key points use for definition:

1. Computer graphics: The field of computer science that deals with the creation, manipulation, and display of images and videos using computers.
2. Graphics Processing Unit (GPU): A specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display.
3. Image processing: The process of transforming an image into a form more suitable for computer-based analysis and manipulation.
4. 3D modelling: The process of creating a virtual representation of a three-dimensional object or scene using specialized software.
5. Ray tracing: A method for rendering images by tracing the path of light as pixels in an image.
6. Virtual reality (VR): A simulated experience that can be similar to or completely different from the real world.
7. Augmented reality (AR): A live direct or indirect view of a physical, real-world environment where elements are augmented by computer-generated sensory input.
8. Computer vision: The field of computer science that deals with how computers can gain high-level understanding from digital images or videos.
9. Machine learning: A subfield of artificial intelligence that focuses on the development of algorithms that can learn from and make predictions on data.

Input mode refers to the general category of input device being used to interact with a computer-generated image, video, or virtual environment, while input method refers to the specific way in which the input device is being used.

For example, the input mode can be a keyboard and mouse, while the input method could be clicking or dragging with the mouse. Another input mode could be a touch screen, with the input method being tapping or swiping with a finger.

So, input modes are the broad categories of input devices (e.g. keyboard and mouse, touch screen, etc.), while input methods are the specific actions taken using these input devices (e.g. clicking, tapping, swiping, etc.).

input modes and methods refer to the ways in which a user can interact with a computer-generated image, video, or virtual environment. Here are some common input modes and methods used in visual computing:

1. Keyboard and Mouse: A traditional input method where the user inputs commands through a keyboard and uses a mouse to point and click.
2. Touchscreen: A type of display screen that can detect the presence and location of touch within the display. The user can interact with the screen by tapping, swiping, or pinching.
3. Joystick: A device that consists of a handheld stick that pivots on a base and is used to control movement in computer games and simulations.
4. Gamepad: A type of input device similar to a joystick, but with additional buttons for specific actions in computer games and simulations.
5. VR controllers: Devices specifically designed for use in virtual reality environments, allowing users to interact with the virtual environment in a more immersive way.
6. Motion sensing: A type of input method that uses sensors to detect movement and translate it into inputs for the computer. This can include technologies like the Kinect for Microsoft Xbox, or the PlayStation Move for the PlayStation.
7. Voice recognition: A type of input method that allows users to input commands or information using spoken words.
8. Pen and tablet: A type of input device that consists of a stylus and a digitizing tablet, allowing the user to draw or write directly on the tablet surface.

These are just a few examples of the various input modes and methods used in visual computing. The choice of input method depends on the specific application and the user's preferences.

Polygon Filling Algorithm

Polygon filling is a process in computer graphics that involves colouring the interior of a 2D polygon. The polygon is defined by a set of vertices and edges that connect those vertices.

The basic idea behind polygon filling is to determine which pixels within the boundary of the polygon should be colored. There are several algorithms used to perform this task, including scanline fill, boundary fill, and flood fill.

The scanline fill algorithm works by scanning the image from top to bottom, keeping track of the intersections of the scanline with the edges of the polygon. Pixels between these intersections are then filled.

The boundary fill algorithm works by starting from a seed pixel within the polygon and following the boundary of the polygon, colouring pixels as it goes.

The flood fill algorithm works by starting from a seed pixel within the polygon and coloring all connected pixels that have the same color.

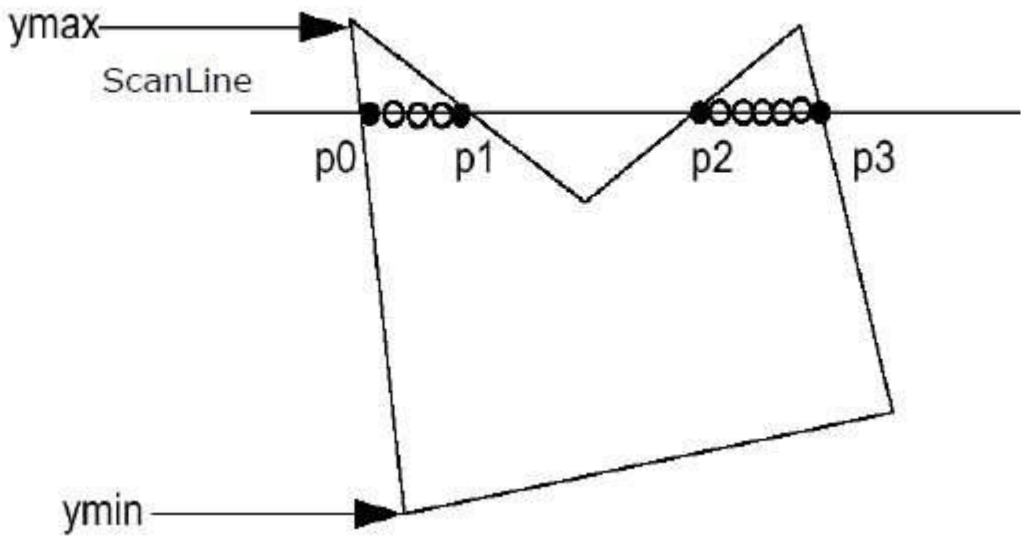
Polygon filling is used in a variety of computer graphics applications, including game development, 3D modeling, and image editing. It is an important step in the process of rendering 2D images and 3D objects on a computer screen.

