Name : Arpan Adhikari
Roll. 231309

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

Assignment 7

Q1) What are the common features of every graphical file format ? Explain any one of the Graphics file format that you are familiar with

→

Some common features of graphical formats are :

(i) Compression
Many formats use lossless (PNG) or lossy (JPEG) compression to reduce the file size.

(ii) Color Depth
(iii) Transparency Support
(iv) Metadata
(v) Scalability
(vi) Support for Animation

PNG (Portable Network Graphics) file format :

PNG is a popular raster image format known for its lossless compression and transparency support.

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

Key features of PNG:
1. Lossless compression
2. Alpha Transparency
3. Supports High Color Depth
4. Better for Graphics and web Use

Q2) Why is it required to follow Graphics standard? Explain any one of the graphics standard that is used for display purpose. what is Language Binding?

→

It is required to follow graphics standard because it helps in
(i) Interoperability
(ii) Portability
(iii) Optimization
(iv) Consistency
(v) Development Efficiency

A common graphics standard is GKS (Graphical Kernel system).

GKS is an early graphics standard developed for 2D graphics programming. It provides a device independent way to create graphical applications.

Key features:
1. Device Independence
2. standardized API

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

3. Support for Multiple Output Devices

Language Binding :-

Language Binding is the process of making a library or API, written in one programming language, usable in another language. It acts as bridge between different programming languages and the graphics library.

For e.g.: OpenGL has bindings for C, C++, Python and Java, allowing developers to use OpenGL functions in their preferred Language.

Q3) Explain the need for machine independent graphical languages.

→

We need machine independent graphical language because:

1. Cross Platform Compatibility
2. Device Independence
3. Standardization in Graphics Development
4. Interoperatability Between systems
5. Efficient Development and Maintenance

Q

Date :
Page :

Q4) Explain the architecture of OpenGL along with the API's used for graphical rendering of objects.

→

OpenGL follows a client-server architecture, where the application (client) sends rendering commands to the OpenGL pipeline (server) for processing.

~~The pipeline~~
Key stages of OpenGL Architecture:

1. Application Stage
- The user's program generates rendering commands using OpenGL API functions.
- Handles input, scene setup and transformations.

2. Geometry Processing Stage
- Converts objects into primitives (points, lines, triangles)
- Performs transformations, lighting and clipping.

3. Rasterization stage
- Converts geometric data into pixels for display.
- Generates fragments (potential pixels) with interpolated colors and textures.

4. Fragment Processing Stage
- Applies shading, texturing and blending to fragments.
- Determines the final colour of each pixel.

Date :
Page :

5. Framebuffer stage
- Processes and stores the final rendered image in the framebuffer for display.

OpenGL API functions for rendering Objects:

1. Object Creation & Transformation
glVertex3f(x,y,z) → Defines vertices of an object
glTranslatef(dx,dy,dz) → Moves an object in 3D space
glRotatef(angle,x,y,z) → Rotates an object around an axis

2. Drawing Primitives
glBegin(GL_TRIANGLES) → starts defining a triangle
glEnd() → Ends drawing the primitive

3. Color & shading
glColor3f(r,g,b) → Sets object color
glEnable(GL_LIGHTING) → Enables lighting effects

4. Texture Mapping
glBindTexture(GL_TEXTURE_2D, textureID) → Binds a texture to an object

5. Rendering & Display
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
→ clears buffer before rendering
glFlush() → Executes openGL commands immediately.

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

Q5) Explain the GLU, GLUT and GL categories of APIs used in OpenGL for rendering Graphical Objects.

→

GL (OpenGL core library)
- Provides the fundamental functions for graphiss rendering
- Key features:
1. Low level API that directly interacts with the GPU
2. Supports primitive rendering (points, lines, triangles, polygons)
3. Provides functions for transformations, shading and texture mapping.
- E.g.:

```
glBegin(GL_TRIANGLES)
glColor3f(r,g,b)
glVertex3f(x,y,z)
```

GLU (OpenGL Utility Library)
- simplifies complex operations like matrix transformations, camera setup and curve generation.

- Key Features:
1. Built on top of the core OpenGL functions.
2. Provides higher level abstraction to make programming clear.
3. Commonly used for projections, viewing transformations, and curves.

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

E.g.
gluPerspective (fov, aspect, near, far)
gluLookAt (eyeX, eyeY, eyeZ, centerX, centerY, centerZ, upX, upY, upZ)

#GLUT (OpenGL utility Toolkit)
- Handles window management, user input and event handling for OpenGL applications.

- key features:
1. Provides a cross platform way to create, and manage OpenGL windows.
2. Handles keyboard, mouse and window events.
3. supports basic 3D object rendering.

E.g.
glutInit (&argc, argv)
glutCreateWindow (":OpenGL window")
glutDisplayFunc (display)

Q6) what are the different types of data structures that can be used for representing graphical objects?

→
The types of data structures are:

1. Arrays
Used to store vertices, colors, and texture coordinates efficiently.

Date :
Page :

2. Linked lists
- Used for storing and managing hierarchial objects in scene graphs.

3. Graphs
Represents complex relationships between graphical elements

4. Trees
Useful for hierarchial modeling of objects

5. Meshes (Graphs + Arrays)
Represents 3D surfaces using polygons (triangles, quads etc.)

6. Spatial Data Structures
Used for efficient rendering and collision detection in large scenes.

Q7) Explain different types of color models.
→

RGB Color model
Type : Additive Color model
Used In : Digital screens (monitors, TVs, projectors, mobile displays)

concept :
- RGB is a light based color model where colors are created by adding different intensities of red, green and blue light.

