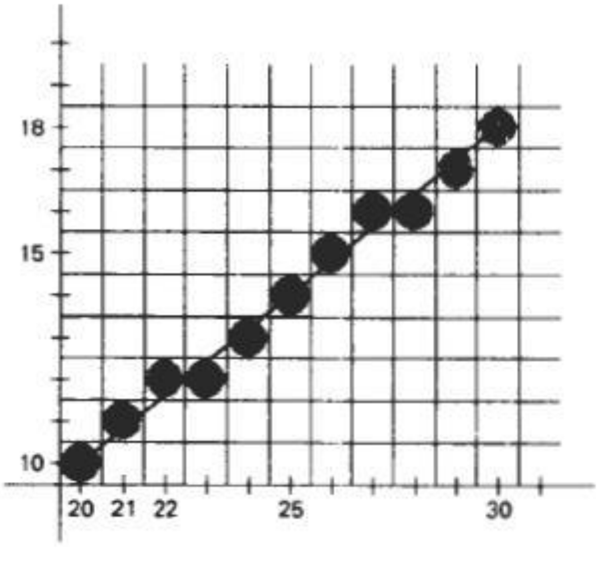
2∆y=16 2∆y-2 ∆x= -4

We plot the initial point (x0,y0) = (20,10) and determine successive pixel positions along the line path from the decision parameter as

TABULATION:

k	pk	(xk+1, yK+1)
0	6	(21,11)
1	2	(22,12)
2	-2	(23,12)
3	14	(24,13)
4	10	(25,14)
5	6	(26,15)
6	2	(27,16)
7	-2	(28,16)
8	14	(29,17)
9	10	(30,18)

RESULT:



Advantages

1. Algorithm is Fast
2. Uses only integer calculations

Disadvantages It is meant only for basic line drawing

2. Explain the midpoint circle drawing algorithm. Assume 10 cm as the radius and co-ordinate origin as the center of the circle.

Given a circle radius r=10

The circle octant in the first quadrant from x=0 to x=y.

The initial value of the decision parameter is

$P_0=1-r = -9$

For the circle centered on the coordinate origin, the initial point

is $(X_0, y0)=(0, 10)$

and initial increment terms for calculating the decision parameters are

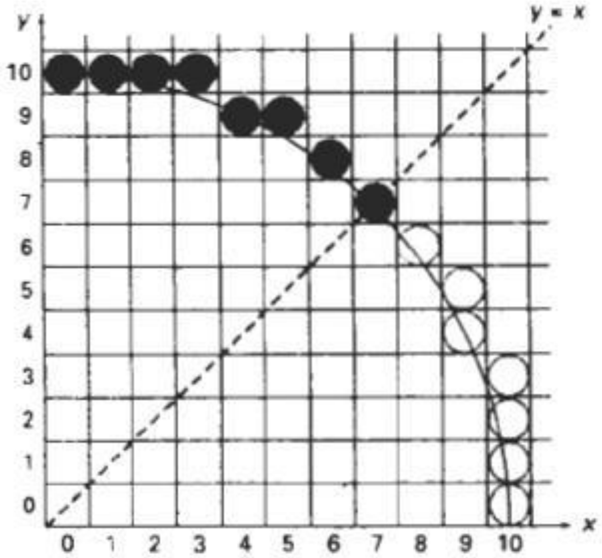
$2x_0=0, 2y_0=20$

Successive midpoint decision parameter values and the corresponding coordinate positions along the circle path are listed in the following table.

TABULATION :

k	**p_k**	**(xk+1, yk-1)**	**$2x_{k+1}$**	**$2y_{k+1}$**
0	-9	(1,10)	2	20
1	-6	(2,10)	4	20
2	-1	(3,10)	6	20
3	6	(4,9)	8	18
4	-3	(5,9)	10	18
5	8	(6,8)	12	16
6	5	(7,7)	14	14

RESULT :



Implementation of Midpoint Circle Algorithm

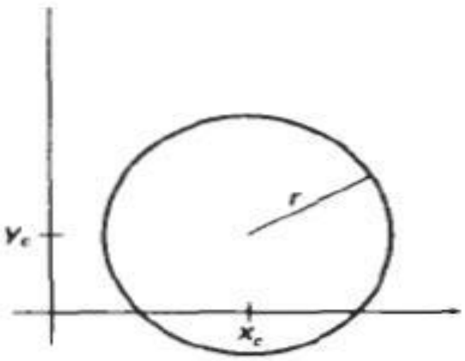
```
voidcircleMidpoint (int xCenter, int yCenter, int
radius) {
int x = 0; int y = radius;
int p = 1 - radius;
voidcirclePlotPoints (int, int, int, int); /* Plot first set of points
*/ irclePlotPoints (xCenter, yCenter, x, y);
while (x < y)
 {
 x++ ;
if (p < 0) p +=2*x +1;

else
{
y--; p +=2* (x - Y) + 1;
}
circlePlotPoints(xCenter, yCenter, x, y)
}
}
voidcirclePlotPolnts (int xCenter, int yCenter, int x, int y)
{
setpixel (xCenter + x, yCenter + y ) ;
setpixel (xCenter - x. yCenter + y);
setpixel (xCenter + x, yCenter - y);
setpixel (xCenter - x, yCenter - y ) ;
setpixel (xCenter + y, yCenter + x);
setpixel (xCenter - y , yCenter + x);
setpixel (xCenter t y , yCenter - x);
setpixel (xCenter - y , yCenter - x);
}
```

3. Explain about Bresenham’s circle generating algorithm with example.

CIRCLE-GENERATING ALGORITHMS

A circle is defined as a set of points that are all the given distance (xc,yc).



$(x – x_c)^2+ (y – y_c)^2 = r^2$

Midpoint circle Algorithm

1. Input radius r and circle center (xc,yc) and obtain the first point on the circumference of the circle centered on the origin as (x0,y0) = (0,r)
2. Calculate the initial value of the decision parameter as P0=(5/4)-r
3. At each xk position, starting at k=0, perform the following test.
If Pk<0 the next point along the circle centered on (0,0) is (xk+1,yk)
and Pk+1=Pk+2xk+1+1
Otherwise the next point along the circle is (xk+1,yk-1) and
Pk+1=Pk+2xk+1+1-2 yk+1 Where 2xk+1=2xk+2 and 2yk+1=2yk-2
4. Determine symmetry points in the other seven octants.
5. Move each calculated pixel position (x,y) onto the circular path centered at (xc,yc) and plot the coordinate values. x=x+xc y=y+yc 6. Repeat step 3 through 5 until x>=y.
6.

4.Explain in detail about Bresenham‘s ellipse generating algorithm. Give example.

BRESENHAM’S ELLIPSE GENERATING ALGORITHM

Mid point Ellipse Algorithm

1. Input rx,ry and ellipse center (xc,yc) and obtain the first point on an ellipse centered on the origin as (x0,y0) = (0,ry)
2. Calculate the initial value of the decision parameter in region 1 as

P10=ry2-rx2ry +(1/4)rx2

At each xk position in region1 starting at k=0 perform the following test.

3. If P1k<0, the next point along the ellipse centered on (0,0) is (xk+1, yk) and

p1k+1 = p1k +2 ry2xk +1 + ry2

Otherwise the next point along the ellipse is (xk+1, yk-1) and
p1k+1 = p1k +2 ry2xk +1 - 2rx2 yk+1 + ry2
with 2 ry2xk +1 = 2 ry2xk + 2ry2 2 rx2yk +1 = 2 rx2yk +
2rx2 And continue until 2ry2 x>=2rx2 y

5. Calculate the initial value of the decision parameter in region 2 using the last point (x0,y0) is the last position calculated in region 1.

p20 = ry2(x0+1/2)2+rx2(yo-1)2 – rx2ry2

6. At each position yk in region 2, starting at k=0 perform the following test, If p2k>0 the next point along the ellipse centered on (0,0) is (xk,yk-1) and

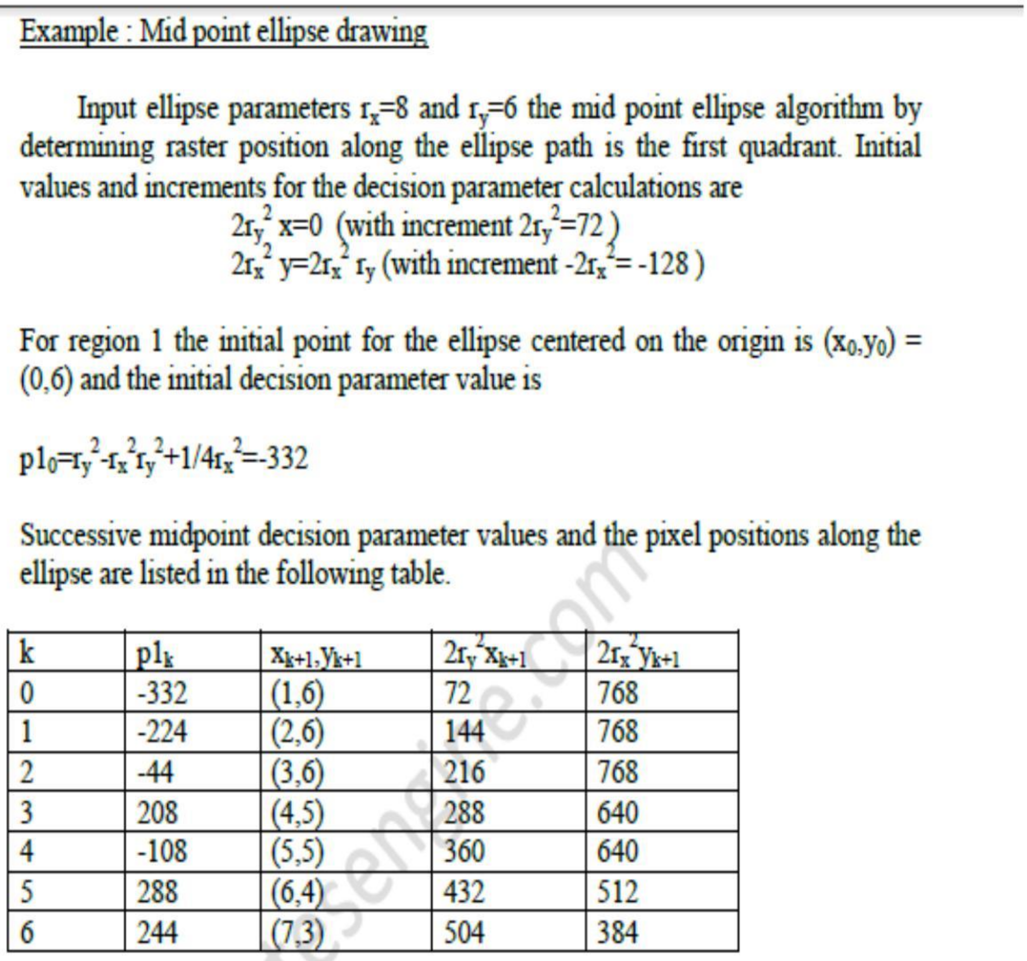
p2k+1 = p2k – 2rx2yk+1+rx2 Otherwise the next point along the ellipse is (xk+1,yk-1) and

p2k+1 = p2k + 2ry2xk+1 – 2rxx2yk+1 + rx2 Using the same incremental calculations for x any y as in region 1.

6. Determine symmetry points in the other three quadrants.
7. Move each calculate pixel position (x,y) onto the elliptical path centered on (xc,yc) and plot the coordinate values

x=x+xc, y=y+yc

1. Repeat the steps for region1 unit 2ry2x>=2rx2y



Example : Mid point ellipse drawing

Input ellipse parameters r_x=8 and r_y=6 the mid point ellipse algorithm by determining raster position along the ellipse path is the first quadrant. Initial values and increments for the decision parameter calculations are

$$2r_y^2\, x=0 \;(\text{with increment } 2r_y^2=72)$$
$$2r_x^2\, y=2r_x^2\, r_y \;(\text{with increment } -2r_x^2=-128)$$

For region 1 the initial point for the ellipse centered on the origin is (x_0,y_0) = (0,6) and the initial decision parameter value is

$p1_0=r_y^2-r_x^2r_y^2+1/4r_x^2=-332$

Successive midpoint decision parameter values and the pixel positions along the ellipse are listed in the following table.

k	$p1_k$	x_{k+1},y_{k+1}	$2r_y^2 x_{k+1}$	$2r_x^2 y_{k+1}$
0	-332	(1,6)	72	768
1	-224	(2,6)	144	768
2	-44	(3,6)	216	768
3	208	(4,5)	288	640
4	-108	(5,5)	360	640
5	288	(6,4)	432	512
6	244	(7,3)	504	384

5. Explain Line drawing algorithm.

DIGITAL DIFFERENTIAL ANALYZER (DDA) ALGORTIHM

Algorithm

```
#define ROUND(a) ((int)(a+0.5))
voidlineDDA (int xa, int ya, int xb, int yb)
 {
int dx = xb - xa, dy = yb - ya, steps, k;
floatxIncrement, yIncrement, x = xa, y = ya;
if (abs (dx) > abs (dy) steps = abs (dx) ;
else steps = abs dy);
xIncrement = dx / (float) steps;
yIncrement = dy / (float) steps
setpixel (ROUND(x), ROUND(y) ) :
for (k=0; k<steps; k++)
{
x += xIncrement;
 y += yIncrement;
setpixel (ROUND(x), ROUND(y));
}
}
```

Algorithm Description:

Step 1 :Accept Input as two endpoint pixel positions
Step 2: Horizontal and vertical differences between the endpoint positions *are* assigned to parameters dx and dy (Calculate dx=xb-xa and dy=yb-ya).
Step 3: The difference with the greater magnitude determines the value of parameter steps.
Step 4 : Starting with pixel position (xa, ya), determine the offset needed at each step to generate the next pixel position along the line path.
Step 5: loop the following process for steps number of times
a. Use a unit of increment or decrement in the x and y direction
b. if xa is less than xb the values of increment in the x and y directions are 1 and
m c. if xa is greater than xb then the decrements -1 and – m are used.

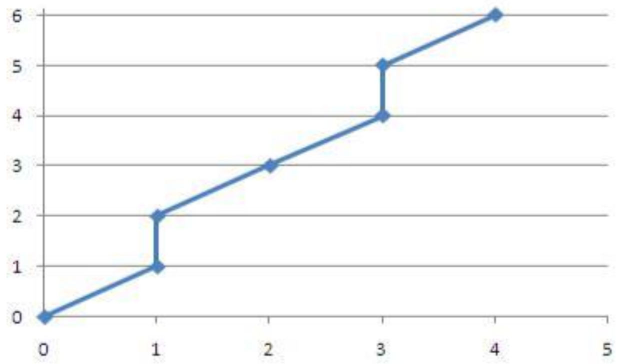
Example: Consider the line from (0,0) to (4,6)

1. xa=0, ya =0 and xb=4 yb=6
2. dx=xb-xa = 4-0 = 4 and dy=yb-ya=6-0= 6
3. x=0 and y=0
4 > 6 (false) so, steps=6
5. Calculate xIncrement = dx/steps = 4 / 6 = 0.66 and yIncrement = dy/steps =6/6=1
6. Setpixel(x,y) = Setpixel(0,0) (Starting Pixel Position)
7. Iterate the calculation for xIncrement and yIncrement for steps(6) number of times

8. Tabulation of the each iteration

k	x	Y	Plotting points (Rounded to Integer)
0	0+0.66=0.66	0+1=1	(1,1)
1	0.66+0.66=1.32	1+1=2	(1,2)
2	1.32+0.66=1.98	2+1=3	(2,3)
3	1.98+0.66=2.64	3+1=4	(3,4)
4	2.64+0.66=3.3	4+1=5	(3,5)

RESULT:



Advantages of DDA Algorithm

1. It is the simplest algorithm
2. It is a is a **faster method** for calculating pixel positions

Disadvantages of DDA Algorithm

1. Floating point arithmetic in DDA algorithm is still time-consuming
2. End point accuracy is poor

6. Write short notes on Video display devices.

2.1 VIDEO DISPLAY DEVICES

Typically, the primary output device in a graphics system is a video monitor.The operation of most video monitors is based on the standard cathode-ray* tube (CRT) design, but several other technologies exist and solid-state* monitors may eventually predominate.

REFRESH CATHODE-RAY TUBES

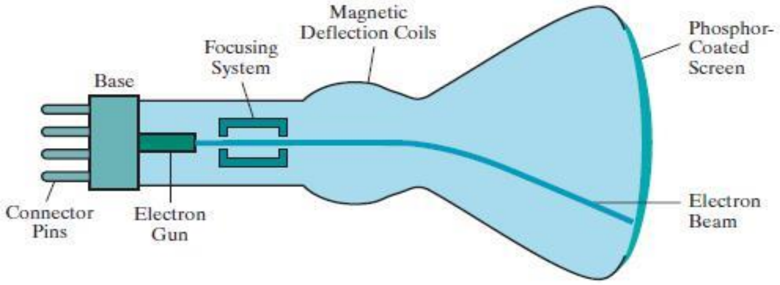


FIGURE 2-2 Basic design of a magnetic-deflection CRT.

