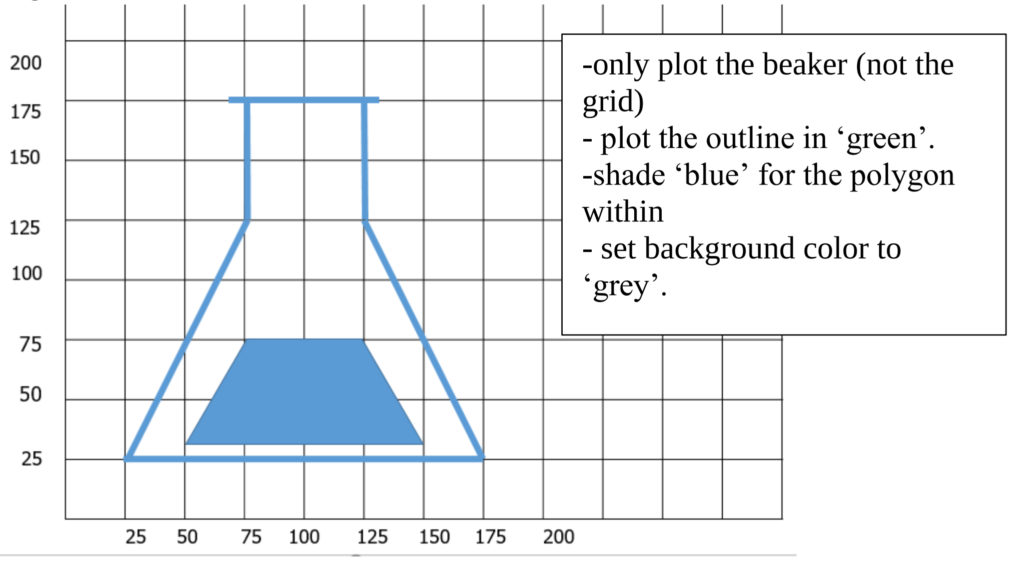
USN | | | | | | | | | | |

Internal Assessment Test 1 – June 2024

Sub:	Computer Graphics and Fundamentals of Image Processing				Sub Code:	21CS63	Branch:	CSE
Date:	04.06.2024	Duration:	90 mins	Max Marks: 50	Sem/Sec:	6 A,B,C		OBE

Answer any FIVE FULL Questions		MARKS	CO	RBT
1 (a)	List the applications of computer graphics and explain any three in detail.	5 M	CO1	L1
(b)	Explain the architecture of a raster-graphics system with a display processor.	5M	CO1	L1
2 (a)	Describe the general structure of an OpenGL program with suitable example.	5M	CO1	L1
(b)	Differentiate GL_LINES, GL_LINE_STRIP and GL_LINE_LOOP with code snippets and neat diagrams.	5M	CO1	L2
3 (a)	With a neat diagram, explain Bresenham's line drawing algorithm for slope >1	5M	CO1	L2
(b)	Trace points generated by Bresenham's line drawing algorithm for k, p_k and (x_{k+1}, y_{k+1}) for the coordinates (2,2) to (10,7). Plot the points generated.	5M	CO1	L3
4 (a)	Explain with neat diagrams, rotation about a pivot point and the derivation for the composite 3x3 homogeneous matrix thereof.	5M	CO2	L2
(b)	Consider the line from (6,2) to (6,6). What are the coordinates of the line when sx = sy = 0.5. Plot the line and coordinates. Show the 3x3 homogenous matrix calculations. Does the line move toward or away from the coordinate origin?	5M	CO2	L3
5 (a)	Explain 3D Rotations transformation with relevant transformation matrix.	5 M	CO 2	L2
(b)	Design transformation matrix to rotate a 3D object about an axis that is not parallel to one of the coordinate axis.	5 M	CO2	L2
6	Write an open GL program using OpenGL primitives to display the following. Code it in a function call beaker(). Write the full openGL code.Choose coordinate system of 200x200 with coordinate origin at bottom left. 	10 M	CO2	L3

CI **CCI** **HOD**

CMRIT — CMR INSTITUTE OF TECHNOLOGY, BENGALURU. ACCREDITED WITH A+ GRADE BY NAAC — CELEBRATING 25 YEARS

Internal Assessment Test 1 – June 2024
Solution

Sub:	Computer Graphics and Fundamentals of Image Processing					Sub Code:	21CS63	Branch:	CSE
Date:	04.06.2024	Duration:	90 mins	Max Marks:	50	Sem/Sec:	6 A,B,C		OBE

	Solution	MARKS	CO	RBT
1 (a)	List the applications of computer graphics and explain any three in detail.	5 M	CO1	L1

a. Graphs and Charts

An early application for computer graphics is the display of simple data graphs usually plotted on a character printer. Data plotting is still one of the most common graphics application.

