ITIC
Information Technology
Innovation Center

Chapter 1
OpenGL Introduction

姚智原

國立台灣科技大學 資訊工程系

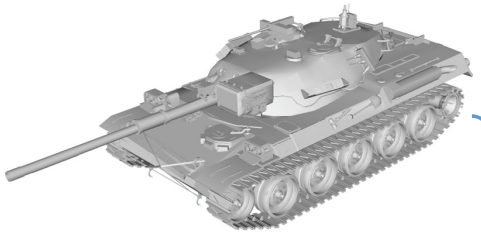
Introducing…

Open Graphics Library

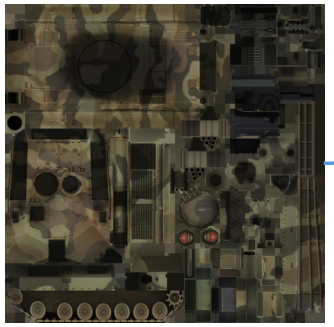
What is OpenGL?

- OpenGL is an *application programming interface (API)*
- Accessing features in graphics hardware
- Over 500 commands to specify image, texture, object and shader programs
- Producing interactive 3-dimensional computer-graphics applications
- Latest version: OpenGL 4.5

What is OpenGL?



3D Model



Texture

Light =(1.5, 0.6, 0.2)
Material=……

OpenGL

Your OpenGL application

Rendered image

An Example of OpenGL API

Name

glClearColor — specify clear values for the color buffers

C Specification

```
void glClearColor( GLfloat red,
                   GLfloat green,
                   GLfloat blue,
                   GLfloat alpha);
```

Parameters

red, green, blue, alpha

Specify the red, green, blue, and alpha values used when the color buffers are cleared. The initial values are all 0.

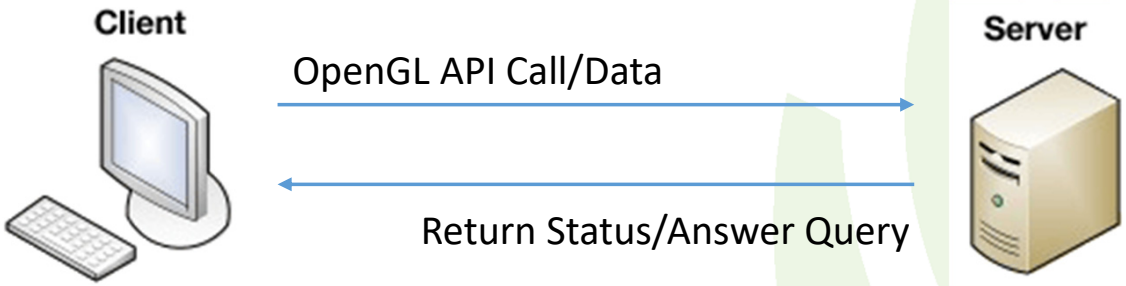
Description

glClearColor specifies the red, green, blue, and alpha values used by glClear to clear the color buffers. Values specified by **glClearColor** are clamped to the range [0, 1].

What is OpenGL?

- OpenGL is designed as:
  - A streamlined, hardware-independent API
  - A primitive-based rendering pipeline
  - A client-server system
- OpenGL is implemented by:
  - Video card manufacturer for each hardware/OS
    - Nvidia, AMD
  - OS provider (software rendering)
  - Yourself!

What is OpenGL?



