Computer Graphics

Date :..2080/02/06..
Page :.....

CHAPTER-1

1. Define computer graphics. Explain the application areas of cimputer graphics.

→ Computer graphics is the discipline of generating images with the aid of computers. It is an art of drawing pictures, lines, charts, etc using computers with the help of programming.

The application areas of computer graphics are :

a. Computer Art :

Computer graphics are used in the field of commercial arts which include animation packages, paint packages. It is used to generate television and advertising commercial. Cartoon drawing, paintings, logo design can also be done.

b. Computer Aided Drawing:

Design of buildings, auto mobile, aircrafts is done with the help of computer aided drawing, this helps in providing minute details to the drawing and producing more accurate and sharp drawings with better specifications.

c. Presentation Graphics :

Example of presentation graphics are bar graphs, pie charts, time charts showing the relationships betweens multiple parameters. Presentation graphics is commonly used to summarize financial, statistical, mathematical, scientific and other types of reports.

Date :.....
Page :.....

d. Entertainment:

Computer graphics are now commonly used in making motion pictures, music videos and television shows. It is also used in making games.

e. Education:

Using computer graphics, many educational models can be created through which more interest can be generated among the students regarding the subject.

f. Visualization:

It is used for visualization of scientists, engineers, medical personnel, business analysts for the study of a large amount of information.

g. Training:

Specialized systems for training like simulators can be used for training the candidates in a way that can be grasped in a short period of time with better understanding. Creation of training modules using computer graphics is simple and very useful.

Date :.....
Page :.....

2. As you are student of computer science (CT), why computer graphics is important in human life. Explain.

→ Computer graphics plays a vital role in various aspects of human life. It has become an integral part of several fields, including entertainment, design, education, simulation, and visualization.

Same as Q.1

3. What is color model in computer graphics? Explain XYZ color model

→ In computer graphics, a color model is a mathematical representation that defines how colors are represented and displayed. It provides a systematic way to describe and manipulate colors.

The XYZ color model, also known as the CIE XYZ color space, is a standardized color model developed by the International commission on Illumination (CIE). It was created to provide a device-independent and perceptually uniform color representation.

The XYZ color model is based on the human visual system's response to different wavelengths of light. It uses three components to represent a color: X, Y, and Z. These components are often referred to as tristimulus values.

The X, Y and Z values in the XYZ color model are derived from a linear transformation of the original RGB color model. The RGB values are first transformed to a linear color space and then converted to XYZ using specific conversion formulas. This conversion takes into account the characteristics of the human eye's perception of color.

Date :.....
Page :.....

In the XYZ color model, the Y component represents the luminance or brightness of the color, while the X and Z components represent the chromaticity or the color's position in the color space. The XYZ color space encompasses all possible colors visible to the human eye.

The XYZ color model is often used as a reference color space in color management systems and color science research. It serves as a foundation for other color models and spaces such as the CIE LAB and CIE LUV color spaces, which are designed to be more perceptually uniform and suitable for color difference calculations.

Overall, the XYZ color model provides a standardized and device-independent representation of colors, enabling accurate color reproduction and consistency across different devices and platforms.

4. Describe the properties of light in brief.

→ The properties of light are:

a. Wave-like Nature: Light exhibits wave-like behavior and can be described in terms of its wavelength, frequency, and amplitude. It propagates through space as an electromagnetic wave, composed of oscillating electric and magnetic fields perpendicular to each other and to the direction of propagation.

Date :.....
Page :.....

b. Speed : Light travels at a constant speed in a vacuum, commonly denotes as "c" and approximately equal to 299,792,458 meters per second (or about 186,282 miles per second). This speed is a fundamental constant of nature.

c. Reflection: when light encounters a boundary between two different materials, it can bounce off the surface. This phenomen is called reflection. The angle of incidence (the angle at which light strikes the surface) is equal to the angle of reflection (the angle at which light is reflected.

d. Refraction : Refraction occurs when light passes from one medium to another, causing a change in direction is due to the change in the speed of light as it enters a different medium. The bending of light during refraction is governed by Snell's law.

e. Transmission: Light can also pass through certain materials without being significantly absorbed or scattered. This phenomenon is known as transmission. Transparent materials allow light to transmit through them, while opaque materials block or absorb light.

f. Absorption : When light interacts with matter, it can be absorbed by the material, converting its energy into internal energy. Different materials have different absorption properties, with some absorbing certain wavelengths more strongly than others.

Date :.....

Page :.....

5. Explain graphics standards and graphical file formats.

→ Graphics standards are specifications and guidelines that define the structure, encoding, and behavior of graphics data. They ensure interoperability and consistency among different software and hardware platforms. Some important graphics standards include:

a. JPEG (Joint photographic Experts Group):

JPEG is a widely used standard for compressing and storing digital images and supports lossy compression, which reduces file size by selectively discarding image data.

b. PNG (Portable Network Graphics):

PNG is a standard format for lossless image compression. It supports transparency and is well-suited for images with sharp edges, such as logos and icons. PNG files don't suffer from the loss of quality associated with compression.

c. GIF (Graphics Interchange Format):

GIF is a standard format for animated images. It supports multiple frames that can be displayed sequentially, creating the illusion of animation. GIF also allows for transparent backgrounds and limited color palettes.

d. SVG (Scalable Vector Graphics):

SVG is a standard for representing vector-based graphics. It uses XML markup to describe shapes, paths, and other graphical elements, allowing for resolution-independent scaling. SVG is widely used for web graphics and icons.

