Q1) Imagine you are designing a computer-generated artwork that features the Koch curve prominently. So first write the properties of Fractal and then give the steps to construct the Koch curve to ensure the accuracy and aesthetic appeal of your design.

Ans - Fractals are complex geometric shapes that exhibit self-similarity at different scales. They possess several key properties:

1. Self-Similarity: Fractals exhibit self-similarity, which means that smaller portions of the fractal resemble the whole structure. This property occurs at various scales, allowing you to zoom in and see similar shapes and patterns repeatedly.
2. Infinite Detail: Fractals exhibit an infinite level of detail. Regardless of how closely you examine a fractal, you will always find smaller and more intricate shapes within it.
3. Iteration: Many fractals are generated through a process of iteration, where a simple geometric shape is repeatedly transformed or replaced with a more complex pattern.
4. Non-integer Dimension: Fractals often have a non-integer or fractal dimension, which means their dimension falls between that of one-dimensional and two-dimensional objects.
5. Scale Invariance: Fractals look the same or exhibit similar patterns at different levels of magnification, which is known as scale invariance.

Now, let's construct the Koch curve, a famous fractal, step by step:

The Koch curve is generated through a recursive process, repeatedly replacing line segments with smaller segments while adding triangular patterns. Here's how to construct it:
Step 1: Start with a straight-line segment. This will be the initial iteration.
Step 2: Divide the line segment into three equal parts.
Step 3: Replace the middle segment with an equilateral triangle, pointing outward.
Step 4: Repeat the process for each of the four-line segments (the original two outer segments and the new two segments formed by the sides of the triangle). For each segment, divide it into three equal parts and replace the middle part with an outward-pointing equilateral triangle.
Step 5: Continue this process for as many iterations as desired. With each iteration, the Koch curve becomes more intricate, and you will see the self-similar pattern emerge at different scales.

The Koch curve is a classic example of a self-replicating fractal, and you'll notice that as you repeat these steps, the curve becomes more and more detailed, exhibiting the properties of self-similarity and infinite complexity. The more iterations you perform, the closer the curve approximates its fractal nature.

Q2) Imagine you're working on an advertising campaign that requires a visually stunning and highly dynamic animation. For that explain different principles of animation to achieve the desired impact and quality.

Ans - The principles of animation are a set of guidelines and techniques that were developed by Disney animators Ollie Johnston and Frank Thomas in their 1981 book "The Illusion of Life: Disney Animation." These principles are fundamental to the art of animation and help create the

illusion of life and movement in animated characters. There are 12 main principles of animation, which include:

1. Squash and Stretch: This principle involves exaggerating the shape and form of an object or character to convey a sense of weight, flexibility, and impact. It adds realism and impact to animations.
2. Anticipation: Anticipation is a technique where an action is preceded by a smaller, opposite movement. It helps the audience anticipate the main action and makes the animation more believable.
3. Staging: Staging involves presenting an action or idea clearly so that the audience can easily understand what is happening. It also includes the use of composition, camera angles, and lighting to focus attention on the most important elements of the scene.
4. Straight Ahead and Pose-to-Pose: These are two different approaches to animating. "Straight Ahead" means animating frame by frame, starting at the beginning and working through to the end. "Pose-to-Pose" involves creating key poses at important moments and then filling in the in-between frames.
5. Follow Through and Overlapping Action: These principles deal with the continuation of motion. "Follow Through" refers to the parts of a character that continue moving after the character has stopped, and "Overlapping Action" involves having different parts of a character move at different times to create a more natural and fluid animation.
6. Slow In and Slow Out: Objects in motion tend to start and stop gradually. This principle emphasizes the need to ease into and out of an action, creating more fluid and realistic movement.
7. Arcs: Most natural movements follow a curved path, so animators often use arcs to create smoother and more natural motion. This principle is particularly important for character movements and camera actions.
8. Secondary Action: Secondary actions are additional movements that support and enhance the main action. These actions add depth and realism to the animation and make characters feel more alive.
9. Timing: Timing is critical for conveying a character's mood, personality, and physical properties. It refers to the speed and spacing of actions and helps create the desired emotional impact.
10. Exaggeration: Exaggeration involves pushing the characteristics and movements of a character or object to make them more dynamic and entertaining. It can be used to emphasize certain features or emotions.
11. Solid Drawing: Solid drawing is the application of traditional drawing and art principles to create a sense of three-dimensionality and volume in 2D animation. This principle adds depth and realism to characters and objects.

