CSCI 420 Computer Graphics
Lecture 2

Introduction to OpenGL

OpenGL API
Core and Compatibility Profiles
Colors
[Angel Ch. 2]

Jernej Barbic
University of Southern California

What is OpenGL

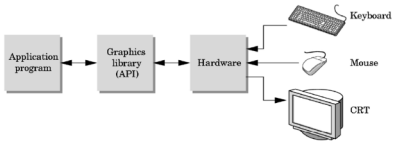
- A low-level graphics library (API) for 2D and 3D interactive graphics.
- Descendent of GL (from SGI)
- First version in 1992; now: 4.6 (July 2017)
- Managed by Khronos Group (non-profit consortium)
- API is governed by Architecture Review Board (part of Khronos)

Where is OpenGL used

- CAD
- Virtual reality
- Scientific visualization
- Flight simulation
- Video games

Graphics library (API)

- Intermediary between applications and graphics hardware



- Other popular APIs:
  Direct3D (Microsoft)
  OpenGL ES (embedded devices)
  X3D (successor of VRML)
  Vulkan (more low-level than OpenGL)

4

OpenGL is cross-platform

- Same code works with little/no modifications
- Windows: default implementation ships with OS
  Improved OpenGL: Nvidia or AMD drivers
- Linux: Mesa, a freeware implementation
  Improved OpenGL: Nvidia or AMD drivers
- Mac: ships with the OS. Apple announced deprecation in 2018, but OpenGL continues to work.

Choice of Programming Language

- OpenGL lives close to the hardware
- OpenGL is not object-oriented
- OpenGL is not a functional language (as in, ML)
- Use C to expose and exploit low-level details
- Use C++, Java, ... for toolkits
- Support for C in assignments

OpenGL is cross-platform

Include file (OpenGL Compatibility Profile) :

```
#if defined(WIN32) || defined(linux)
  #include <GL/gl.h>
  #include <GL/glu.h>
  #include <GL/glut.h>
#elif defined(__APPLE__)
  #include <OpenGL/gl.h>
  #include <OpenGL/glu.h>
  #include <GLUT/glut.h>
#endif
```

OpenGL is cross-platform

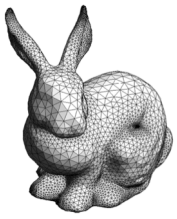
Include file (OpenGL Core Profile) :

```
#if defined(WIN32) || defined(linux)
  #include <GL/glew.h>
  #include <GL/glut.h>
#elif defined(__APPLE__)
  #include <OpenGL/gl3.h>
  #include <OpenGL/gl3ext.h>
  #include <GLUT/glut.h>
#endif
```

How does OpenGL work

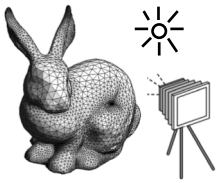
From the programmer's point of view:

1. Specify geometric objects
2. Describe object properties
   - Color
   - How objects reflect light

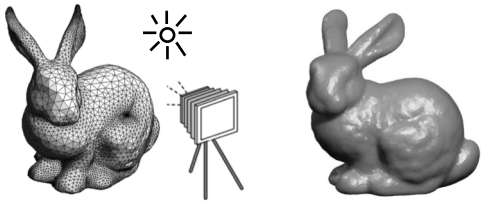


How does OpenGL work (continued)

3. Define how objects should be viewed
   - where is the camera
   - what type of camera
4. Specify light sources
   - where, what kind
5. Move camera or objects around for animation



The result



the result

OpenGL is a state machine

State variables: vertex buffers, camera settings, textures, background color, hidden surface removal settings, the current shader program...

These variables (the *state*) then apply to every subsequent drawing command.

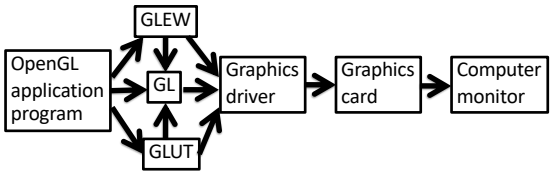
They persist until set to new values by the programmer.

Attributes:
color, shading and reflection properties

- Set before primitives are drawn
- Remain in effect until changed !

