

**D.R. AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY**

**JALANDHAR,**

**PUNJAB, INDIA**



## **ASSIGNMENT**

**Sub: COMPUTER GRAPHICS**

**(CSX-308)**

**SUBMITTED TO:**

**Rahul Aggarwal**

**SUBMITTED BY:**

**Ankit Goyal**

**17103011**

Ques 1-> Apply Liang Barsky line clipping algorithm algorithm to the line with coordinates (30, 60) and (60, 25) against the window.

(Xmin,Ymin)=(10,10)and(Xmax,Ymax)=(50,50)

(Xmin,Ymin)=(10,10) and (Xmax,Ymax)=(50,50)

P1 (30, 60) and P2 = (60, 25)

Set Umin = 0 and Umax = 1

ULeft =  $q1 / p1$

$$\begin{aligned} &= X1 - Xmin / - \Delta X \\ &= 30 - 10 / - (60 - 30) \\ &= 20 / - 30 \\ &= -0.67 \end{aligned}$$

URight =  $q2 / p2$

$$\begin{aligned} &= Xmax - X1 / \Delta X \\ &= 50 - 30 / (60 - 30) \\ &= 20 / 30 \\ &= 0.67 \end{aligned}$$

UBottom =  $q3 / p3$

$$\begin{aligned} &= Y1 - Ymin / - \Delta Y \\ &= 60 - 10 / - (25 - 60) \\ &= 50 / 35 \\ &= 1.43 \end{aligned}$$

UTop =  $q4 / p4$

$$\begin{aligned} &= Ymax - Y1 / \Delta Y \\ &= 50 - 60 / (25 - 60) \\ &= -10 / - 35 \\ &= 0.29 \end{aligned}$$

Since,

$U_{\text{Left}} = -0.57$  which is less than  $U_{\text{min}}$ . Therefore we ignore it.

Similarly

$U_{\text{Bottom}} = 1.43$  which is greater than  $U_{\text{max}}$ . So we ignore it.

$U_{\text{Right}} = U_{\text{min}} = 0.67$  (Entering)

$U_{\text{Top}} = U_{\text{max}} = 0.29$  (Exiting)

We have

$U_{\text{Top}} = 0.29$  and  $U_{\text{Right}} = 0.67$

$Q - P = (\Delta X, \Delta Y) = (30, -35)$

Since  $U_{\text{min}} > U_{\text{max}}$ , there is no line segment to draw.

Ques 2-> Find the normalization transformation window to viewport with window lower left corner at(1,1) and upper right corner at (3,5) onto a viewpoint with lower left corner at (0,0) and upper right corner at [1/2][1/2].

Given window has coordinates :

$$X_{wmin} = 1 \quad X_{wmax} = 3$$

$$Y_{wmin} = 1 \quad Y_{wmax} = 5$$

Coordinates for view port are :

$$X_{vmin} = 0 \quad X_{vmax} = \frac{1}{2} = 0.5$$

$$Y_{vmin} = 0 \quad Y_{vmax} = \frac{1}{2} = 0.5$$

We know that,

$$S_x = (X_{vmax} - X_{vmin}) / (X_{wmax} - X_{wmin})$$

$$= (0.5 - 0) / (3 - 1)$$

$$= 0.25$$

$$S_y = (Y_{vmax} - Y_{vmin}) / (Y_{wmax} - Y_{wmin})$$

$$= (0.5 - 0) / (5 - 1)$$

$$= 0.125$$

So, transformation equation is given by :

$$\begin{aligned}
 T \cdot S \cdot T^{-1} &= \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ X_{vmin} - X_{wmin}S_x & Y_{vmin} - Y_{wmin}S_y & 1 \end{bmatrix} \\
 &= \begin{bmatrix} 0.25 & 0 & 0 \\ 0 & 0.125 & 0 \\ 0 - (1 \times 0.25) & 0 - (1 \times 0.125) & 1 \end{bmatrix} \\
 &= \begin{bmatrix} 0.25 & 0 & 0 \\ 0 & 0.125 & 0 \\ -0.25 & -0.125 & 1 \end{bmatrix}
 \end{aligned}$$

So, This is the normalization transformation window to given viewport.

Ques 3-> Write steps of Liang–Barsky line clipping algorithm.

