CSCD 396

Beginning Graphics

Course Introduction

- This course introduces the basic concept of 3D Computer Graphics and its implementation using OpenGL.

- **Text book:**
  - OpenGL Programming Guide, Eighth Edition (The Red Book) by D. Shreiner. ISBN:978-0-321-77303-6.

Course Syllabus

- **Subject to change:**

- **OpenGL overview**: Introduction to OpenGL pipeline, OpenGL API and 3D system.
- **Introduction to shader and GLSL**: Shader programming basics, data transfer from client to server in shader program.
- **Transformation basics**: different transformation (translation, rotation, scale and shear), matrix and vector operations.
- **3D camera and viewing**: viewing and modeling transformations to display an object on the screen in desired size and orientation.
- **Projection and clipping**: Parallel and perspective projections.
- **Color, lighting and shading**: different color models (RGB, CMY, HLS), different illumination models (ambient, diffuse and phong), shading techniques (flat, gouraud and phong shading) and material properties.
- **Texture mapping**: 1D and 2D texture mapping;
- **Rasterization**: scan conversion, anti-aliasing, blending.

Instructor Information

- Shamima Yasmin
- Office hours: Monday to Thursday: 10:00 am to 10:50 pm
- Office location: CEB 315,
- Email: syasmin@ewu.edu

Course Evaluation

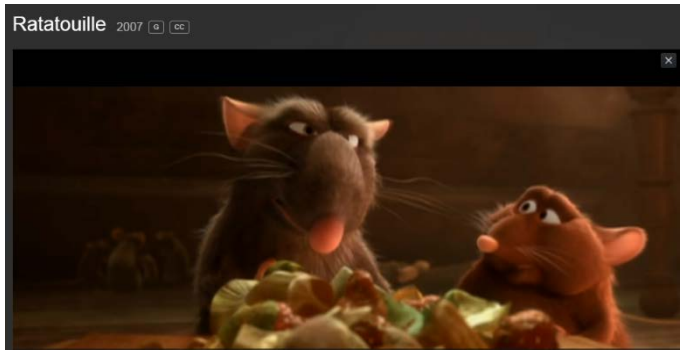
- Seven to eight assignments and one final quiz; grading will be combined for final evaluation;
- C language will be used for the programming assignments;
- Be careful about the university policy on plagiarism (http://access.ewu.edu/undergraduate-studies/academic-integrity);
- Attendance Not graded, but good attendance may have positive influence on your final grades.
  - 95% - 100% is 4.0;
  - For 62% - 94%, subtract 0.1; 60% - 62% is 0.7; 0% - 59% is 0.0;
  - Some specific conversions: 95/4.0, 90/3.5, 85/3.0, 80/2.5, 75/2.0, 70/1.5, 65/1.0.

Introduction

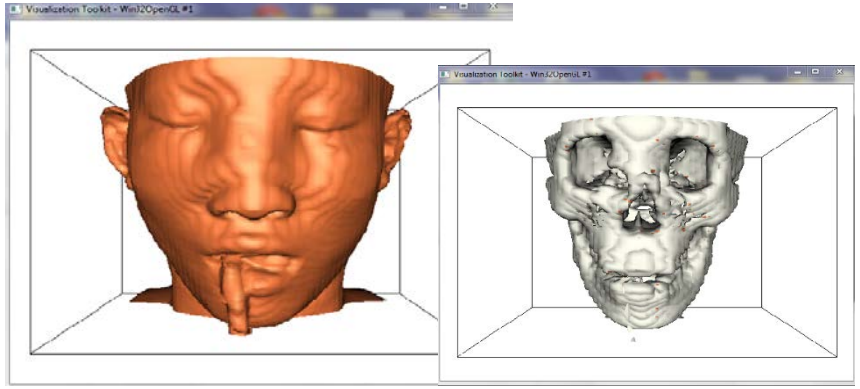
- Computer Graphics deals with all aspects of producing pictures on the computer screen.
- “A picture is worth a thousand words”.
- In this course, we’ll use OpenGL, a graphics software system, widely used for developing graphics application.