Graphical file formats are specific file formats that comply with graphics standards. They determine how graphical data is organized, stored, and accessed within a file. Some popular graphical file formats include:

a. BMP (Bitmap):

BMP is a simple file format that stores images as uncompressed pixel data. It supports various color depths and is widely supported by different platforms, but it tends to result in large file sizes.

b. PSD (Adobe Photoshop Document):

PSD is the native file format of Adobe Photoshop. It supports layers, masks, and other advanced image editing features. PSD files preserve all the editing capabilities and are commonly used in professional graphic design workflows.

c. AI (Adobe Illustrator):

AI is the file format associated with Adobe Illustrator, a popular vector graphics editor. It stores vector-based graphics, including shapes, paths, and other artwork elements. AI files maintain the editing capabilities of the original artwork.

d. EPS (Encapsulated PostScript):

EPS is a file format that encapsulates vector-based graphics in a PostScript language format. It is widely used for printing and sharing graphics across different applications & platforms.

Date :.....
Page :.....

CHAPTER-2

1. Differentiate between raster scan display architecture and random scan display architecture.

→ The difference between raster scan display architecture and random scan display architecture are;

Raster scan display architecture	Random scan display architecture
1. It has less resolution because picture definition is stored as a intensity value.	1. It has high resolution because it stores picture definition as a set of line commands.
2. It is less expensive.	2. It is costlier than Raster scan system.
3. It stores picture definition in Refresh Buffer also called frame buffer.	3. It stores picture definition as a set of line commands called Refresh display file.
4. Zig-zag line is produced because plotted value are discrete.	4. Smooth line is produced because directly the line path is followed by electron beam.
5. It contains shadow, advance shading and hidden surface technique so gives the realistic display of scenes.	5. It doesnot contain shadow & hidden surface technique. so it can not give realistic display of scenes.
6. Refresh rate is 60 to 80 frame per second.	6. Refresh rate depends on the number of lines to be displayed i.e 30 to 60 sec.
7. It uses pixels along scan lines for drawing an image.	7. It is designed for line drawing applications & uses various mathematical function to draw.

Date :.....
Page :.....

2. Describe about hard-copy devices that are used in real life with example.

→ The hard-copy devices that are used in real life with examples are:

a. Printer:

Printers are one of the most widely used hard copy devices. The produce text, images, or both on paper or other printable materials. There are different types of printers available, including inkjet printers, laser printers, and dot matrix printers. They can be connected to computers or used wirelessly.

b. Photocopier:

Photocopiers, also called copiers or copy machines, are used to duplicate documents. They utilize a combination of light, static electricity, and toner to create replicas of printed materials. Photocopiers are often found in offices, libraries, and other shared spaces.

c. Scanner:

Scanners capture physical documents or images and convert them into digital formats. They use light-sensitive sensors to read and translate the information on papes or other objects into electronic files. Scanners are commonly used for archiving documents, creating digital copies, or sending information via email.

Date :.....

Page :.....

d. Fax machine:

Despite the prevalence of digital communication, fax machines are still utilized in certain industries. They transmit printed documents over telephone lines, allowing the recipient to receive a physical copy of the sent information. Fax machines are often used in offices, medical facilities and legal organizations.

e. Plotter:

Plotters are specialized hard copy devices used for printing large-scale graphics, such as engineering drawings, blueprints, or architectural designs. They use pens or markers to draw precise lines on paper, vinyl, or other materials. Plotters are commonly used in engineering, architecture, and design industries.

Hard copy devices, also known as output devices, are peripherals that generate physical copies of digital information. They are commonly used in various settings, including offices, homes, and educational institutions.

Date :.....

Page :.....

3. Why flat display is popular in recent scenario? Short explain about LCD display system.

→ Flat displays have gained popularity in recent years due to several reasons:

a. Aesthetics:

Flat displays provide a sleek and modern look to electronic devices such as televisions, computer monitors, and smartphones. The slim profile and clean lines of flat displays are visually appealing to consumers.

b. Space Efficiency:

Flat displays take up less physical space compared to older bulky CRT (Cathode Ray tube) displays. This makes them more efficient and convenient for use in various settings, including homes, offices, and public spaces.

c. Versatility:

Flat displays can be easily integrated into different devices and applications. They can be made in various sizes, from small screens for smartphones to large screens for televisions. The flexibility and compatibility with different devices make them popular across a wide range of industries and consumer electronics.

Date :.....
Page :.....

An LCD (Liquid Crystal Display) is a type of flat-panel display technology widely used in electronic devices such as televisions, computer monitors, smartphones, and digital signage. It provides a visual output by controlling the properties of liquid crystals placed between two transparent electrodes.

Here's a simplified explanation of how an LCD display system works:

a. Light source: An LCD screen requires a backlight to provide illumination. This can be in the form of a fluorescent tube or LEDs (Light Emitting Diodes) placed at the back of the display panel.

b. Polarizers: The light generated by the backlight passes through a polarizer, which allows only light waves oscillating in a particular direction to pass through.

c. Liquid crystals: Behind the first polarizer, there is a layer of liquid crystals. These crystals are made up of elongated molecules that can change their alignment when an electric field is applied.

d. Control of light: When a voltage is applied to the electrodes corresponding to a particular pixel, the liquid crystal molecules align with the electric field, untwisting the light passing through them. This untwisted light can now pass through the second polarizer.

Date :.....
Page :.....

4. Explain the working principle of CRT monitor.

→ CRT stands for cathode Ray Tube. CRT is a technology used in traditional computer monitors and televisions. The image on CRT display is created by firing electrons from the back of the tube of phosphorous located towards the front of the screen.

Once the electron heats the phosphorous, they light up, and they are projected on a screen. The color you view on the screen is produced by a blend of red, blue and green light.