Graphs & charts are commonly used to summarize functional, statistical, mathematical, engineering and economic data for research reports, managerial summaries and other types of publications.

Typically examples of data plots are line graphs, bar charts, pie charts, surface graphs, contour plots and other displays showing relationships between multiple parameters in two dimensions, three dimensions, or higher-dimensional spaces

✓ ***Computer-Aided Design***

✓ A major use of computer graphics is in design processes-particularly for engineering and architectural systems.

CAD, computer-aided design or CADD, computer-aided drafting and design methods are now routinely used in the automobiles, aircraft, spacecraft, computers, home appliances.

Circuits and networks for communications, water supply or other utilities are constructed with repeated placement of a few geographical shapes.

Animations are often used in CAD applications. Real-time, computer animations using wire-frame shapes are useful for quickly testing the performance of a vehicle or system.

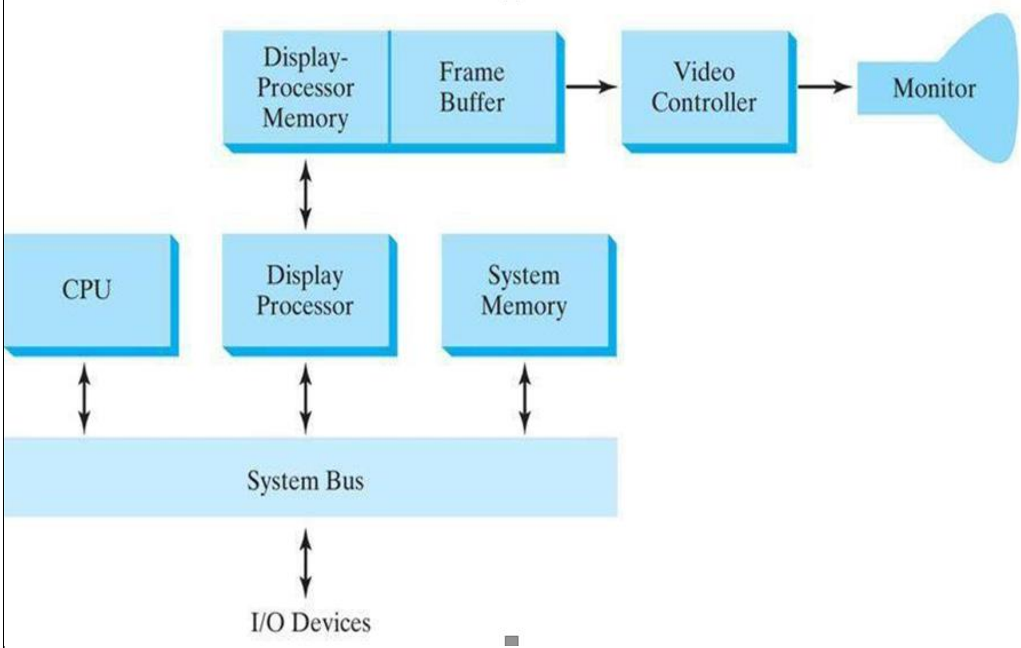
c. Data Visualizations

✓ Producing graphical representations for scientific, engineering and medical data sets and processes is another fairly new application of computer graphics, which is generally referred to as scientific visualization. And the term business visualization is used in connection with data sets related to commerce, industry and other nonscientific areas.

(b)	Explain the architecture of a raster-graphics system with a display processor.	5M	CO1	L1

a) Raster-Scan Display Processor

- ✓ Figure shows one way to organize the components of a raster system that contains a separate display processor, sometimes referred to as a graphics controller or a display coprocessor.



- ✓ The purpose of the display processor is to free the CPU from the graphics chores.
- ✓ In addition to the system memory, a separate display-processor memory area can be provided.

Scan conversion:

- ✓ A major task of the display processor is digitizing a picture definition given in an application program into a set of pixel values for storage in the frame buffer.
- ✓ This digitization process is called scan conversion.