The Liang-Barsky algorithm is a line clipping algorithm. This algorithm is more efficient than Cohen–Sutherland line clipping algorithm and can be extended to 3-Dimensional clipping. This algorithm is considered to be the faster parametric line-clipping algorithm. The following concepts are used in this clipping:

1. The parametric equation of the line.
2. The inequalities describing the range of the clipping window which is used to determine the intersections between the line and the clip window.

The parametric equation of a line can be given by,

$$\begin{aligned} X &= x_1 + t(x_2 - x_1) \\ Y &= y_1 + t(y_2 - y_1) \end{aligned}$$

Where,  $t$  is between 0 and 1.

Then, writing the point-clipping conditions in the parametric form:

$$x_{W_{\min}} \leq x_1 + t(x_2 - x_1) \leq x_{W_{\max}}$$

$$y_{W_{\min}} \leq y_1 + t(y_2 - y_1) \leq y_{W_{\max}}$$

The above 4 inequalities can be expressed as,

$$tp_k \leq q_k$$

Where  $k = 1, 2, 3, 4$  (correspond to the left, right, bottom, and top boundaries, respectively).

The  $p$  and  $q$  are defined as,

$$p_1 = -(x_2 - x_1), \quad q_1 = x_1 - x_{W_{\min}} \text{ (Left Boundary)}$$

$$p_2 = (x_2 - x_1), \quad q_2 = x_{W_{\max}} - x_1 \text{ (Right Boundary)}$$

$$p_3 = -(y_2 - y_1), \quad q_3 = y_1 - y_{W_{\min}} \text{ (Bottom Boundary)}$$

$$p_4 = (y_2 - y_1), \quad q_4 = y_{W_{\max}} - y_1 \text{ (Top Boundary)}$$

When the line is parallel to a view window boundary, the  $p$  value for that boundary is zero.

When  $p_k < 0$ , as  $t$  increase line goes from the outside to inside (entering).

When  $p_k > 0$ , line goes from inside to outside (exiting).

When  $p_k = 0$  and  $q_k < 0$  then line is trivially invisible because it is outside view window.

When  $p_k = 0$  and  $q_k > 0$  then the line is inside the corresponding window boundary.

Following are the conditions that are needed to determine the position of the line

CONDITION	POSITION OF LINE
$p_k = 0$	parallel to the clipping boundaries
$p_k = 0$ and $q_k < 0$	completely outside the boundary
$p_k = 0$ and $q_k \geq 0$	inside the parallel clipping boundary
$p_k < 0$	line proceeds from outside to inside
$p_k > 0$	line proceeds from inside to outside

The Algorithm of the above is as follows :

1. Set  $t_{\min}=0$ ,  $t_{\max}=1$ .
2. Calculate the values of  $t$  ( $t(\text{left})$ ,  $t(\text{right})$ ,  $t(\text{top})$ ,  $t(\text{bottom})$ ),
  - (i) If  $t < t_{\min}$  ignore that and move to the next edge.
  - (ii) else separate the  $t$  values as entering or exiting values using the inner product.
  - (iii) If  $t$  is entering value, set  $t_{\min} = t$ ; if  $t$  is existing value, set  $t_{\max} = t$ .
3. If  $t_{\min} < t_{\max}$ , draw a line from  $(x_1 + t_{\min}(x_2-x_1), y_1 + t_{\min}(y_2-y_1))$  to  $(x_1 + t_{\max}(x_2-x_1), y_1 + t_{\max}(y_2-y_1))$
4. If the line crosses over the window,  $(x_1 + t_{\min}(x_2-x_1), y_1 + t_{\min}(y_2-y_1))$  and  $(x_1 + t_{\max}(x_2-x_1), y_1 + t_{\max}(y_2-y_1))$  are the intersection point of line and edge.

Ques 4-> What do you mean by polygon net or mesh? Explain various ways of representing it along with its merits and demerits.

In computer graphics a polygon mesh is the **collection of vertices, edges, and faces** that make up a 3D object. A polygon mesh defines the shape and contour of every 3D character & object, whether it be used for 3D animated film, advertising, or video games.

Polygon meshes can be used to model almost any object.

A polygonal mesh can be rendered using hidden surface removal algorithms. The polygon mesh can be represented by three ways –

- Explicit representation
- Pointers to a vertex list
- Pointers to an edge list

Below are the merits :

- It can be used to model almost any object.
- They are easy to represent as a collection of vertices.
- They are easy to transform.
- They are easy to draw on computer screen.