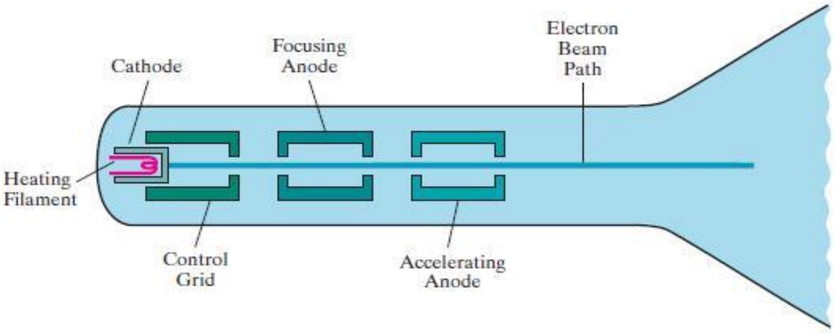


FIGURE 2-3 Operation of an electron gun with an accelerating anode.

1. Figure 2-2 illustrates the basic operation of a CRT. A beam of electrons (*cathode rays*), emitted by an electron gun, passes through focusing and deflection systems that direct the beam toward specified positions on the phosphor-coated screen.
2. The phosphor then emits a small spot of light at each position contacted by the electron beam. Because the light emitted by the phosphor fades very rapidly,

3. Some method is needed for maintaining the screen picture. One way to do this is to store the picture information as a charge distribution within the CRT.
4. This charge distribution can then be used to keep the phosphors activated. However, the most common method now employed for maintaining phosphor glow is to redraw the picture repeatedly by quickly directing the electron beam back over the same screen points.
5. This type of display is called a **refreshCRT,** and the frequency at which a picture is redrawn on the screen is referred to as the **refresh rate.** The primary components of an electron gun in a CRT are the heated metal cathode and a control grid (Fig. 2-3).
6. Heat is supplied to the cathode by directing a current through a coil of wire, called the filament, inside the cylindrical cathode structure. This causes electrons to be ―boiled off‖ the hot cathode surface.
7. In the vacuum inside the CRT envelope, the free, negatively charged electrons are then accelerated toward the phosphor coating by a high positive voltage. The accelerating voltage can be generated with a positively charged metal coating on the inside of the CRT envelope near the phosphor screen, or an accelerating anode, as in Fig. 2-3, can be used to provide the positive voltage.
8. Sometimes the electron gun is designed so that the accelerating anode and focusing system are within the same unit.
9. Intensity of the electron beam is controlled by the voltage at the control grid, which is a metal cylinder that fits over the cathode. A high negative voltage applied to the control

   grid will shut off the beam by repelling electrons and stopping them from passing through the small hole at the end of the control-grid structure.
10. A smaller negative voltage on the control grid simply decreases the number of electrons passing through. Since the amount of light emitted by the phosphor coating depends on the number of electrons striking the screen, the brightness of a display point is controlled by varying the voltage on the control grid.
11. The focusing system in a CRT forces the electron beam to converge to a small cross section as it strikes the phosphor. Otherwise, the electrons would repel eachother, and the beam would spread out as it approaches the screen. Focusing is accomplished with either electric or magnetic fields.
12. With electrostatic focusing, the electron beam is passed through a positively charged metal cylinder so that electrons along the centerline of the cylinder are in an equilibrium position.
13. This arrangement forms an electrostatic lens, as shown in Fig. 2-3, and the electron beam is focused at the center of the screen in the same way that an optical lens focuses a beam of light at a particular focal distance. Similar lens focusing effects can be accomplished with a magnetic field set up by a coil mounted around the outside of the CRT envelope, and magnetic lens focusing usually produces thesmallest spot size on the screen.
14. Additional focusing hardware is used in high-precision systems to keep the beam in focus at all screen positions. The distance that the electron beam must travel to different points on the screen varies because the radius of curvature for most CRTs is greater than the distance from the focusing system to the screen center.
15. Therefore, the electron beam will be focused properly only at the center of the screen. As the beam moves to the outer edges of the screen, displayed images become blurred. To compensate for this, the system can adjust the focusing according to the screen position of the beam.
16. As with focusing, deflection of the electron beam can be controlled with either electric or magnetic fields. Cathode-ray tubes are now commonly constructed with magnetic-deflection coils mounted on the outside of the CRT envelope, as illustrated in Fig. 2-2.
17. Two pairs of coils are used for this purpose. One pair is mounted on the top and bottom of the CRT neck, and the other pair is mounted on opposite sides of the neck. The magnetic field produced by each pair of coils results in a transverse deflection force that is perpendicular to both the direction of the magnetic field and the direction of travel of the electron beam.
18. Horizontal deflection is accomplished with one pair of coils, and vertical deflection with the other pair. The proper deflection amounts are attained by adjusting the currentthrough

the coils. When electrostatic deflection is used, two pairs of parallel plates are mounted inside the CRT envelope.

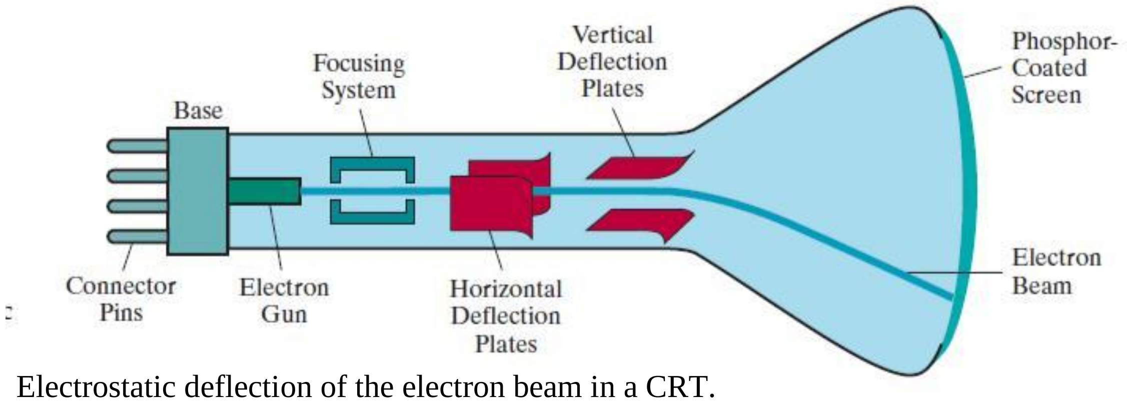


Fig Electrostatic deflection of the electron beam in a CRT.

19. One pair of plates is mounted horizontally to control vertical deflection, and the other pair is mounted vertically to control horizontal deflection (Fig. 2-4). Spots of light are produced on the screen by the transfer of the CRT beam energy to the phosphor. When the electrons in the beam collide with the phosphor coating, they are stopped and their kinetic energy is absorbed by the phosphor. Part of the beam energy is converted by friction into heat energy, and the remainder causes electrons in the phosphor atoms to move up to higher quantum-energy levels. After a short time, the ―excited‖ phosphor electrons begin dropping back to their stable ground state, giving up their extra energy as small quantum‘s of light energy called photons.

20. Different kinds of phosphors are available for use in CRTs. Besides color, a major difference between phosphors is their **persistence:** how long they continue to emit light (that is, how long before all excited electrons have returned to the ground state) after the CRT beam is removed. Persistence is defined as the time that it takes the emitted light from the screen to decay to one-tenth of its original intensity.

21. Lower-persistence phosphors require higher refresh rates to maintain a picture on the screen without flicker. A phosphor with low persistence can be useful for animation, while high-persistence phosphors are better suited for displaying highly complex, static pictures. Although some phosphors have persistence values greater than 1 second, general-purpose graphics monitors are usually constructed with persistence in the range from 10 to 60 microseconds.

22. Figure 2-5 shows the intensity distribution of a spot on the screen. The intensity is greatest at the center of the spot, and it decreases with a Gaussian distribution out to the edges of the spot. This distribution corresponds to the cross-sectional electron density distribution of the CRT beam. The maximum number of points that can be displayed without overlap on aCRT is referred to as the **resolution**

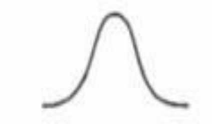


Fig 2.5 Intensity distribution of an illuminated Phosphor spot on a CRT screen

Fig 2.6 Two Illuminated phosphor spots are distinguishable when their separation is greater than the diameter at which a spot intensity has fallen to 60 percent of maximum.

8. Explain about Random scan systems.

RANDOM-SCAN DISPLAYS

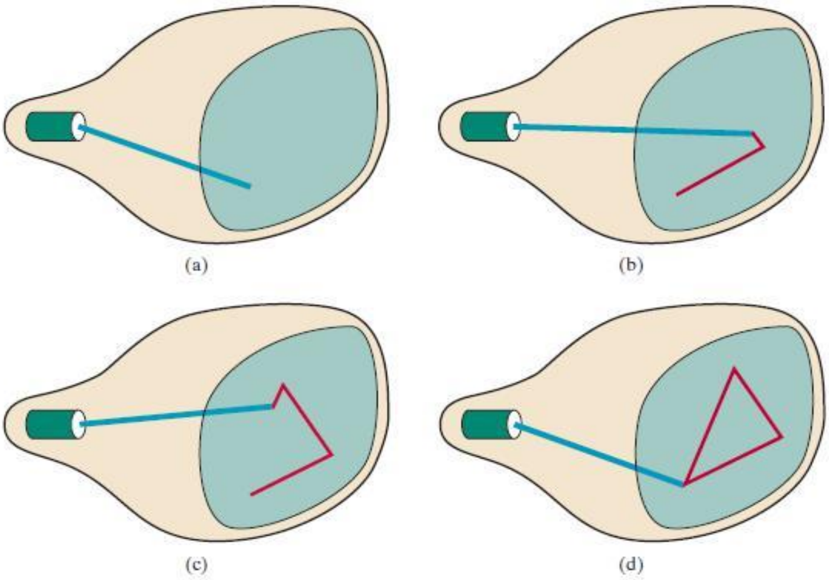
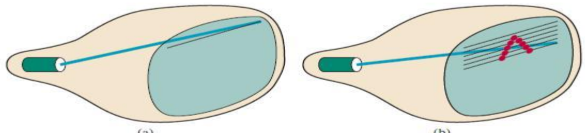


FIGURE 2–9 A random-scan system draws the component lines of an object in any specified order.

1. When operated as a **random-scan display** unit, a CRT has the electron beam directed only to those parts of the screen where a picture is to be displayed.
2. Pictures are generated as line drawings, with the electron beam tracing out the component lines one after the other. For this reason, random-scan monitors are also referred to as **vector displays** (or **stroke-writing displays** or **calligraphic displays**).
3. The component lines of a picture can be drawn and refreshed by a random-scan system in any specified order (Fig. 2-9).A pen plotter operates in a similar way and is an example of a random-scan, hard-copy device.
4. Refresh rate on a random-scan system depends on the number of lines to be displayed on that system. Picture definition is now stored as a set of line-drawing commands in an area of memory referred to as the **display list, refresh display file, vector file,** or **display program.**
5. To display a specified picture, the system cycles through the set of commands in the display file, drawing each component line in turn. After all line-drawing commands have been processed, the system cycles back to the first line command in the list.
6. Random-scan displays are designed to draw all the component lines of a picture 30 to 60 times each second, with up to 100,000 ―short‖ lines in the display list. When a small set of lines is to be displayed, each refresh cycle is delayed to avoid very high refresh rates, which could burn out the phosphor.
7. Random-scan systems were designed for line-drawing applications, such as architectural and engineering layouts, and they cannot display realistic shaded scenes. Since picture definition is stored as a set of line-drawing instructions rather than as a set of intensity values for all screen points, vector displays generally have higher resolutions than raster systems.
8. Also, vector displays produce smooth line drawings because the CRT beam directly follows the line path. A raster system, by contrast, produces jagged lines that are plotted as discrete point sets. However, the greater flexibility and improved line-drawing capabilities of raster systems have resulted in the abandonment of vector technology.
9. **Explain about Raster scan systems.**

RASTER-SCAN DISPLAYS



1. The most common type of graphics monitor employing a CRT is the **raster-scan display,** based on television technology. Ina raster-scan system, the electron beam is swept across the screen, one row at a time, from top to bottom.
2. Each row is referred to as a **scan line.** As the electron beam moves across a scan line, the beam intensity is turned on and off (or set to some intermediate value) to create a pattern of illuminated spots. Picture definition is stored in a memory area called the **refresh buffer** or **frame buffer,** where the term **frame** refers to the total screen area. This memory area holds the set of color values for the screen points.
3. These stored color values are then retrieved from the refresh buffer and used to control the intensity of the electron beam as it moves from spot to spot across the screen. In this way, the picture is ―painted‖ on the screen one scan line at a time, as demonstrated in Fig.
   2-7.
4. Each screen spot that can be illuminated by the electron beam is referred to as a **pixel** or **pel**(shortened forms of **picture element**). Since the refresh buffer is used to store the set of screen color values, it is also sometimes called a **color buffer.**
5. Also, other kinds of pixel information, besides color, are stored in buffer locations, so all the different buffer areas are sometimes referred to collectively as the ―frame buffer‖. The capability of a raster-scan system to store color information for each screen point makes it well suited for the realistic display of scenes containing subtle shading and color patterns.
6. Home television sets and printers are examples of other systems using raster-scan methods. Raster systems are commonly characterized by their resolution, which is the number of pixel positions that can be plotted.
7. Another property of video monitors is **aspect ratio,** which is now often defined as the number of pixel columns divided by the number of scan lines that can be displayed by the system. (Sometimes the term aspect ratio is used to refer to the number of scan lines divided by the number of pixel columns.) Aspect ratio can also be described as the number of horizontal points to vertical points (or vice versa) necessary to produce equal-length lines in both directions on the screen.
8. The number of bits per pixel in a frame buffer is sometimes referred to as either the **depth** of the buffer area or the number of **bit planes.** Also, a frame buffer with one bit per pixel is commonly called a **bitmap,** and a frame buffer with multiple bits per pixel is a **pixmap.**

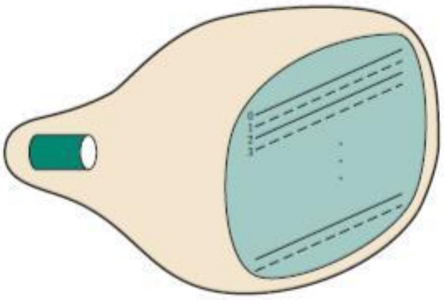


FIGURE 2-8 Interlacing scan lines on a raster-scan display. First, all points on the even-numbered (solid) scan lines are displayed; then all points along the odd-numbered (dashed) lines are displayed.

9. On some raster-scan systems and TV sets, each frame is displayed in two passes using an *interlaced* refresh procedure. In the first pass, the beam sweeps across every other scan line from top to bottom. After the vertical retrace, the beam then sweeps out the remaining scan lines (Fig. 2-8).
10. Interlacing of the scan lines in this way allows us to see the entire screen displayed in one-half the time it would have taken to sweep across all the lines at once from top to bottom. This technique is primarily used with slower refresh rates. On an older, 30 frame per- second, non-interlaced display, for instance, some flicker is noticeable. But with interlacing, each of the two passes can be accomplished in 1/60 of a second, which brings the refresh rate nearer to 60 frames per second. This is an effective technique for avoiding flicker provided that adjacent scan lines contain similar display information.

TWO DIMENSIONAL GRAPHICS

PART-A

1.What are homogeneous co-ordinates?

To express any 2D transformation as a matrix multiplication, each Cartesian co-ordinate position (x, y) is represented with the homogeneous coordinate triple (xh, yh, h) where

$x=\frac{xh}{h}$ and $y=\frac{yh}{h}$. Thus the general homogeneous coordinate representation can also be written as (h.x, h.y, h). The homogeneous parameter h can be any nonzero value. A convenient choice is to set h=1. Each 2D position is then represented by the homogeneous coordinates (x, y, 1).

2.What are the basic transformations?

Translation : Translation is applied to an object by repositioning it along a straight line path from one coordinate location to another. x1=x+Tx y1=y+Ty (Tx,Ty) – translation vector or shift vector

Rotation: A two dimensional rotation is applied to an object by repositioning it along a circularpath in the xy plane.

$$P1=R.P$$

$$R=\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta\cos\theta & \end{bmatrix} \quad \theta\text{- rotation angle}$$

Scaling: A scaling transformation alters the size of an object .

x1=x.Sx y1=y.SySx and Sy are scaling factors.

