Introduction to OpenGL

CSCI 4229/5229 Computer Graphics Summer 2025

History of Graphics Libraries

- PLOT-10: Tektronix 4010 graphics
- PLOT88: PC graphics
- DISSPLA: NCAR graphics
- GINO: Portable graphics
- DIGLIB: LLNL device-independent, open source
- GKS: Graphics Kernel System (2D vector)
- PHIGS: 3D Interactive Graphics
- OpenGL & DirectX

The rise of OpenGL

- Originated as SGI IrisGL
- Vendor-neutral OpenGL controlled by ARB
- Hides the details of hardware
  - Software emulation when necessary
  - Hardware acceleration when possible
- Supports 2D to advanced 3D graphics
- Portable to most hardware and OS with WGL, AGL and GLX
- Hardware range from phones to Big Iron

Focus of OpenGL

- OpenGL 1 (1992)
  - Hardware abstraction
- OpenGL 2 (2004)
  - Add Shaders (Programming the GPU)
- OpenGL 3 (2008)
  - Focus on shaders and new hardware
  - Deprecates many features
- OpenGL 4 (2010)
  - Core & Compatibility Profiles
  - Merge desktop and embedded versions

Gaming and Graphics

- Text based/ASCII graphics (Pong, PacMan)
- 2D monochrome line graphics (Astroids)
- 2D images & sprites (Mario)
- 3D graphics
  - Flight Simulators (2D -> 3D)
  - First Person Shooters
  - Multi-player games
- Games push the envelope
  - Realism
  - Speed

OpenGL by Example

- Learn OpenGL by reading
- nehe.gamedev.net
- lighthouse3d.com
  - Excellent free tutorials
- OpenGL: A Primer (3ed) by Edward Angel
  - Short and sweet
- OpenGL Programming Guide (Vermillion Book)
  - Free older editions as PDF
- OpenGL Superbible
  - Theory and Applications

What is OpenGL?

- Sometimes called a library, actually an Application Programming Interface (API)
- Specification is controlled by Kronos
- Multiple implementations by different vendors
  - Mesa & FreeGLUT free implementations
- OpenGL just does real time graphics
  - Need GLX/WGL/AGL for windowing and input
  - Limited font support (in GLUT)
  - No sound, printing, etc. support

OpenGL Versions

1.0 Initial release (1992)

1.1 Major upgrade (1997)
  - Latest version on some Windows system

1.2 Improves textures (1998)

1.3-1.5 Incremental improvements (2001-2003)

2.0 Relaxes restrictions, adds shader (2004)

2.1-2.3 Incremental improvement (2006-7)

3.0 Support advanced hardware features (2008)

3.1-3.3 Improved shaders (2009)