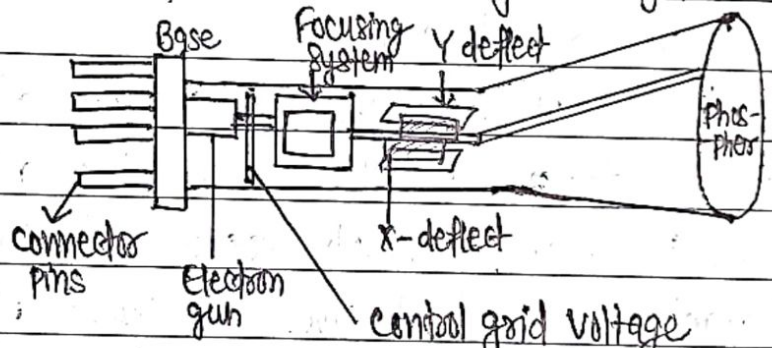

Fig- CRT (Cathode Ray Tube)

Components of CRT :

a. Electron Gun :

Electron gun consisting of a series of elements, primarily a heating filament (heater) and a cathode. The electron gun creates a source of electrons which are focused into a narrow beam directed at the face of the CRT.

b. Control Electrode :

It is used to turn the electron beam on and off.

c. Focusing System :

It is used to create a clear picture by focusing the electrons into a narrow beam.

Date :.....

Page :.....

d. Deflection Yoke:

It is used to control the direction of the electron beam. It creates an electric or magnetic field which will bend the electron beam as it passes through the area. In a conventional CRT, the yoke is linked to a sweep or scan generator. The deflection yoke which is connected to the sweep generator creates a fluctuating electric or magnetic potential.

e. Phosphorus-coated screen:

The inside front surface of every CRT is coated with phosphorus. Phosphors glow when a high-energy electron beam hits them. Phosphorescence is the term used to characterize the light given off by a phosphor after it has been exposed to an electron beam.

Date :.....
Page :.....

CHAPTER-3

1. What are HSR algorithms ? Explain any three HSR algorithms.

→ HSR (Hierarchical Spatial Reasoning) algorithms are used in computer vision and image processing to analyze and understand the spatial relationships between objects in an image or scene. These algorithms aim to capture the hierarchical structure of objects, their arrangement, and the relationships between them.

The any three HSR algorithms are:

a. Region Quadtree (R-Quadtree): The R-Quadtree algorithm is used to partition an image into regions based on color or intensity similarities. It starts with the entire image as a single region & recursively subdivides it into four quadrants until a desired level of detail or a termination condition is reached. The resulting quadtree structure represents the hierarchical arrangement of regions in the image. This algorithm is useful for tasks such as image compression, region-based analysis, & object recognition.

b. Binary Space Partitioning (BSP): BSP is a spatial partitioning algorithm that divides a scene into two regions based on a splitting line or plane. It recursively subdivides the regions, creating a binary tree structure. BSP trees can be used to efficiently determine visibility in computer graphics, perform collision detection in virtual environments, and accelerate ray-tracing algorithms. The splitting planes can be defined based on the spatial relationships between objects, such as their position, orientation, or distance.

Date :.....
Page :.....

c. Octree: An octree is a three-dimensional extension of the quadtree. It is used to partition a 3D space into smaller octants. Similar to the R-Quadtree, an octree starts with the entire space as a single octant and recursively subdivides it until a termination condition is met. Octrees are particularly useful in 3D computer graphics and spatial data structures, enabling efficient representation and processing of volumetric data, such as 3D models, medical images, and point clouds.

2. What are VSD algorithms? Explain any three VSD algorithms.

→ VSD (Variation-Sensitive Design) algorithms are computational methods used in the field of computer-aided design (CAD) and computer-aided engineering (CAE). These algorithms are specially designed to handle variations in the design parameters and optimize the performance of the resulting product under different operating conditions.

The any three VSD algorithms are:

a. Monte Carlo Simulation:

Monte Carlo simulation is a statistical method that uses random sampling to estimate the behavior of a system under uncertainty. In VSD, it is used to analyze the performance of a design by generating random variations in the design parameters and simulating the system multiple times. By running the simulation repeatedly with different parameter values, designers can obtain statistical distributions

Date :.....
Page :.....

of the performance metrics and assess the variability of the design.

b. Response Surface Methodology (RSM):

RSM is a technique used to model and optimize the response of a system with respect to multiple input variables. In VSD, RSM is employed to build mathematical models that describe the relationship between design parameters and the performance metrics of interest. These models are then used to identify the optimal design settings that maximize performance or minimize variability.

c. Sensitivity Analysis:

Sensitivity Analysis is a method used to assess the impact of variations in design parameters on the performance of a system. In VSD, sensitivity analysis helps designers understand which parameters have the most significant influence on the system's response and identify critical design variables. By quantifying the sensitivity of the performance metrics to different parameters, designers can prioritize their efforts in controlling or optimizing specific variables to achieve desired performance goals.

Date :.....
Page :.....

3. Digitize the line with end points (20,10) to (30,18) using Bresenham's line drawing algorithm.

→ The line with end points (20,10) to (30,18) using Bresenham's line drawing algorithm. Here's the step-by-step process:

a. Calculate the differences between the x-coordinates and y-coordinates of the two endpoints:

dx = 30 - 20 = 10

dy = 18 - 10 = 8

b. Determine the increments for each coordinates direction (either +1 or -1):

let dx1 = 1 and dy1 = 1 (positive increments).

c. Calculate the decisions parameters:

d = 2 * dy - dx

d1 = 2 * dy

d2 = 2 * (dy - dx)

d. Initialize the starting point:

x = 20 and Y = 10

e. Begin the line-drawing algorithm:

start a loop from 0 to dx and perform the following steps inside the loop:

(i) plot the current point (x,y).

(ii) If the decision parameter (d) is greater than 0, increment y by dy1 and update d as follows:

y = y + dy1

d = d + d2

(iii) If the decision parameter (d) is less than or equal to 0, update d as follows:

$$d = d + d_1$$

(iv) Increment x by dx1.

f. Repeat steps(i) to(iv) until the loop ends (i.e when the value of dx is reached).

Following the above steps, we can apply Bresenham's algorithm to digitize the line from (20,10) to (30,18). Here's the list of plotted points:

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

These points represent the line from (20,10) to (30,18) when using Bresenham's line drawing algorithms.

Date :
Page :

4. Explain Digital Differential Analyzer Algorithm (DDA) with example. Calculate the coordinates of line using DDA algorithm of between points (0,0) and (4,5).