12. Appeal: Appeal refers to the attractiveness and relatability of characters and their actions. Characters should be designed and animated in a way that captures the audience's attention and makes them emotionally invested in the story.

These principles are foundational for animators and can be applied in various forms of animation, including traditional hand-drawn animation, 3D computer animation, and stop-motion animation. They help create engaging and lifelike animations that connect with the audience.

Q3) You are a civil engineer working on the development of a modern urban infrastructure project that includes the construction of a complex underground subway system. The efficient visualization and analysis of the intricate network of tunnels, stations, and utility conduits are paramount to ensure safety and functionality.

In this context, please elucidate the principles of Warnock's hidden surface removal algorithm.

Ans -

Warnock's Algorithm (Area Subdivision Algorithm)

An interesting approach to the hidden-surface problem was developed by Warnock. He developed area subdivision algorithm which subdivides each area into four equal squares. At each stage in the recursive-subdivision process, the relationship between projection of each polygon and the area of interest is checked for four possible relationships :

1. Surrounding Polygon - One that completely encloses the (shaded) area of interest
2. Overlapping or Intersecting Polygon - One that is partly inside and partly outside the area
3. Inside or Contained Polygon - One that is completely inside the area
4. Outside or Disjoint Polygon - One that is completely outside the area

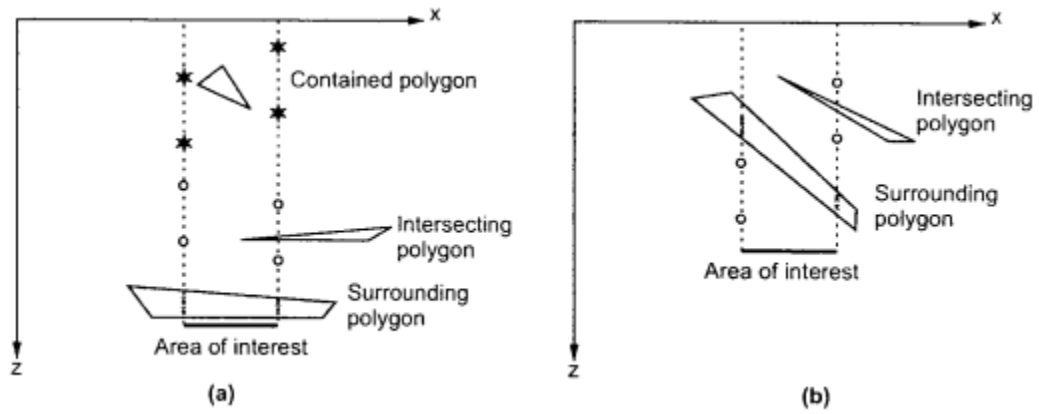
(a) Surrounding (b) Overlapping (c) Inside or Contained (d) Outside or Disjoint

Fig. Possible relationships with polygon surfaces and the area of interest

After checking four relationships we can handle each relationship as follows :

1. If all the polygons are disjoint from the area, then the background colour is displayed in the area.
2. If there is only one intersecting or only one contained polygon, then the area is first filled with the background colour, and then the part of the polygon contained in the area is filled with colour of polygon.
3. If there is a single surrounding polygon, but no intersecting or contained polygons, then the area is filled with the colour of the surrounding polygon.
4. If there are more than one polygon intersecting, contained in, or surrounding the area then we have to do some more processing.