Your Application Code

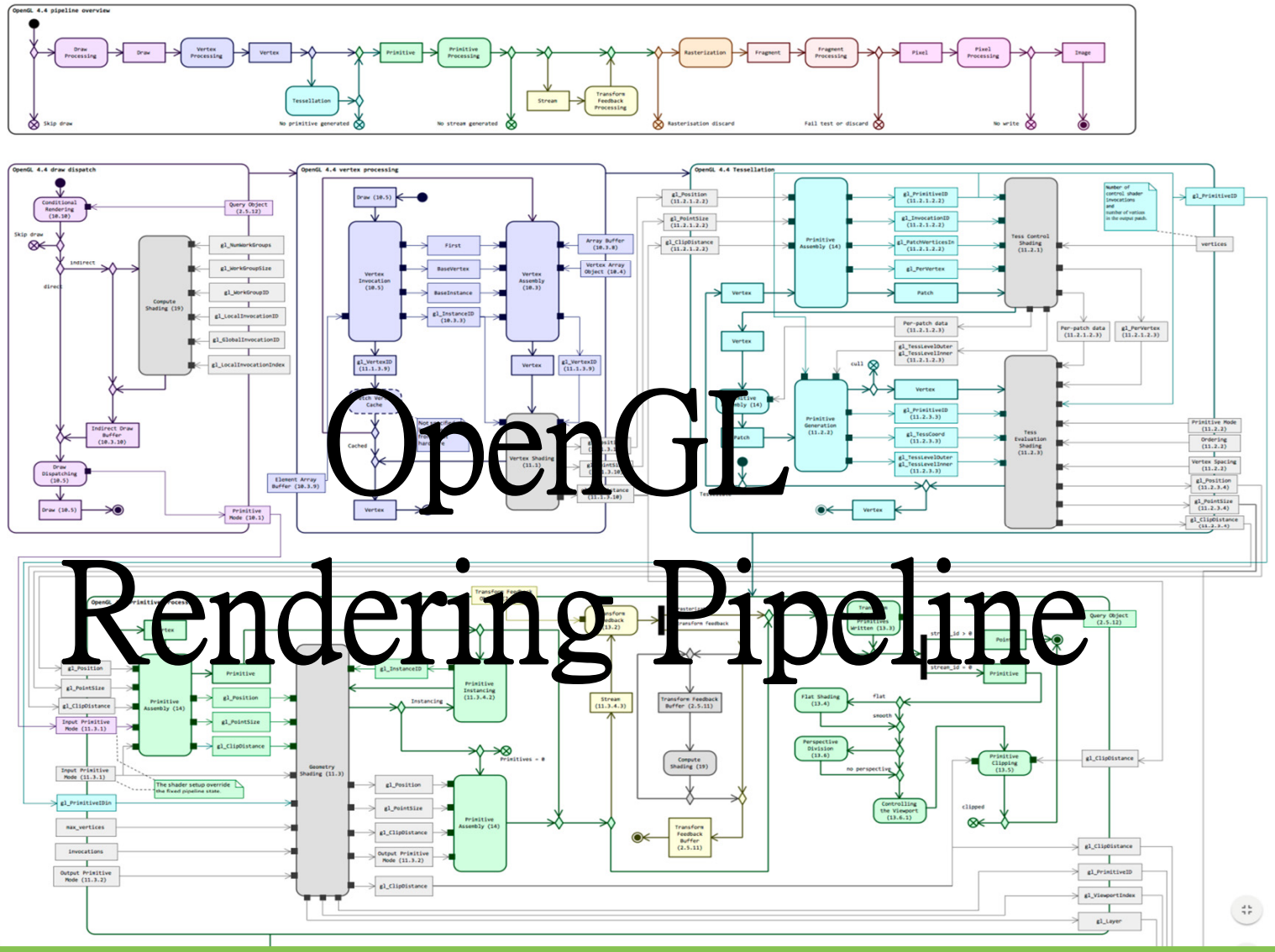
Rendering Algorithm
Load Data from Hard Disk
…

OpenGL Implementation
(AMD/NVIDIA Driver)

OpenGL States:
DEPTH_TEST=ON
Shader ID=1
…

Vulkan

- A.K.A. OpenGL Next
- Idea similar to AMD Mantle and Direct3D 12
- A Fully cross-platform API
  - Windows, Linux, Android…
  - Google has acquired the mobile group of LunarG, the develop team of the Vulkan driver