→ The Digital Differential Analyzer (DDA) algorithm is another method used for line drawing and rasterization. It uses the concept of incremental steps along the line to determine the coordinates of the pixels to be plotted. Herels an explanation of the DDA algorithm with an example:

Step 1: Determine the coordinates of the end points

$(x_1, y_1) = (20, 10)$

$(x_2, y_2) = (30, 18)$.

Step 2: Calculate the differences between the end points

$dx = x_2 - x_1 = 30 - 20 = 10$

$dy = y_2 - y_1 = 18 - 10 = 8$

Step 3: Determine the number of steps required.

Lets the number of steps be n, where n is the greater absolute difference between dx and dy.

$n = \max(|dx|, |dy|) = \max(10, 8) = 10$

Step 4: Calculate the incremental values for x and y.

$ix = dx/n = 10/10 = 1$

$iy = dy/n = 8/10 = 0.8$ (rounded to the nearest integer).

Step 5: Initialize the starting point

$x = x_1$ and $y = y_1$.

Step 6: Begin the line drawing algorithm.

For i = 1 to n:

plot the point (x, y).

Date :.....
Page :.....

$x = x + ix$ and $y = y + iy$

Using the DDA algorithm, let's apply the steps to the given endpoints:

Step 1: $(x_1, y_1) = (20, 10)$, $(x_2, y_2) = (30, 18)$.

Step 2: $dx = 10$, $dy = 8$

Step 3: $n = 10$.

Step 4: $ix = 1$, $iy = 1$ (rounded from 0.8).

Now, we'll apply step 6 for $i = 1$ to 10:

Iteration 1: Plot the point (20, 10).

$x = 20 + 1 = 21$ and $y = 10 + 1 = 11$

Iteration 2: Plot the point (21, 11).

$x = 21 + 1 = 22$ and $y = 11 + 1 = 12$

Iteration 3: Plot the point (22, 12).

$x = 22 + 1 = 23$ and $y = 12 + 1 = 13$

Iteration 4: Plot the point (23, 13).

$x = 23 + 1 = 24$ and $y = 13 + 1 = 14$

Iteration 5: Plot the point (24, 14).

$x = 24 + 1 = 25$ and $y = 14 + 1 = 15$

Iteration 6: Plot the point (25, 15).

$x = 25 + 1 = 26$ and $y = 15 + 1 = 16$

Iteration 7: Plot the point (26, 16).

$x = 26 + 1 = 27$ and $y = 16 + 1 = 17$

Iteration 8: Plot the point (27, 17).

$x = 27 + 1 = 28$ and $y = 17 + 1 = 18$

Iteration 9: Plot the point (28, 18).

$x = 28 + 1 = 29$ and $y = 18 + 1 = 19$.

→ To calculate the coordinates of a line using the DDA (Digital Differential Analyzer) algorithm between two points (x_0, y_0) and (x_1, y_1), we need to determine the number of steps required to traverse the longer distance in either the x-axis or the y-axis. Then, we incrementally calculate the coordinates of each step.

Lets apply the DDA algorithm to find the coordinates of the line between points (0,0) and (4,5):

a. Calculate the differences in the x and y coordinates:

$dx = x_1 - x_0 = 4 - 0 = 4$

$dy = y_1 - y_0 = 5 - 0 = 5$

b. Determine the number of steps:

$Steps = max(|dx|, |dy|) = max(4, 5) = 5$

c. Calculate the increments in the x and y directions:

x increment = dx/steps = 4/5 = 0.8

y increment = dy/steps = 5/5 = 1

d. Initialize the starting coordinates:

$x = x_0 = 0$ and $y = y_0 = 0$

e. Generate the coordinates of each step:

Step 1 : (0,0)

Step 2 : (0 + x increment, 0 + y increment) = (0 + 0.8, 0 + 1) = (0.8, 1)

Step 3 : (0.8 + 0.8, 1 + 1) = (1.6, 2)

Step 4 : (1.6 + 0.8, 2 + 1) = (2.4, 3)

Step 5 : (2.4 + 0.8, 3 + 1) = (3.2, 4)

Step 6 : (3.2 + 0.8, 4 + 1) = (4, 5).

Thus, the coordinates of the line using the DDA algorithm between points (0,0) & (4,5) are (0,0), (0.8, 1), (1.6, 2), (2.4, 3), (3.2, 4), (4,5).

Date :.....
Page :.....

5. Compute points on arc of circle using mid-point circle drawing algorithm for a circle with radius R = 8. [When R = 10]

→ The mid point circle drawing algorithm is a simple algorithm used to generate points on the circumference of a circle. Here's how we can compute the points on the arc of a circle using the mid-point circle drawing algorithm for a circle with radius of R = 8. [R = 10]

Algorithm:

a. Initialize the center of the circle as (0, 0).

b. Set the initial point on the circumference of the circle as (0, R).

c. Initialize the decision parameter as P = 1 - R.

d. Iterate while the x-coordinates is less than or equal to the y-coordinate:

- If the decision parameter is less than 0, update the next point as (x+1, y) and the decision parameter as P = P + 2x + 3.
- If the decision parameter is greater than or equal to 0, update the next point as (x+1, y-1) and the decision parameter as P = P + 2x - 2y + 5.
  - Increment x by 1.
  - Decrement y by 1.

e. Repeat the above ~~two~~ steps for one octant of the circle and plot the corresponding points in the other seven octants symmetrically.

Here's an example implementation in python that compute points on the arc of a circle with Radius = 8. [R = 10]

Date :
Page :

```
import matplotlib.pyplot as plt
def draw_circle(radius):
    x = 0
    y = radius
    decision_param = 1 - radius
    points = []
    while x <= y:
        points.append((x, y))
        if decision_param < 0:
            decision_param += 2 * x + 3
        else:
            decision_param += 2 * (x - y) + 5
            y -= 1
        x += 1
    # Generate points in other octants symmetrically
    all_points = points + [(y, x) for x, y in points] + [(-y, x)
    for x, y in points] + [(x, -y) for x, y in points] + [(-x, -y) for
    x, y in points] + [(-y, -x) for x, y in points] + [(y, -x) for
    x, y in points] + [(-x, y) for x, y in points]
    return all_points
radius = 8
circle_points = draw_circle(radius)
# plotting the points
x_coords, y_coords = zip(*circle_points)
plt.scatter(x_coords, y_coords)
plt.axis('equal')
plt.show()
```

Date :.....
Page :.....