And the demerits :

- Curved surfaces can only be approximately described.
- It is difficult to simulate some type of objects like hair or liquid.

Ques 5-> What are the limitations of Bezier curves?

Bezier curves are the curves which uses approximate tangents and controlling points are used to generate curves

Following are the limitation of the Bezier curves :

1. It although unlimited but contains a finite number of control points.
2. There are many curves which cannot be completely expressed by Bezier curves.
3. Most cubic rational Bezier curves may be included in that class, and most curves of higher degree also.
4. It is liable to approximate in nature.
5. By means of an arbitrary number of points those could probably be approximated arbitrarily not exactly.

Ques 6-> Explain B-Spline curves along with its properties

The Bezier-curve produced by the Bernstein basis function has limited flexibility.

- First, the number of specified polygon vertices fixes the order of the resulting polynomial which defines the curve.
- The second limiting characteristic is that the value of the blending function is nonzero for all parameter values over the entire curve.

The B-spline basis contains the Bernstein basis as the special case. The B-spline basis is non-global.

A B-spline curve is defined as a linear combination of control points  $P_i$  and B-spline basis function  $N_{i,k}(t)$  given by

$$C(t) = \sum_{i=0}^n P_i N_{i,k}(t), \quad n \geq k - 1,$$

$$t \in [t_{k-1}, t_{n+1}]$$

Where,

- $\{P_i: i=0, 1, 2, \dots, n\}$  are the control points
- $k$  is the order of the polynomial segments of the B-spline curve. Order  $k$  means that the curve is made up of piecewise polynomial segments of degree  $k - 1$ ,
- the  $N_{i,k}(t)$  are the “normalized B-spline blending functions”. They are described by the order  $k$  and by a non-decreasing sequence of real numbers normally called the “knot sequence”.

$$t_i : i = 0, \dots, n + K$$

The  $N_{i,k}$  functions are described as follows –

$$N_{i,1}(t) = \begin{cases} 1, & \text{if } t \in [t_i, t_{i+1}) \\ 0, & \text{Otherwise} \end{cases}$$

and if  $k > 1$ ,

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t)$$

and

$$t \in [t_{k-1}, t_{n+1})$$



## Properties of B-spline Curve

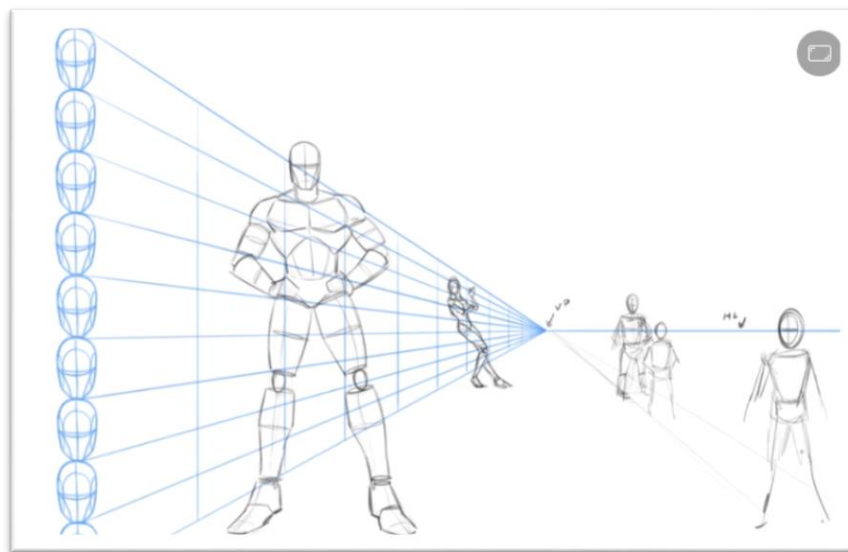
B-spline curves have the following properties –

- The sum of the B-spline basis functions for any parameter value is 1.
- Each basis function is positive or zero for all parameter values.
- Each basis function has precisely one maximum value, except for  $k=1$ .
- The maximum order of the curve is equal to the number of vertices of defining polygon.
- The degree of B-spline polynomial is independent on the number of vertices of defining polygon.
- B-spline allows the local control over the curve surface because each vertex affects the shape of a curve only over a range of parameter values where its associated basis function is nonzero.
- The curve exhibits the variation diminishing property.
- The curve generally follows the shape of defining polygon.
- Any affine transformation can be applied to the curve by applying it to the vertices of defining polygon.
- The curve line within the convex hull of its defining polygon.