3.How can we express a two dimensional geometric transformation?

We can express two-dimensional geometric transformations as 3 by 3 matrixoperators, so that sequences of transformations can be concatenated into a single composite matrix. This is an efficient formulation, since it allows us to reducecomputations by applying the composite matrix to the initial coordinate positionsof an object to obtain the final transformed positions

4.What is uniform and differential scaling?

Uniform scaling: Sx and Sy are assigned the same value.

Differential scaling: unequal values for Sx and Sy.

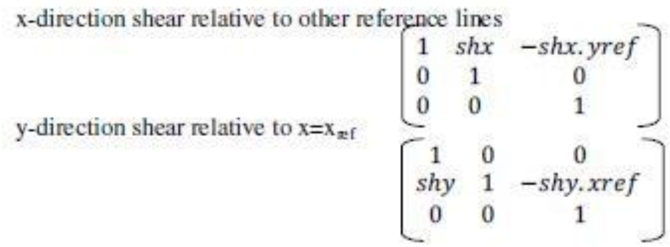
5.Define reflection.

A reflection is a transformation that produces a mirror image of an object.
By line y =0(x-axis)

Transformation matrix =

6.Write down the shear transformation matrix. (AU NOV/DEC 2012)

A transformation that distorts the shape of an object such that the transformed shape appears as if theobject is composed of internal layers that had been caused to slide over each other is called shear.x-direction shear relative to x axis



x-direction shear relative to other reference lines

$$\begin{pmatrix} 1 & shx & -shx.yref \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

y-direction shear relative to $x=x_{ref}$

$$\begin{pmatrix} 1 & 0 & 0 \\ shy & 1 & -shy.xref \\ 0 & 0 & 1 \end{pmatrix}$$

7.What is the rule of clipping? (AU MAY/JUNE 2012)

For the viewing transformation, we are needed to display only those picture parts that are within thewindow area. Everything outside the window is discarded. Clipping algorithms are applied in world coordinates,so that only the contents of the window interior are mapped to device co-ordinates.

8.Define clipping.(AU NOV/DEC 2012)

Any procedure that identifies those portions of a picture that are either inside or outside of a specifiedregion of space is referred to as a clipping algorithm or clipping. The region against which an object is tobe clipped is called as the clip window.

9.Define Translation?

A translation is applied to an object by repositioning it along a straight-line path from one coordinate location to another. We translate a two-dimensional point by adding translation distances, *f,* and t,, to the original coordinate position (x, y) to move the point to a new position

(x ' , y')

x' = *x* + *t*,, y' = y + *t,*

The translation distance pair *(t,, t,)* is called a translation vector or shift vector.

10.Define scaling?

A scaling transformation alters the size of an object. This operation can be carried out for polygons by multiplying the coordinate values ***(x,*** y) of each vertex by scaling factors s, and s, to produce the transformed coordinates **(x',** y'):

Scaling factor s, scales objects in the **x** direction, while sy scales in their direction.

11. Define reflection?

A reflection is a transformation that produces a mirror image of an object. The mirror image for a two-dimensional reflection is generated relative to an **axis** of reflection by rotating the object 180" about the reflection axis. We can choose an axis of reflection in the *xy*plane or perpendicular to the *xy*plane. When the reflection axis is a line in the *xy*plane, the rotation path about this axis is in a plane perpendicular to the *xy*plane. For reflection axes that are perpendicular to the *xy*plane, the rotation path is in the *xy*plane.

12. What is shear?

A transformation that distorts the shape of an object such that the transformed shape appears as if the object were composed of internal layers that had been caused to slide over each other is called a shear. Two common shearing transformations are those that shift coordinate w values and those that shift y values.

13. What is affine transformation?

A coordinate transformation of the form

$$x' = a_{xx}x + a_{xy}y + b_x, \qquad y' = a_{yx}x + a_{yy}y + b_y$$

is called a two-dimensional affine transformation. Each of the transformed coordinates *x'* and y ' is a linear function of the original coordinates *x* and y, and parameters a,, and *bk*are constants determined by the transformation type. Affine transformations have the general properties that parallel lines are transformed into parallel lines and finite points map to finite points.Translation, rotation, scaling, reflection, and shear are examples of two-dimensional affine transformations.

14. What is viewing transformation?

The window defines *what* is to be viewed; the viewport defines *where* it is to be displayed.Often, windows and viewports are rectangles in standard position, with the rectangle edges parallel to the coordinate axes. Other window or viewport geometries,such as general polygon shapes and circles, are used in some applications,but these shapes take longer to process. In general, the mapping of a part of a world-coordinate scene to device coordinates is referred to as a **viewing transformation**

15. What are the various line clipping algorithm?

Cohen-Sutherland line clipping

Liang-Barsky line clipping

Nicholl-Lee-Nicholl line clipping

16. Differentiate window and viewport(AU NOV/DEC 2011)

Window	Viewport
A window is a world coordinate area selected for display	A viewport is an area on a display device to which the window is mapped
The window defines what is to be viewed	The viewport defines where it is to be displayed

17. What are the various polygon clipping algorithms?

Sutherland-Hodgenialpolvgon Clipping

Welter-Atherton Polygon Clipping

18. List the different types of text clipping methods available?

There are several techniques that can be used to provide text clipping in a graphics package. The clipping technique used will depend on the methods used to generate characters and the requirements of a particular application.The simplest method for processing character strings relative to a window boundary is to use the **all- or-none string-clipping** strategy. An alternative to rejecting an entire character string that overlaps a window boundary is to use the all-or-none **character-clipping** strategy.**A** final method for handling text clipping is to clip the components of individual characters. We now treat characters in much the same way that we treated lines. If an individual character overlaps a clip window boundary, we clip off the parts of the character that are outside the window (Fig. 6-30). Outline character fonts formed with line segments can be processed in this way using a line clipping algorithm.

19. How will you clip a point?

Assuming that the clip window is a rectangle in standard position, we save a point P = *(x,* y) for display if the following inequalities are satisfied:

$$xw_{min} \le x \le xw_{max}$$

$$yw_{min} \le y \le yw_{max}$$

Where the *edges of* the clip window (numi,, mum,, *yw,,, yiu,,,)* can be either the world-coordinate window boundaries or viewport boundaries. If any one of these four inequalities is not satisfied, the point is clipped (not saved for display).Although point clipping is applied less often than line or polygon clipping,some .applications may require a point clipping procedure. For example, point clipping can be applied to scenes involving explosions or sea foam that are modeled with particles (points) distributed in some region of the scene.

20. Give an example for text clipping?

We now treat characters in much the same way that we treated lines. If an individual character overlaps a clip window boundary, we clip off the parts of the character that are outside the window. Outline character fonts formed with line segments can be processed in this way using a line clipping algorithm.

21. Define Exterior clipping.

We have considered only procedures for clipping a picture to the interior of a screen by eliminating everything outside the clipping region. What is saved by these procedures is *inside* the region. In some cases, we want to do the reverse,that is, we want to clip a picture to the exterior of a specified region. The picture parts to be saved are those that are *outside* the region. This is referred to as exterior clipping.

22. Define curve clipping.

Curve-clipping procedures will involve non- linear equations, however, and this requires more processing than for objects with linear boundaries.The bounding rectangle for a circle or other curved object can beused first to test for overlap with a rectangular clip window. If the bounding rectangle for the object is completely inside the window, we save the object. If the rectangle is determined to be completely outside the window, we discard the object. In either case, there is no further computation necessary. But if the bounding rectangle test fails, we can look for other computation-saving approaches. For a circle, we can use the coordinate extents of individual quadrants and then octants for preliminary testing before calculating curve-window intersections.

23. Define window to viewport coordinate transformation.

Once object descriptions have been transferred to the viewing reference frame,we choose the window extents in viewing coordinates and select the viewport limits in normalized coordinates Object descriptions are then transferredto normalized device coordinates. We do this using a transformation that maintains the same relative placement of objects in normalized space as they had in viewing coordinates. If a coordinate position is at the center of the viewing window, for instance, it will be displayed at the center of the viewport.

1. Parametric representation of Line segment with endpoints (x1, y1) and (x2, y2)

$x = x_1 + u(x_2-x_1)$

$y = y_1 + u(y_2-y_1)$; $0<=u<=1$

2. Exterior of the window

– Intersection with outside the range u

3. Interior of the window

– Intersection with inside the range u

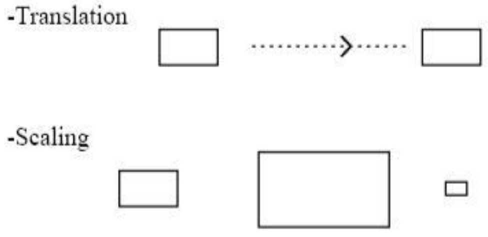
2. **Explain the rotational transformations.**

BASIC TWO-DIMENSIONAL GEOMETRIC TRANSFORMATIONS

Operations that are applied to the geometric description of an object to change its position, orientation, or size are called ***geometric transformations***. Geometric transformations can be used to describe how objects might move around in a scene during an animation sequence or simply to view them from another angle

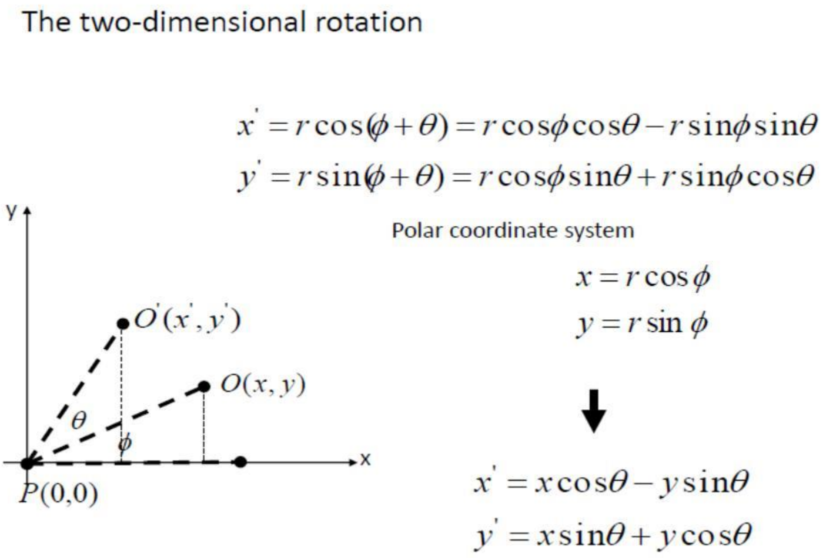
Geometric transformations

1. Translation
2. Rotation
3. Scaling
4. Reflection
5. shearing



Two-Dimensional Rotation

1. We generate a rotation transformation of an object by specifying a rotation axis and a rotation angle.
2. A two-dimensional rotation of an object is obtained by repositioning the object along a circular path in the xy plane.
3. Parameters for the two-dimensional rotation are
   –The rotation angle θ
   –A position (x,y) – rotation point (pivot point)



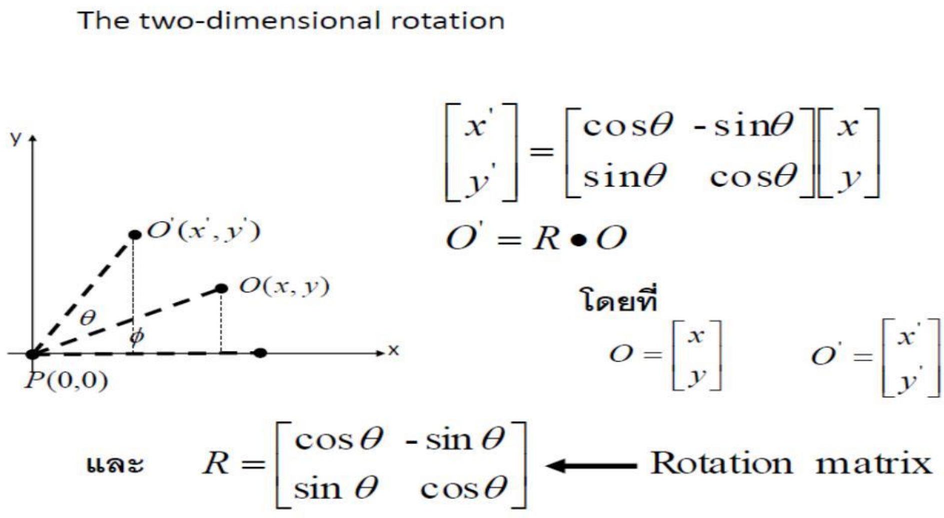
The two-dimensional rotation

$$x' = r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta$$
$$y' = r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta$$

Polar coordinate system

$$x = r\cos\phi$$
$$y = r\sin\phi$$

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$



The two-dimensional rotation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

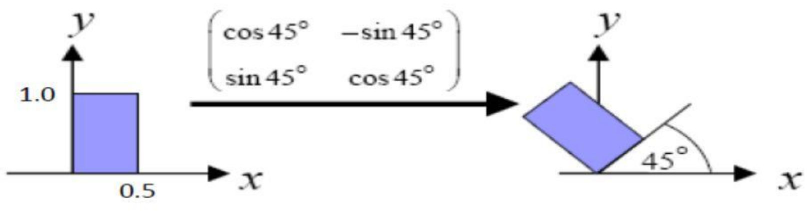
$$O' = R \bullet O$$

โดยที่

$$O = \begin{bmatrix} x \\ y \end{bmatrix} \qquad O' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

และ $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ ← Rotation matrix

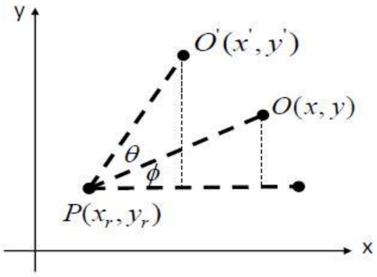
Ex. 1



y
1.0
0.5
x
$\begin{pmatrix} \cos 45° & -\sin 45° \\ \sin 45° & \cos 45° \end{pmatrix}$
y
45°
x

Rotation of a point about an arbitrary pivot position

$$x' = (x - x_r)\cos\theta - (y - y_r)\sin\theta + x_r$$
$$y' = (x - x_r)\sin\theta + (y - y_r)\cos\theta + y_r$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x - x_r \\ y - y_r \end{bmatrix} + \begin{bmatrix} x_r \\ y_r \end{bmatrix}$$

$$O' = R \bullet O^* + P$$

โดยที่

$$O^* = \begin{bmatrix} x - x' \\ y - y' \end{bmatrix}$$

3. **Explain Curve clipping algorithm**

Areas with curved boundaries can be clipped with methods similar to those discussed in the previous .sections. Curve-clipping procedures will involve nonlinear equations, however, and this requires more processing than for objects with linear boundaries. The bounding rectangle for a circle or other curved object can **be** used first to test for overlap with a rectangular clip window. If the bounding rectangle for
the object is completely inside the window, we save the object. If the rectangle is determined to be completely outs~deth e window, we discard the object. In either case, there is no further computation necessary. But if the bounding rectangle test fails, we can lwk for other computation-saving approaches. For a circle, we can use the coordinate extents of individual quadrants and then octants for preliminary
testing before calculating curve-window intersections. For an ellipse, we can test the coordinate extents of individual quadrants. Figure 6-27 illustrates circle clipping against a rectangular window.
Similar procedures can **be** applied when clipping a curved object against a general polygon clip region. On the first pass, we can clip the bounding rectangle of the object against the bounding rectangle of the clip region. If the two regions overlap, we will need to solve the simultaneous line-curve equations to obtain the clipping intersection points

4. **Write a detailed note on the basic two dimensional transformations**

BASIC TWO-DIMENSIONAL GEOMETRIC TRANSFORMATIONS

1. Operations that are applied to the geometric description of an object to change its position, orientation, or size are called ***geometric transformations***.
2. Geometric transformations can be used to describe how objects might move around in a scene during an animation sequence or simply to view them from another angle