This code will generate the points on the arc of a circle with a radius of 8 using the midpoint circle drawing algorithm and plot them using matplotlib.

Q: [When Radius = 10]

```
import matplotlib.pyplot as plt
def plot_symmetric_points(x, y):
 # Plot the symmetric points
    plt.plot(x, y, 'bo')
    plt.plot(y, x, 'bo')
    plt.plot(-x, y, 'bo')
    plt.plot(y, -x, 'bo')
    plt.plot(-x, -y, 'bo')
    plt.plot(-y, -x, 'bo')
    plt.plot(x, -y, 'bo')
    plt.plot(-y, x, 'bo')
def draw_circle(radius):
    x = 0 and y = radius
    decision_param = 1 - radius
    while x <= y:
     if decision_param >= 0:
        y -= 1
        decision_param += 2 * (x - y) + 5
    else:
        decision_param += 2 * x + 3
        x += 1
```

Date :.....
Page :.....

```
plot_symmetric_points(x,y)
draw_circle(10)
plt.axis('equal')
plt.show()
```

This code uses the 'matplotlib' library to plot the symmetric points on the circle arc. When we run the code, it will display a plot with the points forming the arc of the circle with radius R = 10.

6. Write about Bresenham's circle drawing algorithm.

→ Bresenham's circle drawing algorithm is an efficient method for approximating the points on the circumference of a circle using integer arithmetic. The algorithm was developed by Jack E. Bresenham in 1965 and is widely used in computer graphics and image processing.

The basic idea behind Bresenham's algorithm is to determine the appropriate positions to plot on the circumference of a circle based on the octant (one-eight of a circle) in which the point lies. By selecting the most appropriate points, the algorithm ensures that the resulting circle is as close as possible to the ideal mathematical circle.

Here is a step-by-step explanation of the Bresenham's circle drawing algorithm:

a. Start by initializing the center of the circle at coordinates (x_0, y_0) and the radius r.

b. Set the initial values of two variables: $x = 0$ and $y = r$.

Date :.....
Page :.....

c. Calculate the initial decision parameter as $P = 3 - 2r$.

d. Enter a loop that iterates until x becomes greater than or equal to y:

(i) Plot the points $(x_0 + x, y_0 + y)$ and $(x_0 + y, y_0 + x)$ in the current octant of the circle.

(ii) If the decision parameter P is greater than or equal to zero, decrement y and update P as $P = P + 4(x - y) + 10$.
Otherwise, update P as $P = P + 4x + 6$.

(iii) Increment x.

e. After the loop ends, the circle will be plotted in the current octant. Repeat the same process for the remaining seven octants by using symmetry and reflections.

By following these steps, Bresenham's algorithm allows us to draw circles efficiently using integer increments and avoid costly floating-point calculations. The algorithm is particularly useful in situations where integer operations are performed or where floating-point operations are not available or expensive.

Note that the above explanation assumes that the center coordinates (x_0, y_0) and the radius r are given as input. The algorithm can be modified to handle other variations, such as plotting a circle using a different center point or using a different starting octant.

Date :
Page :

7. Write a mid-point circle drawing algorithm.

→ The mid point circle drawing algorithm is a widely used algorithm in computer graphics to efficiently draw a circle on a raster display. It calculates the coordinates of the points along the circumference of a circle by utilizing the symmetry properties of circles.

The algorithm starts with the center coordinates of the circle and its radius. It then iteratively determines the coordinates of points along the circumference by dividing the circle into eight octants and plotting points in each octant. The algorithm takes advantage of the fact that the circle is symmetric in all eight octants.

Here is a step-by-step description of the mid point circle drawing algorithm:

a. Input the center coordinates (x_c, y_c) of the circle and its radius (r).

b. Set the initial values of the variables: $x = 0$, $y = r$, and the decision parameter $p = 1 - r$.

c. Repeat the following steps until $x <= y$:

(i) Plot the points (x_c+x, y_c+y), (x_c-x, y_c+y), (x_c+x, y_c-y), (x_c-x, y_c-y), (x_c+y, y_c+x), (x_c-y, y_c+x), (x_c+y, y_c-x), (x_c-y, y_c-x) to draw the circle points in all eight octants.

(ii) If $p < 0$, update the decision parameter: $p = p + 2x + 1$.

(iii) If $p >= 0$, update the decision parameter: $p = p + 2x - 2y + 1$ and decrement y by 1.

(iv) Increment x by 1.

Date :.....
Page :.....

d: Repeat the above steps until $x > y$.

The algorithm calculates the decision parameter at each step based on the current point (x, y). If the decision parameter is less than 0, it means the next point to be plotted lies inside the circle, so x is incremented, and the decision parameter is updated accordingly. If the decision parameter is greater than or equal to 0, it means the next point to be plotted lies outside the circle, so both x and y are incremented, and the decision parameter is updated accordingly.

By using this algorithm, the circle can be drawn efficiently without having to calculate the coordinates of all the points along its circumference. The midpoint circle drawing algorithm is a simple and effective technique for rendering circles in computer graphics and has been widely used in various applications, including graphics libraries and software.

Date :.....

Page :.....

CHAPTER-4

1: Find the rotation matrix R for rotation of a point from position (X, Y) to position (X', Y') through an angle θ relative to the coordinates origin.

→ To find the rotation matrix R for rotating a point from position (X, Y) to position (X', Y') through an angle θ relative to the coordinate origin, we can use the following formula:

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

This matrix represents a counterclockwise rotation by an angle θ.