Ques 7-> Write short notes on the following –

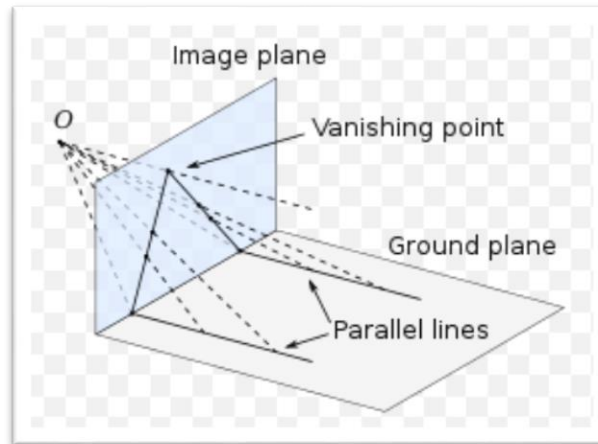
a. Perspective foreshortening

Foreshortening, is the visual effect or the optical illusion that causes an object or the given distance to appeared as if it is smaller /shorter than actually is as it appears shorter because as it is angled towards the viewer



b. Principle vanishing points

In graphical perspective, a **vanishing point** is an abstract **point** on the image plane where 2D projections (or drawings) of a set of parallel lines in 3D space appear to converge. The **vanishing point** may also be referred to as the “direction **point**”.

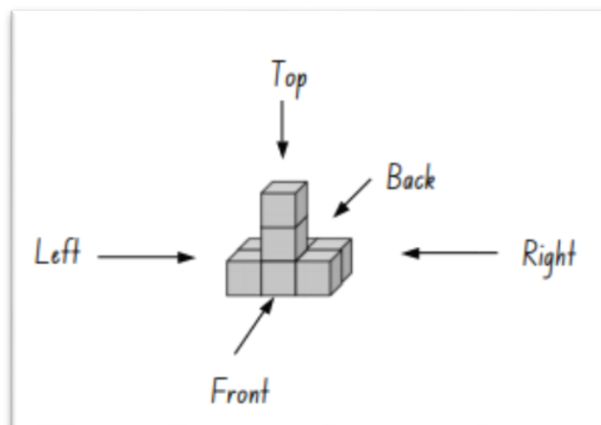


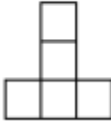


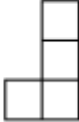
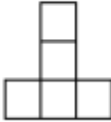
c. Front, top and side views of an object

Front view, the view in which the viewing angle is in front of the object to be looked at

Top view, the view in which the viewing angle is from the top of the object

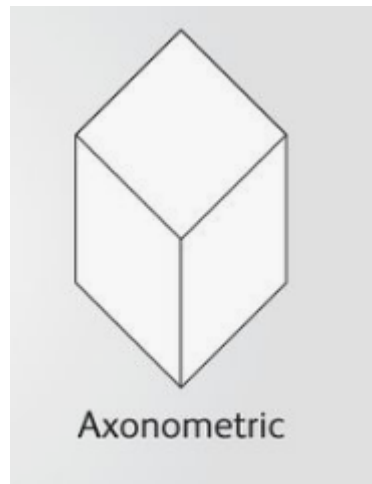
Side view, the view in which the viewing angle is on the side ways



Front View	Top View	Right Side View	Left Side View	Back View
				

d. Axonometric projections

**Axonometric projection** is a type of orthographic **projection** used for creating a pictorial drawing of an object, where the lines of sight are perpendicular to the plane of **projection**, and the object is rotated around one or more of its axes to reveal multiple sides.