Applications

- Education: computer graphics is widely used in engineering, architecture, CAD, flight simulation;
- Industry: entertainment, manufacturing companies



Medical applications

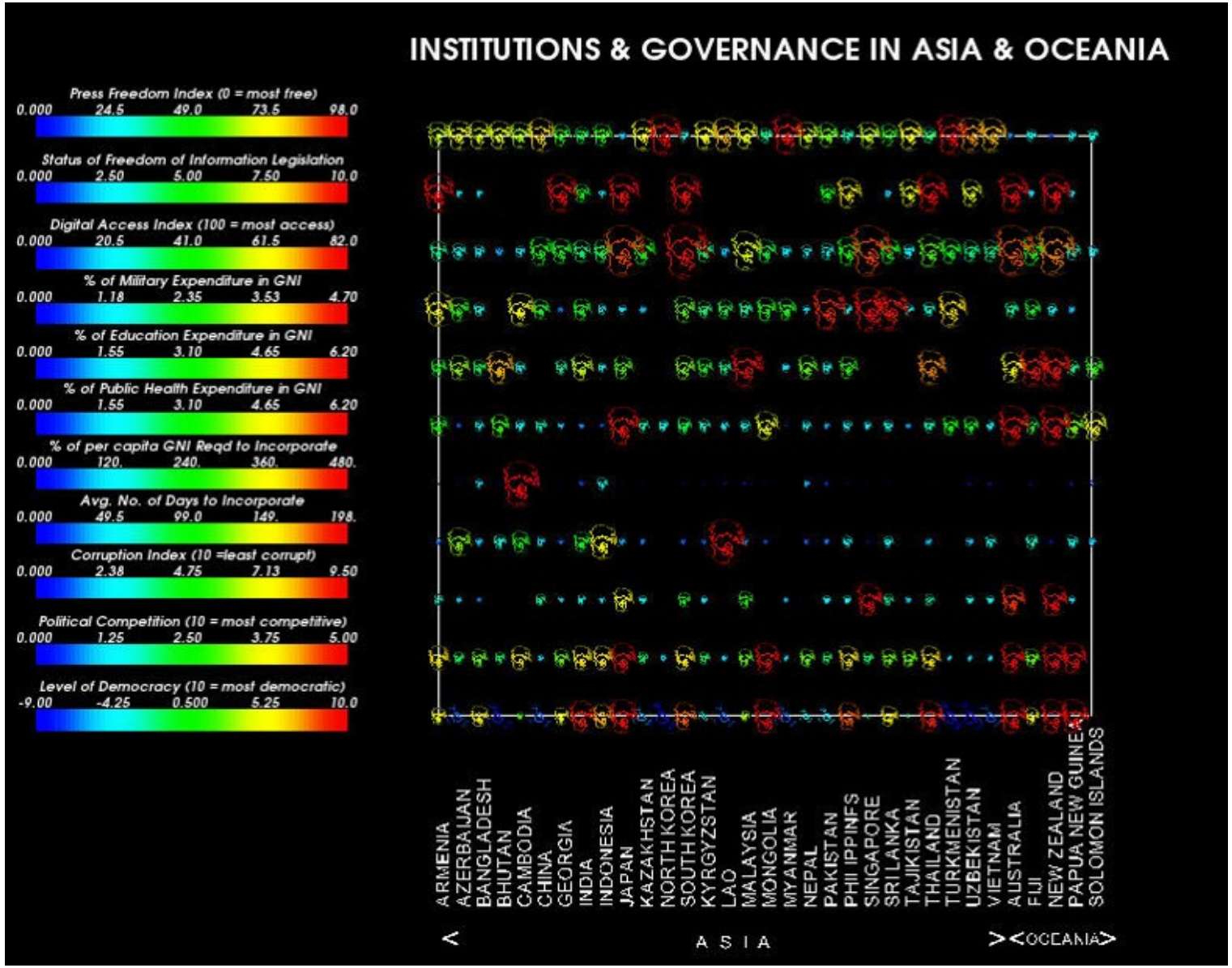
- 3D Doctor helps reconstruction of surface from images;
- vtk / paraview are powerful visualization tools;
-  (Taken from vtk examples)
- Pillcam is used to detect abnormalities in intestine;
- Medical practitioners use virtual environment as a safe practice tool, i.e. virtual palpation, virtual surgery etc.

