Introduction to OpenGL and GLUT

What is OpenGL?
- An application programming interface (API)
- A (low-level) Graphics rendering API
- Generate high-quality color images composed of geometric and image primitives

Maximal Portability
- Display device independent
- Window system independent
- Operating system independent

Without a standard API (such as OpenGL) - impossible to port

- Line(100,50,150,80) - device/lib 1
- Moveto(100,50)
- Lineto(150,100) - device/lib 2
OpenGL Basics

- OpenGL’s primary function - Rendering
- Rendering? - converting geometric/mathematical object descriptions into frame buffer values
- OpenGL can render:
  - Geometric primitives
  - Bitmaps and Images (Raster primitives)

Code Example

```c
void Display()
{
  glClear(GL_COLOR_BUFFER_BIT);
  glColor4f(1.0, 0.0, 0.0);
  glBegin(GL_POLYGON);
  glVertex2f(-0.5, -0.5);
  glVertex2f(-0.5, 0.5);
  glVertex2f(0.5, 0.5);
  glVertex2f(0.5, -0.5);
  glEnd();
  glFlush();
}
```

Specifying Geometric primitives

- Primitives are specified using
  ```c
  glBegin(primitive);
  // define your primitives here
  ...
  glEnd();
  ```
- `primitive`: GL_POINTS, GL_LINES, GL_TRIANGLES, GL_QUADS, ...

Primitive Types
Sample Example

```c
void DrawQuad(GLfloat color[])
{
    glColor3f(0.0, 1.0);
    glBegin(GL_QUADS);
    glVertex2f(0, 0);
    glVertex2f(1.0, 0);
    glVertex2f(1.0, 1.0);
    glVertex2f(0.0, 1.0);
    glEnd();
}
```

OpenGL Command Formats

- `glVertex2f(x, y)`
- Add 'v' for vector form
- `glVertex2fv(v)`

Shape Example

Window-based programming

- Most of the modern graphics systems are window-based
Window system independent

- OpenGL is window system independent
  - No window management functions – create windows, resize windows, event handling, etc
  - This is to ensure the application’s portability
  - Create some headache though – just a pure OpenGL program won’t work anywhere.

More APIs are needed

- X window system: GLX
- Apple Macintosh: AGL
- Microsoft Windows: WGL

These libraries provide complete functionality to create Graphics User Interface (GUI) such as sliders, buttons, menus etc.

Problem – you need to learn and implement them all to write a true portable software

Use GLUT (OpenGL Utility Toolkit)

- For fast prototyping, we can use GLUT to interface with different window systems
- GLUT is a window independent API – programs written using OpenGL and GLUT can be ported to X windows, MS windows, and Macintosh with no effort
- GLUT does not contain all the bells and whistles though (no sliders, no dialog boxes, no menu bar, etc)

GLUT Basics

Program Structure
1. Configure and open window (GLUT)
2. Initialize OpenGL (Optional)
3. Register input callback functions (GLUT)
   - Render
   - Resize
   - Input: keyboard, mouse, etc
4. Enter event processing loop (GLUT)
Sample Program

```c
#include <GL/glut.h>
#include <GL/gl.h>

Void main(int argc, char** argv)
{
    int mode = GLUT_RGB|GLUT_SINGLE;
    glutInitDisplayMode(mode);
    glutInitWindowSize(500,500);
    glutCreateWindow(argv[0]);
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

Specify the display Mode - RGB or color Index, single or double Buffer

Create a window Named "simple" with resolution 500 x 500

Your OpenGL initialization code (Optional)
Sample Program

```c
#include <GL/glut.h>
#include <GL/gl.h>

void main(int argc, char** argv)
{
    int mode = GLUT_RGB | GLUT_SINGLE;
    glutInitDisplayMode(mode);
    glutInitWindowSize(500, 500);
    glutCreateWindow("Simple");
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMainLoop();
}
```

Callback functions?

- Most of window-based programs are event-driven
  - which means do nothing until an event happens, and then execute some pre-defined functions
- Events - key press, mouse button press and release, window resize, etc.

void display() - the function you provide. It contains all the OpenGL drawing function calls and will be called when pixels in the window need to be refreshed.
And many more ...

- glutKeyboardFunc() – register the callback that will be called when a key is pressed
- glutMouseFunc() – register the callback that will be called when a mouse button is pressed
- glutMotionFunc() – register the callback that will be called when the mouse is in motion while a button is pressed
- glutIdleFunc() – register the callback that will be called when nothing is going on (no event)

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

Void main(int argc, char** argv) {
    int mode = GLUT_RGB|GLUT_SINGLE;
    glutInitDisplayMode(mode);
    glutInitWindowSize(500,500);
    glutCreateWindow("Simple");
    init();
    glutDisplayFunc(display);
    glutReshapeFunc(resize);
    glutKeyboardFunc(key);
    glutMainLoop();
}

The program goes into a infinite loop waiting for events