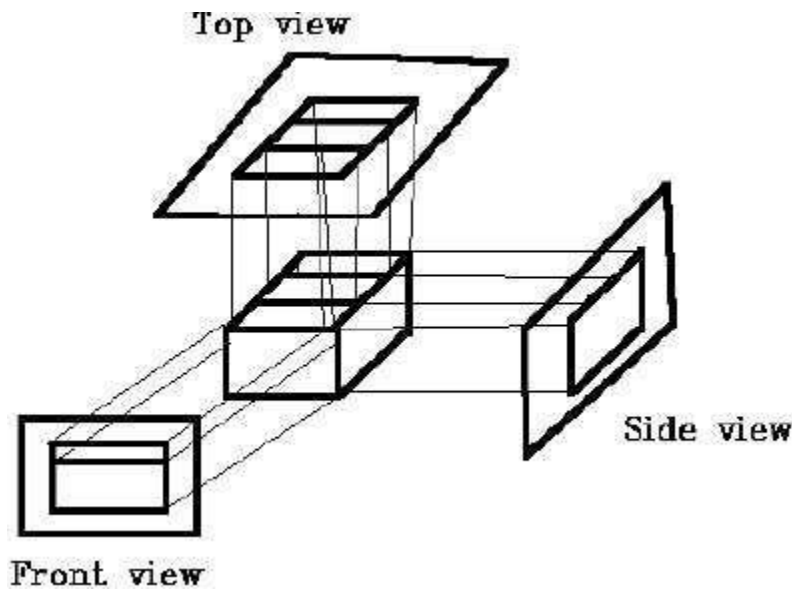
a) Orthographic projections

Orthographic projections are characterized by the fact that the direction of projection is perpendicular to the view plane. When the direction of projection is parallel to any of the principal axes, this produces the front, top, and side views of mechanical drawings (also referred to as multi view drawings)

In orthographic projection the direction of projection is normal to the projection of the plane.

There are three types of orthographic projections –

- Front Projection
- Top Projection
- Side Projection



#### b) Oblique projections

In oblique projection, the direction of projection is not normal to the projection of plane. In oblique projection, we can view the object better than orthographic projection.

There are two types of oblique projections – Cavalier and Cabinet. The Cavalier projection makes  $45^\circ$  angle with the projection plane. The projection of a line perpendicular to the view plane has the same length as the line itself in Cavalier projection. In a cavalier projection, the foreshortening factors for all three principal directions are equal.

#### c) Dimetric projections

In dimetric projection, the direction of viewing is such that two of the three axes of space appear equally foreshortened, of which the attendant scale and angles of presentation are determined according to the angle of viewing; the scale of the third direction is determined separately.

#### d) Cabinet projections

The term cabinet projection (sometimes cabinet perspective) stems from its use in illustrations by the furniture industry. Like cavalier perspective, one face of the projected object is parallel to the viewing plane, and the third axis is projected as going off in an angle (typically  $30^\circ$  or  $45^\circ$  or  $\arctan(2) = 63.4^\circ$ ). It is an oblique projection in mechanical drawing in which dimensions parallel to the third axis of the object are shortened one half to overcome apparent distortion.

#### e) Cavalier projections

The Cavalier projection makes  $45^\circ$  angle with the projection plane. The projection of a line perpendicular to the view plane has the same length as the line itself in Cavalier projection. In a cavalier projection, the foreshortening factors for all three principal directions are equal. Cavalier

Projection CAVALIER PROJECTION is a form of oblique projection in which the projection lines are presumed to make a 45-degree vertical and a 45-degree horizontal angle with the plane of projection.

---

### Q.8)

#### a) Z Buffer Algorithm

It is also called a Depth Buffer Algorithm. Depth buffer algorithm is simplest image space algorithm. For each pixel on the display screen, we keep a record of the depth of an object within the pixel that lies closest to the observer. In addition to depth, we also record the intensity that should be displayed to show the object. Depth buffer is an extension of the frame buffer. Depth buffer algorithm requires 2 arrays, intensity and depth each of which is indexed by pixel coordinates (x, y)

#### Algorithm

For all pixels on the screen, set depth [x, y] to 1.0 and intensity [x, y] to a background value.

