

# Open GL Hands-On Workshop

OpenGL is the premier environment for developing portable, interactive 2D and 3D graphics applications. Since its introduction in 1992, OpenGL has become the industry's most widely used and supported 2D and 3D graphics application programming interface (API), bringing thousands of applications to a wide variety of computer platforms. OpenGL fosters innovation and speeds application development by incorporating a broad set of rendering, texture mapping, special effects, and other powerful visualization functions. Developers can leverage the power of OpenGL across all popular desktop and workstation platforms, ensuring wide application deployment.

Any visual computing application requiring maximum performance—from 3D animation to CAD to visual simulation—can exploit high-quality, high-performance OpenGL capabilities. These capabilities allow developers in diverse markets such as broadcasting, CAD/CAM/CAE, entertainment, medical imaging, and virtual reality to produce and display incredibly compelling 2D and 3D graphics.

Advance topics like OPENGL ES and Stereographic will be covered on the last day however these are just introductory topics full functionality/labs depend on availability of hardware/drivers else will be done as lecture alone topics.

- **DAY 1**
- Getting Started
  - Setting up the Development environment
- Introduction to 3-D Graphics
  - Basic Terminology
  - 3-D Graphics Pipeine
- OpenGL API
  - What is OpenGL
  - Brief History and Evolution of API
  - OpenGL Standard libraries and headers
  - Naming conventions
  - OpenGL Rendering Pipeline
  - OpenGL State machine
  - Hardware acceleration vs software
  - OpenGL vs DirectX
  - Structure of OpenGL Program
  - Simple OpenGL Program
- Introduction to GLUT
  - GLUT Standard headers and libraries
  - Window management
  - Mouse handling
  - Keyboard handling
- Basic Animation
  - Double buffering
  - Timers
- Drawing Basics
  - 2-D Coordinate System
  - 3-D Coordinate System
  - Drawing States
  - Normalized Coordinates
- Drawing Primitives
  - Points

- Lines
- Triangles
- Polygons
- Display Lists
- **DAY 2**
- Alternative ways of passing geometry to OpenGL
  - Vertex Arrays
  - Buffer Objects
  - Vertex Buffer Objects
- Viewing
  - Viewing and Modelling Transformations
  - Projection Transformation
  - Viewport Transformation
  - Clipping planes
  - Hidden Surface Removal
  - Culling
- Colors, Material & Lighting
  - RGBA vs Color Index
  - Color Shade model
  - Defining Material properties
  - Lighting Basics
  - Light Models in OpenGL
  - Creating light sources
- Images
  - Imaging Pipeline
  - Bitmaps
  - Pixmaps
- Alpha Blending & Fog
  - Blending Basics
  - Blending Equation
  - Fog
- **DAY 3**
- Texture Mapping
  - Texture basics
  - Loading Textures
  - Texture Filtering
  - Texture Objects
  - Mapping Texture to geometry
- Advanced Texture Mapping
  - Multitexturing
  - Depth Textures
  - Cube Mapping
  - Point Sprites
- Interactive Graphics

- Rendering Modes
  - Selecting and Picking Objects on screen
  - Feedback rendering mode
- Fonts
- Anti-Aliasing
  - Basics
  - Anti-aliasing points and lines
  - Anti-aliasing polygons
  - Multi sampling
- **DAY 4**
- Programmable Pipeline and GL Shading Language (GLSL)
  - Introduction to Programmable OpenGL Pipeline
  - Comparing Fixed function and programmable pipeline
  - OpenGL Shader programming model
  - Introduction to GLSL
  - Simple Shader example
- Vertex Shader
  - Vertex Shader basics
  - Customized Vertex Transformation
  - Lighting
- Fragment Shader
  - Fragment shader basics
  - Manipulate Color
  - Image processing
- **DAY 5**
- Framebuffer
  - Components of Framebuffer
  - Stencil Buffer
  - Accumulation Buffer
- Pixel Buffer Objects
  - Why PBOs
  - Using Pixel Buffer Objects
- Frame Buffer Objects
  - Introduction to Offscreen Rendering
  - Using Frame Buffer Objects
  - Rendering on Textures
- Debugging OpenGL
- OpenGL ES-fundamentals
  - Introduction to OpenGL ES
  - OpenGL ES 1.x vs OpenGL 2.0
  - Introduction to EGL
  - Examples - OpenGL ES 1.x
  - Examples – OpenGL ES 2.0
- StereographicsFundamentals

- (subject to suitable HW /Graphics Driver available)
- Introduction
- Stereo display technologies
- Camera model for Stereo
- Stereographics using OpenGL
- Example

\* \* \* \* \*