Polygon is an ordered list of vertices as shown in the following figure. For filling polygons with particular colors, you need to determine the pixels falling on the border of the polygon and those which fall inside the polygon. In this chapter, we will see how we can fill polygons using different techniques.

Scan Line Algorithm

This algorithm works by intersecting scanline with polygon edges and fills the polygon between pairs of intersections. The following steps depict how this algorithm works.

Step 1 − Find out the Ymin and Ymax from the given polygon.



Step 2 − ScanLine intersects with each edge of the polygon from Ymin to Ymax. Name each intersection point of the polygon. As per the figure shown above, they are named as p0, p1, p2, p3.

Step 3 − Sort the intersection point in the increasing order of X coordinate i.e. p0,p1, p1,p2 and p2,p3.

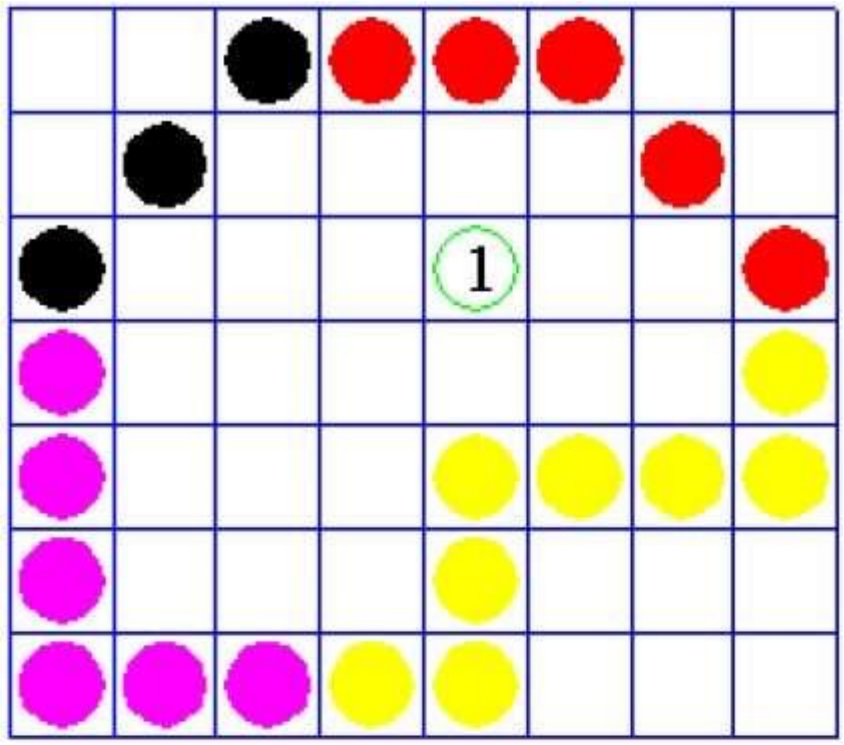
Step 4 − Fill all those pair of coordinates that are inside polygons and ignore the alternate pairs.

Flood Fill Algorithm

Sometimes we come across an object where we want to fill the area and its boundary with different colors. We can paint such objects with a specified interior color instead of searching for particular boundary color as in boundary filling algorithm.

Instead of relying on the boundary of the object, it relies on the fill color. In other words, it replaces the interior color of the object with the fill color. When no more pixels of the original interior color exist, the algorithm is completed.

Once again, this algorithm relies on the Four-connect or Eight-connect method of filling in the pixels. But instead of looking for the boundary color, it is looking for all adjacent pixels that are a part of the interior.



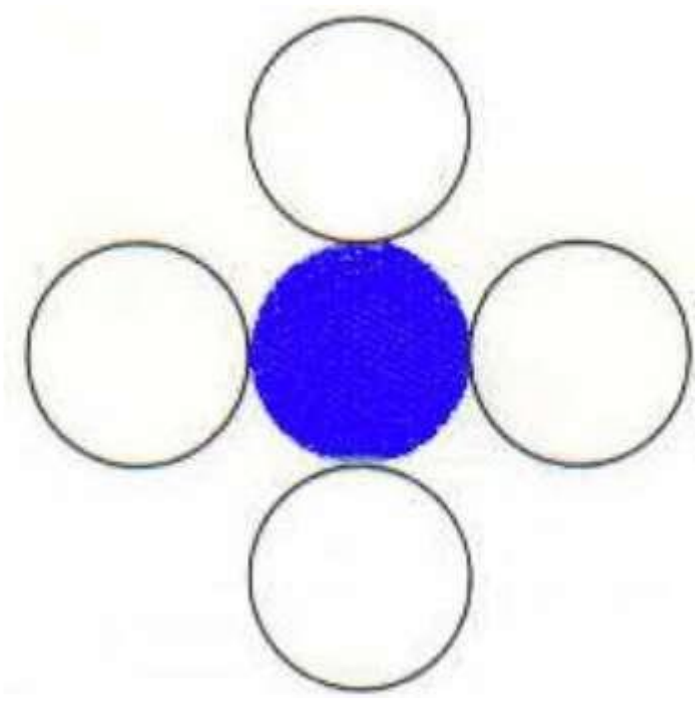
Boundary Fill Algorithm

The boundary fill algorithm works as its name. This algorithm picks a point inside an object and starts to fill until it hits the boundary of the object. The color of the boundary and the color that we fill should be different for this algorithm to work.