4.0 Merge desktop and devices (2010)

4.1-4.6 Additional shaders

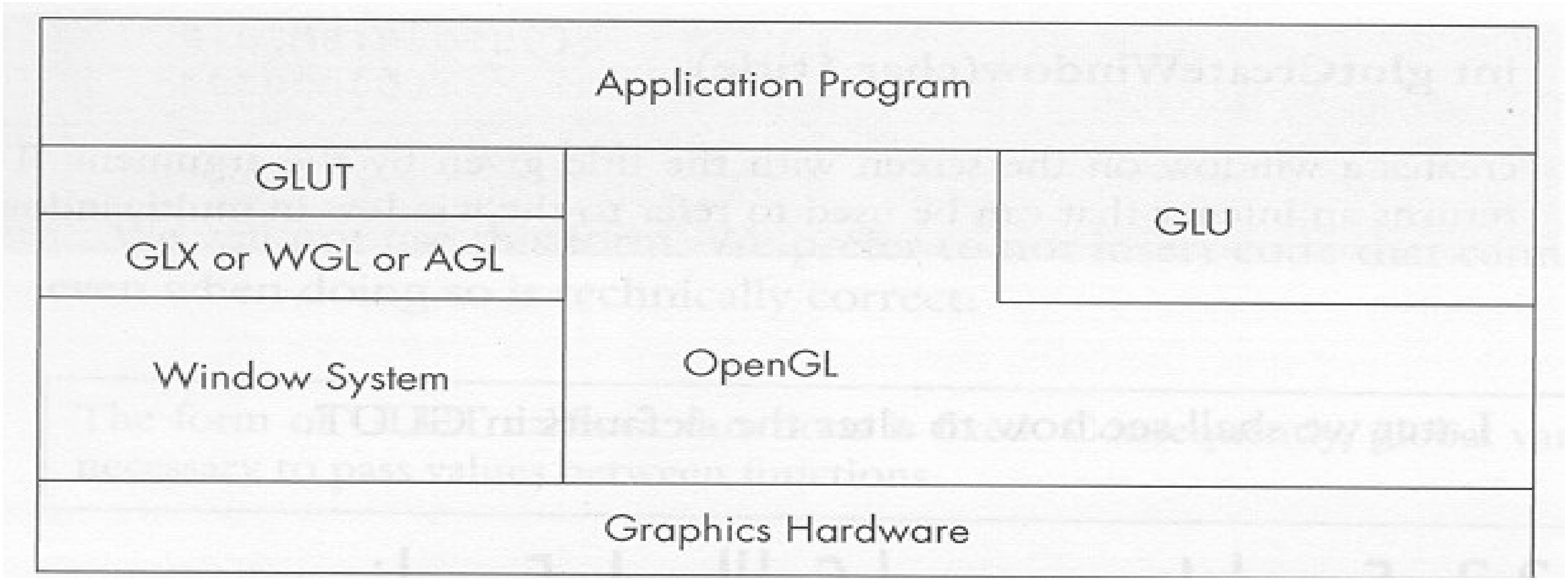
OpenGL Deprecation

- I will mostly use OpenGL 2.0
  - Feature rich, flat learning curve
  - I will use GL3 or GL4 only as needed
- OpenGL Core Profile concentrates on rendering
  - Improved execution time performance
- User must provide deprecated functionality
  - Steepens the learning curve
  - Deprecated features in Compatibility Profile
  - Increases reliance on third party libraries
  - Adds development time until tools mature

OpenGL APIs

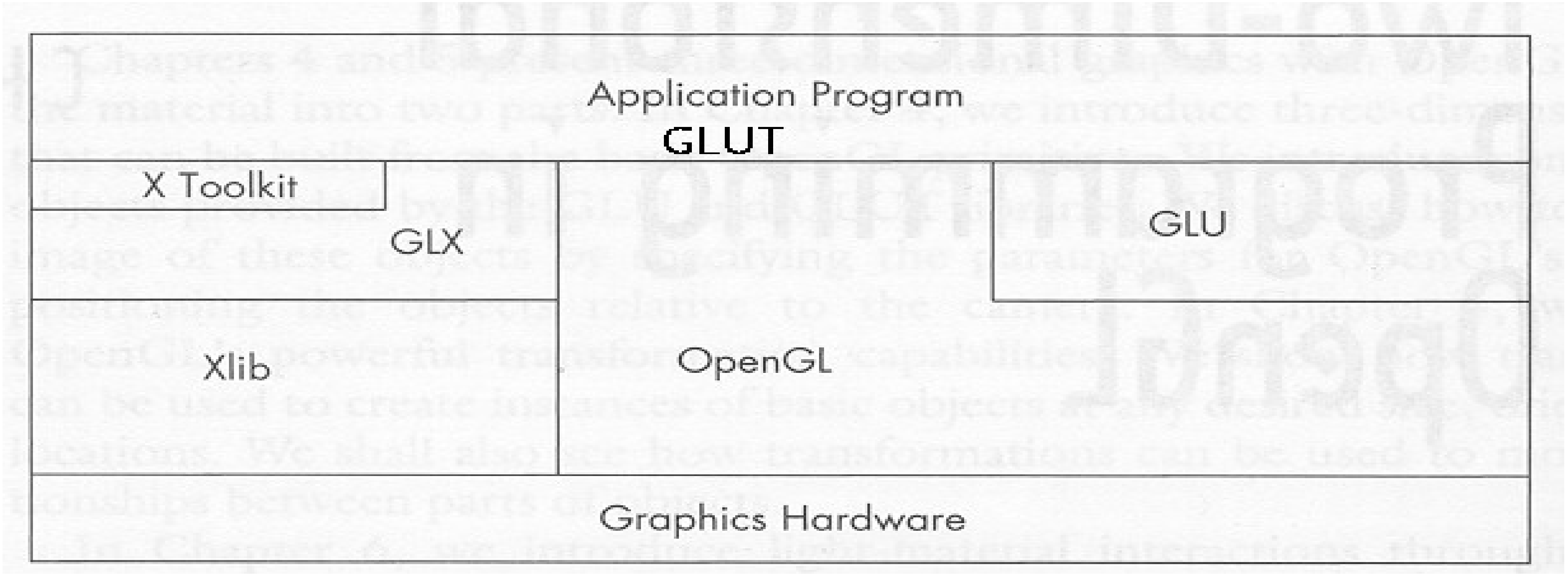
- Languages
  - C, C++, C#
  - FORTRAN
  - Java
  - Perl
  - Python
  - Ada
- Packages
  - Qt (QOpenGLWidget)
  - SDL, glfw, etc
  - Many others