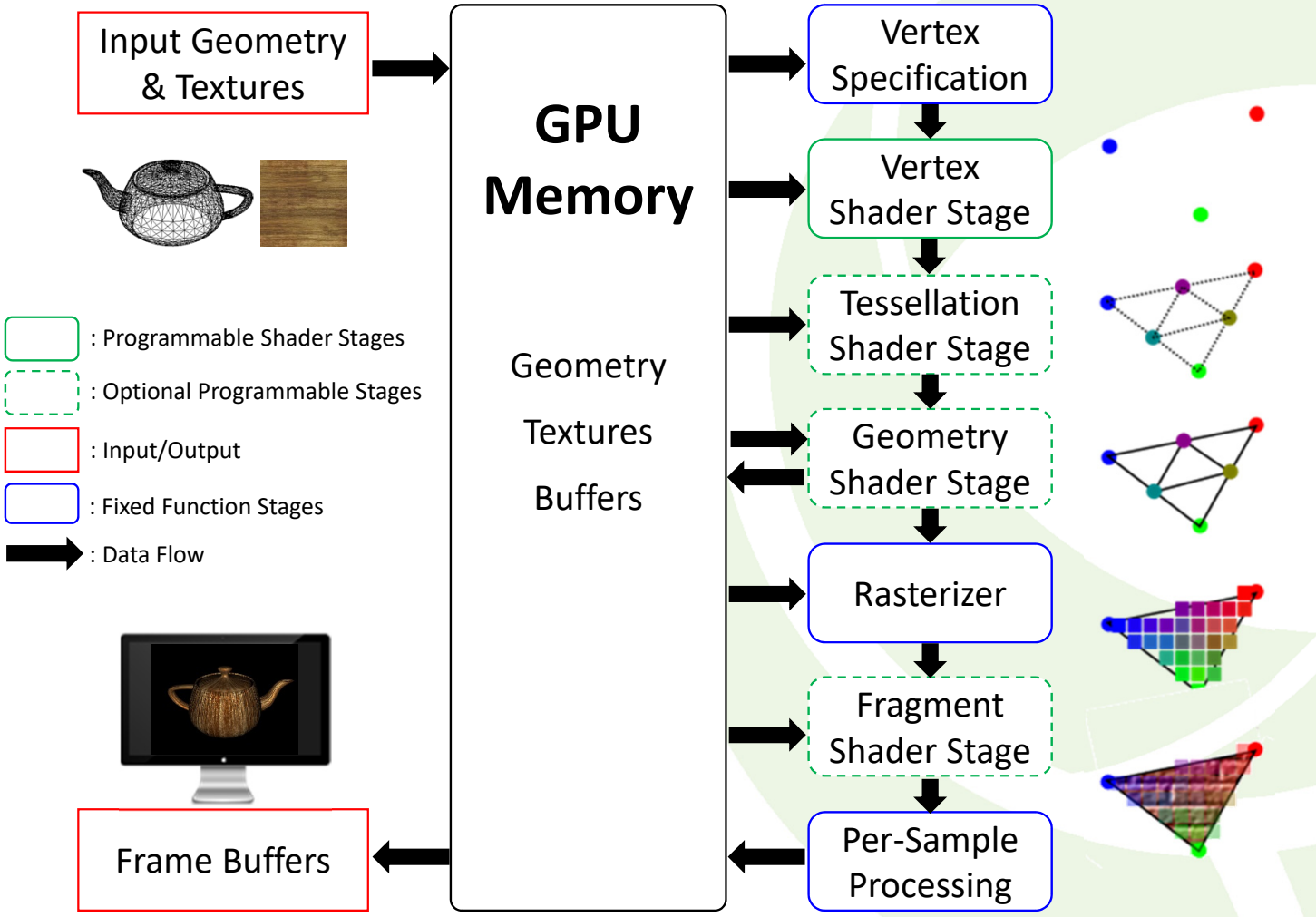


OpenGL Rendering Pipeline

Rendering Pipeline

- Pipeline: consists of multiple stages. Data flows in, being processed in each stages, then flows out
- Stages: each stage represents an unique function to process the input data
  - **Fixed function** stages: limited customization capability, typically exposes states for configuration
  - **Programmable shader** stages: allow custom shader programs to be executed within, providing broader capability of customization