In this algorithm, we assume that color of the boundary is same for the entire object. The boundary fill algorithm can be implemented by 4-connected pixels or 8-connected pixels.

4-Connected Polygon

In this technique 4-connected pixels are used as shown in the figure. We are putting the pixels above, below, to the right, and to the left side of the current pixels and this process will continue until we find a boundary with different color.



Algorithm

Step 1 − Initialize the value of seed point seedx,seedy, fcolor and dcol.

Step 2 − Define the boundary values of the polygon.

Step 3 − Check if the current seed point is of default color, then repeat the steps 4 and 5 till the boundary pixels reached.

```
If getpixel(x, y) = dcol then repeat step 4 and 5
```

Step 4 − Change the default color with the fill color at the seed point.

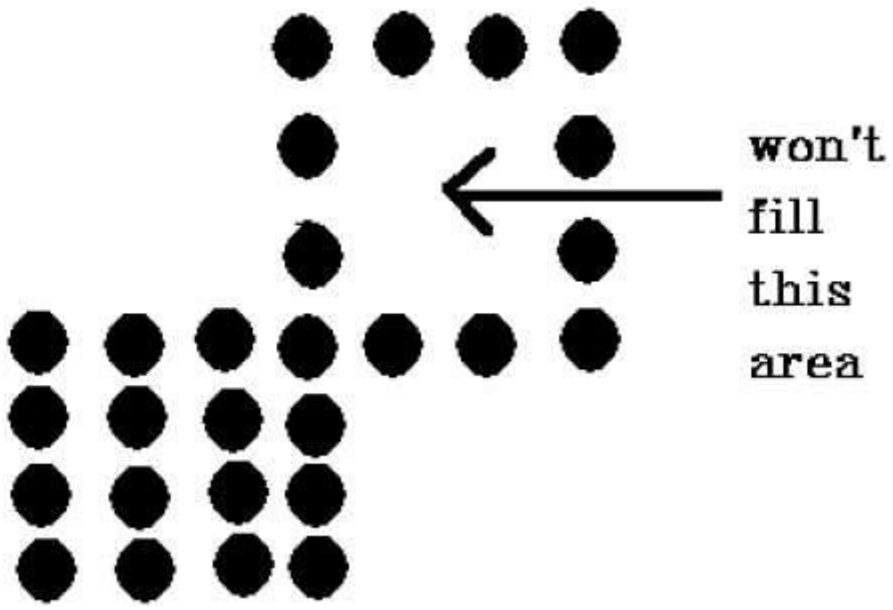
```
setPixel(seedx, seedy, fcol)
```

Step 5 − Recursively follow the procedure with four neighborhood points.

```
FloodFill (seedx – 1, seedy, fcol, dcol)
FloodFill (seedx + 1, seedy, fcol, dcol)
FloodFill (seedx, seedy - 1, fcol, dcol)
FloodFill (seedx – 1, seedy + 1, fcol, dcol)
```

Step 6 − Exit

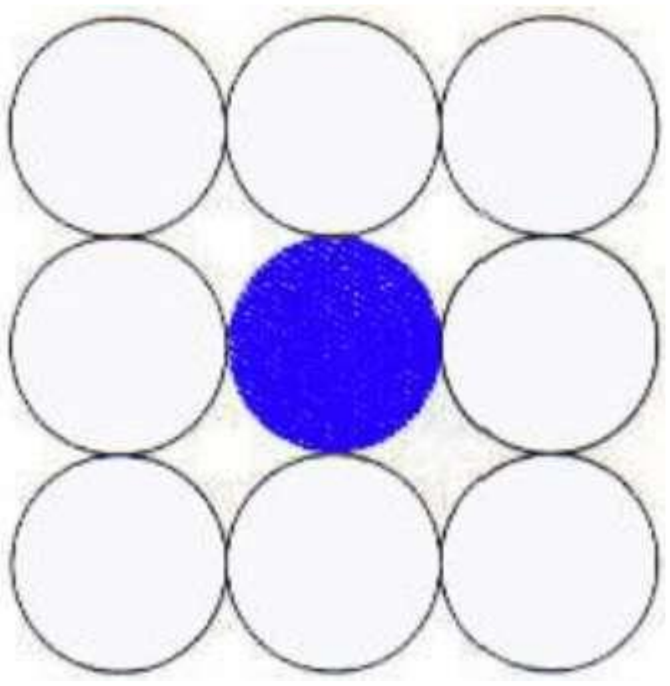
There is a problem with this technique. Consider the case as shown below where we tried to fill the entire region. Here, the image is filled only partially. In such cases, 4-connected pixels technique cannot be used.



8-Connected Polygon

In this technique 8-connected pixels are used as shown in the figure. We are putting pixels above, below, right and left side of the current pixels as we were doing in 4-connected technique.

In addition to this, we are also putting pixels in diagonals so that entire area of the current pixel is covered. This process will continue until we find a boundary with different color.



Algorithm

Step 1 − Initialize the value of seed point seedx,seedy, fcolor and dcol.

Step 2 − Define the boundary values of the polygon.

Step 3 − Check if the current seed point is of default color then repeat the steps 4 and 5 till the boundary pixels reached

If getpixel(x,y) = dcol then repeat step 4 and 5

Step 4 − Change the default color with the fill color at the seed point.