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

- All at full intensity → white light
- No light present → Black

Key Properties:
→ Additive mixing
→ Used in screens
→ 24-bit color Depth

CMYK color model
Type : subtractive color model
Used In : Printing (books, magazines)

- CMYK is an ink-based color model where colors are created by subtracting different amounts of cyan, magenta, and yellow ink from white paper.
- Black (K) is used to improve depth and contrast because mixing CMY alone donot create a pure black.

Key Properties :
→ subtractive mixing
→ Used in printing
→ Four color Process

8. Why does a Printing-Press make use of CMYK colour model? What are the characteristics of CMYK & RGB color Model?

⇒ A printing press uses the CMYK (Cyan, Magenta, Yellow & Black) color model because it is a subtractive color model designed specially for print media. Unlike digital screens which emit light, printed materials absorb & reflect light. It works by layering ink on a white surface, subtracting certain wavelengths of light to produce colours. It is ideal for printing books, newspapers, posters & paging since it ensures accurate color representation/reproduction on paper.

Characteristics of CMYK & RGB color Model are:-

S.N	CMYK	RGB
1.	It is a subtractive color model.	It is additive color model.
2.	Its primary color ~~red~~ are cyan, magenta, yellow, black.	Its primary color are red, green & ~~black~~ blue.
3.	Used in printing (paper, banners, magazines etc).	Used in digital screens (TVs, monitors, mobile device etc).
4.	It absorbs light ~~to emit~~ & reflects color.	It emits light to create color.
5.	It has limited color shades ie: smaller gamut.	It has larger color shades ie: larger gamut.

PUSPANJALI
Date / /
Page

What is a call back function in openGL? How are they used?

A callback function in openGL is a function that is registered & automatically invoked by the system (ie: openGL library) in response to specific events. These functions help manage rendering, user ip, windw events & error handling. They are used to execute code in response to an event like a mouse click, keyboard press etc.

Some callback functions are:-

glutReshapeFunc(void (* func)
glut.KeyboardFunc( ) // detect keypress
glutMouseFunc( ) // detect mouse clicks/movements
glutIdleFunc( )

Call back function are used by:-

1) Registering a callback function
To use callback function in openGL, we register it using an appropriate function provided by GLUT.

glutdisplayFunc( ) // registers the display fun^n to handle windw drawing.
glutkeyboardfunc( ) // registers the keyboard func^n to detect key presses.
glutReshapeFunc( ) // ensures the viewport resizes properly.

10. What is the difference between PHIGS, GKS & OpenGL?

⇒ They are:-

S.N	PHIGS	GKS	OpenGL
1.	Programmers Hierarchical Interactive Graphics System)	Graphical kernal System.	Low-level Graphics API open Graphics Library.
2.	It is Hierarchical Graphics System.	It is standard Graphics API.	It is low level Graphics API.
3.	Developed by ANSI & ISO.	Developed by ISO.	Developed by Silicon Graphics Inc.
4.	Primarly used in 3D graphics with hierarchical structure.	Primarly used in basic 2D graphics.	Primarly used in High-performance 2D & 3D rendering.
5.	Performance is slower due to structure traversal.	It is also mainly for static graphics.	Faster due to hardware acceleration.

11. Explain the graphical rendering pipeline used in openGL.

⇒ They are:-

a) Vertex Processing

Each vertex in the vertex array is processed in turn into an output vertex. It then goes through tessellation stages & geometry shader primitive processing for producing a sequence of primitives.

PUSPANJALI
Date / /
Page

Vertex Post-Processing

Outputs of the last stage are adjusted or shipped to different locations. Transform feedback happens here like Primitive Assembly, Primitive clipping, the perspective dive & the viewport transformation to windas space.

Scan conversion & primitive parameter interpolation which generates a number of fragments.

A fragment shader processes each fragment. Each fragment generates a number of o/p.

Pre-sampling processing.

12. Explain the APIs use in openGl for:

1) Projection

The ~~Glut~~ gluLookAt function creates a viewing matrix derived from an eye point, a reference point indicating the center of the scene & an up vector.

⇒ void gluLookAt(Gldouble eyex, Gldouble eyey, Gldouble eyez, Gldouble centerx, Gldouble centery, Gldouble centerz, Gldouble upx, Gldouble upy, Gldouble upz);

where, eyex, eyez, eyez → position of the eye point.
centerx, centery, centerz → position of the reference point.
upx, upy, upz → direction of the up vector.

ii) Lighting
glLightfv sets the values of individual light source parameters.

glLightfv(GL-LIGHT0, GL-POSITION, position);
void glLightfv(GLenum light, GLenum pname,
const GLfloat * params);

iii) Input handling
for keyboard:-

void keyboard(unsigned char key, int x, int y)
{
if (key==27) ex
exit(0);
}
glutKeyboardFunc(keyboard);

for mouse:-

void mouse(int button, int state, int x, int y)
{
if (button == GLUT-LEFT-BUTTON && state == GLUT-DOWN)
{ //...
}
}

glutMouseFunc(mouse);

iv) Performing Basic Transformations

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(x,y,z);  // move object
glRotatef(angle,x,y,z); // rotate object
glScalef(Sx,Sy,Sz);  // scale object
```

v) Drawing Basic Output Primitives

Primitives like points, lines & polygons form the basic shapes in openGL.

```
glBegin(GL_TRIANGLES); // start defining a shape
glVertex3f(0.0,1.0,0.0);  // 1st vertex
glVertex3f(-1.0,-1.0,0.0);  // 2nd vertex
glVertex3f(1.0,-1.0,0.0);  // 3rd vertex
glEnd();  // ends the primitive drawing
```