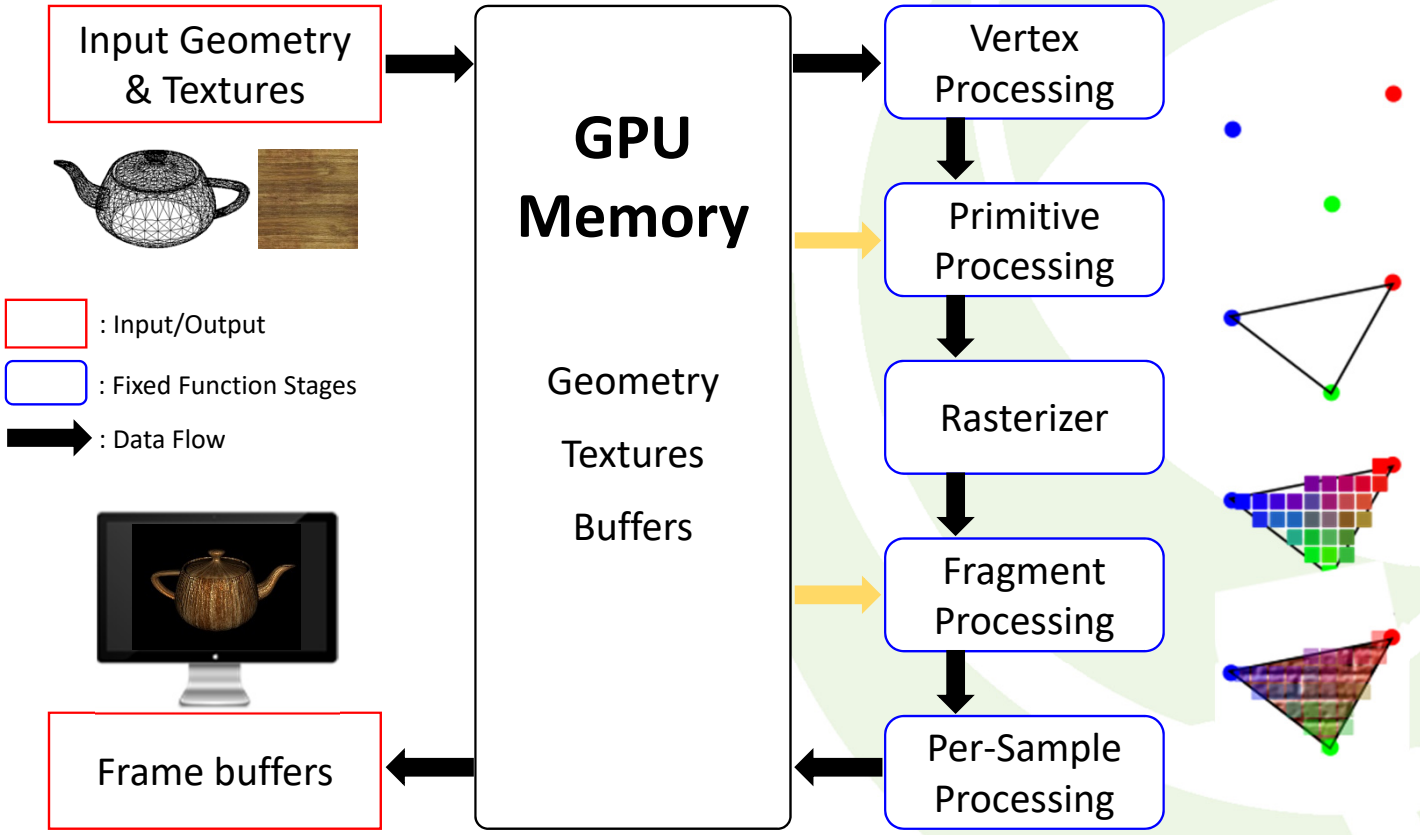
The OpenGL Rendering Pipeline



Fixed Function Pipeline (Legacy)

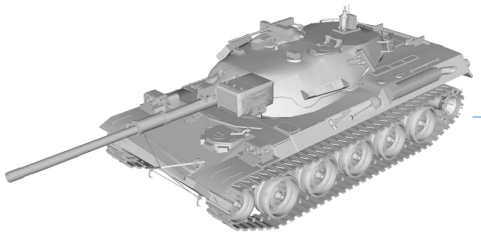
- Before OpenGL 3.0, OpenGL rendering is done in a fixed function pipeline
- Fixed pipeline is like an machine with a lot of switches/values to configure
- One cannot change how the function is implemented as well as the order of execution