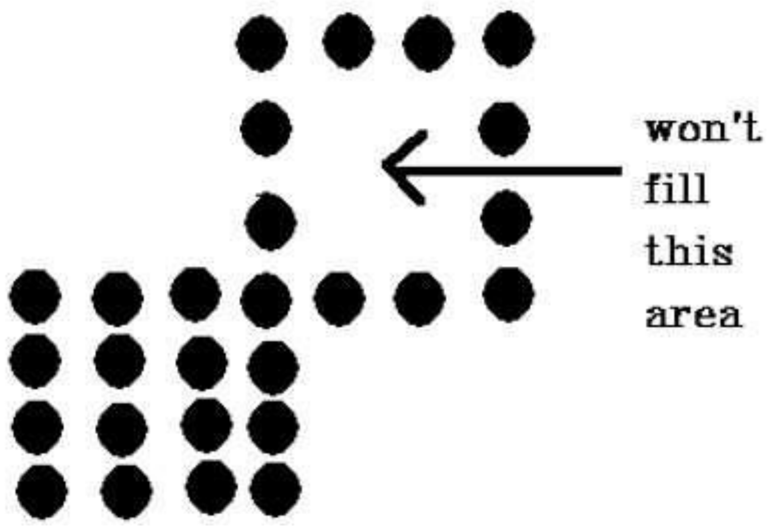
setPixel(seedx, seedy, fcol)

Step 5 − Recursively follow the procedure with four neighbourhood points

```
FloodFill (seedx – 1, seedy, fcol, dcol)
FloodFill (seedx + 1, seedy, fcol, dcol)
FloodFill (seedx, seedy - 1, fcol, dcol)
FloodFill (seedx, seedy + 1, fcol, dcol)
FloodFill (seedx – 1, seedy + 1, fcol, dcol)
FloodFill (seedx + 1, seedy + 1, fcol, dcol)
FloodFill (seedx + 1, seedy - 1, fcol, dcol)
FloodFill (seedx – 1, seedy - 1, fcol, dcol)
```

Step 6 − Exit

The 4-connected pixel technique failed to fill the area as marked in the following figure which won't happen with the 8-connected technique.



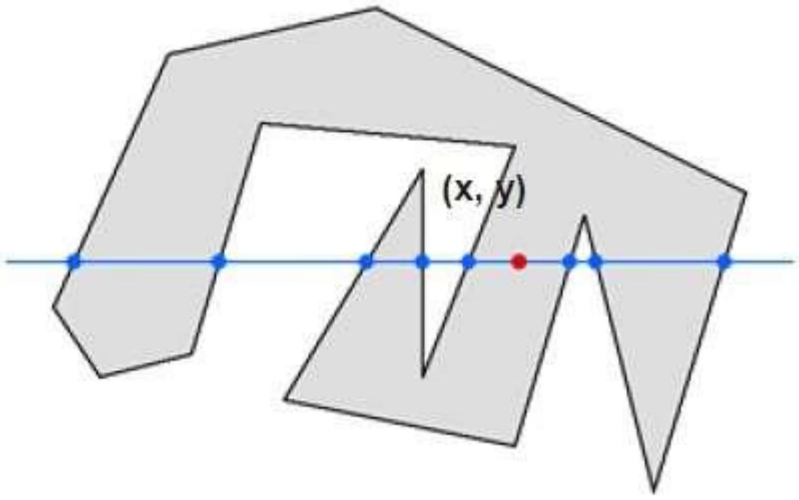
Inside-outside Test

This method is also known as **counting number method**. While filling an object, we often need to identify whether particular point is inside the object or outside it. There are two methods by which we can identify whether particular point is inside an object or outside.

- Odd-Even Rule
- Nonzero winding number rule

Odd-Even Rule

In this technique, we will count the edge crossing along the line from any point x,y to infinity. If the number of interactions is odd, then the point x,y is an interior point; and if the number of interactions is even, then the point x,y is an exterior point. The following example depicts this concept.

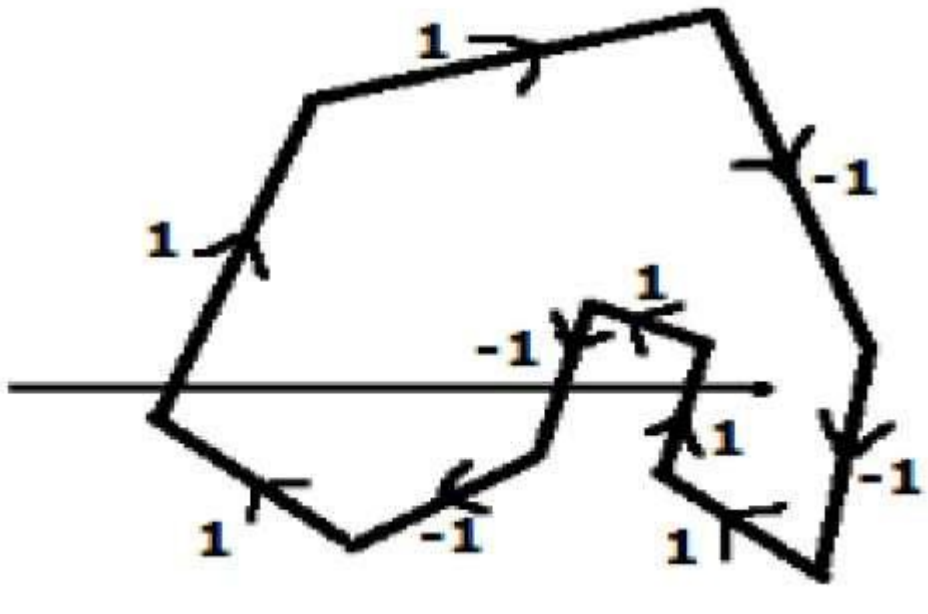


From the above figure, we can see that from the point x,y�,�, the number of interactions point on the left side is 5 and on the right side is 3. From both ends, the number of interaction points is odd, so the point is considered within the object.