2 (a)	Describe the general structure of an OpenGL program with suitable example.	5M	CO1	L1

✓ In windows to include OpenGL core libraries and GLU we can use the following header files:-

#include <windows.h> //precedes other header files for including Microsoft windows ver of OpenGL libraries

#include<GL/gl.h> #include <GL/glu.h>

✓ The above lines can be replaced by using GLUT header file which ensures gl.h and glu.h are included correctly,

✓ #include <GL/glut.h> //GL in windows

✓ In Apple OS X systems, the header file inclusion statement will be, ✓ #include <GLUT/glut.h>

Display-Window Management Using GLUT

✓ We can consider a simplified example, minimal number of operations for displaying a

picture.

Step 1: initialization of GLUT

- We are using the OpenGL Utility Toolkit, our first step is to initialize GLUT.

- This initialization function could also process any command line arguments, but we will not need to use these parameters for our first example programs.

- We perform the GLUT initialization with the statement

glutInit (&argc, argv);

Step 2: title

- We can state that a display window is to be created on the screen with a given caption for the title bar. This is accomplished with the function

glutCreateWindow ("An Example OpenGL Program");

- where the single argument for this function can be any character string that we want to

use for the display-window title.

Step 3: Specification of the display window

- Then we need to specify what the display window is to contain.

- For this, we create a picture using OpenGL functions and pass the picture definition to the GLUT routine glutDisplayFunc, which assigns our picture to the display window.

- Example: suppose we have the OpenGL code for describing a line segment in a procedure called lineSegment.

- Then the following function call passes the line-segment description to the display window:

glutDisplayFunc (lineSegment);

Step 4: one more GLUT function

- But the display window is not yet on the screen.
- We need one more GLUT function to complete the window-processing operations.
- After execution of the following statement, all display windows that we have created, including their graphic content, are now activated:

glutMainLoop ( );

- This function must be the last one in our program. It displays the initial graphics and puts the program into an infinite loop that checks for input from devices such as a mouse or keyboard.

Step 5: these parameters using additional GLUT functions

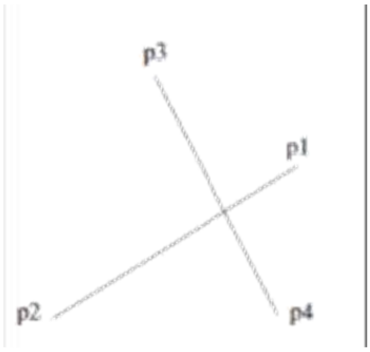
- Although the display window that we created will be in some default location and size,

we can set these parameters using additional GLUT functions.

(b)	Differentiate GL_LINES, GL_LINE_STRIP and GL_LINE_LOOP with code snippets and neat diagrams.	5M	CO1	L2

Case 1: Lines *glBegin (GL_LINES);*

glVertex2iv (p1);
glVertex2iv (p2);
glVertex2iv (p3);
glVertex2iv (p4);
glVertex2iv (p5);
glEnd ( );

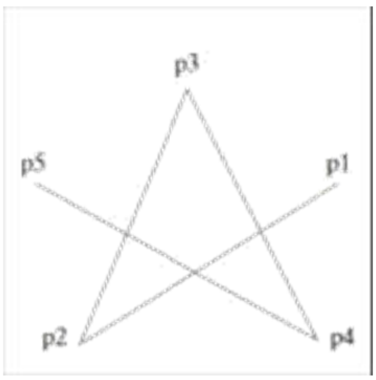


Case 2: GL_LINE_STRIP:

Successive vertices are connected using line segments. However, the final vertex is not connected to the initial vertex.

glBegin (GL_LINES_STRIP);

glVertex2iv (p1);
glVertex2iv (p2);
glVertex2iv (p3);
glVertex2iv (p4);
glVertex2iv (p5);
glEnd ( );



Case 3: GL_LINE_LOOP:

Successive vertices are connected using line segments to form a closed path or loop i.e., final vertex is connected to the initial vertex.

glBegin (GL_LINES_LOOP);

glVertex2iv (p1);

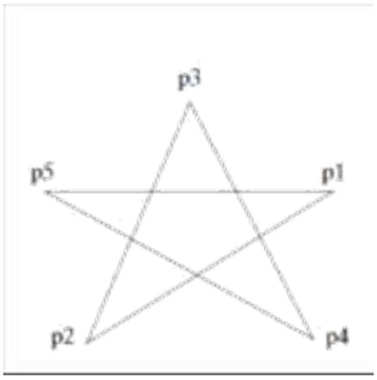
glVertex2iv (p2);

glVertex2iv (p3);

glVertex2iv (p4);

glVertex2iv (p5);

glEnd ();