In Fig. (a), the four intersections of surrounding polygon are all closer to the viewpoint than any of the other intersections. Therefore, the entire area is filled with the colour of the surrounding polygon.



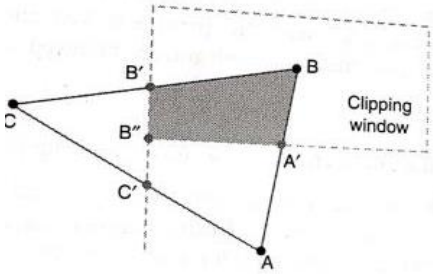
However, Fig. 8.12 (b) shows that surrounding polygon is not completely in front of the intersecting polygon. In such case we cannot make any decision and hence Warnock's algorithm subdivides the area to simplify the problem. This is illustrated in Fig. 8.13. As shown in the Fig. 8.13 (a) we can not make any decision about which polygon is in front of the other. But after dividing area of interest polygon 1 is ahead of the polygon 2 in left area and polygon 2 is ahead of polygon 1 in the right area. Now we can fill these two areas with corresponding colours of the polygons.

The Warnock's algorithm stops subdivision of area only when the problem is simplified or when area is only a single pixel.

Algorithm

1. Initialize the area to be the whole screen.
2. Create the list of polygons by sorting them with their z-values of vertices. Don't include disjoint polygons in the list because they are not visible.
3. Find the relationship of each polygon.
4. Perform the visibility decision test
   a) If all the polygons are disjoint from the area, then fill area with background colour.
   b) If there is only one intersecting or only one contained polygon then first fill entire area with background colour and then fill the part of the polygon contained in the area with the colour of polygon.
   c) If there is a single surrounding polygon, but no intersecting or contained polygons, then fill the area with the colour of the surrounding polygon.
   d) If surrounding polygon is closer to the viewpoint than all other polygons, so that all other polygons are hidden by it, fill the area with the colour of the surrounding polygon.
   e) If the **area is the pixel** (x, y), and neither a,b,c, nor d applies, compute the z coordinate at pixel (x, y) of all polygons in the list. The pixel is then set to colour of the polygon which is closer to the viewpoint.
5. If none of the above tests are true then subdivide the area and go to step 2.

Q4) Consider the polygon ABC as shown in the fig. Clip this polygon using Sutherland-Hodgeman polygon clipping algorithm against the rectangular window and obtain the resulting visible portion of the polygon.



Ans -

Following the algorithm, the clipped polygon is obtained as hereunder:

Input edge	→	Left clipper	→	Right clipper	→	Bottom clipper	→	Top clipper
{A, B}:		{in-in}→{B}						
{B, C}:		{in-out}→{B′}		{B′, B}:{in-in}→{B′}				
{C, A}:		{out-in}→{C′, A}		{B′, C}:{in-in}→{C′}		{B′, C′}:{in-out}→{B″}		
				{C′, A}:{in-in}→{A}		{C′, A}: ({out-out}→{ }		
				{A, B}:{in-in}→{B}		{A, B}: {out-in}→{A′, B}		{B″, A′}::{in-in}→{A′}
						{B, B′}:{in-in}→{B′}		{A′, B}:{in-in}→{B}
								{B, B′}:{in-in}→{B′}
								{B′, B″}:{in-in}→{B′}

The algorithm results into a clipped polygon with a vertex sequence A′, B, B′, B″. Note that convex polygons are corrected clipped, but concave polygons are not, they show erroneous line as shown in Figure .

A concave polygon produces two or more connected areas. This occurs when a clipped polygon has two or more separate components but overlaps a single vertex list, and the list

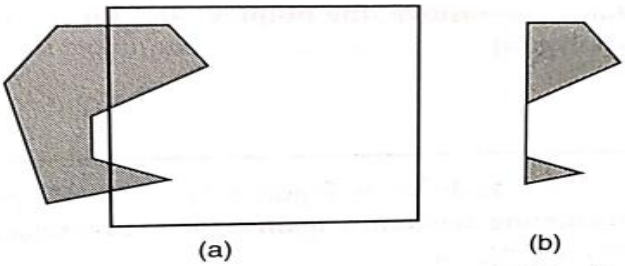


Figure (a) Clipping of a concave polygon; (b) the clip polygon.

vertex is always joined with the previous vertex of the current component. To overcome this, the concave polygon should be split into two or more convex polygon and each convex polygon should be clipped separately.