Geometric transformations

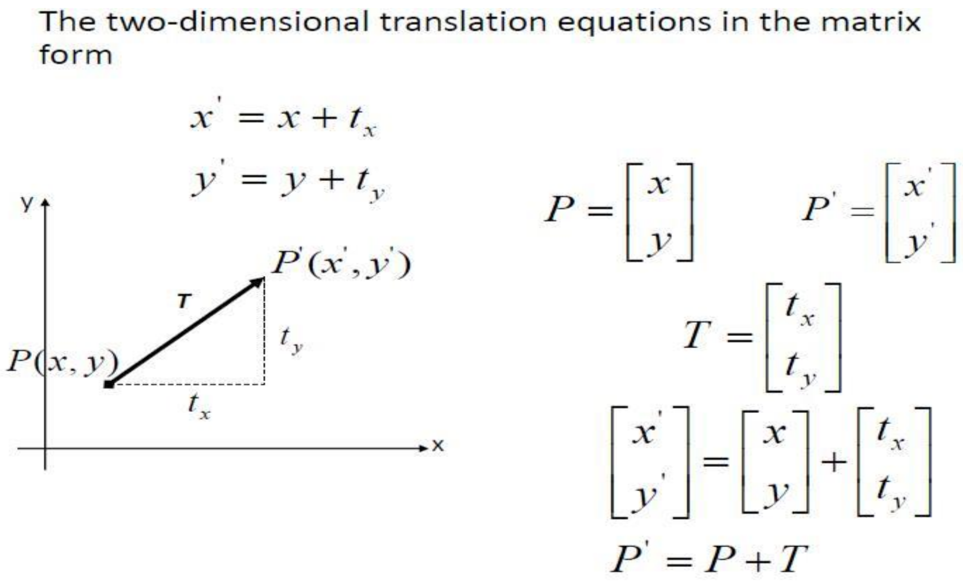
6. Translation
7. Rotation
8. Scaling
9. Reflection
10. shearing

Two-Dimensional Translation

1. We perform a translation on a single coordinate point by adding offsets to its coordinates so as to generate a new coordinate position.
2. To translate a two-dimensional position, we add translation distances, tx and ty to the original coordinates (x,y) to obtain the new coordinate position (x‘,y‘),

$$X' = X + t_x$$

$$Y' = Y + t_y$$



The two-dimensional translation equations in the matrix form

$$x' = x + t_x$$

$$y' = y + t_y$$

$$P = \begin{bmatrix} x \\ y \end{bmatrix} \qquad P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$P' = P + T$$

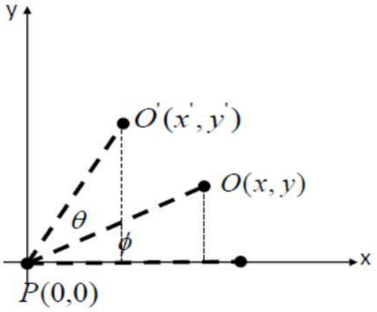
Two-Dimensional Rotation

1. We generate a rotation transformation of an object by specifying a rotation axis and a rotation angle.
2. A two-dimensional rotation of an object is obtained by repositioning the object along a circular path in the xy plane.
3. Parameters for the two-dimensional rotation are
   –The rotation angle θ
   –A position (x,y) – rotation point (pivot point)

The two-dimensional rotation

$$x' = r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta$$

$$y' = r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta$$



Polar coordinate system

$$x = r\cos\phi$$

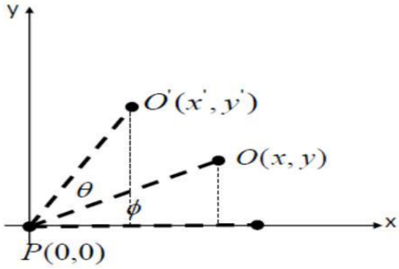
$$y = r\sin\phi$$

$$\Downarrow$$

$$x' = x\cos\theta - y\sin\theta$$

$$y' = x\sin\theta + y\cos\theta$$

The two-dimensional rotation



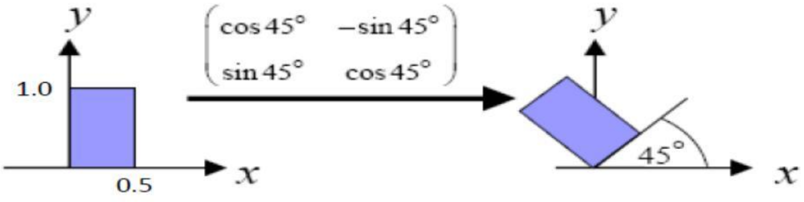
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$O' = R \bullet O$$

โดยที่ $O = \begin{bmatrix} x \\ y \end{bmatrix}$ $O' = \begin{bmatrix} x' \\ y' \end{bmatrix}$

และ $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \longleftarrow$ Rotation matrix

Ex. 1



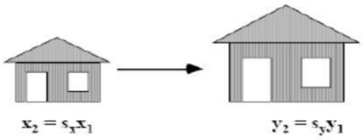
$$\begin{pmatrix} \cos 45^\circ & -\sin 45^\circ \\ \sin 45^\circ & \cos 45^\circ \end{pmatrix}$$

Rotation of a point about an arbitrary pivot position

$$x' = (x - x_r)\cos\theta - (y - y_r)\sin\theta + x_r$$

Two-Dimensional Scaling

1. To alter the size of an object, we apply a scaling transformation.
2. A simple two-dimensional scaling operation is performed by multiplying object positions (x,y) by scaling factors sx and sy to produce the transformed coordinates (x‘,y‘).
3. Any positive values can be assigned to the scaling factors.
   - x Values less than 1 reduce the size of object;
   - x Values greater than 1 produce enlargements.
   - x Uniform scaling – scaling values have the same value
   - x Differential scaling – unequal of the scaling factor



$$x = x \cdot s_x$$

$$y' = y \cdot s_y$$

$$\Downarrow$$

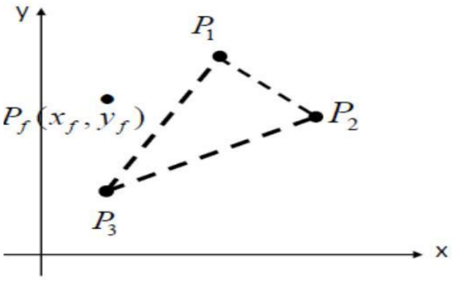
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$P' = S \bullet P$$

Scaling relative to a chosen fixed point

$$x' = x \cdot s_x + x_f (1 - s_x)$$

$$y' = y \cdot s_y + y_f (1 - s_y)$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 1 - s_x & 0 \\ 0 & 1\text{-}s_y \end{bmatrix} \begin{bmatrix} x_f \\ y_f \end{bmatrix}$$

$$P' = S \bullet P + S^* \bullet P_f$$

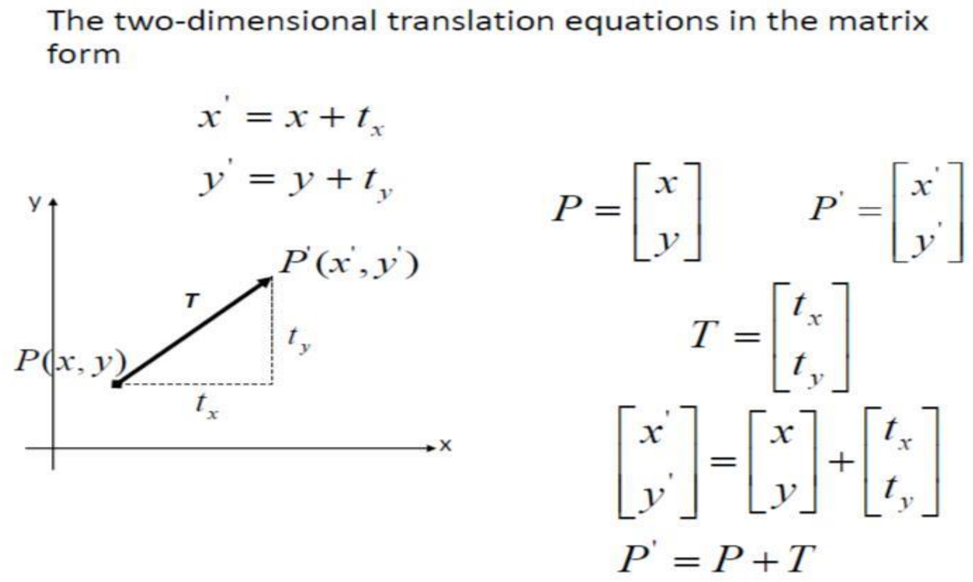
5. **Explain the two dimensional Translation and scaling with example** **Two-Dimensional Translation**

1. We perform a translation on a single coordinate point by adding offsets to its coordinates so as to generate a new coordinate position.

2. To translate a two-dimensional position, we add translation distances, tx and ty to the original coordinates (x,y) to obtain the new coordinate position (x‘,y‘),

$X' = X + t_X$

$Y' = Y + t_Y$



The two-dimensional translation equations in the matrix form

$$x' = x + t_x$$

$$y' = y + t_y$$

$$P = \begin{bmatrix} x \\ y \end{bmatrix} \qquad P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$P' = P + T$$

Two-Dimensional Scaling

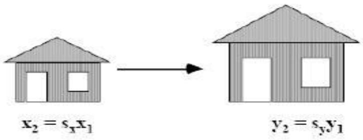
4. To alter the size of an object, we apply a scaling transformation.
5. A simple two-dimensional scaling operation is performed by multiplying object positions (x,y) by scaling factors sx and sy to produce the transformed coordinates (x‘,y‘).
6. Any positive values can be assigned to the scaling factors. x

   Values less than 1 reduce the size of object;

   x Values greater than 1 produce enlargements.

   x Uniform scaling – scaling values have the same value

   x Differential scaling – unequal of the scaling factor



$$x = x \cdot s_x$$

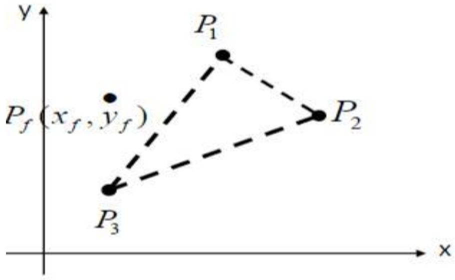
$$y' = y \cdot s_y$$

$$\Downarrow$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$P' = S \bullet P$$

Scaling relative to a chosen fixed point



$$x' = x \cdot s_x + x_f(1 - s_x)$$

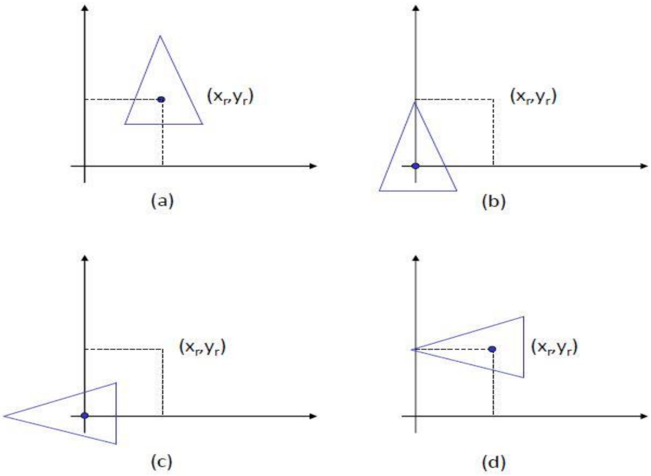
$$y' = y \cdot s_y + y_f(1 - s_y)$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 1 - s_x & 0 \\ 0 & 1\text{-}s_y \end{bmatrix} \begin{bmatrix} x_f \\ y_f \end{bmatrix}$$

$$P' = S \bullet P + S^* \bullet P_f$$

6. **Obtain a transformation matrix for rotating an object about a specified pivot point <u>General Two-dimensional Pivot-Point Rotation</u>**

1. A transformation sequence for rotating an object about a specified pivot point using the rotation matrix **R**(θ).
2. Translate the object so that the pivot-point position is moved to the coordinate origin.
3. Rotate the object about the coordinate origin.
4. Translate the object so that the pivot point is returned to its original position.



(a) (b) (c) (d)

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r \sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r \sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

$$T(x_r, y_r) \cdot R(\theta) \cdot T(-x_r, -y_r) = R(x_r, y_r, \theta)$$

7. Explain Cohen-Sutherland Line clipping algorithm.

Cohen -Sutherland Line Clipping :

1. Region Code Creation

   –Region Code

   •Bit 1: left

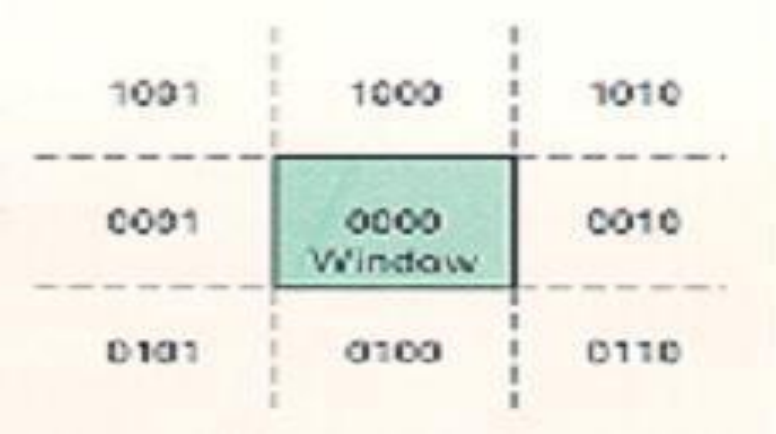
   •Bit 2: right

   •Bit 3: below

   •Bit 4: above

   –Calculate differences between endpoint coordinates and clipping boundaries

   –Use the resultant sign bit of each difference calculation to set the corresponding value in the region code



1. Outside Line Removal Test

   –A method that can be used to test lines total clipping is to perform the logical and operation with both region codes

   –Not 0000
2. Completely outside the clipping region!!
3. Lines that cannot be identified as completely inside or outside a clip window by this test.
4. Calculate Intersection Point