3 (a)	With a neat diagram, explain Bresenham's line drawing algorithm for slope >1	5M	CO1	L2

<u>Bresenham's Algorithm:</u>

- ➔ It is an efficient raster scan generating algorithm that uses incremental integral calculations
- ➔ **To illustrate Bresenham's approach, we first consider the scan-** conversion process for lines with positive slope less than 1.0.
- ➔ Pixel positions along a line path are then determined by sampling at unit x intervals. Starting from the left endpoint (x0, y0) of a given line, we step to each successive column (x position) and plot the pixel whose scan-line y value is closest to the line path.
- ➔ Consider the equation of a straight line y=mx+c where m=dy/dx

<u>Bresenham's Line-Drawing Algorithm for |m| > 1.0</u>

1. Input the two line endpoints and store the left endpoint in (x0, y0).
2. Set the color for frame-buffer position (x0, y0); i.e., plot the first point.
3. **Calculate the constants Δx, Δy, 2Δy, and 2Δy − 2Δx, and obtain the starting value for**

the decision parameter as

$$p_0 = 2\Delta x - \Delta y$$

4. At each xk along the line, starting at k = 0, perform the following test:

If pk < 0, the next point to plot is (xk + 1, yk) and

$$p_{k+1} = p_k + 2\Delta x$$

Otherwise, the next point to plot is (xk + 1, yk + 1) and

$$p_{k+1} = p_k + 2\Delta x - 2\Delta y$$

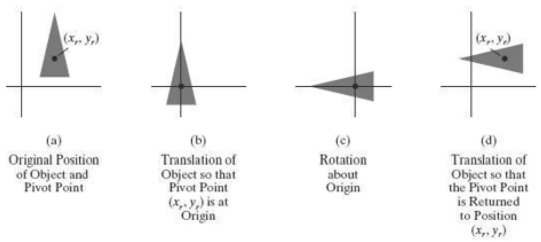
5. Repeat step 4 **Δy − 1 more times.**

(b)	Trace points generated by Bresenham's line drawing algorithm for k, p_k and (x_{k+1}, y_{k+1}) for the coordinates (2,2) to (10,7). Plot the points generated.	5M	CO1	L3

k	P_k	(x_{k+1}, y_{k+1})	P_k
0	2	(3, 3) ✓	
1	−4	(4, 3) ✓	2 + 10 − 16 = −4
2	6	(5, 4) ✓	−4 + 10 = 6
3	0	(6, 5) ✓	6 + 10 − 16 = 0
4	−6	(7, 5) ✓	0 + 10 − 16 = −6
5	4	(8, 6) ✓	−6 + 10 = 4
6	−2	(9, 6) ✓	4 + 10 − 16 = −2
7	8	(10, 7) ✓	−2 + 10 = 8

4 (a)	Explain with neat diagrams, rotation about a pivot point and the derivation for the composite 3x3 homogeneous matrix thereof.	5M	CO2	L2

General Two-Dimensional Pivot-Point Rotation



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

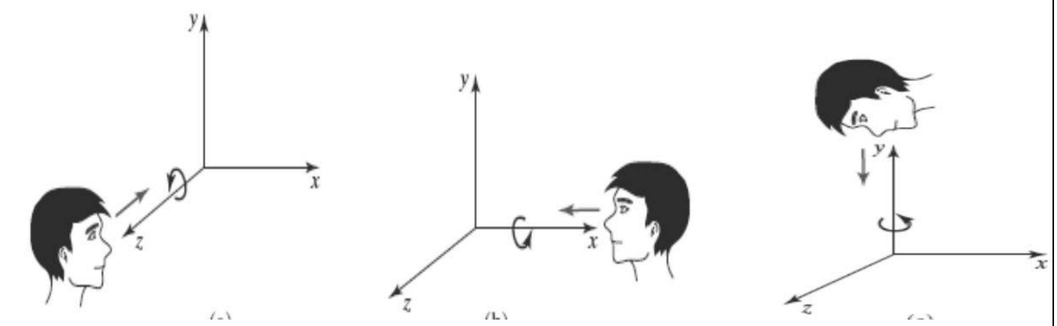
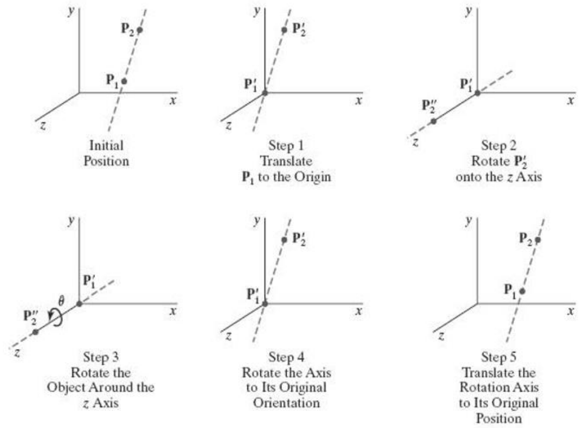
$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r \sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r \sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