Non-zero Winding Number Rule

This method is also used with the simple polygons to test the given point is interior or not. It can be simply understood with the help of a pin and a rubber band. Fix up the pin on one of the edges of the polygon and tie-up the rubber band in it and then stretch the rubber band along the edges of the polygon.

When all the edges of the polygon are covered by the rubber band, check out the pin which has been fixed up at the point to be test. If we find at least one wind at the point consider it within the polygon, else we can say that the point is not inside the polygon.



In another alternative method, give directions to all the edges of the polygon. Draw a scan line from the point to be test towards the left most of X direction.

- Give the value 1 to all the edges which are going to upward direction and all other -1 as direction values.

- Check the edge direction values from which the scan line is passing and sum up them.
- If the total sum of this direction value is non-zero, then this point to be tested is an **interior point,** otherwise it is an **exterior point**.
- In the above figure, we sum up the direction values from which the scan line is passing then the total is 1 – 1 + 1 = 1; which is non-zero. So the point is said to be an interior point.

Degree of freedom (DoF): refers to the number of independent variables or parameters that can be controlled to manipulate an object in a virtual environment. It describes the range of movement and freedom of motion that an object has in the virtual space.

- For example, in 3D graphics, an object with six degrees of freedom would allow the user to rotate and translate the object in three dimensions (x, y, z) with full control. This would allow the object to be positioned and rotated in any direction within the virtual environment.

- In animation and simulations, the degree of freedom is an important factor in determining the realism and versatility of an object. For example, in a character animation, the number of degrees of freedom in the joints and limbs of the character would determine how realistic and fluid the character's movements can be.

- In virtual reality, the degree of freedom of the user's movements and inputs is critical for creating an immersive experience. A high degree of freedom allows for a wide range of movements and actions, making the experience more natural and intuitive for the user.

- Overall, degree of freedom plays a crucial role in visual computing by determining the range of movement and freedom of motion for objects in a virtual environment, and therefore impacting the realism, versatility, and immersion of the visual computing experience.
- A 3D object's polygon surface refers to the outer surface of the object that is made up of a series of polygons. Polygons are flat, 2D shapes with straight sides and angles, and when these polygons are arranged and connected, they form the surface of a 3D object.

- In computer graphics, 3D objects are often modeled using a technique called polygonal modeling, where the surface of the object is represented by a large number of small, flat polygons. These polygons are then rendered and shaded to create the illusion of a solid 3D object.

- The number of polygons used to model a 3D object affects its level of detail and smoothness. More polygons result in a higher level of detail and smoother surfaces, while fewer polygons result in a lower level of detail and rougher surfaces.

- In addition to determining the level of detail and smoothness, the size and shape of the polygons on a 3D object's surface can also impact its performance and visual quality in real-time rendering environments, such as video games.

- Overall, the polygon surface of a 3D object is a critical aspect in computer graphics, as it determines the visual quality, level of detail, and performance of the object in the virtual environment.

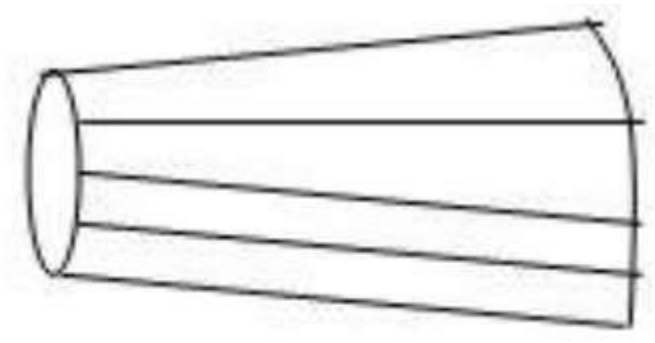
Polygon Surfaces

Objects are represented as a collection of surfaces. 3D object representation is divided into two categories.

- **Boundary Representations -**It describes a 3D object as a set of surfaces that separates the object interior from the environment.
- **Space–partitioning representations** − It is used to describe interior properties, by partitioning the spatial region containing an object into a set of small, non-overlapping.

The most commonly used boundary representation for a 3D graphics object is a set of surface polygons that enclose the object interior. Many graphics system use this method. Set of polygons are stored for object description. This simplifies and speeds up the surface rendering and display of object since all surfaces can be described with linear equations.

The polygon surfaces are common in design and solid-modeling applications, since their **wireframe display** can be done quickly to give general indication of surface structure. Then realistic scenes are produced by interpolating shading patterns across polygon surface to illuminate.