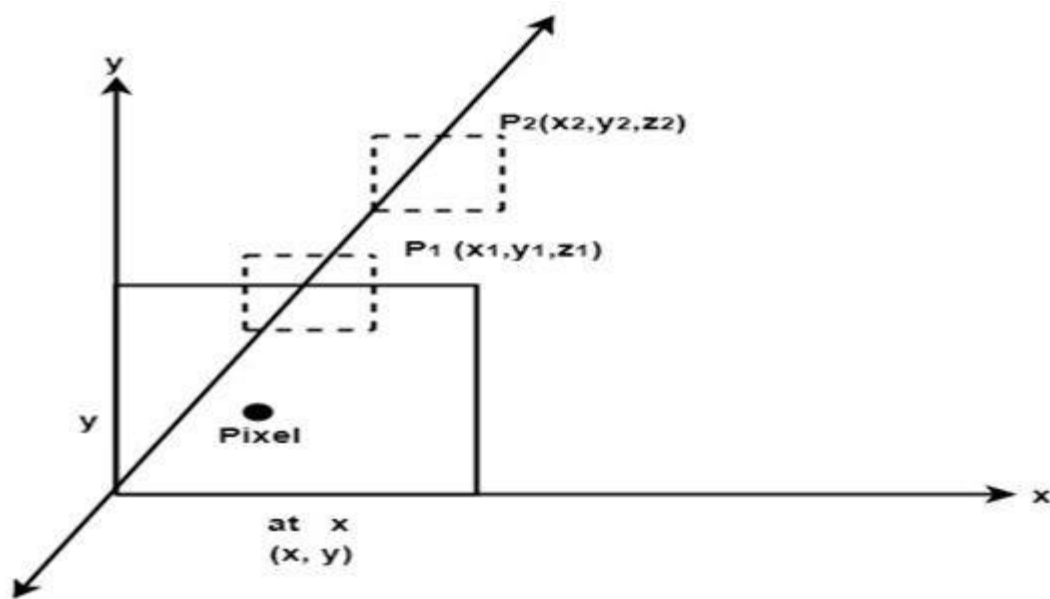
For each polygon in the scene, find all pixels (x, y) that lie within the boundaries of a polygon when projected onto the screen. For each of these pixels:

(a) Calculate the depth z of the polygon at (x, y)

(b) If  $z < \text{depth}[x, y]$ , this polygon is closer to the observer than others already recorded for this pixel. In this case, set depth [x, y] to z and intensity [x, y] to a value corresponding to polygon's shading. If instead  $z > \text{depth}[x, y]$ , the polygon already recorded at (x, y) lies closer to the observer than does this new polygon, and no action is taken.

3. After all, polygons have been processed; the intensity array will contain the solution.

4. The depth buffer algorithm illustrates several features common to all hidden surface algorithms.



5. First, it requires a representation of all opaque surface in scene polygon in this case.

6. These polygons may be faces of polyhedral recorded in the model of scene or may simply represent thin opaque 'sheets' in the scene.

7. The most important feature of the algorithm is its use of a screen coordinate system. Before step 1, all polygons in the scene are transformed into a screen coordinate system using matrix multiplication.

#### b) Painter's or depth sort algorithm

Steps performed in-depth sort

1. Sort all polygons according to z coordinate.
2. Find ambiguities of any, find whether z coordinate overlap, split polygon if necessary.
3. Scan convert each polygon in increasing order of z coordinate.

Algorithm

Step1: Start Algorithm

Step2: Sort all polygons by z value keep the largest value of z first.

Step3: Scan converts polygons in this order.

Test is applied

1. Does A is behind and non-overlapping B in the dimension of Z as shown in fig (a)
2. Does A is behind B in z and no overlapping in x or y as shown in fig (b)

3. If A is behind B in Z and totally outside B with respect to view plane as shown in fig (c)  
If A is behind B in Z and B is totally inside A with respect to view plane as shown in fig (d)

The success of any test with single overlapping polygon allows F to be painted.