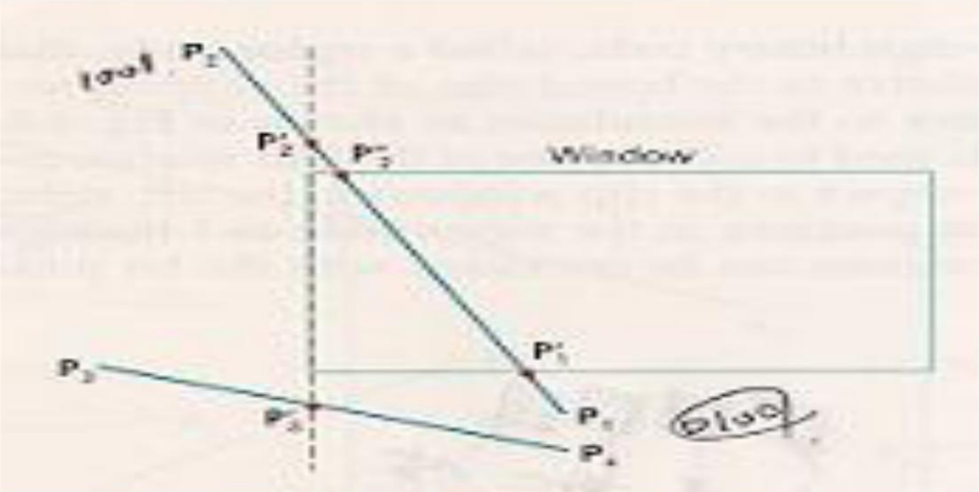
   – Using the slope-intercept form

   – Vertical Boundary, $y = y_1 + m (x – x_1)$

   – Horizontal Boundary

$$x = x_1 + \frac{y - y_1}{m}$$

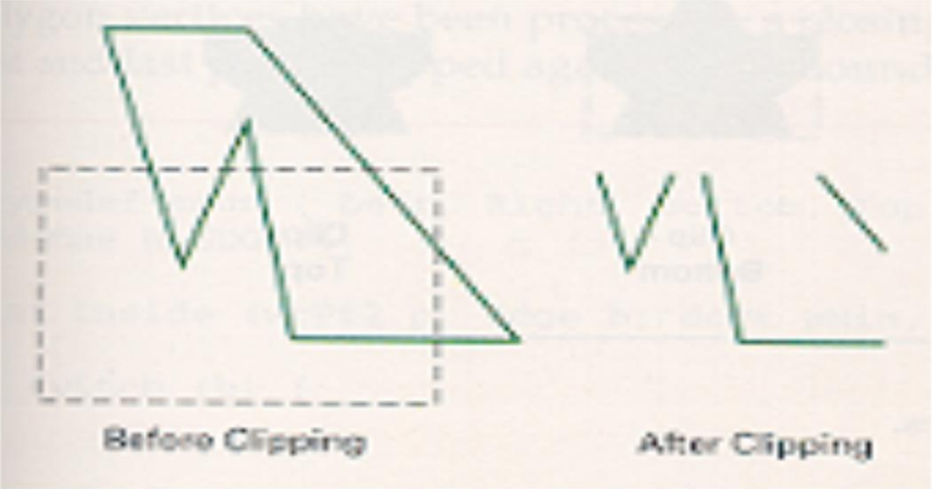
$$m = (y_2 - y_1)/(x_2 - x_1$$

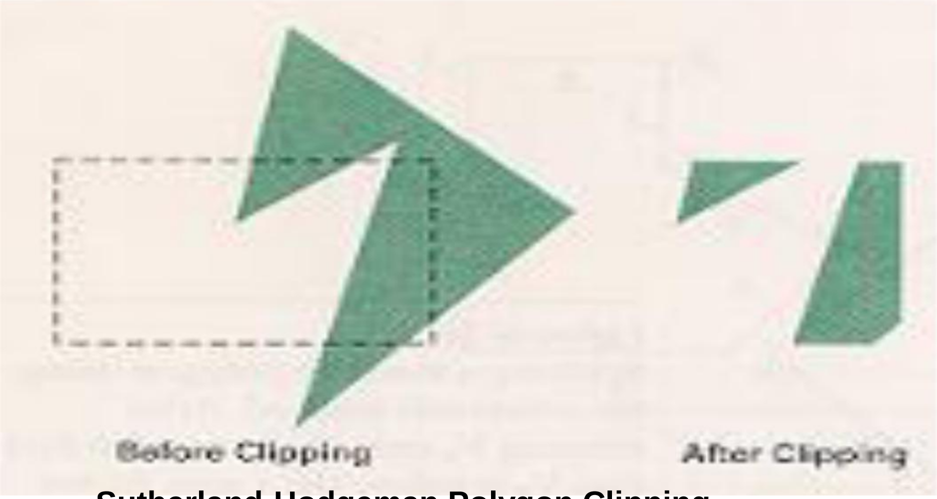


8. **Explain the various polygon clipping algorithm.**

POLYGON CLIPPING

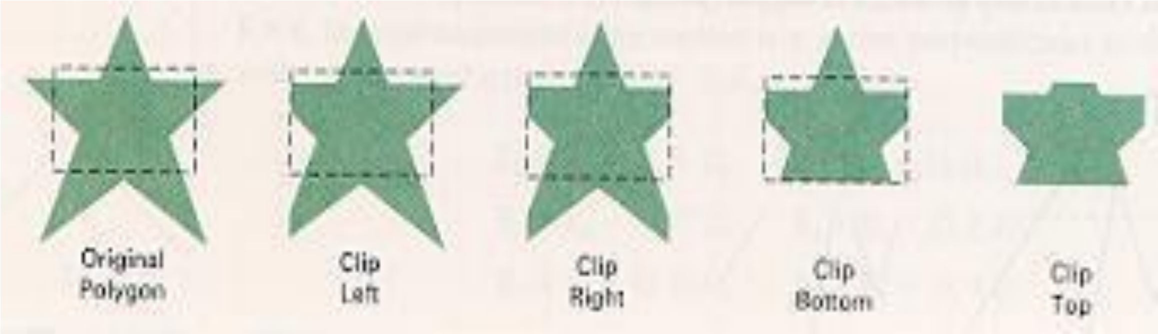
Display of polygon processed by a line-clipping algorithm



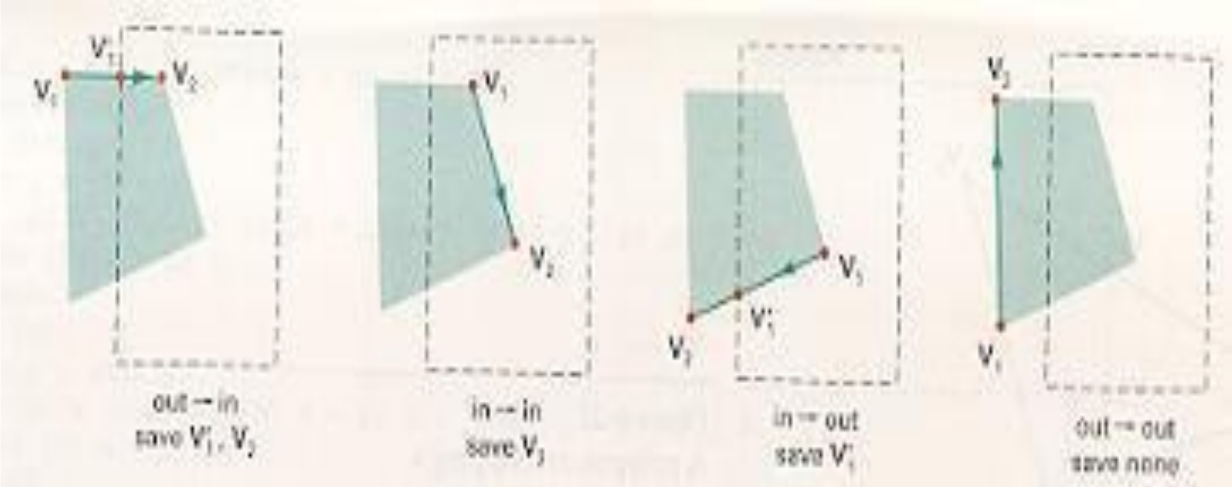


Sutherland-Hodgeman Polygon Clipping

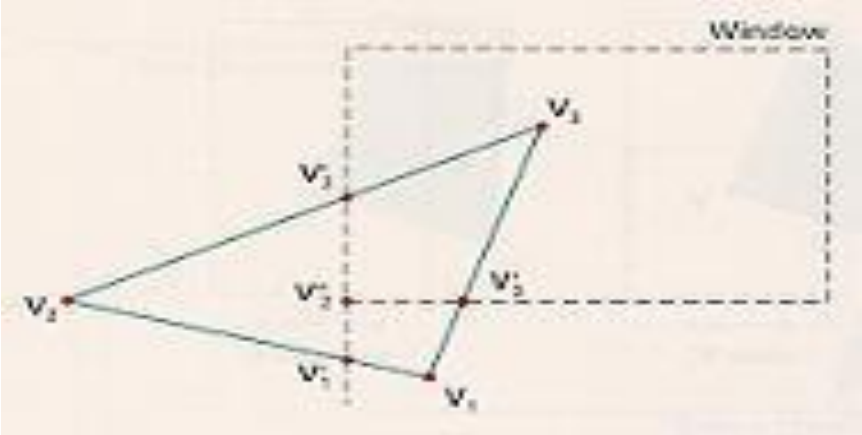
Clipping a polygon against successive window boundary



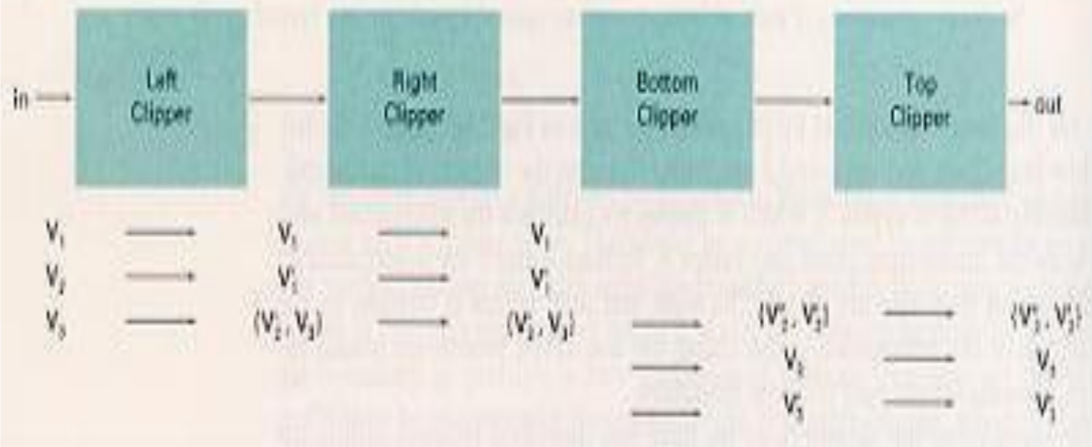
Successive processing of pairs of polygon vertices against the left window boundary



A polygon overlapping a rectangular clip window

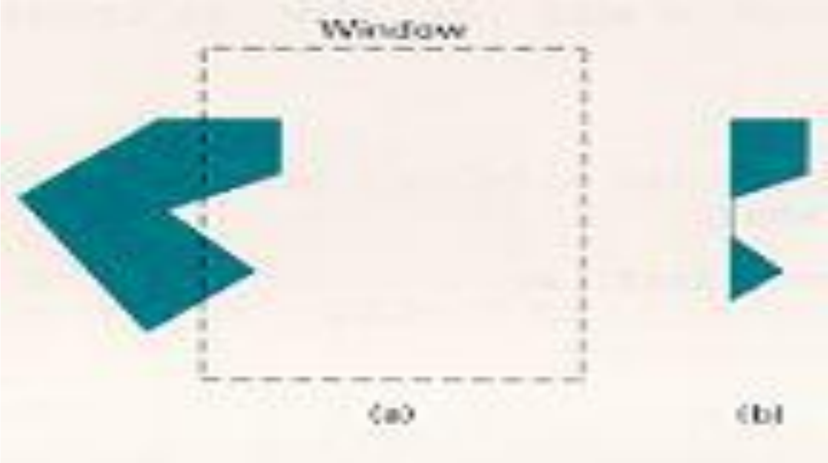


Processing the vertices of the polygon

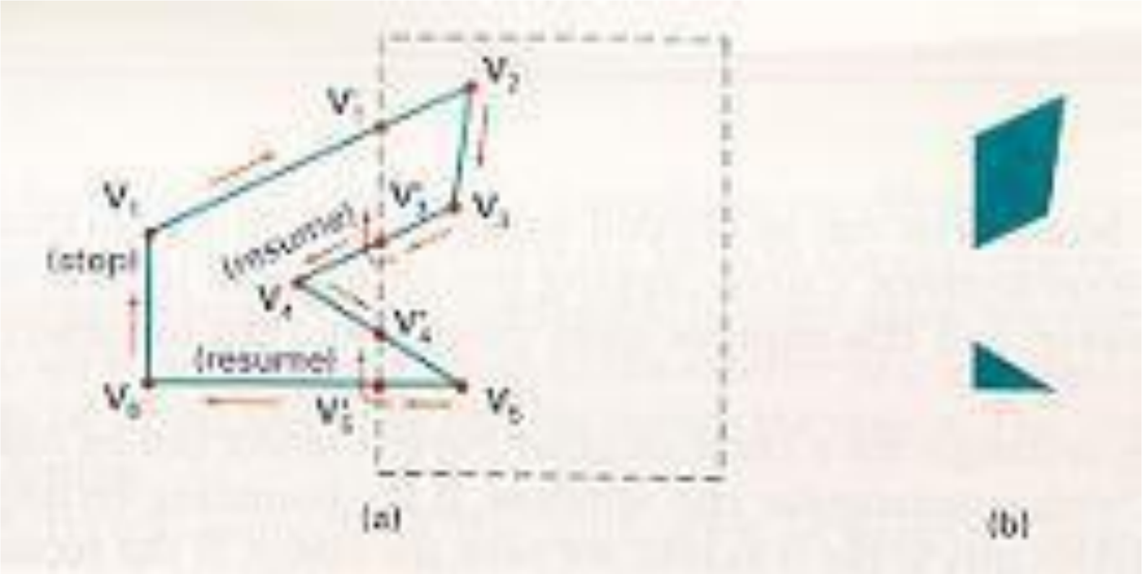


Weiler-Atherton Polygon Clipping

1. Problem of Sutherland-Hodgeman clipping
   –Displaying extraneous line

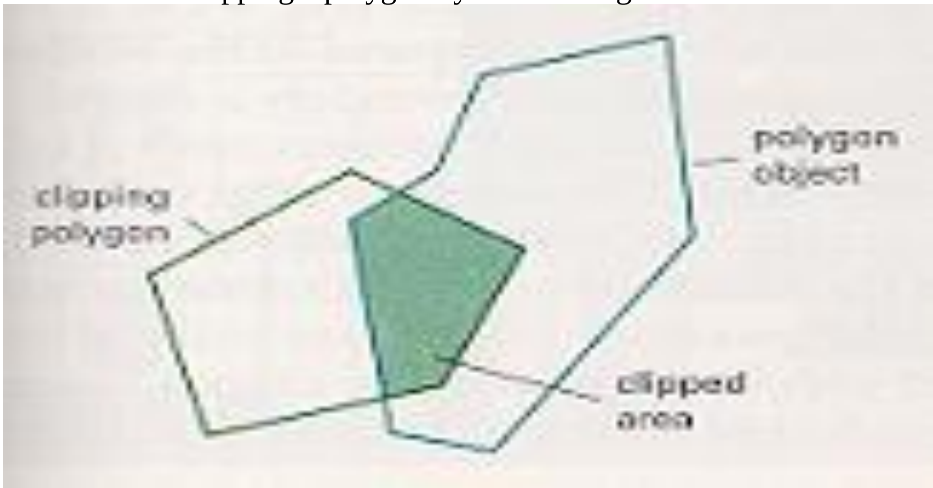


1. Rules
   –For an outside-to-inside pair of vertices, follow the polygon boundary
   –For an inside-to-outside pair of vertices, follow the window boundary in clockwise direction



Other Polygon-Clipping Algorithm

1. Extending parametric line-clipping method
   –Well suited for convex polygon-clipping
   –Using region testing procedures
2. Clipping a polygon by determining the intersection of two polygon areas



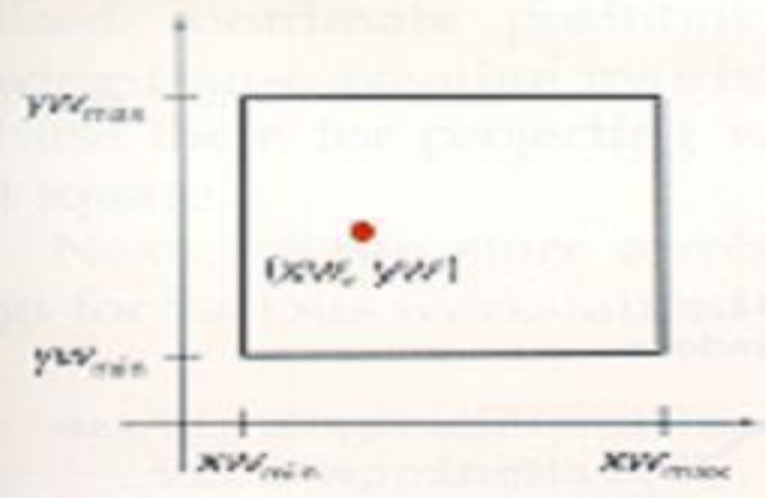
9. **Explain the window to viewport coordinate transformation..**

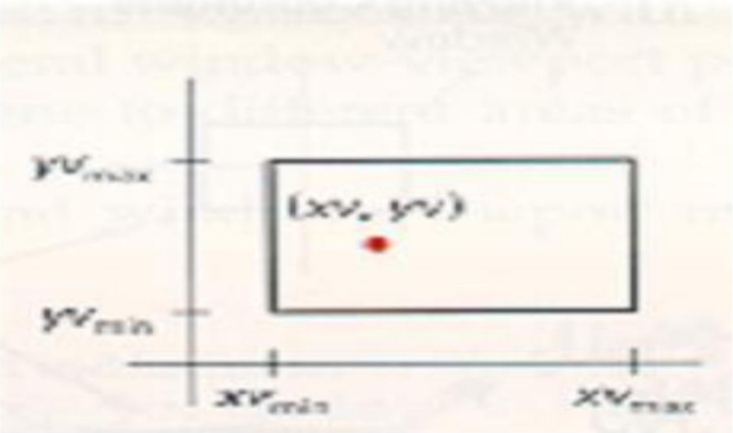
WINDOW-TO-VIEWPORT COORDINATE TRANSFORMATION

1. Window-to-viewport mapping ?

A point at position (xw, yw) in a designated window is mapped to viewport coordinates