OpenGL and Friends



From *OpenGL: A Primer*

OpenGL on X11



From *OpenGL: A Primer*

GLU: OpenGL Utility

- Higher Level and Convenience Functions
  - Projections
  - Creating texture maps
  - NURBS, quadrics, tessalation
  - Predefined objects (sphere, cylinder, teapot)
- Collections of calls for convenience
- Standard with all OpenGL implementations

GLUT: GL Utility Toolkit

- Provides access to OS and Window System
  - Open windows and setting size and capabilities
  - Register and triggers callbacks
  - Keyboard and mouse interaction
  - Elementary fonts
- Not part of OpenGL, but provides a portable abstraction of the OS
  - FreeGLUT
  - OpenGLUT
- Alternatives: SDL, Qt, glfw, …

Header Files and Libraries

- Usually you only need
  - #define GL_GLEXT_PROTOTYPES
  - #include <GL/glut.h>
- Header file locations
  - /usr/include/GL on most systems
- Linking may only need
  - -l glut -l GLU -lGL
- Special cases
  - OS/X separates GL and GLUT
  - Windows differs depending on the compiler

OpenGL Naming Convention

- glSomethingNt()
  - *Something* is the name of the function
  - N is 2 or 3 or 4 for the dimension
  - t is for the the variable type

b	GLbyte	(signed char)	8 bit
s	GLshort	(signed short)	16 bit
i	GLint	(signed int)	32 bit
ub	GLubyte	(unsigned char)	8 bit
us	GLushort	(unsigned short)	16 bit
ui	GLuint	(unsigned int)	32 bit
f	GLfloat	(float)	32 bit
d	GLdouble	(double)	64 bit

OpenGL Naming Example

- Vertex
  - glVertex3i(0 , 0 , 1)
  - glVertex2d(27.34 , 88.12)
  - glVertex3dv(array)
- Few functions return a value
- Most functions created by name mangling
- Constants are GL_SOMETHING
- Variable types are GLsomething

GLUT and GLU Naming

- Functions
  - glutSomething
  - gluSomething
- Constants
  - GLUT_SOMETHING
  - GLU_SOMETHING
- You can always tell by the name which API supplies a function or constant
- Avoid things starting with glx, wgl & agl

GLUT: GL Utility Toolkit

- Supplies interface to OS
  - Windowing
  - Interaction
- Hello World in GLUT (well sorta)

```
int main(int argc,char* argv[ ])
{
   glutInit(&argc,argv);
   glutCreateWindow(“Hello World”);
   glutDisplayFunc(display);
   glutMainLoop();
}
```

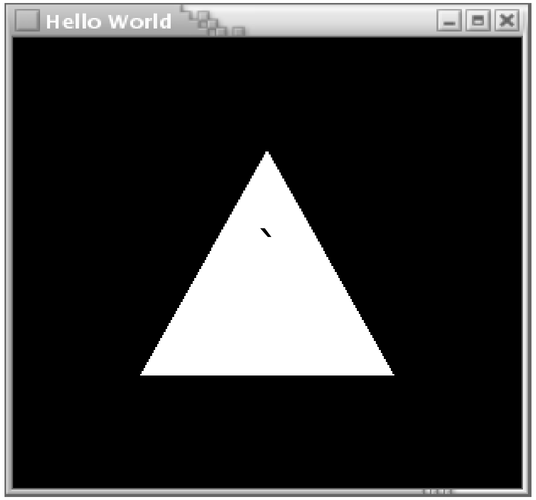
Completing Hello World

- Draw a triangle

```
#include <GL/glut.h>
void display()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glBegin(GL_POLYGON);
   glVertex2f(0.0,0.5);
   glVertex2f(0.5,-0.5);
   glVertex2f(-0.5,-0.5);
   glEnd();
   glFlush();
}
```

Compile, link and run

- gcc -Wall -o ex1 ex1.c -lglut -lGL
- Heavily relies on defaults
  - Window
  - Viewport
  - Projection
  - Color


Hello World