Q5) How the line between (30, 60), and (60, 20) is clipped against window with (Xwmin, Ywmin) = (10, 10) and (Xwmax, Ywmax) = (50, 50) using Liang-Barsky line clipping algorithm.

Here $x_{wmin} = 10$, $x_{wmax} = 50$

$y_{wmin} = 10$, $y_{wmax} = 50$

End points of line are $A = (x_1, y_1) = (30, 60)$ &

$B = (x_2, y_2) = (60, 20)$

Let us compare p_i, q_i and r_i for $i = 1, 2, 3, 4$

$p_1 = (-\Delta x) = -(x_2 - x_1) = -(60 - 30) = -30$

$p_2 = (\Delta x) = 30$

$p_3 = (\Delta y) = -(y_2 - y_1) = -(20 - 60) = 40$

$p_4 = (\Delta y) = -40$

None of the pi is zero, so line is neither horizontal nor vertical

Let us compute q_i ,

$q_1 = x_1 - x_{wmin} = 30 - 10 = 20$

$q_2 = x_{wmax} - x_1 = 50 - 30 = 20$

$q_3 = y_1 - y_{wmin} = 60 - 10 = 50$

$q_4 = y_{wmax} - y_1 = 50 - 60 = -10$

$\therefore r_1 = q_1 / p_1 = -0.67$

$\therefore r_2 = q_2 / p_2 = 0.67$

$\therefore r_3 = q_3 / p_3 = 1.25$

$\therefore r_4 = q_4 / p_4 = 0.25$.

$u_1 = \max(0, \text{all } r_i \text{ having } p_i < 0)$

$= \max(0, r_1, r_4)$

$= \max(0, -0.67, 0.25) = 0.25$

$u_2 = \min(1, \text{all } r_i \text{ having } p_i > 0)$

$= \min(1, r_2, r_3)$

$= \min(1, 0.67, 0.125) = 0.67$

$x'_1 = x_1 + \Delta x \cdot u_1 = 30 + 30\,(0.25) = 37.5$

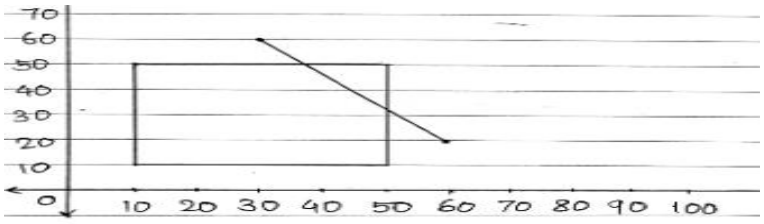
$y'_1 = y_1 + \Delta y \cdot u_1 = 60 + (-40)(0.25) = 50$

$x'_2 = x_2 + \Delta x\, u_2 = 30 + (30)(0.67) = 50$.

$y'_2 = y_2 + \Delta y\, u_2 = 60 + (-40)(0.67) = 33.2$

$\therefore$ Original line Segment : A (30,60) & B (60,20)

$\therefore$ Visible line Segment : A'(37.5, 50) & B' (50, 33.2)



Q6) Imagine you are designing a graphic user interface (GUI) for a vector-based drawing application. One of the features you want to implement is the ability for users to create smooth, free-form curves. To achieve this, you decide to use a Bezier curve of order 3. So, explain the concept of a Bezier curve of order 3 and how it is represented mathematically. Also write important properties.