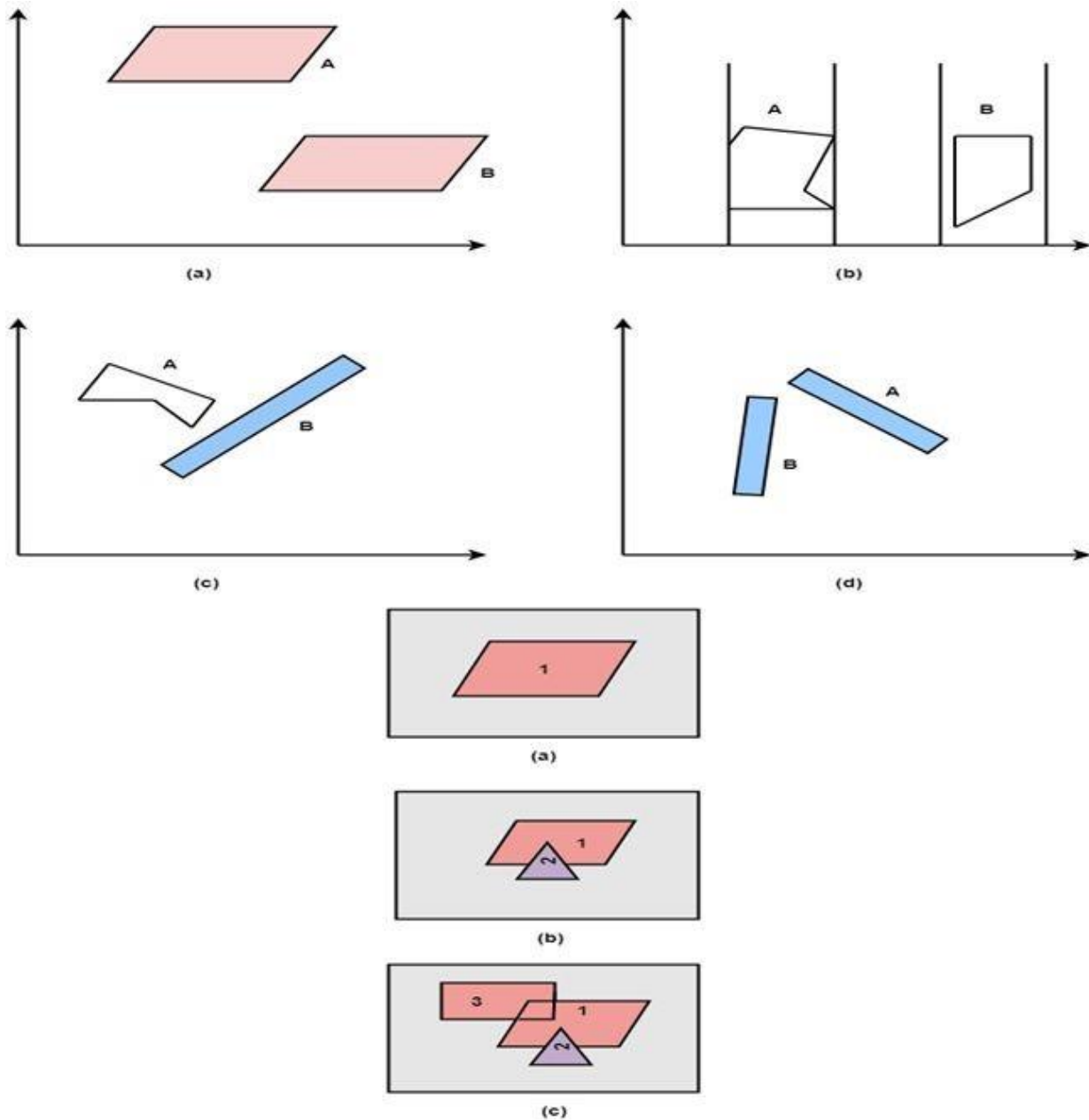


Figure showing addition of surface one by one and then painting done using painter algorithm

### c) Scan line algorithm

Step1: Start algorithm

Step2: Initialize the desired data structure

1. Create a polygon table having color, edge pointers, coefficients
2. Establish edge table contains information regarding, the endpoint of edges, pointer to polygon, inverse slope.
3. Create Active edge list. This will be sorted in increasing order of x.
4. Create a flag F. It will have two values either on or off.

Step3: Perform the following steps for all scan lines

1. Enter values in Active edge list (AEL) in sorted order using y as value
2. Scan until the flag, i.e. F is on using a background color
3. When one polygon flag is on, and this is for surface  $S_1$  enter color intensity as  $I_1$  into refresh buffer
4. When two or image surface flag are on, sort the surfaces according to depth and use intensity value  $S_n$  for the nth surface. This surface will have least z depth value
5. Use the concept of coherence for remaining planes.

Step4: Stop Algorithm

### d) Area Subdivision Algorithm

It was invented by John Warnock and also called a Warnock Algorithm. It is based on a divide & conquer method. It uses fundamental of area coherence. It is used to resolve the visibility of algorithms. It classifies polygons in two cases i.e. trivial and non-trivial.