Information visualization

- Data visualization: numeric/ non-numeric data from science, engineering and medicine;
- GIS, GPS, multi-variate data visualization;
- MATLAB, lifeline can be used for graph, chart etc.
- Again “ A picture is worth a thousand words”, as it has
  - better question answering capability;
  - scalability;
  - modularity and portability.

Information Visualization

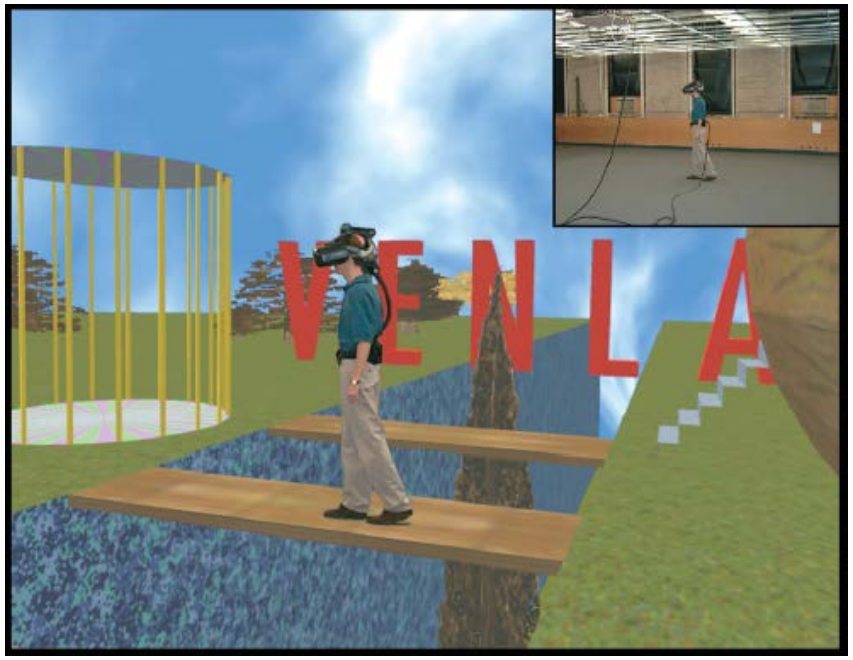
- Institutions and Governance in Asia and Oceania



Virtual Reality

- Creates the illusion that users are in someplace other than where they really are
- Observers can move freely and have the system respond to his/her actions

CyberTouch



"Virtual reality in behavioral neuroscience and beyond", Michael J. Tarr and William J. Warren