Fixed Function Pipeline (Legacy)



Fixed Function Pipeline: Metaphor

OpenGL Fixed Function Pipeline

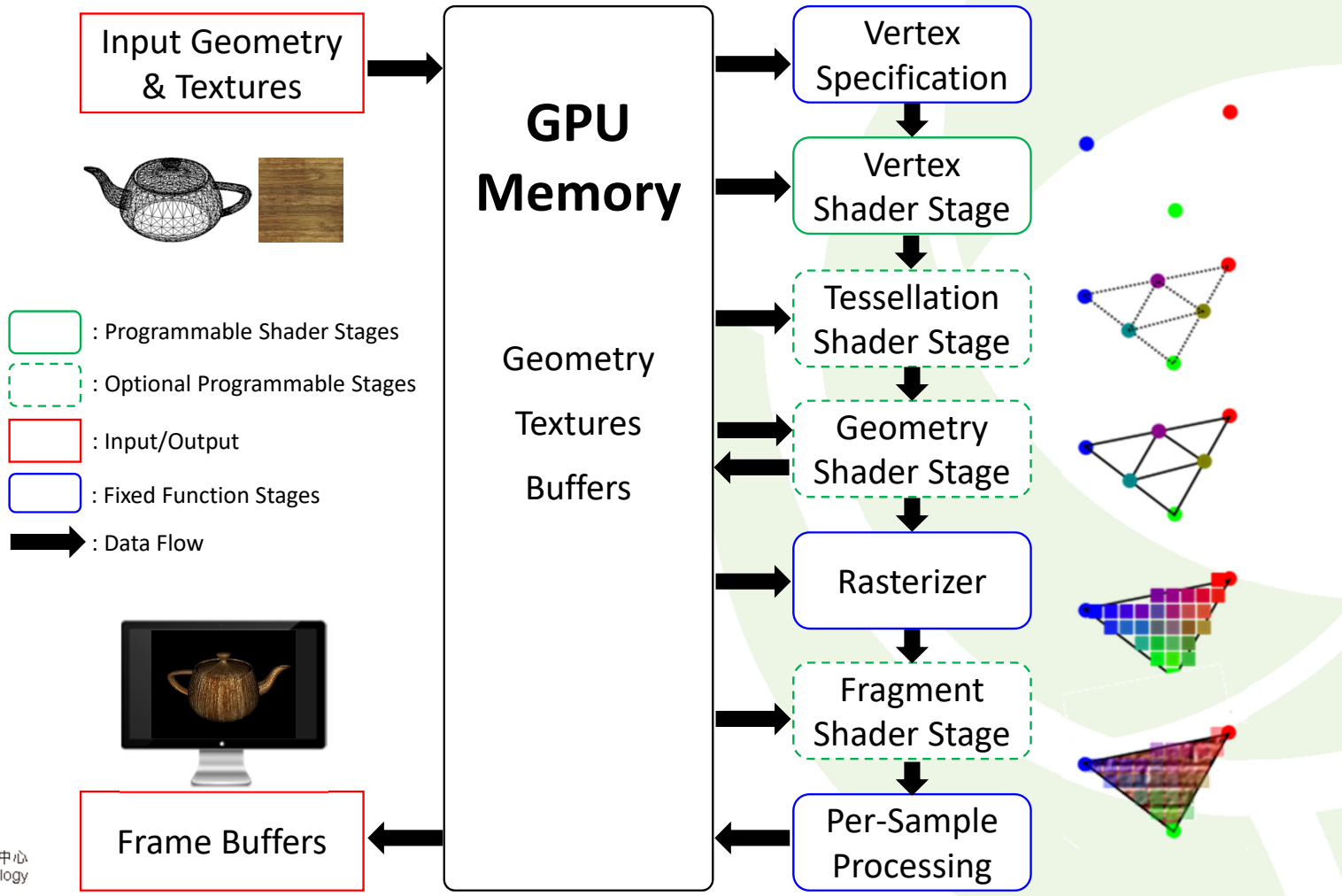


How do I press these buttons to get desired effect?