Bezier curve is an another approach for the construction of the curve. A Bezier curve is determined by a defining polygon. Bezier curves have a number of properties that make them highly useful and convenient for curve and surface design. They are also easy to implement. Therefore Bezier curves are widely available in various CAD systems and in general graphic packages. In this section we will discuss the cubic Bezier curve. The reason for choosing cubic Bezier curve is that they provide reasonable design flexibility and also avoid the large number of calculations.

Properties of Bezier curve

1. The basis functions are real.
2. Bezier curve always passes through the first and last control points i.e. curve has same end points as the guiding polygon.
3. The degree of the polynomial defining the curve segment is one less than the number of defining polygon point. Therefore, for 4 control points, the degree of the polynomial is three, i.e. cubic polynomial.
4. The curve generally follows the shape of the defining polygon.
5. The direction of the tangent vector at the end points is the same as that of the vector determined by first and last segments.
6. The curve lies entirely within the convex hull formed by four control points.
7. The convex hull property for a Bezier curve ensures that the polynomial smoothly follows the control points.
8. The curve exhibits the variation diminishing property. This means that the curve does not oscillate about any straight line more often than the defining polygon.
9. The curve is invariant under an affine transformation.

In cubic Bezier curve four control points are used to specify complete curve. Unlike the B-spline curve, we do not add intermediate points and smoothly extend Bezier curve, but we pick four more points and construct a second curve which can be attached to the first.

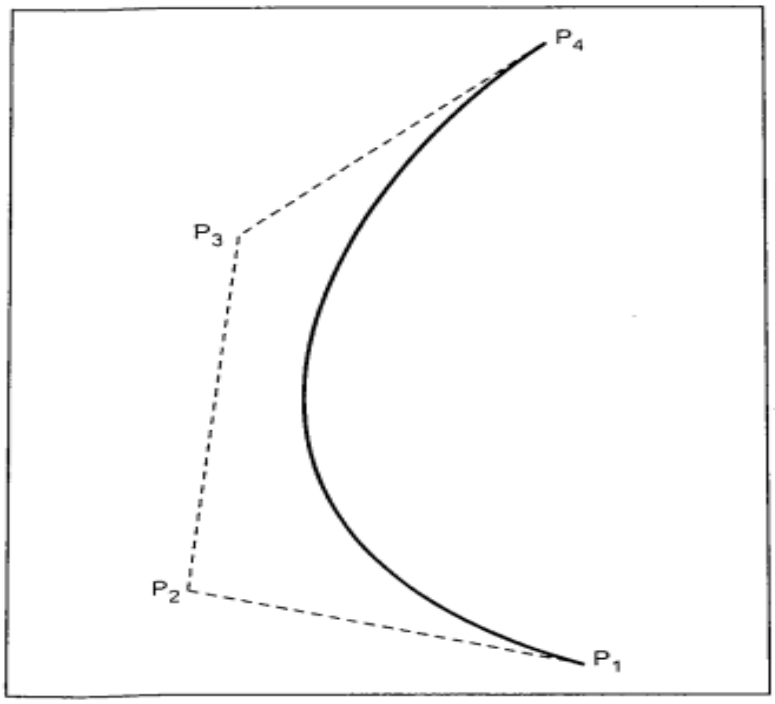


Fig. 9.6 A cubic Bezier spline

The second curve can be attached to the first curve smoothly by selecting appropriate control points.

Fig. 9.6 shows the Bezier curve and its four control points. As shown in the Fig. 9.6, Bezier curve begins at the first control point and ends at the fourth control point. This means that if we want to connect two Bezier curves, we have to make the first control point of the second Bezier curve match the last control point of the first curve. We can also observe that at the start of the curve, the curve is tangent to the line connecting first and second control points. Similarly at the end of curve, the curve is tangent to the line connecting the third and fourth control point. This means that, to join two Bezier curves smoothly we have to place the third and the fourth control points of the first curve on the same line specified by the first and the second control points of the second curve.

curve on the same line specified by the first and the second control points of the second curve.