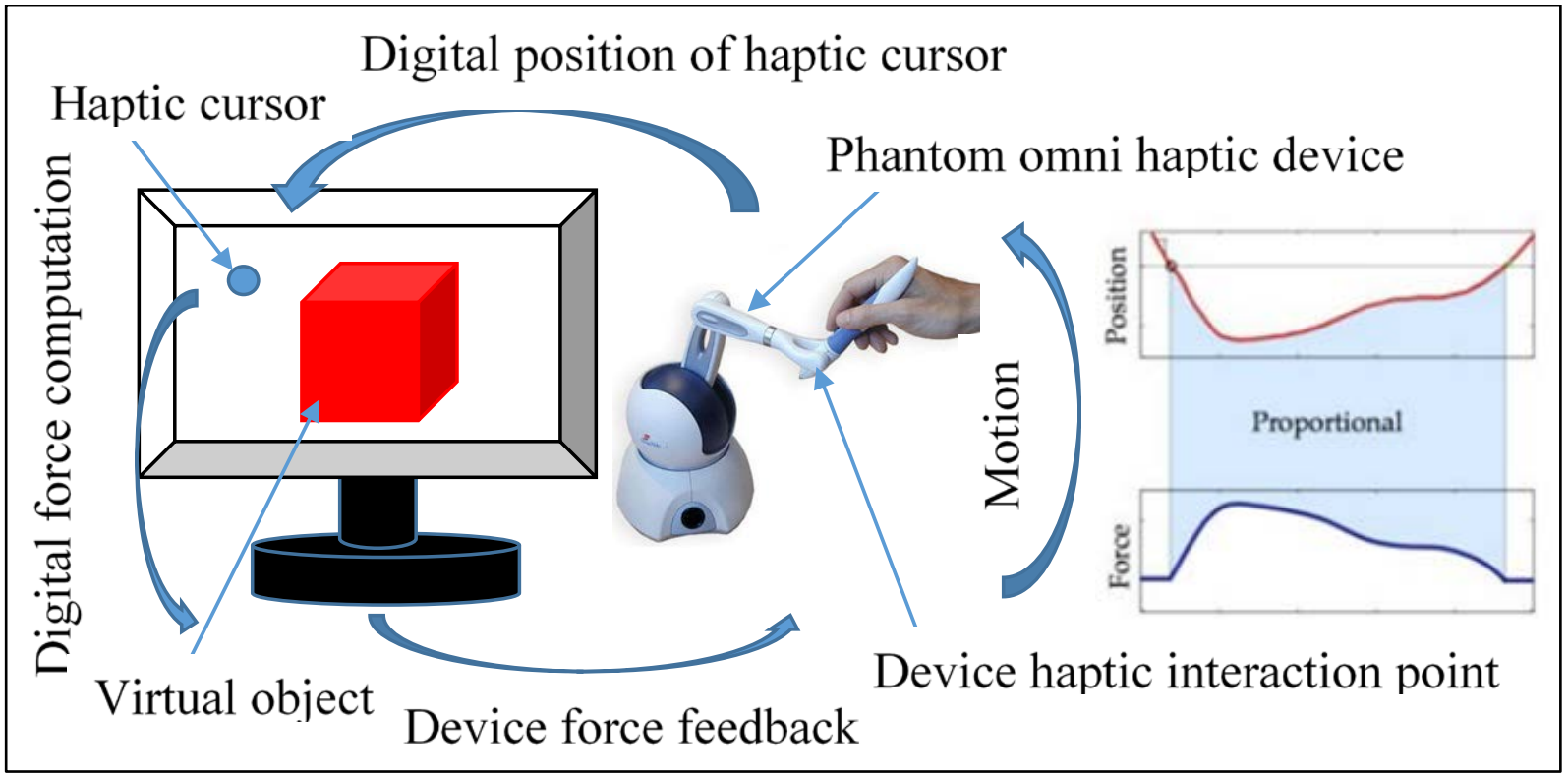
Visual immersion with exercise equipment

- Gait rehabilitation using treadmills
  - Visual cues (optical flow) to modulate the gait characteristics of patients;

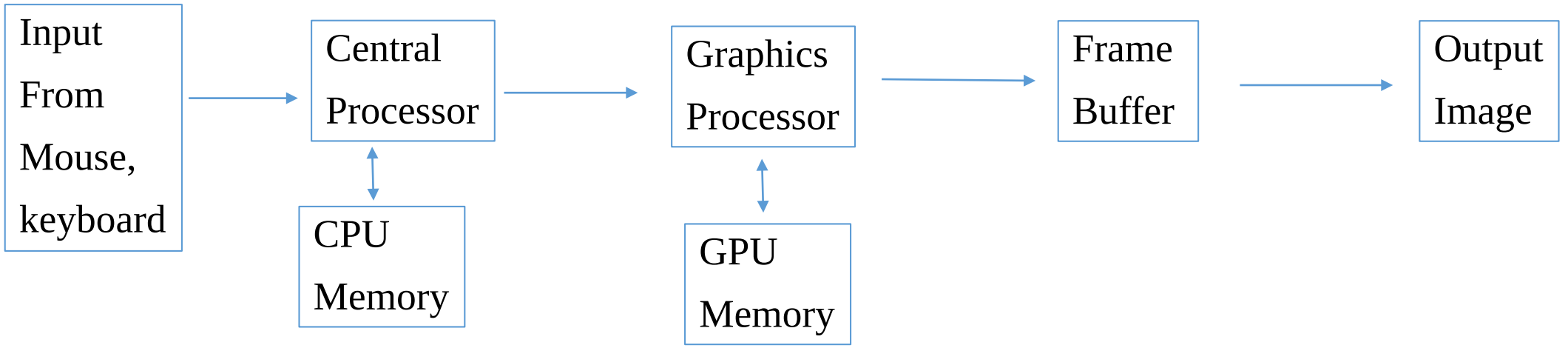
(V-gait system, Motek Medical)

