INTRODUCTION TO OPENGL (1)
CONTENT

• What is OpenGL
• Getting Start
• Matrix Manipulation
• Drawing
• Lighting
WHAT IS OPENGL

• OpenGL is a platform-independent API
• Related utility libraries
  • OpenGL core
  • OpenGL Utility Library (GLU): higher-level drawing routines.
  • OpenGL Utility Toolkit (GLUT): I/O with host OS.
WHAT IS OPENGL

• OpenGL Core (GL: Graphics Library)
  • Mostly low level graphics functions.
  • Create vertices, matrix manipulation, position cameras, etc.
  • Prefixed with gl, e.g. glVertex(), glBegin()

```
glVertex3f(x, y, z)
```

- belongs to core GL library
- x, y, z are floats

```
glVertex3fv(p)
```

- p is a pointer to an array
WHAT IS OPENGL

• GLU (OpenGL Utility Library)
  • Slightly higher level.
  • Prefixed with glu, e.g. gluSphere(), gluLookAt()
  • Draws more complex shapes, e.g. cylinders, spheres, cones.
  • Tessellation

• GLUT (OpenGL Utility Toolkit)
  • Handles windowing API
  • OS-independent
  • Functions prefixed with glut, e.g. glutCreateWindow()
    glutPostRedisplay()
GETTING START

• Structure of a basic OpenGL program:
  • Create a window
  • Initialize OpenGL states:
    • Lighting, camera, etc.
  • Per frame display function
    • e.g. set local transformations, clear buffers, swap buffers, etc.
  • Handle UI
    • Keyboard, mouse input
  • Loop: keep calling display function
void main(int argc, char *argv[]) {
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize (500,500);
    glutInitWindowPosition (100,100);
    glutCreateWindow (argv[0]);

    glutIdleFunc (myIdle);
    glutDisplayFunc (myDisplay);
    glutReshapeFunc (myReshape);
    glutKeyboardFunc (myKeyboard);
    glutMouseFunc (myMouse);

    mglInit();
    glutMainLoop();
    return;
}
GETTING START

Programming in OpenGL

- State machine: It means that once the value of a property is set, the value persists until a new value is set.
- You set various properties via function calls, e.g. position/types of lights, the current material, the current modeling transformation, etc.
- When you make a call to draw some geometry, you enter a draw state, draw some geometry and then exit the draw state.
  - glBegin(GL_TRIANGLES)
  - glVertex3d(....);
  - glVertex3d(....);
  - glVertex3d(....);
  - glEnd();
MATRIX MANIPULATION

- Used to define transformations
- 3 matrix stacks:
  - GL_MODELVIEW (Scene graphs)
  - GL_PROJECTION
  - GL_TEXTURE
- The current matrix is the top of the stack
  - Initialized to the identity matrix
MATRIX MANIPULATION

• Modifying matrices:
  • Select matrix stack:
    • `glMatrixMode(GL_MODELVIEW/PROJECTION/TEXTURE)`
  • Stack operations: changes current.
    • `glPushMatrix()`
    • `glPopMatrix()`
  • Loading matrices: overwrite current
    • `glLoadIdentity()`
    • `glLoadMatrix()`
  • Matrix operations: manipulate current
    • `glMultMatrix()`
    • `glTranslate()`, `glRotate()`, `glScale()`
MATRIX MANIPULATION

- **GL_PROJECTION**
  - Sets the 3D to 2D matrix
  - Usually only set it once.
  - Helper functions:
    - `gluPerspective()`
    - `glFrustum()`
    - `glOrtho()`
MATRICES MANIPULATION

- **GL_MODELVIEW:**
  - Sets the viewpoint. Camera setup.
    - `gluLookAt(eyeXYZ,centerXYZ,upXYZ)`
  - Object transformations:
    - `glTranslatef()`
    - `glScalef()`
    - `glRotatef()`
  - All transformations are based on the origin of the coordinate.
DEMO
DRAWING

• Primitives are drawn by declaring vertices:
  • `glBegin(primitive type)`
  • `glVertex3f(...) ... glEnd()`

• Can also declare normal, color and texture info for (BEFORE!) each vertex.
  • `glNormal()`, `glColor()`, `glTexCoords()`
DRAWING

• Primitive types:
  • GL_POINTS
  • GL_LINES
  • GL_TRIANGLES
  • GL_TRIANGLE_STRIP
  • GL_QUADS
  • GL_POLYGON

• More complex primitives:
  • gluCylinder()
  • gluDisk()
  • gluSphere()
DEMO
LIGHTING

• Lighting
  • glLightfv(GL_LIGHT0, param, val)
    • Total 8 lights, GL_LIGHT0, GL_LIGHT1...
    • GL_AMBIENT/DIFFUSE/SPECULAR/POSITION(x,y,z,1/-1)
    • GL_SPOT*, GL_*_ATTENUATION

• Materials
  • glMaterialfv(GL_FRONT, param, val)
    • GL_AMBIENT/DIFFUSE/SPECULAR/EMISSION
    • GL_SHININESS (specular falloff)

• Set the material parameters then draw.
USEFUL LINKS

- http://www.glprogramming.com/red/
- http://www.glprogramming.com/blue
QUESTION?