The Bezier matrix for periodic cubic polynomial is

$$M_B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\therefore \quad P(u) = U \cdot M_B \cdot G_B$$

where $$G_B = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$

and the product $P(u) = U.M_B.G_B$ is

$$P(u) = (1-u)^3 P_1 + 3u(1-u)^2 P_2 + 3u^2(1-u) P_3 + u^3 P_4$$

Q7) Write a scaling matrix with scaling factors 1, 2 and 3 in x, y and z directions, respectively. Apply the transformation matrix on a unit cube situated at the origin.

Q8) Imagine you are a mechanical engineer working on the design and analysis of a complex engine assembly for a cutting-edge electric vehicle. The engine components are intricate, with many parts that need to be accurately positioned and assessed for interference during the design phase. Ensuring that these components fit together perfectly and do not collide is crucial to the engine's efficiency and safety.

Explore the Z-buffer method and provide a step-by-step illustration how the Z-buffer algorithm helps in detecting and resolving depth conflicts between the various engine components, ensuring they are assembled without interference.

Ans -

One of the simplest and commonly used image space approach to eliminate hidden surfaces is the **Z-buffer** or **depth buffer** algorithm. It is developed by Catmull. This algorithm compares surface depths at each pixel position on the projection plane. The surface depth is measured from the view plane along the z axis of a viewing system. When object description is converted to projection coordinates (x, y, z), each pixel position on the view plane is specified by x and y coordinate, and z value gives the depth information. Thus object depths can be compared by comparing the z- values.

The Z-buffer algorithm is usually implemented in the normalized coordinates, so that z values range from 0 at the back clipping plane to 1 at the front clipping plane. The implementation requires another buffer memory called Z-buffer along with the frame buffer memory required for raster display devices. A Z-buffer is used to store depth values for each

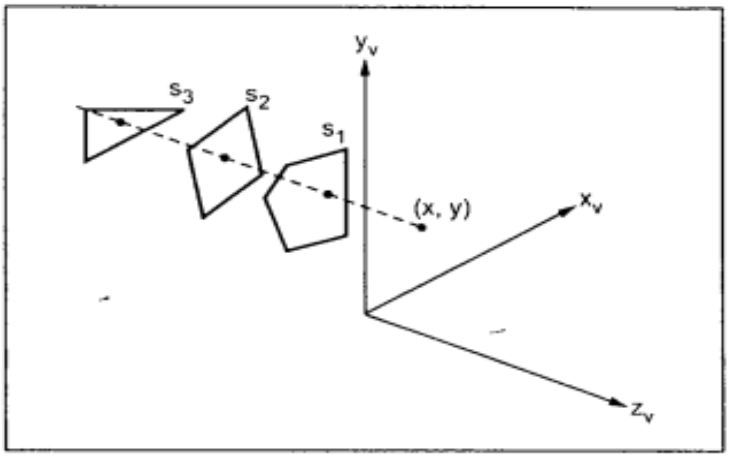


Fig. 8.10

(x, y) position as surfaces are processed, and the frame buffer stores the intensity values for each position. At the beginning Z-buffer is initialized to zero, representing the z-value at the back clipping plane, and the frame buffer is initialized to the background colour. Each surface listed in the display file is then processed, one scan line at a time, calculating the depth (z-value) at each (x, y) pixel position. The calculated depth value is compared to the value previously stored in the Z-buffer at that position.

If the calculated depth values is greater than the value stored in the Z-buffer, the new depth value is stored, and the surface intensity at that position is determined and placed in the same xy location in the frame buffer.

For example, in Fig. 8.10 among three surfaces, surface S_1 has the smallest depth at view position (x, y) and hence highest z value. So it is visible at that position.

Z-buffer Algorithm

1. Initialize the Z-buffer and frame buffer so that for all buffer positions

   Z-buffer (x, y) = 0 and frame-buffer (x, y) = $I_{background}$
2. During scan conversion process, for each position on each polygon surface, compare depth values to previously stored values in the depth buffer to determine visibility.