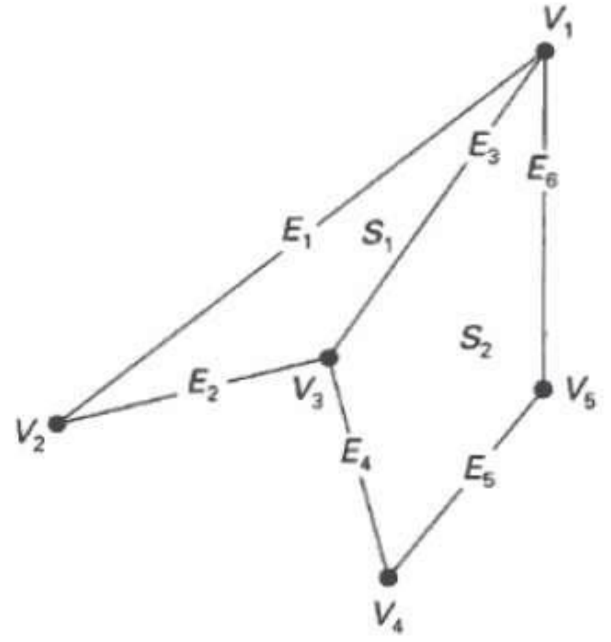


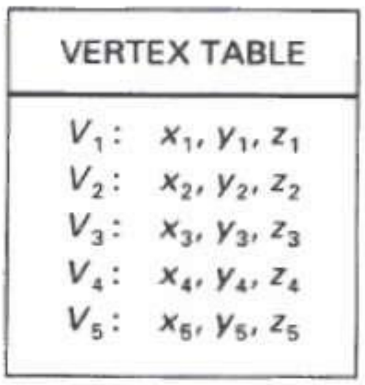
A 3D object represented by polygons

Polygon Tables

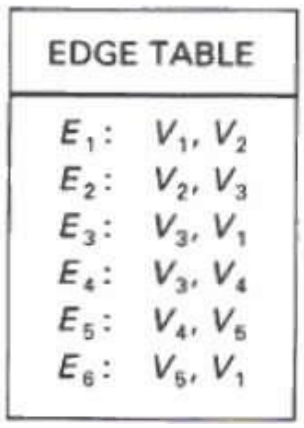
In this method, the surface is specified by the set of vertex coordinates and associated attributes. As shown in the following figure, there are five vertices, from v_1 to v_5.

- Each vertex stores x, y, and z coordinate information which is represented in the table as v_1: x_1, y_1, z_1.
- The Edge table is used to store the edge information of polygon. In the following figure, edge E_1 lies between vertex v_1 and v_2 which is represented in the table as E_1: v_1, v_2.
- Polygon surface table stores the number of surfaces present in the polygon. From the following figure, surface S_1 is covered by edges E_1, E_2 and E_3 which can be represented in the polygon surface table as S_1: E_1, E_2, and E_3.

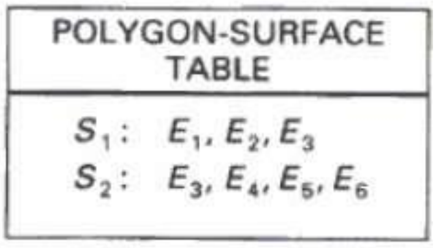




VERTEX TABLE	
V_1:	x_1, y_1, z_1
V_2:	x_2, y_2, z_2
V_3:	x_3, y_3, z_3
V_4:	x_4, y_4, z_4
V_5:	x_5, y_5, z_5



EDGE TABLE	
E_1:	V_1, V_2
E_2:	V_2, V_3
E_3:	V_3, V_1
E_4:	V_3, V_4
E_5:	V_4, V_5
E_6:	V_5, V_1



POLYGON-SURFACE TABLE	
S_1:	E_1, E_2, E_3
S_2:	E_3, E_4, E_5, E_6

Plane Equations

The equation for plane surface can be expressed as −

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

Where x,y,z is any point on the plane, and the coefficients A, B, C, and D are constants describing the spatial properties of the plane. We can obtain the values of A, B, C, and D by solving a set of three plane equations using the coordinate values for three non collinear points in the plane. Let us assume that three vertices of the plane are (x_1, y_1, z_1), (x_2, y_2, z_2) and (x_3, y_3, z_3).

Let us solve the following simultaneous equations for ratios A/D, B/D, and C/D. You get the values of A, B, C, and D.

$$A/D \; x_1 + B/D \; y_1 + C/D \; z_1 = -1$$

$$A/D \; x_2 + B/D \; y_2 + C/D \; z_2 = -1$$

$$A/D \; x_3 + B/D \; y_3 + C/D \; z_3 = -1$$

To obtain the above equations in determinant form, apply Cramer's rule to the above equations.

$$A = \begin{bmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{bmatrix} B = \begin{bmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{bmatrix} C = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} D = - \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$$