Virtual Reality

- My area of research!



Graphics System at a Glance



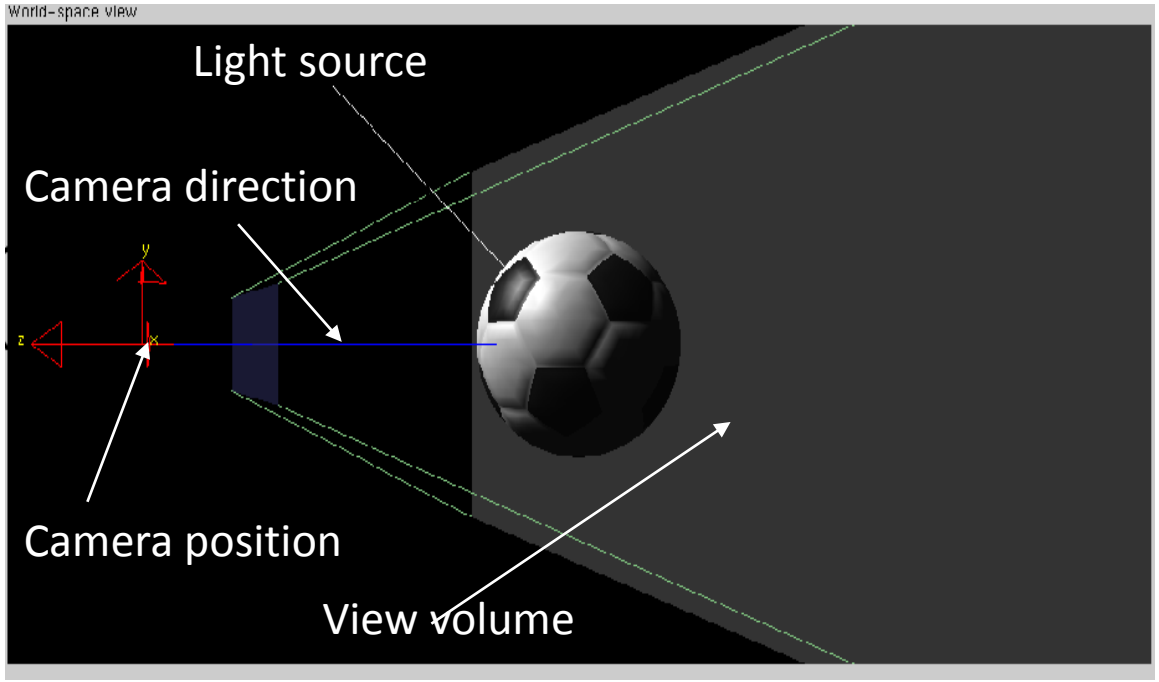
Introduction

In Computer Graphics, modeling our surroundings involves

- Setting up a virtual workspace or view volume
- Placing the models within the view volume
- Setting up a virtual camera to focus the models within the view volume
- Applying proper illumination (lighting) to generate real-life effect
- Consider material properties;
- Generating the 2D image representation of the 3D scene considering the projective geometry.