   Calculate z-value for each (x, y) position on the polygon

   If z > Z-buffer (x, y), then set

   Z-buffer (x, y) = z, frame-buffer (x, y) = $I_{surface}$ (x, y)
3. Stop

Note that, $I_{background}$ is the value for the background intensity, and $I_{surface}$ is the projected intensity value for the surface at pixel position (x, y). After processing of all surfaces, the Z-buffer contains depth values for the visible surfaces and the frame buffer contains the corresponding intensity values for those surfaces.

To calculate z-values, the plane equation

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

is used where (x, y, z) is any point on the plane, and the coefficient A, B, C and D are constants describing the spatial properties of the plane. (Refer Appendix A for details)

Therefore, we can write

$$z = \frac{-Ax - By - D}{C}$$

Note, if at (x, y) the above equation evaluates to z_1, then at (x + Δx, y) the value of z, is

$$z_1 = \frac{A}{C}(\Delta x)$$

Only one subtraction is needed to calculate z(x + 1, y), given z(x, y), since the quotient A/C is constant and $\Delta x = 1$. A similar incremental calculation can be performed to determine the first value of z on the next scan line, decrementing by B/C for each Δy.

Advantages

1. It is easy to implement.
2. It can be implemented in hardware to overcome the speed problem.
3. Since the algorithm processes objects one at a time, the total number of polygons in a picture can be arbitrarily large.

Disadvantages

1. It requires an additional buffer and hence the large memory.
2. It is a time consuming process as it requires comparison for each pixel instead of for the entire polygon.

Q9) A Point has coordinates P(2, 3, 4) in x, y, z-direction. The Rotation angle is 90 degrees. Apply the rotation in x, y, z direction, and find out the new coordinates of the point?

For x-axis :
Let the new coordinates (x_1, y_1, z_1) then,

$x_1 = x_0 = 2$

$y_1 = y_0 \cos\theta - z_0 \sin\theta$
$= 3\cos 90° - 4\sin 90°$
$y_1 = -4$

$z_1 = y_0 \sin\theta + z_0 \cos\theta$
$= 3\sin 90 + 4\cos 90°$
$z_1 = 3$.

New coordinates $\Longrightarrow$ $P_1(2, -4, -3)$

For y-axis.
Let the new coordinates (x_2, y_2, z_2) then,

$x_2 = z_0 \sin\theta + x_0 \cos\theta$
$= 4\sin 90 + 2\cos 90$
$x_2 = 4$

$z_2 = z_0 \cos\theta - x_0 \sin\theta$
$= 4\cos 90° - 2\sin 90°$
$z_2 = -2$

$y_2 = y_0 = 3$

New coordinates $\longrightarrow$ $P_2(4, 3, -2)$

For z axis,
Let the new coordinates be (x_3, y_3, z_3) then,

$x_3 = x_0 \cos\theta - y_0 \sin\theta$
$= 2\cos 90 - 3\sin 90$
$x_3 = -3$

$y_3 = x_0 \sin\theta + y_0 \sin\theta$
$= 2\sin 90° + 3\sin 90°$
$y_3 = 5$

$z_3 = z_0 = 4$

New coordinates $\Longrightarrow$ $P_3(-3, 5, 4)$

Q10) Write the comparison between Conventional/Traditional and Computer based animation techniques.

Traditional Animation :-
(1) Traditional 2-D cel animation and stop-motion animation both fall under category of tradition animation.
(2) Cel animation involves hand drawing - inking, painting thousands of frames on clear cells that display in rapid sequence.
(3) For large scale projects, the amount of time, labour and equipement requires is more.

Computer Animation :-
(1) Computer animation simplifies animation toolkit, one need system to run 2D - 3D softwaare applications for animation.
(2) It is much less labour - intensive and much cheaper. It comes with greater margin of error because one can undo mistakes they made.
(3) 3D computer animation employs hybrid work-flow following traditional timelines adopted working it in virtual space.