For any point x, y, z with parameters A, B, C, and D, we can say that −

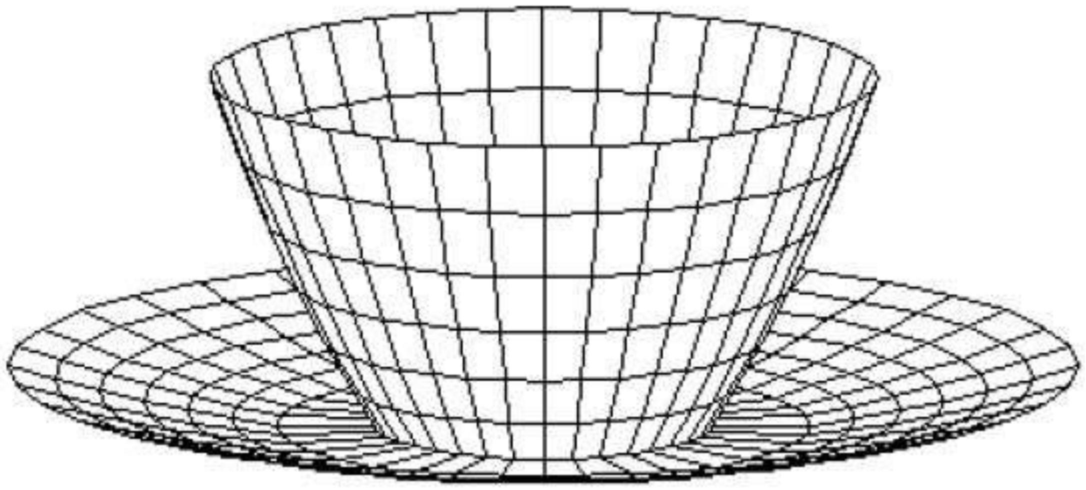
- $Ax + By + Cz + D \neq 0$ means the point is not on the plane.
- $Ax + By + Cz + D < 0$ means the point is inside the surface.
- $Ax + By + Cz + D > 0$ means the point is outside the surface.

Polygon Meshes

3D surfaces and solids can be approximated by a set of polygonal and line elements. Such surfaces are called **polygonal meshes**. In polygon mesh, each edge is shared by at most two polygons. The set of polygons or faces, together form the "skin" of the object.

This method can be used to represent a broad class of solids/surfaces in graphics. 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



Advantages

- 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.

Disadvantages

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

Painter’s Algorithm in Computer Graphics

Painter’s algorithm is the algorithm which is introduced by **Hewells** in **1972**.

The techniques used by these algorithms are **image space** and **object space**.

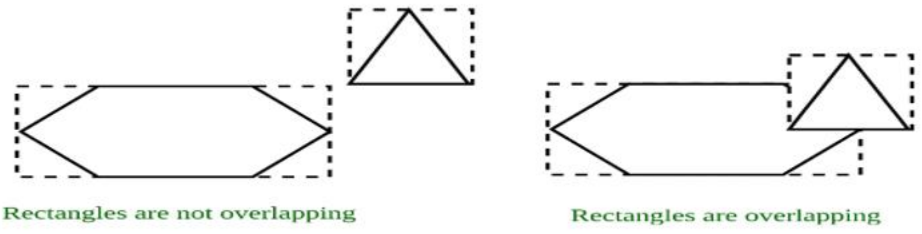
The name of this algorithm is Painter’s because it’s working is like a painter who creating an **oil painting**. Just like an artist paint, he start his painting with an empty canvas, the first thing the artist will do is to create a **background layer** for the painting, after this layer he start creating **another layers** of objects **one-by-one**. In this way he completes his painting, by covering the previous layer partially or fully according to the requirement of the painting.

This algorithm is basically used to paint the **polygons** in the view plane by considering their distance from the viewer. The polygons which are at more distance from the viewer are painted first. After that the nearer polygons are started painted on or over more distant polygons according to the requirement.

In this algorithm the polygons or surfaces in the scene are firstly scanned or then painted in the frame buffer in the **decreasing distance** from view point

of the viewer starting with the polygons of **maximum depth** or we can say minimum z-value.
Firstly the **depth sort** is performed in which the polygons are listed according to their **visibility order** or depth priority.
As this algorithm uses the concept of depth priority so it is also called as **depth priority algorithm** or **priority algorithm**.
The frame buffer is painted with the background color. After that the polygon which is farthest enter to the frame buffer. For this, the pixel information will get changed i.e. information of the background which has the farthest polygon get replaced with that of the background. This is going to be repeatedly changed as we move from one polygon to the other and end up with the nearest polygon.

Usually **comparisons** are performed whenever the polygons are going to overlap each other. The most common method used for the comparison is called as **mini-max method**. For this purpose, the rectangles are drawn around the polygons such that the rectangles exactly fit the polygons. Then the rectangles are going to check to see whether they overlap each other or not. If the rectangles are observed as they do not overlap then we consider that the surfaces are also not overlap. If the rectangles are overlapped then the surfaces are also overlapped which is as shown in the following figure:



To find out which rectangle is to be overlap, we need to find the minimum and maximum x and y values to the rectangles which are going to test for overlapping. If the minimum value of y of one of the rectangles is larger than the maximum y value of the other rectangle then they are not going to be overlapped and as the rectangles are not overlapped then the surfaces are also not overlapped.

The same test is to be performed for the x-coordinates.

If the surfaces are overlapping, we do not know which surface should be present on the top of the other. To find out which surface is to be present on the top the **principle of mini-max** is used on the depth values of both the overlapping surfaces.
Example:

Distant Background is painted first

Objects closer than the background are painted

Finally, Nearest Objects are painted

Algorithm for Painter’s method

The following are the steps of this algorithm:

1. Sorting of the various surfaces which is on the basis of their decreasing depth or we can say the largest value of z.
2. Now scanning to convert the various surfaces which is in the order starting with the surface which has greatest depth.
3. Comparing is to be done on the basis of various overlapping surfaces so that the user will determine which surface is to be kept visible.
4. In the refresh buffer enter the intensity value for the determined surface i.e. the surface which is determined to be visible.
5. The above process is going to be repeat for all the available surfaces.