Programmable Pipeline

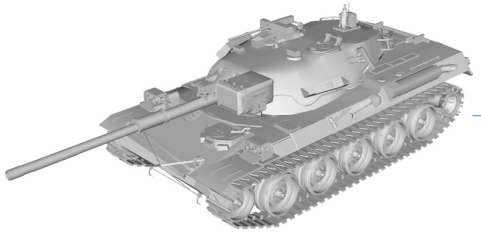
- Shader programs are introduced in OpenGL 2.0, and included in the core profile in OpenGL 3.0
- Fixed function pipeline is deprecated since OpenGL 3.0
- Shader programs, written in OpenGL Shading Language(GLSL), allow the programmers to customize certain stages in the OpenGL rendering pipeline

The OpenGL Rendering Pipeline



Programmable Pipeline: Metaphor

Shader Programs



```
uniform float t;    // Time (passed in from the application)
attribute vec4 vel; // particle velocity

const vec4 g = vec4( 0.0, -9.80, 0.0 );

void main()
{
    vec4 position = gl_Vertex;
    position += t*vel + t*t*g;

    gl_Position = gl_ModelViewProjectionMatrix * position;
}
```

Let's write a program to create the effect!

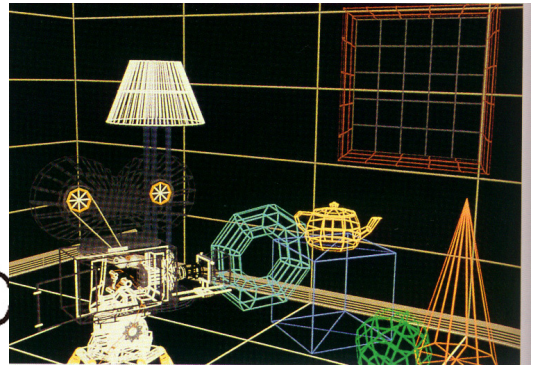
Programmable Pipeline V.S. Fixed Function Pipeline

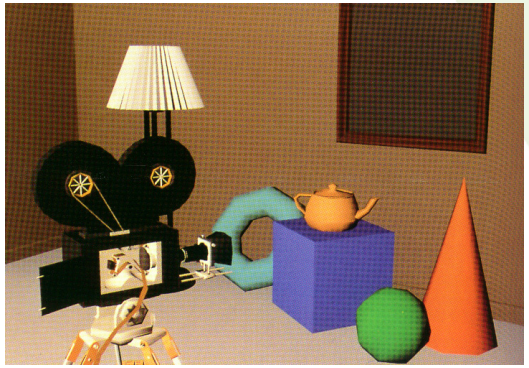
	Programmable Pipeline	Fixed Function Pipeline
Flexibility	+implement various algorithms in shader programs	-limited customization capability
For Simple Application	-must configure the whole pipeline	+performs simple task with less configurations
For Complex Application	+can achieve various effects	-most advanced effects are impossible
Learning Curve	-one must have full knowledge of the pipeline and GLSL before writing an application	+can create simple applications without much knowledge
Deploy	-should consider all graphics driver environment	+works in most graphics driver environment

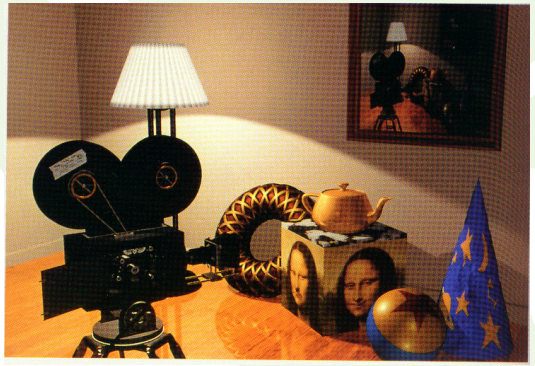
Course outlines

Fundamental of Computer Graphics

- 3D geometry representation
- 3D projection and transformation
- Color, material and lighting
- Texture mapping