Modeling an object in Computer Graphics

Output of the object on the screen

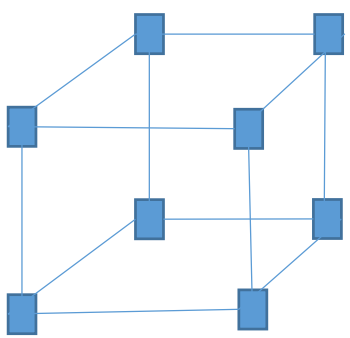


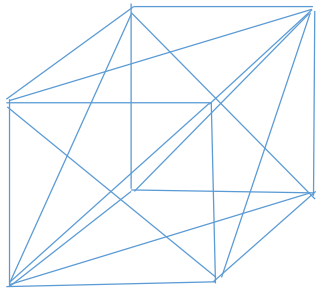
Position of camera, object, light source and view volume

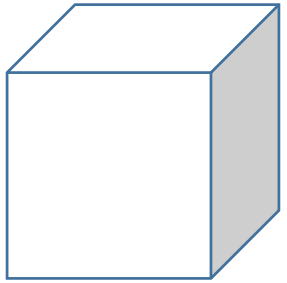
From Nate Robin's tutorial (http://iel.ucdavis.edu/projects/chopengl/demos.html)

Graphics Primitives

- In Computer Graphics, a smooth surface is approximated by polygon patches called polygonal approximation;
- Surface is presented as a large collection of primitives, i.e. triangles, quads;

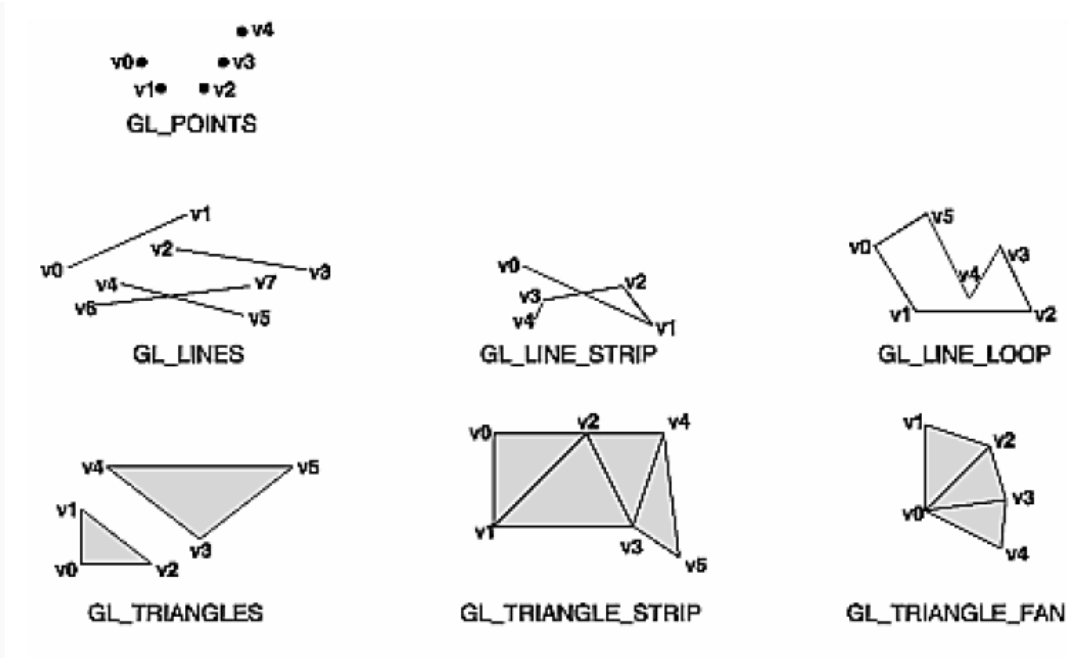






Graphics primitives

- Points: represented by a single vertex in homogeneous coordinates; i.e. (x, y, z, w); I'll talk about homogeneous coordinates later;
- Lines : individual lines are represented by a pair of vertices, one for each end point of the line
- Triangles: made up of three vertices with three edges.



What is OpenGL

- OpenGl is a renderer, an Application Programming Interface (API) , a software library for accessing features in graphics hardware;
- OpenGL is a streamlined, hardware-independent interface that can be implemented on many different types of graphics hardware systems;
- OpenGL is independent of a computer’s operating/ windowing system

What is OpenGL

- OpenGL is implemented as a “client-server” system;
- The application being written is considered as the “client”;
- OpenGL implementation provided by computer graphics hardware is considered as the server.