(xv, yv) so that relative positions in the two areas are the same





- To maintain the same relative placement

$$\frac{xv-xv_{min}}{xv_{max}-xv_{min}}=\frac{xw-xw_{min}}{xw_{max}-xw_{min}}$$

$$\frac{yv-yv_{min}}{yv_{max}-yv_{min}}=\frac{yw-yw_{min}}{yw_{max}-yw_{min}}$$

- Solving these expressions for the viewport position (xv, yv)

$$xv=xv_{min}(xw-xw_{min})sx$$

$$yv=yv_{min}(yw-yw_{min})sy$$

The scaling factors

$$sx=\frac{xv_{max}-xv_{min}}{xw_{max}-xw_{min}}$$

$$sy=\frac{yv_{max}-yv_{min}}{yw_{max}-yw_{min}}$$

Conversion sequence of transformation

The way of character string mapping

–Maintaining character size

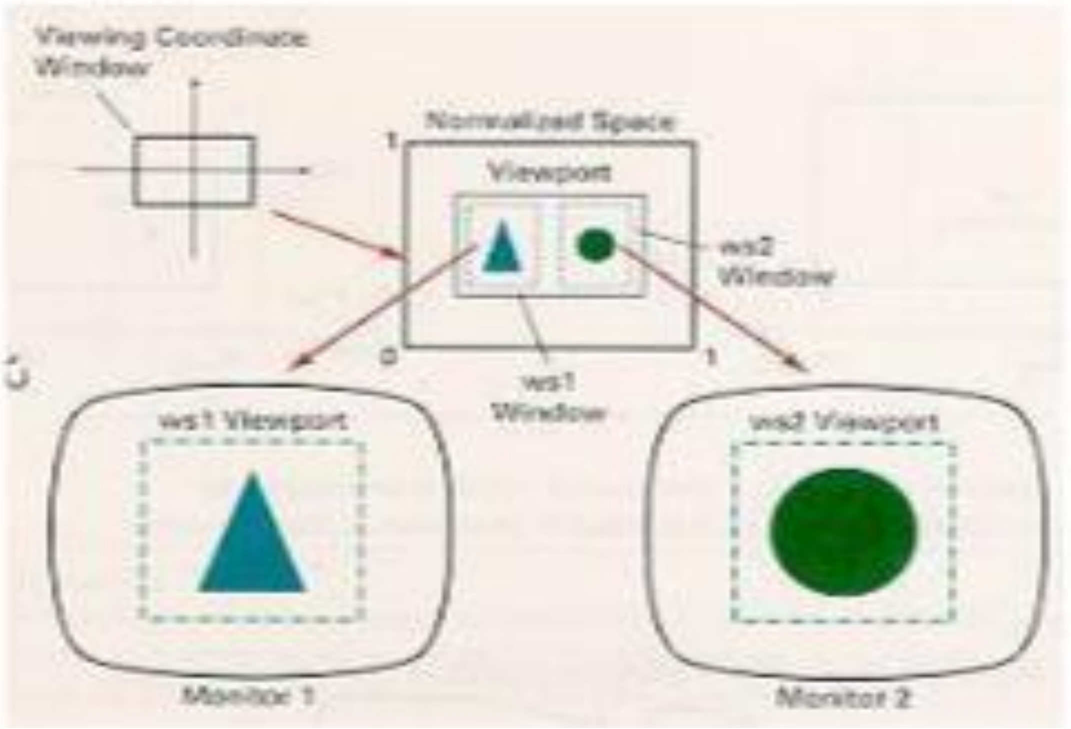
1. Using standard character fonts

–Changing character size

2. Using characters formed with line segments

Workstation transformation

3. –Opening any number of output devices in a particular application
4. –Performing another window-to-viewport transformation for each open output device
5. Mapping selected parts of a scene in normalized coordinates to different video monitors with Workstation transformation.



10. **Explain the various clipping operations.**

CLIPPING OPERATIONS

Any procedure that identifies those portions of a picture that is either inside or outside of a specified region of space

1. Applied in World Coordinates
2. Adapting Primitive Types

–Point

–Line

–Area (or Polygons)

–Curve, Text

Point Clipping:

1. Assuming that the clip window is a rectangle in standard position
2. Saving a point P=(x, y) for display

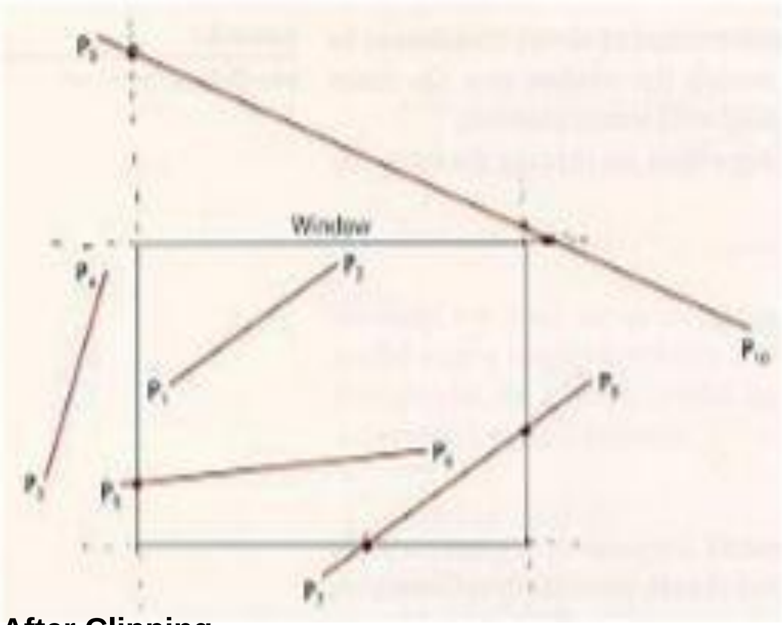
$$xw_{min} <= x <= xw_{max}$$

$$yw_{min} <= y <= yw_{max}$$

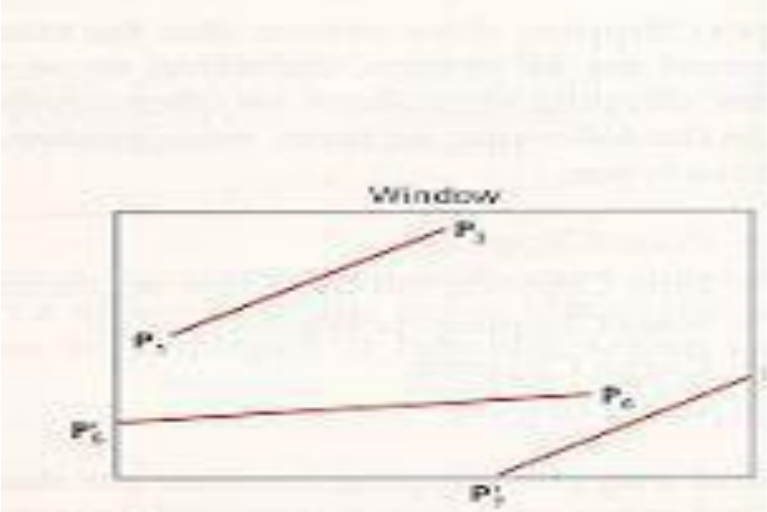
3. Appling Fields - Particles (explosion, sea foam)

Line Clipping:

Before Clipping



After Clipping



1. Parametric representation of Line segment with endpoints (x1, y1) and (x2, y2)

$x = x_1 + u(x_2-x_1)$

$y = y_1 + u(y_2-y_1)$; 0<=u<=1

2. Exterior of the window

– Intersection with outside the range u

3. Interior of the window

– Intersection with inside the range u

THREE DIMENSIONAL GRAPHICS
PART-A

1.What are blobby objects?

Some objects do not maintain a fixed shape, but change their surface characteristics in certain motions or when in proximity with other objects. These objects are referred to as blobby objects, since their shapes show a certain degree of fluidness.

2.What are spline curves? (AU NOV/DEC 2011 & NOV/DEC 2012)

The term spline is a flexible strip used to produce a smooth curve through a designated set of points. Incomputer graphics, the term spline curve refers to any composite curve formed with polynomial sectionssatisfying specified continuity conditions at the boundary of the pieces.

3.How to generate a spline curve?

A spline curve is specified by giving a set of coordinate positions called as control points. These controlpoints are then fitted with piece wise continuous parametric polynomial functions in one of the two ways.When polynomial sections are fitted so that the curve passes through each control point, the resultingcurve is said to interpolate the set of control points. When the polynomials are fitted to the generalcontrol point path without necessarily passing through any control point the resulting curve is said toapproximate the set control points.

4.What are called control points?

The spline curve is specified by giving a set of coordinate positions, called controlpoints, which indicatesthe general shape of the curve.

5.When is the curve said to interpolate the set of control points?

When polynomial sections are fitted so that the curve passes through each control point, the resultingcurve is said to interpolate the set of control points.

6.When is the curve said to approximate the set of control points?

When the polynomials are fitted to the general control-point path without necessarily passing through anycontrol point, the resulting curve is said to approximate the set of control points.

7.What is called a convex hull?

The convex polygon boundary that encloses a set of control points is called the convex hull.

8. Explain about Bezier curves.

This is a spline approximation method. A beizer curve section can be fitted to any number of controlpoints. The number of control points to be approximated and their relative position determine the degreeof the Beizer polynomial. As with the interpolation splines, a beizer curve can be specified withboundary conditions, with a characterization matrix, or with blending functions.

9.What are the various 3D transformations?

The various 3D transformations are translation, reflection, scaling, rotation and shearing.

10.What is shear transformation? (AU MAY/JUNE 2012 IT)

Shearing transformations can be used to modify object shapes. They are also used in 3D viewing forobtaining general projection transformation. A z-axis 3D shear:

$$SH_Z = \begin{matrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

Parameters a and b can be assigned any real value.

11.Define viewing. (AU MAY/JUNE 2012)

Viewing in 3D have more parameters to select when specifying how a 3D scene is to be mapped to adisplay device. The scene description must be processed through the viewing coordinate transformationand projection routines that transform the 3D viewing coordinate into 2D device coordinates.

12.Mention some surface detection methods.

Back-face detection, depth-buffer method, A-buffer method, scan-line method, depth-sorting method,BSP-tree method, area subdivision, octree method, ray casting.

13.What is ray casting?

Ray casting methods are commonly used to implement constructive solid geometry operations whenobjects are described with boundary representations. Ray casting is applied by constructing compositeobjects in world coordinates with the xy plane corresponding to the pixel plane of a video monitor. Thisplane is referred to as ―firing plane‖, since each pixel emits a ray through the objects that are combined.Then the surface intersections along each ray path, and they are sorted according to the distance from thefiring plane. The surface limits for the composite objects are determined by specified set operations.

14.What are the two types of projections?

Parallel projection: coordinate positions are transformed to the view plane along parallel lines.

Perspective projection: object positions are transformed to the view plane along lines that converge to apoint called projection reference point.

15.Differentiate parallel projection from perspective projection. (AU MAY/JUNE 2012)

Parallel Projection	Perspective Projection
In parallel projection, coordinate positions aretransformed to the view plane along parallel lines.	In perspective projection, object positions are transformed to the view plane along lines that converge to a point called projection reference point or center of projection
Preserves the relative proportions of objects.	Produce realistic views but does not preserve relative proportions.
Used in drafting to produce scale drawings of 3Dobjects.	Projections of distant objects are smaller than theprojections of objects of the same size that arecloser to the projection plane.

16. Differentiate oblique and orthographic parallel projections. (AU MAY/JUNE 2012 IT &NOV/DEC 2012)

Orthographic Parallel Projection	Oblique Parallel projection
Projection is perpendicular to the view plane.	Projection is not perpendicular to the view plane.
Used to produce front, side and top views of objectcalled as elevations.	An oblique projection vector is specified with twoangles, and .

17. What are the two types of parallel projection?

Orthographic parallel projection: projection is perpendicular to the view plane.

Oblique parallel projection: projection is not perpendicular to the view plane.

18.What is axonometric projection?

Orthogonal projections that display more than one face of an object are axonometric projection.

19.What is isometric projection?

Isometric projection is obtained by aligning the projection plane so that it intersects each coordinate axisin which the object is defined at the same distance from the origin.

20.What is cavalier projections?

Point (x,y,z) is projected tp position (xp,yp) on the view plane. The projection line from (x,y,z) and (xp,yp) makes and angle α with the line on the projection plane that joins(xp,yp) and (x,y). when α = 45 the views obtained are cavalier projections. All lines perpendicular to the projection plane are projected with no change in length.

21.What are the representation schemes for solid objects?

Boundary representations: they describe a 3D object as a set of surfaces that separate the object interiorfrom environment. Example: polygon facets Space partitioning representations: they are used to describeinterior properties, by partitioning the spatial region containing an object into a set of small, non-overlapping,contiguous solids. Example: octree

22. Define quadric surfaces. (AU NOV/DEC 2011)

Quadric surfaces are described with second degree equations (quadrics). They include sphere, ellipsoids,tori, paraboloids and hyperboloids. Spheres and ellipsoids are common elements of graphic scenes, theyare often available in graphics packages from which more complex objects can be constructed.

23.What is an ellipsoid?

An ellipsoid surface can be described as an extension of a spherical surface, where the radii in threemutually perpendicular directions can have different values.

24. Define Octree.

Hierarchical tree structures called octrees are used to represent solid objects in some graphics system. Thetree structure is organized so that each node corresponds to a region of 3D space. This representation forsolids takes advantage of spatial coherence to reduce storage requirements for 3D objects.

25. Write about sweep representations.

Sweep representations are useful for constructing three- dimensional objects that possess translational,rotational or other symmetries. One can represent such objects by specifying a 2D shape and a sweep thatmoves the shape through a region of space. A set of 2D primitives ,such as circle and rectangles, can beprovided for sweep representations as menu options.

PART - B

1. Differentiate parallel and perspective projections and derive their projection matrices. (AU
NOV/DEC 2011 & MAY/JUNE 2012 IT & NOV/DEC 2012)

- Parallel projections:
  - – no shortening due to distance
  - – several kinds, depending on orientation:
    - isometric, cavalier,…
- Perspective projections:
  - – shortening of objects in the distance
  - – several kind, depending on orientation:
    - one, two, three vanishing points

Parallel Projection Matrix

- Parallel projection onto z=0

plane: x‘=x, y‘=y, w‘=w

Matrix for this projection:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Perspective Projection Matrix

Projection onto plane z=0, with center of projection at z=-d:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix}$$

Perspective projections pros and cons:

Size varies inversely with distance - looks realistic – Distance and angles are not (in general) preserved – Parallel lines do not (in general) remain parallel

Parallel projection pros and cons:

Less realistic looking + Good for exact measurements + Parallel lines remain parallel – Angles not (in general) preserved

Parallel projections

For parallel projections, we specify a direction of projection (DOP) instead of a COP. There are two types of parallel projections: w Orthographic projection — DOP perpendicular to PP w Oblique projection — DOP not perpendicular to PP There are two especially useful kinds of oblique projections: w Cavalier projection • DOP makes 45° angle with PP • Does not foreshorten lines perpendicular to PP w Cabinet projection • DOP makes 63.4° angle with PP • Foreshortens lines perpendicular to PP by one half

Perspective in the graphic arts is an approximate representation, on a flat surface (such as paper), of an image as it is seen by the eye. The two most characteristic features of perspective are that objects are smaller as their distance from the observer increases; and that they are foreshortened, meaning that an object's dimensions along the line of sight are shorter than its dimensions across the line of sight.

2. Explain about 3D object representations. (AU MAY/JUNE 2012)

Polygon surfaces-polygon tables-plane equations-polygon meshes
Object descriptions are stored as sets of surface polygons
The surfaces are described with linear equations
Polygon table
data is placed into the polygon table for processing
Polygon data table can be organised into two groups
geometric table
attribute table
Quadric surfaces-sphere-ellipsoid-torus
Described with second degree eqns.
Ex. Sphere,ellipsoids,tori,paraboloids,hyperboloids
Sphere
A spherical surface with radius _r‘ centered on the coordinate origin is defined as a set of points(x,y,z) that satisfy the equation
x2 + y2 + z2 = r2
In parametric form,
x = r CosΦCosΘ
y = r Cos ΦSinΘ
z = r Sin Φ
Blobby objects-definition and example
Don‘t maintain a fixed shape
Change surface characteristics in certain motions
Ex. Water droplet, Molecular structures
T =some threshold
a,b used to adjust the amount of bloobiness.
Spline-representation-interpolation
it is a composite curve formed with polynomial pieces satisfying a specified continuity conditions at the boundary of the pieces Bezier curves

can be fitted to any no. of control points
degree of bezier polynomial is determined by the number of control points and their relative position
Bezier curve is specified by
Boundary conditions
Characterising matrix

Blending function

3. **How are polygon surfaces represented in 3d?** Polygon tables-Basic concept

Polygon table data is placed into the polygon table for processing Polygon data table can be organised into two groups geometric table attribute table Storing geometric data To store geometric data three lists are created Vertex table – contains coordinate values for each vertex

Edge table – contains pointers back into the vertex table

Polygon table – contains pointers back into the edge table

Advantages of three table

efficient display of objects For faster info. Extraction expand edge table to include forward pointers to the polygon table Plane Equation

$$Ax + By + Cz + D = 0$$

eqn. is solved by Cramer‘s rule

Identification of points

if $Ax + By + Cz + D < 0$,the points (x,y,z) is inside the surface if

$Ax + By + Cz + D > 0$,the points (x,y,z) is outside the surface

4. **Write notes on quadric surfaces. 2012) Quadric** surfaces-definition

Described with second degree eqns.