To apply this rotation matrix to a point (X, Y), we can use matrix multiplication as follows:

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} * \begin{bmatrix} X \\ Y \end{bmatrix}$$

The resulting vector (X', Y') will be the coordinates of the rotated point.

Note that the angle θ is usually measured in radians. If we have the angle in degrees, we need to convert it to radians by multiplying it by $\frac{\pi}{180}$.

Date :.....
Page :.....

2. What is transformation and 3D transformation? List out different types of transformations. Explain in brief.

→ Transformation refers to the process of changing the position, orientation, size, or shape of an object in a coordinate plane. Transformations are used to manipulate geometric figures while preserving certain properties, such as angles, lengths & parallelism.

3D transformation refers to the process of changing the position, orientation, and scale of a three-dimensional space. It involves manipulating the objects coordinates and vertices to achieve the desired transformation, such as translation, rotation, scaling, shearing, and perspective projection.

The different types of transformations are:

a. Translation:

A translation involves moving an object from one location to another without changing its size, shape, or orientation. This can be done by shifting the object horizontally and/or vertically.

b. Rotation:

Rotation involves rotating an object around a fixed point called the center of rotation. The object remains the same size and shape, but its orientation changes.

c. Reflection:

Reflection involves flipping an object over a line called the line of reflection. The line of reflection acts as a mirror, and the object's image is a flipped version of itself.

Date :.....
Page :.....

d. Scaling :

Scaling involves changing the size of an object. It can be done by multiplying or dividing the coordinates of the object's points by a scale factor. Scaling can either enlarge (stretch) or reduce (shrink) the object.

e. Shearing:

Shearing involves distorting an object by shifting its parts along a specific direction. This creates a slanted or skewed version of the original object.

f. Dilation :

Dilation is a type of scaling the changes both the size and shape of an object. It involves enlarging or reducing an object while keeping its properties intact. Dilation is performed relative to a fixed point called the center of dilation.

Each type of transformation has specific characteristics and rules associated with it. These transformations are fundamental concepts in geometry and have applications in various fields, including computer graphics, robotics and architecture.

Date :.....
Page :.....

3. Let ABCD be the rectangular window with A(10,10), B(80,10), C(80,60) and D(10,60). Find the region code for end points and use it when Cohen-Sutherland algorithm to clip the line P_1P_2 with $P_1(8,30)$ and $P_2(20,70)$.

→ To apply the Cohen-Sutherland algorithm for line clipping, we need to determine the region codes for the endpoints of the line segment relative to the rectangular window. The region codes are typically represented using 4 bits, each bit indicating the location of the point in relation to the window.

Let's calculate the region codes for the endpoints $P_1(8,30)$ and $P_2(20,70)$ with respect to the rectangular window ABCD.

For a point (x,y) inside the window, the region code is 0000. For a point (x,y) above the window, the region code has the first bit set to 1 (binary representation : 1). For a point (x,y) below the window, the region code has the second bit set to 1 (binary representation : 2). For a point (x,y) to the left of the window, the region code has the third bit set to 1 (binary representation : 4). For a point (x,y) to the right of the window, the region code has the fourth bit set to 1 (binary representation : 8).

Let's calculate the region codes for the end points:

Endpoints $P_1(8,30)$.

- It is to the left of the window, so the third bit is set to 1.
- It is within the vertical boundaries of the window, so the first and second bits are set to 0.

Date :.....
Page :.....

- It is within the horizontal boundaries of the window, so the fourth bit is set to 0.

Region code for P_1 : 0100 (binary representation : 4)

Endpoints P_2 (20, 70).

- It is within the vertical boundaries of the window, so the first and second bits are set to 0.
- It is to the right of the window, so the fourth bit is set to 1.
- It is below the window, so the second bit is set to 1.

Region code for P_2 : 1002 (binary representation : 10)

Now, that we have determined the region codes for the endpoints, we can apply the cohen-Sutherland algorithm to clip the line segment P_1P_2.

Date :.....
Page :.....

4. State Cohen-Sutherland line clipping algorithm with example.

→ The Cohen-Sutherland line clipping algorithm is a popular algorithm used for line clipping in computer graphics. It divides the 2D space into nine regions and determines which parts of a line lie inside or outside the clipping window. The algorithm uses bitwise operations to quickly determine the region codes for the line endpoints and performs several tests to determine if the line is completely outside, partially inside, or completely inside, partially inside, or completely inside the clipping window. Here's the step-by-step process of the Cohen-Sutherland line clipping algorithm.

a. Define the clipping window : This is usually a rectangular region defined by two points : (x_{min}, y_{min}) for the bottom-left corner & (x_{max}, y_{max}) for the top-right corner.

b. Assign region codes : Each point (X, Y) in the 2D space is assigned as 4-bit region code, indicating its location with respect to the clipping window. The region codes are determined as follows :
- Bit 1 (LSB) : set to 1 if the point is above the clipping window.
- Bit 2 : Set of 1 if the point is below the clipping window.
- Bit 3 : Set of 1 if the point is to the right of the clipping window.
- Bit 4 (MSB) : Set of 1 if the point is to the left of the clipping window.

c. Clip the line : Consider a line segment defined by two endpoints $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$. Perform the following steps to clip the line :
- (i) Compute the region codes for P_1 and P_2.
- (ii) Check if both endpoints are inside the clipping window

Date :.....
Page :.....

(region codes = 0000). If so, the line is completely visible and can be drawn.

(iii) Check if both endpoints are outside the same region (logical AND of region codes != 0000). If so, the line is completely outside the clipping window and can be rejected.

(iv) If neither of the above conditions is met, the line is partially visible. Repeat steps (i) and (ii) until the line is completely inside or outside the clipping window.

d. Draw the clipped line: If the line is partially visible, draw the line segment between the updated endpoints.