OpenGL Library Organization

- **GL** (Graphics Library): core graphics capabilities
- **GLUT** (OpenGL Utility Toolkit): input and windowing
- **GLEW** (Extension Wrangler): removes OS dependencies
- **GLU** (OpenGL Utility Library; compatibility profile only): utilities on top of GL



Core vs Compatibility Profile

• Core Profile:
  • "Modern" OpenGL
  • Introduced in OpenGL 3.2 (August 2009)
  • Optimized in modern graphics drivers
  • Shader-based
  • Used in our homeworks

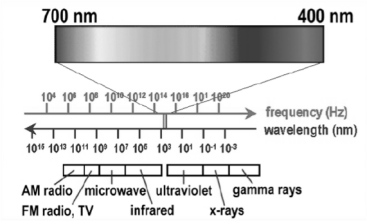
• Compatibility Profile:
  • "Classic" OpenGL
  • Supports the "old" (pre-3.2) OpenGL API
  • Fixed-function (non-shader) pipeline
  • Not as optimized as Core Profile

Mixing core and compatibility profiles

- Windows, Linux:
  Can mix core and compatibility profile OpenGL commands
  ➔ can lead to confusion
  (is the specific OpenGL command optimized?)
  ➔ advantage: more flexible (can re-use old code)
- Mac:
  Can only choose one profile (in each application)

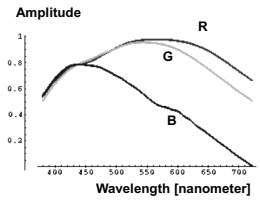
Physics of Color

- Electromagnetic radiation
- Can see only a tiny piece of the spectrum



Color Filters

- Eye can perceive only 3 basic colors
- Computer screens designed accordingly



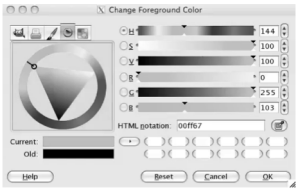
Source: Vos & Walraven

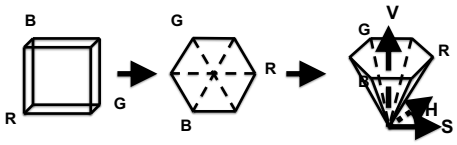
Color Spaces

- RGB (Red, Green, Blue)
  - Convenient for display
  - Can be unintuitive (3 floats in OpenGL)
- HSV (Hue, Saturation, Value)
  - Hue: what color
  - Saturation: how far away from gray
  - Value: how bright
- Other formats for movies and printing

RGB vs HSV

Gimp Color Picker

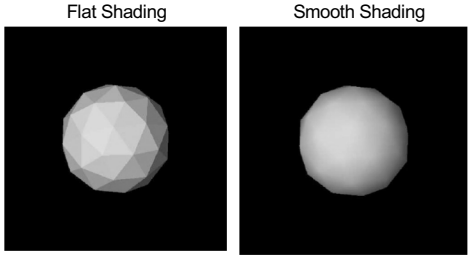




Flat vs Smooth Shading

Flat Shading

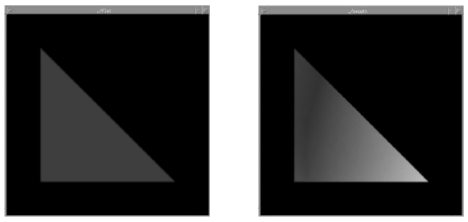
Smooth Shading



Flat vs Smooth Shading

color of last vertex

each vertex separate color smoothly interpolated



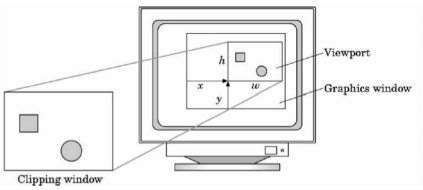
Compatibility profile:
glShadeModel(GL_FLAT)

Compatibility profile:
glShadeModel(GL_SMOOTH)

Core profile: use interpolation qualifiers in the fragment shader

Viewport

- Determines clipping in window coordinates
- glViewport(x, y, w, h) (usually in reshape function)



Summary

1. OpenGL API
2. Core and compatibility profiles
3. Colors
4. Flat and smooth shading