Ex. Sphere,ellipsoids,tori,paraboloids,hyperboloids

Sphere-definition-equations-diagram

Sphere

A spherical surface with radius _r‘ centered on the coordinate origin is defined as a set of points(x,y,z) that satisfy the equation

$$x2 + y2 + z2 = r2$$

In parametric form,

$$x = r \, \text{Cos}\Phi\text{Cos}\Theta$$
$$y = r \, \text{Cos} \, \Phi\text{Sin}\Theta$$
$$z = r \, \text{Sin} \, \Phi$$

Ellipsoid-definition-equations-diagram

Ellipsoid

Extension of spherical surface ,where the radii in three mutually perpendicular directions have different values

$$(x/rx)2 + (y/ry)2 + (z/rz)2 = 1$$

5. **With suitable examples, explain all 3D transformations.**

Transformation-definition and types Translation-definition-equations-diagram-matrix representation Translation

$$PI = T .P$$
$$xI = x + tx$$
$$yI = y + ty$$
$$zI = z + tz$$

Inverse translation

- obtained by negating translation distances Rotation-definition-equations-diagram-matrix representation Rotation

To perform rotation we need,

An axis

Rotation angle

+ve rotation angles produce counter clockwise rotation -ve rotation angles produce clockwise rotation

Coordinate axis rotation Z-axis, Y-axis and X-axis

Z axis rotation

$$xI = x\text{Cos}\Theta – y\text{Sin}\Theta$$
$$yI = x\text{Sin}\Theta + y\text{Cos}\Theta$$
$$zI = z$$
$$PI = Rz(\Theta).P$$

Scaling Reflection Shearing -definition

Scaling:

alters the size of the object

coordinate values of the vertex is multiplied by scaling factors Sx & Sy

xI = x . Sx

yI = y . Sy

Reflection

produces mirror image

obtained by rotating the object 180 degrees about the reflection axis.

Shear

distorts the shape of an object.

can be with respect to both axis

Reflection-definition-equations-diagram-matrix representation

Shearing-definition-equations-diagram-matrix representation

6. **Write notes on 3D viewing.**

Viewing – transfers positions from world coordinate plane to pixels positions in the plane of the output device

Viewing pipeline:

MC ➦ MT ➦WC ➦VT ➦VC ➦PT ➦PC ➦WT ➦DC

Transformation from world to viewing coordinates:

sequences

Translate view reference point to the origin of world coordinate system

Apply rotation to align xv , yv , zv axes with the world xw ,yw ,zw axes

7. **Discuss the various surface detection methods in detail.**

Back face detection

A point (x,y,z) is inside a polygon surface with plane parameters A,B,C and D

if Ax+By+Cz+D < 0

When an inside point is along the line of sight to the surface , the polygon must be a back-face

Conditions for back face:

A polygon is a back-face if V.N > 0

Depth buffer method

Steps

Initialize the depth buffer and refresh buffer so that for all the buffer positions (x,y) depth(x,y) = 0 , refresh(x,y) = I backgnd

For each position on each polygon surface listed on the polygon table calculate the depth value and compare the depth vaslue to the previously stored values in the depth buffer to determine visibility

Let the calculated depth be Z for each position (x,y)

If Z > depth(x,y) , then set) depth(x,y) = Z , refresh(x,y) = Isurf (x,y)

Scan-line method-concept-example-diagram

Extension of scan line algorithm for filling polygon interiors

All polygon surfaces intersecting the scan lines are examined

Depth calculations are made for each overlapping surface across every scan line to determine the nearest surface to the view plane

After the visible surface is determined the intensity value for the position is entered into the refresh buffer

Depth-sorting method

Steps:

Surfaces are ordered according to the largest Z value

Surface S with greatest depth is compared with other surfaces to determine whether there are any overlaps in depth

If no depth overlap occurs , S is scan converted
This process is repeated for the next surface as long as no overlap occurs
If depth overlaps occurred additional comparisons are used to determine whether reordering of surfaces are needed or not

Ray casting method
- it is a variation of depth buffer method
- process pixels one at a time and calculate depths for all surfaces along the projection path to that pixel

Wireframe method

visible edges are displayed and hidden edges are either eliminated or displayed differently from the visible edges .Procedures for determining visibility of object edges are referred to as wireframe visibility methods / visible line detection methods / hidden line detection methods

8. Explain in detail about depth buffer method and A-buffer method for visible surface detection.

Depth buffer method

Steps

1. Initialize the depth buffer and refresh buffer so that for all the buffer positions (x,y) depth(x,y) = 0 , refresh(x,y) = I backgnd
2. For each position on each polygon surface listed on the polygon table calculate the depth value and compare the depth vaslue to the previously stored values in the depth buffer to determine visibility

Let the calculated depth be Z for each position (x,y)
If Z > depth(x,y) , then set) depth(x,y) = Z , refresh(x,y) = Isurf (x,y)

9. Explain in detail about B-Spline curves and surfaces.

Control Points

- A set of points that influence the curve‘s shape

Knots

- Control points that lie on the curve

Interpolating Splines

- Curves that pass through the control points (knots)

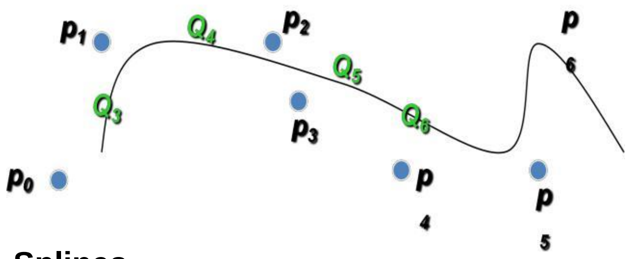
Approximating Splines

Control points merely influence shape
B-splines consist of curve segments whose polynomial coefficients depend on just a few control points

- Local control

Examples of Splines
Start with a sequence of control points
Select four from middle of sequence (pi-2, pi-1, pi, pi+1) d

- Bezier and Hermite goes between pi-2 and pi+1
- B-Spline doesn‘t interpolate (touch) any of them but approximates the going through pi-
1 and pi



Uniform B -Splines

Approximating Splines
Approximates n+1 control points

- P0, P1, …, Pn, n ¸ 3

Curve consists of n –2 cubic polynomial segments

- Q3, Q4, … Qn

t varies along B-spline as Qi: ti <= t < ti+1

ti (i = integer) are knot points that join segment Qi-1 to Qi

Curve is uniform because knots are spaced at equal intervals of parameter,

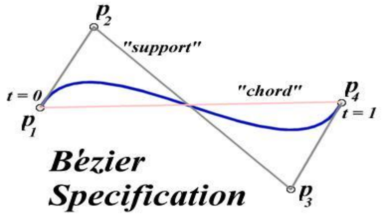
t First curve segment, Q3, is defined by first four control points

Last curve segment, Qm, is defined by last four control points, Pm-3, Pm-2, Pm-1, Pm

Each control point affects four curve segments

10. Explain in detail about Bezier curves and surfaces.

Four control points, two of which are knots



The derivative values of the Bezier Curve at the knots are dependent on the adjacent points

$$\nabla p_1 = 3(p_2 - p_1)$$
$$\nabla p_4 = 3(p_4 - p_3)$$

The scalar 3 was selected just for this curve

x Bézier Blending Functions

x Look at the blending functions

$$p(t) = \begin{bmatrix} (1-t)^3 \\ 3t(1-t)^2 \\ 3t^2(1-t) \\ t^3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

x This family of polynomials is calledorder-3 Bernstein Polynomials

- C(3, k) t^k $(1\text{-}t)^{3\text{-}k}$; 0<= k <= 3
- They are all positive in interval [0,1]
- Their sum is equal to 1

ANIMATIONS & REALISM
PART-A

1.What is animation?

Computer animation generally refers to any time sequence of visual changes in a scene. In addition tochanging object positions with translations or rotations, a computer generated animation could displaytime variations in object size, color, transparency or surface texture. Animations often transition from oneobject shape to another.

2.Mention the steps in animation sequence.

Storyboard layout, Object definitions, Key-frame specifications, Generation of in-between frame.

3. Explain about frame-by-frame animation.

Frame-by-frame animation, each frame of the scene is separately generated and stored. Later the framescan be recorded on film or they can be consecutively displayed in ―real time playback‖ mode.

4.Define keyframes. (AU NOV/DEC 2011)

A key frame is a detailed drawing of the scene at a certain time in the animation sequence. Within eachkeyframe, each object is positioned according to the time for that frame. Some key frames are chosen atextreme positions in action; others are spaced so that the time interval between key frames is not toogreat. More key frames are specified for intricate motions than for simple, slowly varying motions.

5.What are in between frames?

In-betweens are the intermediate frames between the key frames. The number of in-betweens needed isdetermined by the media to be used to display the animation. Film requires 24 frames per second andgraphics terminals are refreshed at the rate of 30 to 60 frames per second.

6.Mention the different types of animation.

The different types of animation are:

¾ Raster animation

x Raster operations: generate real-time animation in limited applications using raster operations.

x Color-table transformations: animate objects along 2D motion paths

¾ Key-frame system: specialized animation languages designed to generate the in-between frames
from user specified key frames.

¾ Parameterized systems: allow object motion characteristics to be specified as part of the objectdefinitions. The adjustable parameter control such as object characteristics as degrees of freedom,motion limitations and allowable shape changes.

¾ Scripting systems: allow object specifications and animation sequences to be defined with a user input script.

7.What are key frame systems?

Key-frame systems are specialized animation languages designed to generate the in- between frames fromuser specified key frames. Each object in the scene is defined as a set of rigid bodies connected at thejoints and with a limited number of degrees of freedom. In-between frames are generated from thespecification of two or more fey frames. Motion paths can be given by kinematic description as a set ofspline curves or physically based by specifying the forces acting on the objects to be animated.

8.What is Morphing?

Transformation of object shape from one form to another is called morphing.

9.What are fractals?

Fractals are those which have the property of a shape that has the same degree of roughness no matterhow much it is magnified. A fractal appears ―the same‖ at every scale. No matter how much one enlargesa picture of the curve, it has the same level of detail.

10.Define self-similar. Mention the types of self-similar.

Most of the curves and pictures have a particularly important property: they are self-similar. This meansthat they appear ―the same‖ at every scale. No matter how much one enlarges a picture of the curve, it hasthe same level of detail.The types are: exactly self-similar and statistically self-similar.

11.What is Koch curve?

Very complex curves can be fashioned recursively by repeatedly ―refining‖ a simple curve. The simplestis the Koch curve. This curve stirred great interest in the mathematical world because it produces ainfinitely long line within a region of finite area.

12.What are Peanocurves?List down the properties of Peano curves. (AU NOV/DEC 2012)

A fractal curve can in fact ―fill the plane‖ and therefore have a dimension of 2. Such curves are called
Peano curves.

13.What is a L-System?

A large number of interesting curves can be generated by refining line segments. A particularly simpleapproach to generating these curves uses so called L-Systems to draw complex curves based on simple setof rules.

14.Mention the dragon rules.

The dragon rules are ‗F' ‗ ‗F', ‗X' ‗ ―X+YF+‖, ‗Y' ‗ ―-FX-Y‖, atom=‖FX‖

15.What is space-filling?

Such curves have a fractal dimension of 2 and they completely fill a region of space.

16.What are the two famous Peano curves?

The two famous Peano curves are: Hilbert and Sierpinski curves.

17.Define fractal trees.

Fractal trees provide an interesting family of shapes that resemble actual trees. Such shrubbery can be
used to ornament various drawings.

18.What is periodic tiling and dihedral tiling?

In periodic tiling, the notion is to take many copies of some shape such as polygon and to fit themtogether as a jigsaw puzzle so that they cover the entire plane with no gaps. In dihedral tiling, they permitthe use of two prototiles and therefore offer many more possibilities.

19.How can a black and white image be described?

A black and white image I can be described simply as the set of its black points: I =set of all black
points={(x,y) such that (x,y) is coloured black}.

20.What is the "Chaos Game"?

It is also known as the random iteration algorithm which offers a simple non recursive way to produce apicture of the attractor of an IFS.

21.What is fractal image compression?

The original image is processed to create the list of affine maps, resulting in a greatly compressed representation of the image.

22.What is Mandelbrot set?

A very famous fractal shape is obtained from the Mandelbrot set, which is a set of complex values z thatdo not diverge under the squaring transformation: z0 = z, zk = z2k-1 + z0, k = 1, 2, 3, . . .It is the blackinner portion, which appears to consist of a cardioid along with a number of wart like circles glued to it.Its border is complicated and this complexity can be explored by zooming in on a portion of the border.

23.What is Julia sets?

For some functions, the boundary between those points that move towards infinity and those that tendtoward a finite limit is a fractal. The boundary of the fractal object is called the Julia set. Julia sets are extremely complicated sets of points in the complex plane. There is a different Julia set Jc for each value of c.

24.Differentiate Mandelbrot sets and Julia sets.

Mandelbrot sets	Julia sets
A very famous fractal shape is obtained from the Mandelbrot set, which is a set of complex values z that do not diverge under the squaring transformation: z0 = z, zk = z2k-1 + z0, k = 1, 2,3. .	For some functions, the boundary between those points that move towards infinity and those that tend toward a finite limit is a fractal. The boundary of the fractal object is called the Julia set.
It is the black inner portion, which appears to consist of a cardioid along with a number of wart like circles glued to it. Its border is complicated and this complexity can be explored by zooming in on a portion of the border.	Julia sets are extremely complicated sets of points in the complex plane. There is a different Julia set Jc for each value of c.

25.What are the steps in ray tracing process? How to incorporate texture into a ray tracer?

The steps are: build the rc-thary, fins the intersections with the object, identify intersections that lie closeto and in front of the eye, compute the hit point, find the color of the light and place the color in the rc- thpixel.Two principal kinds of texture are used: with image texture a 2D image is pasted onto each surfaceof the object, with solid texture the object is considered to be carved out of a block of some material.

26.List down the ray tracing methods.

The various ray tracing methods are:

x Basic ray-tracing algorithm

x Ray-surface intersection calculations

x Reducing object-intersection calculations

x Space-subdivision methods

x Anti-aliased ray tracing

x Distributed ray tracing

PART B

1. Explain about fractals and self-similarity.

Characteristics of a fractal object

1. Infinite detail at every point
2. Self similarity between object

parts Types of self similarity

- Exact self similarity
- Statistical self

similarity Exact self similarity

if a region of a curve is enlarged the enlargement looks exactly like the original

Successive refinement of curves

by repeatedly refining a simple curve very complex curves can be fashioned

Ex. Koch curve

Koch curve

produces an infinitely long line within a region of finite

area Generations

Successive generations are denoted by K0 , K1, K2 , …

The zeroth generation shape K0 is a horizontal line of unit length

The curve K1 is generated by dividing K0 into three equal parts and replacing the middle section with a triangular bump

The bump should have sides of length 1/3 so the length of the line is now 4/3

The second order curve K2 is generated by building a bump on each of the 4 line segments of K1

```
Void drawKoch(double dir , double len , int n)
      {doubledirRad = 0.0174533 * dir ; // direction in radians if (
            n==0)
            lineRel(len * Cos(dirRad) , len * Sin(dirRad));
            else      {     n--; // reduce the order & length
            len    /=3;
            drawKoch(dir , len ,n);
            dir  += 60;
            drawKoch(dir , len ,n);
            dir  -= 120;
            drawKoch(dir , len ,n);
            dir  += 60;
            drawKoch(dir , len ,n);
                    }
        }
```

Fractal Dimension

Estimated by box covering method

$D = \log(N) / \log(1/r)$

N = no. of equal

segments r = 1/N

For Koch curve the fractal dimension is in between 1 &

2 For Peanocurve D is 2

2. Write notes on Peano curves.

Peano curves

Space filling curves (completely fill a region of space

) Have fractal dimension of 2

Ex. Hilbert curves, Sierpinski curves

Polya‘sPeano curve

Generated by replacing each segment of a generation by a right angled elbow

Direction of the elbow alternate in a L , LR , LRLR ,… fashion

To save the current state and to restore characters ‗]‘ and ‗]‘ are added to the language L – Systems

Approach to generate curves

Generate curves based on simple set of rules (productions)

String Production rules

F ➩F – F ++ F – F

F means forward (1,1)

+ means turn(A)

- means turn(-A)

➩ Means that every F is replaced by F – F ++ F – F

Atom – initial string

Production rules are applied to atom to produce the first generation string (S1)

To generate the second generation string the same production is applied to the first generation string

Generation of richer set of curves

by adding more rules to string production process richer set of curves can be generated Ex. Dragon curves

F ➩F

X ➩X + YF +

Y ➩ – FX – Y

Atom = FX

Order1 String S1	Order2 String S2
Atom = FX	
S1 = FX + YF +	S2 = FX + YF ++ – FX – YF +

Write notes on raster animation. (AU MAY/JUNE 2012 IT)

Raster animation-definition

This is the most common animation technique

Frames are copied very fast from off-screen memory to the frame buffer

Copying usually done with bitBLT-type operations Copying can be applied to

complete frames

only parts of the frame which contain some movement Example with diagram

Procedure

A part of the frame in the frame buffer needs to be erased

The static part of the frame is re-projected as a whole, and the animated part is over-projected

3. Discuss the computer animation techniques.

Computer animation-definition

:Raster animations-concept

This is the most common animation technique Frames are copied very fast from off-screen memory to the frame buffer Copying usually done with bitBLT-type operations Copying can be applied to complete frames only parts of the frame which contain some movement Example with diagram Procedure A part of the frame in the frame buffer needs to be erased The static part of the frame is re-projected as a whole, and the animated part is over-projected Keyframe systems-concept Compute first a small number of key frames Interpolate the remaining frames in-between these key frames (in-betweening) Key frames can be computed at equal time intervals according to some other rules for example when the direction of the path changes rapidly

4. Explain in detail about the approaches for object motion specifications.

In simple manual systems, the objects can be simply the artist drawings • In computer-generated animations, models are used • Examples of models: – a "flying logo" in a TV advertisement – a walking stick-man – a dinosaur attacking its prey in Jurassic Park Models Can Be

Rigid (i.e. they have no moving parts) • Articulated (subparts are rigid, but movement is allowed between the sub- parts) • Dynamic (using physical laws to simulate the motion) • Particle based (animating individual particles using the statistics of behaviour • Behaviour based (e.g. based on behaviour of real animals)

Path Specification:

Impression of movement can be created for two basic situations, or for their combination: – static object, moving camera – static camera, moving object • The path defines the sequence of locations (for either the camera or the object) for the consecutive time frames.