Example: Consider a clipping window defined by (xmin = 50, ymin = 50) and (xmax = 300, ymax = 200). Let's clip a line segment with endpoints $P_1(20, 80)$ and $P_2(200, 250)$.

(i) Define the clipping window: (xmin = 50, ymin = 50) & (xmax = 300, ymax = 200).

(ii) Assign region codes:

- $P_1(20, 80)$: 0001 (outside the left edge).
- $P_2(200, 250)$: 0010 (outside the bottom edge).

(iii) Clip the line: a. Compute region codes: • P_1: 0001 • P_2: 0010

b. Both end points are outside the same region, so the line is completely outside the clipping window. Reject the line.

Thus, the line segment ($P_1(20, 80)$, $P_2(200, 250)$) is rejected as it is entirely outside the clipping window.

Date :.....
Page :.....

5. State Liang-Barsky line clipping algorithm with example.

→ The Liang-Barsky line clipping algorithm is another commonly used algorithm for line clipping in computer graphics. It works by determining the intersection points between a line segment and the clipping window edges. The algorithm uses parameterization to calculate the values of t that represent the position of the intersection points along the line segment. Heres the step-by-step process of the Liang-Barsky line clipping algorithm:

a. Define the clipping window: This is usually a rectangular region defined by two points: (xmin, ymin) for the bottom-left corner and (xmax, ymax) for the top-right corner.

b. Define the line segment: Specify the endpoints of the line segment as $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$.

c. Calculate the values of t:
- Calculate $dx = x_2 - x_1$ and $dy = y_2 - y_1$.
- Initialize $t_0 = 0$ (for the first intersection point) and $t_1 = 1$ (for the second intersection point).
- Calculate the values of $P = (-dx, dx, -dy, dy)$ and $q = (x_1 - x_{min}, x_{max} - x_1, y_1 - y_{min}, y_{max} - y_1)$.

d. Check for line parallel to clipping window edges:
- If $P = (0, 0, 0, 0)$, the line is parallel to one or more of the clipping window edges. Check if the line is outside the clipping window. If it is, reject the line. Otherwise, the line is completely inside the clipping window, and it can be drawn.

Date :.....
Page :.....

e. Perform parameter adjustments:
- For each of the four clipping window edges:
  - If P is negative, update t_0 to be the maximum of t_0 and q/p.
  - If P is positive, update t_1 to be the minimum of t_1 and q/p.

f. Check if the line is outside the clipping window:
- If $t_0 > t_1$, the line is completely outside the clipping window. Reject the line.

g. Calculate the intersection points:
- Calculate the intersection points of the line segment with the clipping window edges using t_0 and t_1.

h. Draw the clipped line:
- Draw the line segment between the intersection points obtained in step g.

Example: Consider a clipping window defined by (Xmin = 50, ymin = 50) and (Xmax = 300, ymax = 200). Let's clip a line segment with end points P_1(20, 80) and P_2(200, 250).

a. Define the clipping window: (Xmin = 50, ymin = 50) & (Xmax = 300, ymax = 200)
b. Define the line segment: P_1(20, 80) and P_2(200, 250).
c. Calculate the values of t:
- dx = 200 − 20 = 180
- dy = 250 − 80 = 170
- P = (−180, 180, −170, 170)
- q = (20 − 50, 300 − 20, 80 − 50, 200 − 80) = (−30, 280, 30, 120).

d. check for line parallel to clipping window edges: P = (−180, 180, −170, 170), which is not (0, 0, 0, 0). So the line is not parallel to the clipping window edges.

e. Perform parameter adjustments:
- For the left edge (x = xmin):
  - P = −180, q = −30

Date :.....
Page :.....

CHAPTER-5

1. Explain the basic object Illumination models.

→ Object illumination models are used in computer graphics and computer vision to simulate how light interacts with objects in a scene. These models take into account various factors such as the properties of the light sources, the surfaces properties of the ~~lights~~ objects, and the geometry of the scene to determine how light is reflected, transmitted, and scattered by the objects.

There are several basic objects illumination models commonly used in computer graphics. Here are three of the most fundamental models:

a. Lambertian Reflection Model: The lambertian model assumes that light is diffusely by a matte (non-shiny) surface. It is based on lambert's cosine law, which states that the amount of light reflected by a surface is proportional to the cosine of the angle between the surface normal and the direction of the incident light. This model is simple and widely used because it provides a good approximation for many materials with rough surfaces.

b. Phong Reflection Model: The phong model extends the lambertian model by considering specular (shiny) reflections in addition to diffuse reflections. It consists of three components: ambient, diffuse, and specular. The ambient components represents the light that is scattered and reflected by the environment. The diffuse component represents the light that is scattered equally in all directions by a rough surface. The specular component.

Date :.....
Page :.....

represents the highlight or mirror-like reflection that occurs on shiny surfaces. The phong model also introduces a shininess exponent, called the specular exponent, which controls the size and sharpness of the specular highlights.

e. Blinn-Phong Reflection model: The Blinn-Phong Model is a variation of the Phong model that replaces the specular exponent with a specular power parameter. This change provides a more computationally efficient way to calculate the specular highlights. The Blinn-Phong model approximates the specular reflection by using the halfway vector between the direction of the incident light and the direction towards the viewer. The specular power parameter controls the size and concentration of the specular highlights, with higher values producing smaller and more concentrated highlights.

2. What is projection? Explain parallel and perspective projections in computer graphics.

→ Projection is a fundamental concept in computer graphics that involves transforming a three-dimensional (3D) scene into a two-dimensional (2D) representation for display on a screen. It is necessary because computer screens or images are flat surfaces and cannot directly represent the depth information present in a 3D scene. Projection techniques allow us to create a 2D image that accurately represents the spatial relationships and perspective of the original 3D scene.

Date :.....

Page :.....