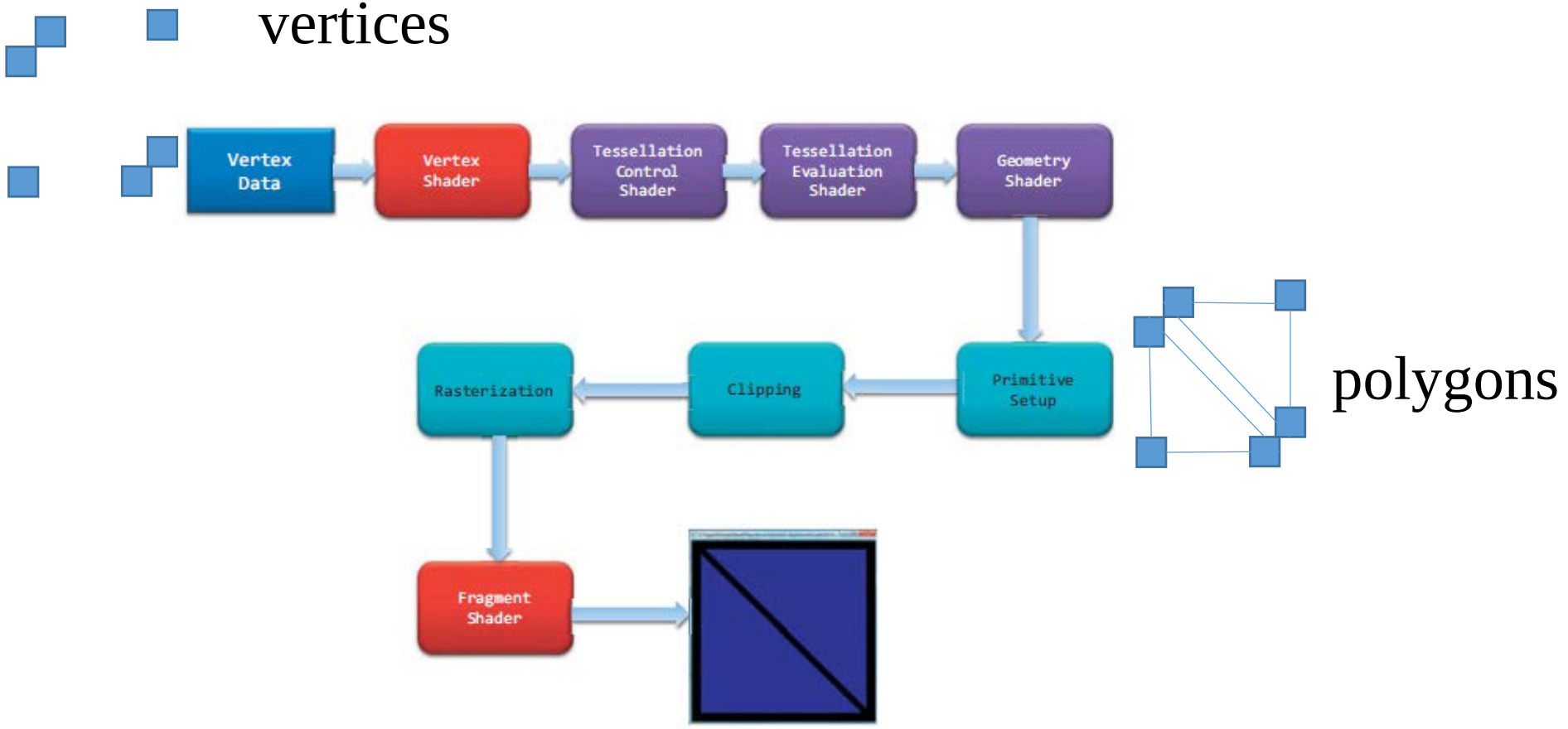
What is OpenGL

- OpenGL requires that all data be stored in *buffer objects*, which are just chunks of memory managed by the OpenGL server.