Illumination models are mathematical representations of how light interacts with objects and surfaces in a 3D environment. These models are used in computer graphics to simulate the behaviour of light and create realistic-looking images and animations.

An illumination model typically includes definitions of the following components:

1. Light sources: The types and number of light sources in the environment, such as point lights, directional lights, and spot lights.

2. Material properties: The properties of objects and surfaces in the environment, such as color, reflectivity, and transparency.
3. Light interaction: The way light interacts with objects and surfaces, including reflection, refraction, and absorption.
4. Shadows: The calculation of shadows and how they are cast by objects and light sources in the environment.
5. Global illumination: The calculation of indirect light in the environment, such as ambient light and light that bounces off of surfaces.

There are several different types of illumination models, each with their own strengths and weaknesses. Some of the most commonly used models include the Phong model, the Blinn-Phong model, and the Cook-Torrance model.

The choice of illumination model depends on the desired level of realism and computational resources available. More complex models can result in higher quality images, but also require more processing power and time to render.

Overall, the illumination model is a critical aspect in computer graphics and plays a major role in creating realistic-looking images and animations in a virtual environment.

Basic Illumination Models

Illumination model, also known as Shading model or Lightning model, is used to calculate the intensity of light that is reflected at a given point on surface. There are three factors on which lightning effect depends on:

1. **Light Source:**
   Light source is the light emitting source. There are three types of light sources:
   1. **Point Sources –** The source that emit rays in all directions (A bulb in a room).
   2. **Parallel Sources –** Can be considered as a point source which is far from the surface (The sun).
   3. **Distributed Sources –** Rays originate from a finite area (A tubelight).

   Their position, electromagnetic spectrum and shape determine the lightning effect.

2. **Surface :**
   When light falls on a surface part of it is reflected and part of it is absorbed. Now the surface structure decides the amount of reflection and absorption of light. The position of the surface and positions of all the nearby surfaces also determine the lightning effect.
3. **Observer :**
   The observer's position and sensor spectrum sensitivities also affect the lightning effect.

1. Ambient Illumination :
Assume you are standing on a road, facing a building with glass exterior and sun rays are falling on that building reflecting back from it and the falling on the object under observation. This would be **Ambient Illumination**. In simple words, Ambient Illumination is the one where source of light is indirect.
The reflected intensity I_{amb} of any point on the surface is:

2. Diffuse Reflection:
Diffuse reflection occurs on the surfaces which are rough or grainy. In this reflection the brightness of a point depends upon the angle made by the light source and the surface.
The reflected intensity I_{diff} of a point on the surface is:

3. Specular Reflection :
When light falls on any shiny or glossy surface most of it is reflected back, such reflection is known as Specular Reflection.
Phong Model is an empirical model for Specular Reflection which provides us with the formula for calculation the reflected intensity I_{spec}:

Halftone and dithering: techniques are methods used in digital printing and graphics to simulate the appearance of continuous tone images, such as photographs, on devices with limited color depth, such as printers or computer monitors.

Halftoning refers to the process of converting a continuous-tone image into a series of dots or patterns of different sizes and shapes, which can be printed or displayed on a device with limited color depth. The size and arrangement of the dots determines the perceived intensity of the color in the final image.

Dithering refers to a specific type of halftoning technique that uses a pattern of pixels of different colors to approximate a desired color. In dithering, the colors of adjacent pixels are chosen to create the illusion of a smooth gradient, even though the device is only capable of displaying a limited number of colors.

Both halftone and dithering techniques are used to overcome the limitations of digital printing and display devices, which are not able to produce the full range of colors and shades found in a continuous-tone image. These techniques allow for a wider range of colors and shades to be represented in the final image, resulting in a more visually appealing and realistic representation of the original image.

Overall, halftone and dithering techniques are important methods used in digital printing and graphics to overcome the limitations of limited color depth devices and create a more visually appealing representation of images.
The anatomy of a digital camera can be divided into several main components:

1. Lens: The lens is responsible for collecting and focusing light onto the image sensor. The type and quality of the lens determines the sharpness, clarity, and perspective of the images captured by the camera.
2. Image sensor: The image sensor is the component of the camera that captures the light and converts it into digital data. The size and type of the image sensor determines the image quality, resolution, and sensitivity to light.
3. Processor: The processor is the central processing unit of the camera that controls the functions of the camera and processes the data captured by the image sensor into a final image.
4. Storage: Digital cameras have built-in storage, usually in the form of memory cards, that store the images captured by the camera.
5. Display: Most digital cameras have a built-in display screen that allows you to view your images and navigate the camera's menu system.
6. Shutter: The shutter is the mechanism that opens and closes to allow light to reach the image sensor. It determines the length of time that the image sensor is exposed to light and is an important factor in controlling the exposure of the image.
7. Flash: Some digital cameras have built-in flash units that provide additional lighting for low-light situations.
8. Battery: Digital cameras are powered by a built-in rechargeable battery that provides the energy needed to operate the camera.
9. Ports: Many digital cameras have ports for connecting to other devices, such as computers or printers, for transferring images and charging the battery.

Overall, these are the main components of a digital camera and each plays a critical role in determining the quality, performance, and functionality of the camera.