Parallel projection : In parallel projection, all lines of sight from the viewer are parallel, resulting in a uniform scaling of the 3D scene onto the 2D plane. This projection type preserves the relative sizes and shapes of objects in the scene but doesnot accurately represent depth or the sense of perspective. Parallel projection is often used in the technical drawings, engineering designs, and architectural blueprints. One common for of parallel projection is the orthographic projection, where the projection rays are perpendicular to the projection plane. In orthographic projection, all objects appear the same size regardless of their distance from the viewer.

Perspective Projection: Perspective projection aim to create a more realistic representation by simulating the way objects appear in our visual perception. It mimics the behavior of our eyes when viewing a 3D scene. In perspective projection, lines of sight from the viewer converge to a single vanishing point, creating the illusion of depth and distance. In perspective projection, objects that are closer to the viewer appear larger, while objects that are farther away appear smaller. This scaling effect emphasizes the perception of depth and spatial relationships. Perspective projection is commonly used in 3D computer graphics, virtual reality, and video games to create a more immersive and realistic experience.

Date :.....
Page :.....

3. What is hidden surface removal techniques? Explain brief.

→ Hidden surface removal (HSR) techniques, also known as visibility algorithms, are methods used in computer graphics to determine which surfaces or parts of surfaces or parts of surfaces are visible or hidden from a specific viewpoint. The goal is to accurately render a scene by removing the surfaces that are occluded or blocked from view by other objects in the scene.

There are several commonly used techniques for hidden surface removal:

a. Back-face culling: This technique assumes that all polygons have a consistent orientation (usually defined by a normal face away from the viewer. Back-face culling assumes that the viewer is always looking at the front face of objects and ignores polygons that face away from the viewer's perspective.

b. Z-Buffer Algorithm: The z-buffer algorithm maintains a separate buffer, called the z-buffer or depth buffer, which stores the depth buffer, which stores the depth value for each pixel on the screen. As polygons are rendered, their depth values are compared with the corresponding values in the z-buffer. If the new depth value is closer to the viewer, the pixel is considered visible and drawn; otherwise, it is discarded.

Date :.....
Page :.....

c. Scanline Algorithm : The scanline algorithm works by scanning each horizontal line of the screen and determining which polygons intersect that line. It then computes the intersection points and draws the visible segments. This algorithm requires polygons to be sorted based on their y-coordinates to handle overlapping polygons correctly.

d. Painter's Algorithm : The painter's algorithm sorts polygons based on their distance from the viewer and renders them from back to front. By drawing objects in the order of decreasing distance, closer objects will cover those farther away, simulating the correct visibility.

4. Explain the basic polygon rendering methods.

→ Polygon rendering methods are techniques used to represent and display polygons, which are fundamental building blocks in computer graphics. These methods involve converting polygon data into pixel-based representations that can be displayed on a screen. Here are some basic polygon rendering methods :

a. Wireframe Rendering : In wireframe rendering, only the edges of polygons are drawn, resulting in a simplified representation of the object. This method provides a basic outline of the shape but doesnot include any shading or surface details.

b. Flat shading : Flat shading assigns a single color to each polygon, usually based on the color of the polygon's surface normal or a

material property. All pixels within a polygon are filled with this color, giving the appearance of a flat, uniformly colored surface. This method doesnot take into account variations in shading across a polygon's surface.

c. Gouraud shading: Gouraud shading calculates the color values at the vertices of a polygon and interpolates those colors across the polygon's surface. The vertex colors are computed based on lighting models, material properties, or other factors. The resulting interpolated colors create a smoother shading effect across the polygon.

d. Phong shading: Phong shading improves upon Gouraud shading by interpolating the normal vectors at the vertices of a polygon and using those interpolated normals to calculate the shading of each pixel. This method provides more accurate shading and captures finer surface details compared to Gouraud shading.

e. Texture Mapping: Texture mapping involves applying a 2D image, called a texture, onto the surface of a polygon. The texture contains color or pattern information that is mapped onto the polygon's vertices and interpolated across its surface. This method allows for realistic surface details, such as wood grains, brick patterns, or complex surface textures.

Date :.....
Page :.....

5. Explain methods of generating non planar surfaces in 3D graphics.

→ Generating non-planar surfaces in 3D graphics involves creating complex, curved, or irregular surfaces that go beyond simple flat planes. Here are some methods commonly used to generate non-planar surfaces:

a. Parametric Surfaces: Parametric surfaces are defined using mathematical functions or equations that describe how the surface coordinates vary based on a set of parameters. These surfaces can be defined using equations such as Bezier curves, B-splines, NURBS (Non-Uniform Rational B-Splines), or parametric equations for geometric shapes like spheres, cylinders, torus, etc. Parametric surfaces allow for precise control over the shape and curvature of the surfaces.

b. Subdivision Surfaces: Subdivision surfaces are generated iteratively by subdividing and refining initial control meshes or polygons. Starting with a coarse mesh, the surface is divided into smaller elements, and each new vertex is positioned based on the surrounding vertices positions. The subdivision process is repeated multiple times resulting in a smoother and more detailed surface. Subdivision surfaces offer flexibility in modeling complex organic shapes with varying levels of detail.

c. Implicit Surfaces: Implicit surfaces define a surface as the zero set of a function. The function takes the 3D coordinates as input and outputs a scalar value. Points on the surface have a zero value, while points outside or inside the surface have positive or negative

Date :.....
Page :.....

values respectively. Implicit surfaces can be defined using mathematical equations or procedures such as distance functions, fractal algorithms (e.g. fractal landscapes), or constructive solid geometry (CSG) operations.

d. Procedural Techniques: Procedural techniques involve generating non-planar surfaces using algorithms or rules rather than explicit geometric representations. Procedural techniques can include fractal-based algorithms (such as Perlin noise or fractal subdivision), procedural textures, L-systems, or other algorithms that generate complex shapes or patterns based on rules or parameters. These methods are often used to create realistic terrains, landscapes, foliage, and other natural or organic forms.

These are just a few examples of methods used to generate non-planar surfaces in 3D graphics. The choice of method depends on the desired shape, complexity, control requirements, and computational resources available for modeling and rendering the surfaces.