Static Object, Moving Camera:

xThe path specifies the spatial coordinates along which the camera moves

• The path is usually specified for a single point, e.g. the view reference point

5. Explain in detail about morphing.

Morphing is an image processing technique typically used as an animation tool for the metamorphosis from one image to another. The whole metamorphosis from one image to the other consists of fading out the source image and fading in the destination image. Thus, the early images in the sequence are much like the source image and the later images are more like the destination image. The middle image of the sequence is the average of the source image distorted halfway toward the destination image and the destination image distorted halfway back to the source image. This middle image is rather important for the whole morphing process. If it looks good then probably the entire animated sequence will look good. For example, when morphing between faces, the middle "face" often looks strikingly "life-like" but is neither the first nor the second person in the image.

6. Describe the creation of images by iterated functions.

1. Experimental copier

❺ Take an initial image I_0

❺ Produce new image I_1 by superimposing several reduced versions of I_0

❺ Feed I_1 back to the copier to generate I_2

❺ Repeat the process to obtain a sequence of images I_0, I_1, I_2,… called **orbit** of I_0 2.S-copier

- Superimposes three smaller versions of whatever image is fed onto it and repeated iterations may result in Sierpinski triangle

Generating images

❺ The 3 lenses present in the copier reduces the i/p image to one half of its size and moves it to a new position

❺ The reduced and shifted images are superimposed on printed o/p

❺ Each lens performs its own affine transformation

IFS – it is a collection of ‗N' affine transformations T_i for I = 1,2,…N *Theory of copying process*

Method:

- Using the lenses present in the copier draw the o/p image by transforming the points present in the i/p image

Working:

Let I be the i/p image to the copier, then the i-th lens builds the new set of points denoted by T_i (I) and adds them to the image being produced at the current iteration.

The o/p image is obtained by superimposing the three transformed images created by the three lenses

$$\text{o/p image} = T_1(I) \cup T_2(I) \cup T_3(I)$$

Overall mapping from i/p to o/p is denoted by W(.)

$$W(.) = T_1(.) \cup T_2(.) \cup T_3(.)$$

Drawing the k-th iterate

Choices of Initial images (I_0)

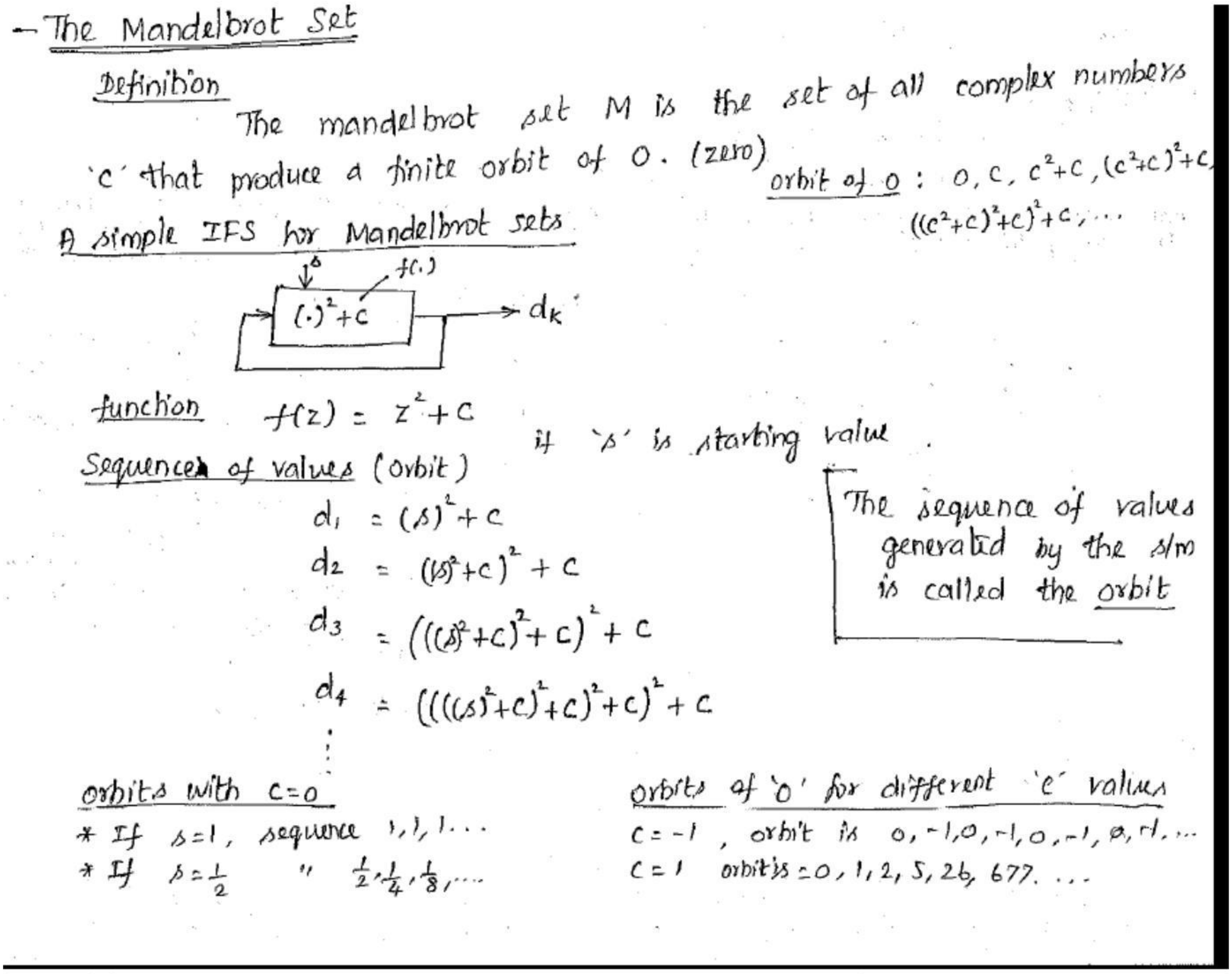
1. Polyline
2. Single point

Chaos Game (Random Iteration Algorithm)

- produces the picture of an attractor
- produces pictures in a non recursive

way Ex. Sierpinski gasket

7. Write notes on Mandelbrot sets.



– The Mandelbrot Set

Definition

The mandelbrot set M is the set of all complex numbers 'c' that produce a finite orbit of 0. (zero)

orbit of 0 : $0, c, c^2+c, (c^2+c)^2+c, ((c^2+c)^2+c)^2+c, \dots$

A simple IFS for Mandelbrot sets

s
f(.)
$(\cdot)^2+c$
d_k

function $f(z) = z^2 + c$

Sequences of values (orbit) if 's' is starting value.

$$d_1 = (s)^2 + c$$
$$d_2 = ((s)^2+c)^2 + c$$
$$d_3 = (((s)^2+c)^2+c)^2 + c$$
$$d_4 = ((((s)^2+c)^2+c)^2+c)^2 + c$$
$$\vdots$$

The sequence of values generated by the s/m is called the orbit

orbits with c=0

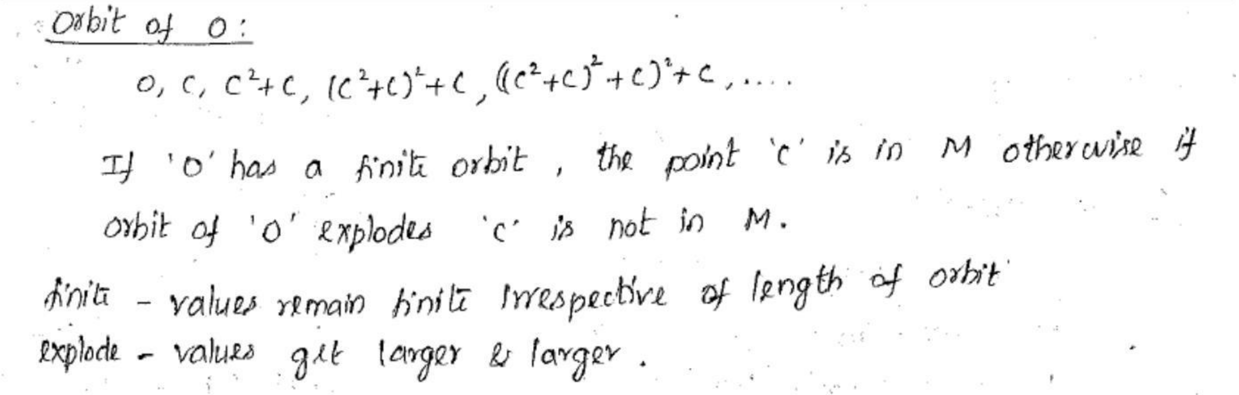
* If s=1, sequence 1, 1, 1 . . .

* If s = $\frac{1}{2}$ " $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \dots$

orbits of '0' for different 'c' values

c = -1 , orbit is 0, -1, 0, -1, 0, -1, 0, -1, ...

c = 1 orbit is = 0, 1, 2, 5, 26, 677. . . .



Orbit of 0 :

$0, c, c^2+c, (c^2+c)^2+c, ((c^2+c)^2+c)^2+c, \dots$

If '0' has a finite orbit, the point 'c' is in M otherwise if orbit of '0' explodes 'c' is not in M.

finite – values remain finite irrespective of length of orbit

explode – values get larger & larger.

Computing whether point c is in Mandelbrot Set Idea:

Check whether the orbit explodes or not if the orbit explodes then the point is not in M

Test for checking: If $|d_k|$ value ever exceeds the value of 2 , then the orbit will definitely explode.

Dwell: The number of iterations $|d_k|$ takes to exceed 2 is called the dwell of the orbit

Approach:

□ Set some upper limit Num for the number of iterations

If $|d_k|$ hasn‘t exceeded 2 after Num iterations assume that it will never exceed and conclude that ‗c‘ is in M

Dwell function:

For a given value of c = $c_x + c_y i$ the routine returns the no. of iterations $|d_k|$ required to exceed 2 or simply the no. of iterations

```
int dwell (double cx, double cy)
{// return true dwell of Num whichever is smaller
   #define Num 100
doubletmp, dx = cx , dy = cy , fsq = cx * cx + cy * cy;
for (int count = 0; count <= Num&&fsq<=4 ; count++)
        {       tmp = dx;     // save old real part
                dx = dx * dx – dy * dy + cx; // new real part
                dy = 2.0 * tmp * dy +cy ;   // new imaginary part
                fsq = dx * dx + dy * dy ;
        }
return count; // no. of iterations used
}
```

Drawing Mandelbrot Sets

Techniques:

❺ Assign black to points inside M & white to those outside M

❺ Use a range of colors (for small value sof ‗d‘ use blue as ‗d‘ approaches ‗Num‘ use red & green component (together form yellow)

8. Write notes on Julia sets.

Filled in Julia sets

The filed in Julia set at c, K_c , is the set of all starting points whose orbits are finite

Difference between Mandelbrot set &filled in Julia set

□ ‗c‘ can take different values	‗c‘ can take a single value
□ Use the same starting point 0	use different starting points

Basin of attraction:

If an orbit starts close enough to an attracting fixed point , it is sucked into that point. The set of points that are sucked in forms a basin of attraction for the fixed point P

Types of filled in Julia set

1. Connected
2. Cantor set

Julia sets ,J$_C$

For any given value of c, the Julia set **J$_C$**is the boundary of filled in Julia set K$_C$

Preimage:

☐ The point just before ‗s‘ in the sequence is called preimage

☐ Preimage is the inverse of the function **f(.) = (.)^2 + c**

☐ The collection of all preimages of any point in **J$_C$** is dense in **J$_C$** Drawing Julia set

Methodology:

find a point and place a dot at all of the point‘s preimages

Problem:

1. Finding the point
2. Keeping track of all the preimages Solution:

☐ Use backward iteration method Backward iteration method

9 Choose some point ‗z‘ in complex plane which may / may not in J$_C$

9 Begin iteration backwards

9 At each iteration choose on e of the two square roots randomly

9 Produce a new ‗z‘ value

9 Repeat the process until **J$_C$** emerge

Pseudo code:

```
do {
        if ( coin flip is heads )
                z = +√ z – c ;
        else
                z = - √ z – c ;
        draw dot at z;
}while(not bored);
```

9. Explain about random fractals.

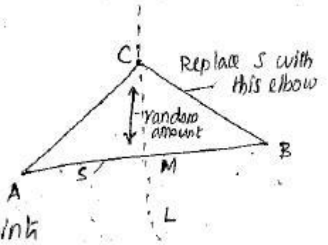
- Random Fractals

random fractals are used to provide naturalistic shapes for representing objects like coastlines, rugged mountains, grass & fire

* Fractalizing a segment

Approach

Replace each line segment with a random elbow at each step.



ex

Line segment S having endpoints A & B is replaced by two segments

10. Write notes on ray tracing.

Ray Tracing / Ray Casting

- Provides a related , powerful approach to render scenes

Overview of Ray Tracing Process

Pseudo code of a ray tracer:

```
for(int r = 0; r <nRows ; r++)
  for(int c = 0; c <nCols ; c++)
  {
    1. build the rc-th ray
    2. find all intersections of the rc-th ray with objects in the scene
    3. Identify the intersection that lies closest to and in front of the eye
    4. Compute the hit point where the ray hits this object and the normal vector at that point
    5. Find the color of the light returning to the eye along the ray from the point of intersection
    6. place the color in the rc-th pixel
  }
```

Intersection of a Ray with an Object

Common shapes used in ray tracing

- □ Sphere
- □ Cylinder
- □ Cone cube
- □ hex cone

If ‗S‘ is the starting point of a ray and ‗c‘ is its direction then the ray that intersects with a shape is given by

$r(t) = S + ct$ // implicit form of shape is $F(P)$ Condition for $r(t)$ to coincide with a point of the surface is $F(r(t)) = 0$

The hit time t_{hit} can be found by solving

$F(S + c\, t_{hit}) = 0$

Intersection of a Ray with the Generic Plane

- □ generic plane – xy plane or $z = 0$
- □ Implicit form is $F(x ,y , z) = z$
- □ The ray S+ct intersects the generic plane when
  $S_z + c_z t_h = 0$, where $t_h = -(S_z / c_z)$

If $c_z = 0$, the ray is moving parallel to the plane & there is no intersection Otherwise , the ray hits the plane at the point $P_{hit} = S – c(S_z / c_z)$

Intersection of a Ray with the Generic Shape

□ Consider a generic shape whose implicit form is $F(P) = |P|^2 – 1$

□ The point of intersection of the ray is given by $|S + ct|^2 - 1 = 0$ $|c|^2T^2 + 2.(S.C)t + (|S|^2 - 1) = 0$ which is of the form $At^2 + 2Bt + c = 0$

by solving $t_h = -(B/A) \pm(\surd B^2 – AC)/ A$

9 If $\mathbf{B^2 - Ac}$ is ‗-‘ ve , the ray misses the sphere

9 If $\mathbf{B^2 - Ac}$ is zero , the ray grazes the sphere at one point

9 If $\mathbf{B^2 - Ac}$ is ‗+‘ ve , there are 2 hit times t_1& t_2