- Populating these buffers with data can occur in numerous ways, but one of the most common is using the **glBufferData()** command;

- After we've initialized our buffers, we can request geometric primitives be rendered by calling one of OpenGL's drawing commands, such as **glDrawArrays()**;

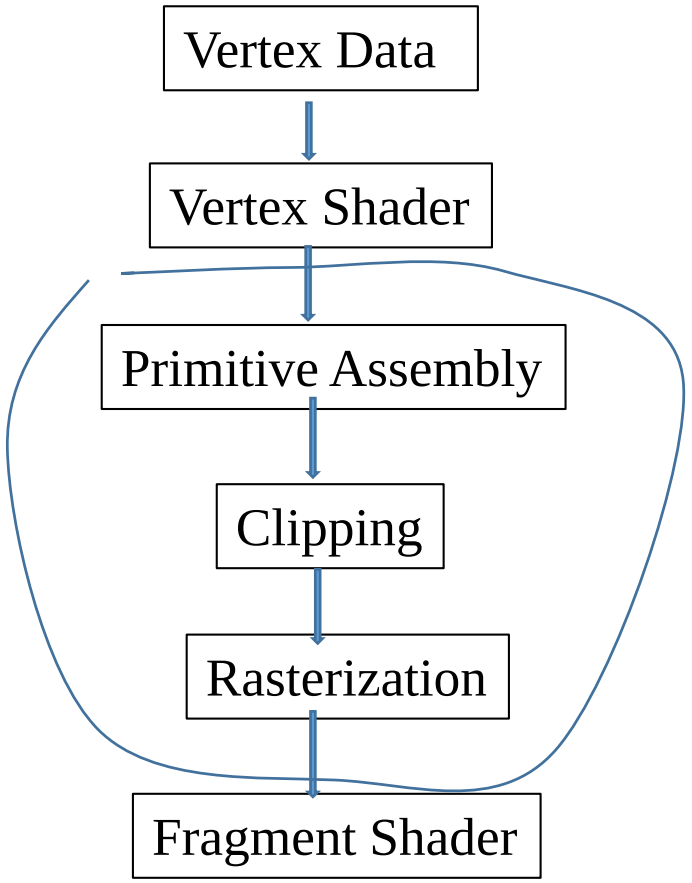
OpenGL Rendering Pipeline



OpenGL Rendering Pipeline

- Not all stages are required; in fact, only vertex shaders and fragment shaders must be included;
- Tessellation and geometry shaders are optional.

OpenGL Pipeline Using Shaders



OpenGL Libraries

OpenGL Utility Toolkit (GLUT) library allows the user to create and manage windows as well as monitor keyboard and mouse input.

freeglut— like GLUT, a cross platform windowing and keyboard/mouse handler. It is more up to date than GLUT

OpenGL Extension Wrangler Library (**GLEW**) is a cross-platform C/C++ library that helps in querying and loading OpenGL extensions.

Installation in Windows

- Download freeglut and glew from sourceforge.net
- Place the header files (freeglut.h, freeglut_std.h, freeglut_ext.h, glew.h etc.) under C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\include
- Place the lib files (freeglut.lib, glew32.lib) under C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\lib
- Place the dlls (freeglut.dll, glew32.dll) in C:\Windows\System32

Installation in Ubuntu

- sudo apt-get update
- sudo apt-get install freeglut3
- sudo apt-get install freeglut3-dev
- sudo apt-get install binutils-gold
- sudo apt-get install libglew-dev
- sudo apt-get install g++
- sudo apt-get install mesa-common-dev
- sudo apt-get install build-essential
- sudo apt-get install libglew1.5-dev libglm-dev

Installing in Ubuntu

- Compile the code “hello.c” using the following command

  g++ hello.c -lglut -lGL -lGLEW -lGLU -o hello

- Run the executable hello using the following command

  ./hello

Linking problem:

sudo ln -s /usr/lib/libGL.so.1 /usr/lib/libGL.so