Trivial cases are easily handled. Non trivial cases are divided into four equal subwindows. The windows are again further subdivided using recursion until all polygons classified trivial and non trivial.

It divides or classifies polygons in four categories:

1. Inside surface
2. Outside surface
3. Overlapping surface
4. Surrounding surface

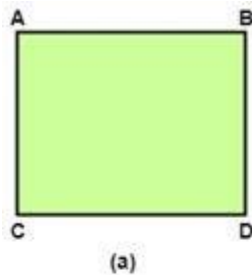
1. Inside surface: It is surface which is completely inside the surrounding window or specified boundary as shown in fig (c)

2. Outside surface: The polygon surface completely outside the surrounding window as shown in fig (a)

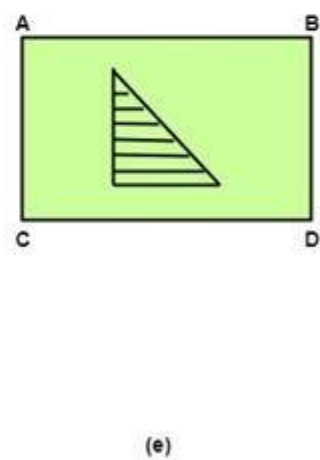
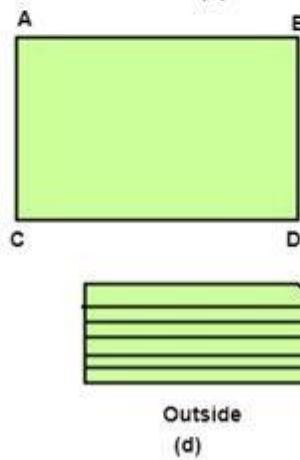
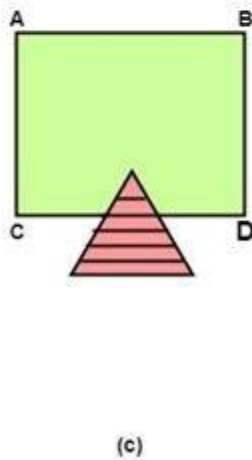
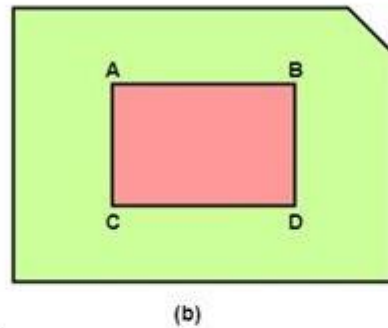


3. Overlapping surface: It is polygon surface which completely encloses the surrounding window as shown in fig (b)

4. Overlapping surface: It is surface partially inside or partially outside the surface area as shown in fig (c)



ABCD is current window against which particular window is determined to be of either of four categories



Surface of object intersecting desired window

Surface is outside specified window

Surface is inside specified window

Q.9).

**Steps for designing animation sequences :**

a) **Storyboard Layout:**

This standard approach for animated cartoons is applied to other animation applications as well, although there are many special applications that do not follow this sequence. Real-time computer animations produced by Bight simulators, for instance, display motion sequences in response to settings on the aircraft controls. And visualization applications are generated by the solutions of the numerical models. For frame-by-frame animation, each frame of the scene is separately generate and stored. Later, the frames can be recoded on film or they can be consecutively displayed in "real-time playback" Mode. It is the outline of the action. It defines the motion sequence as a set of basic events that are to take place. Depending on the type of animation to be produced, the storyboard could consist of a set of rough sketches or it could be a list of the basis ideas for the motion.

b) **Object definitions:**

Each object participating in the action is given object definition, such as terms of basic shapes, such as polygons or splines.

c) **Key frame specifications**

A key frame in animation and filmmaking is a drawing that defines the starting and ending points of any smooth transition. A sequence of key frames defines which movement the spectator will see, but the position of the key frames on the film, defines the timing of the movement. 2 or 3 key frames can be present for a span of a second.

d) **Generation of in-between frames**

Animation packages, such as wave front, provide special functions for designing the animation and processing individual objects. Some steps included in the development of animation sequence are Object manipulation and rendering Camera motions Generation of in – between One function available in animation packages is provided to store and manage the database (object shapes and associated parameters are stored and updated in the data