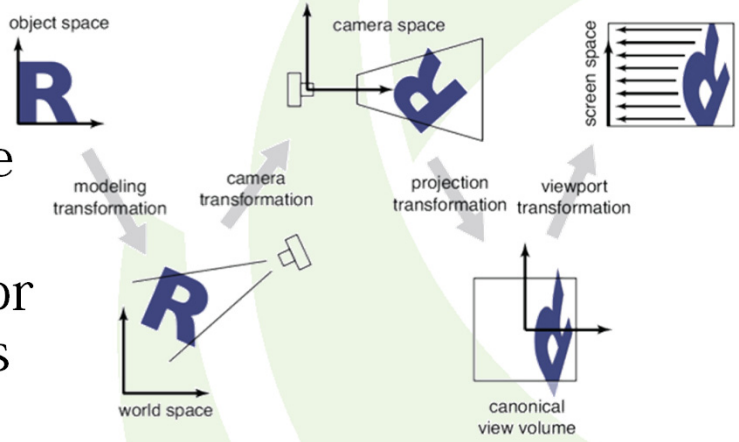


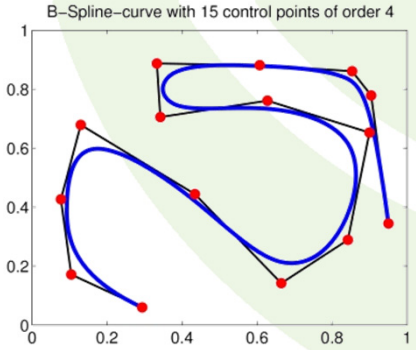


ITIO

Shader Programming (OpenGL)

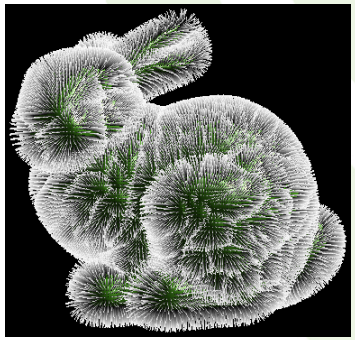
- Rendering Pipeline
  - Configuring the pipeline
  - OpenGL Shading Language (GLSL) basics
  - Writing shader program for each programmable stages
- Math
  - Vectors and matrices
  - Coordinate spaces and transformations
  - Higher-order lines, curves, splines

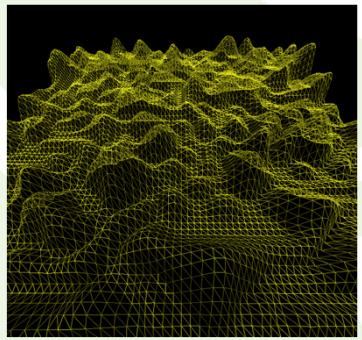




Shader Programming (OpenGL)

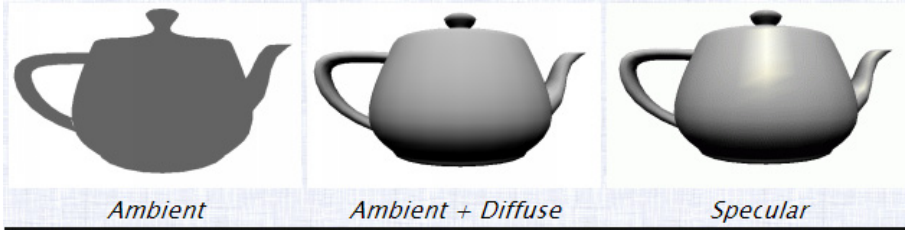
- Data
  - Buffer
  - Texture
- Vertex Processing
  - Vertex shader
  - Drawing commands
- Primitive Processing
  - Geometry shader
  - Tessellation shader





Shader Programming (OpenGL)

- Fragment processing
  - Color
  - Frame buffer processing
- Rendering techniques
  - Lighting models
  - Non-photorealistic rendering




thank you!
Question?