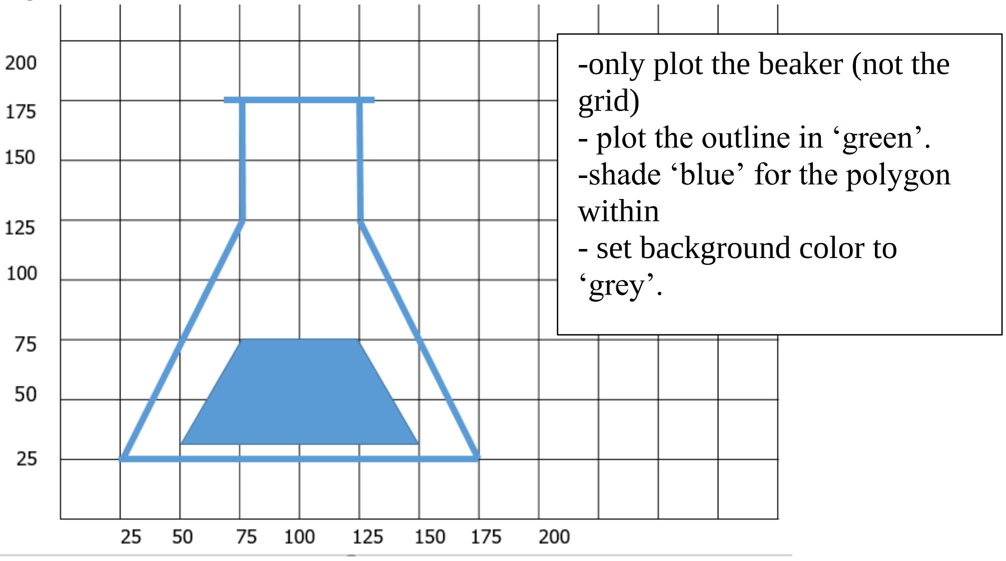
which can be expressed in the form

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

where $\mathbf{T}(-x_r, -y_r) = \mathbf{T}^{-1}(x_r, y_r)$.

(b)	Consider the line from (6,2) to (6,6). What are the coordinates of the line when sx = sy = 0.5. Plot the line and coordinates. Show the 3x3 homogenous matrix calculations. Does the line move toward or away from the coordinate origin?	5M	CO2	L3

5 (a)	Explain 3D Rotations transformation with relevant transformation matrix.  • **Along z axis:** $x' = x\cos\theta - y\sin\theta$ $y' = x\sin\theta + y\cos\theta$ $z' = z$ $\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$ • **x**→ *y*→ *z*→ *x* ***Along z axis*** $x' = x\cos\theta - y\sin\theta$ $y' = x\sin\theta + y\cos\theta$ $z' = z$ ***Along x axis*** $y' = y\cos\theta - z\sin\theta$ $z' = y\sin\theta + z\cos\theta$ $x' = x$ ***Along y axis*** $z' = z\cos\theta - x\sin\theta$ $x' = z\sin\theta + x\cos\theta$ $y' = y$	5 M	CO 2	L2
(b)	Design transformation matrix to rotate a 3D object about an axis that is not parallel to one of the coordinate axis.  1. Translate the object so that the rotation axis passes through the coordinate origin. 2. Rotate the object so that the axis of rotation coincides with one of the coordinate axes. 3. Perform the specified rotation about the selected coordinate axis. 4. Apply inverse rotations to bring the rotation axis back to its original orientation.	5 M	CO2	L2

	5. Apply the inverse translation to bring the rotation axis back to its original spatial position. $R(\theta) = T^{-1} \cdot R_x^{-1}(\alpha) \cdot R_y^{-1}(\beta) \cdot R_z(\theta) \cdot R_y(\beta) \cdot R_x(\alpha) \cdot T$			
6	Write an open GL program using OpenGL primitives to display the following. Code it in a function call beaker(). Write the full openGL code.Choose coordinate system of 200x200 with coordinate origin at bottom left.  	10 M	CO2	L3

CI **CCI** **HOD**