Practical Computer Graphics Skills for Technical Artist

2019 Spring

National Cheng Kung University
Prof. Min-Chun Hu

+ UDREAM Entertainment

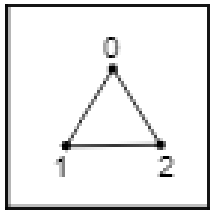
Homework 1
(for CSIE/EE students)

OpenGL Philosophy

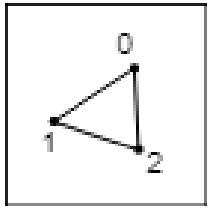
- Context = Big State Machine
- Operations on binding object
  - glGenBuffers(1, &id)
  - glBindBuffer(id)
  - glBufferData(…)
- What's the problem?
  - Binding is expensive
  - >=4.5, use named buffer
  - glCreateBuffers/glCreateTextures
  - glNamedBufferData(id, ….)

Rendering Pipeline

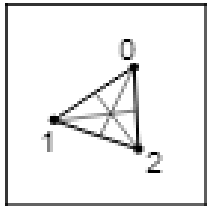
Input assembler



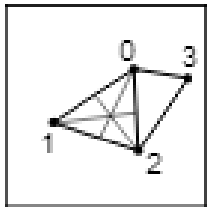
Vertex shader



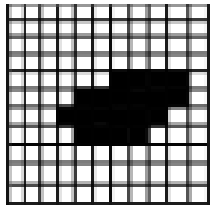
Tessellation



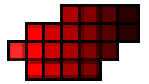
Geometry shader



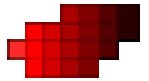
Rasterization



Fragment shader



Color blending



MiS UDREAM Entertainment

Before we start to draw… THINK

- What do we need for drawing?
- Suppose we have an array of struct that hold {vec3 pos, vec2 uv}

- What type of primitive should be drawn? And how many?
- How to upload buffer?
- How to specify buffer format for gpu?

Uploading vertex to GPU/Driver

- Vertex Buffer
  - An object that **hold data**
- Vertex Attribute
  - Vertices Position/Texcoord/Normal/Color
  - Each of them is **Vertex Buffer Object** in OpenGL
    - They can be in the same buffer
  - We use vertex attribute pointer to describe structure format
- Array Of Structure
- Structure Of Array

Create Buffer and upload

- glBindBuffer(GL_ARRAY_BUFFER, vbo)
- glBufferData(GL_ARRAY_BUFFER, size, data_ptr, **GL_STATIC_DRAW**)

- Nowadays, we use glBufferStorage more often
  - GL_MAP_READ_BIT
  - GL_MAP_WRITE_BIT
  - GL_MAP_PERSISTENT_BIT
  - GL_MAP_COHERENT_BIT

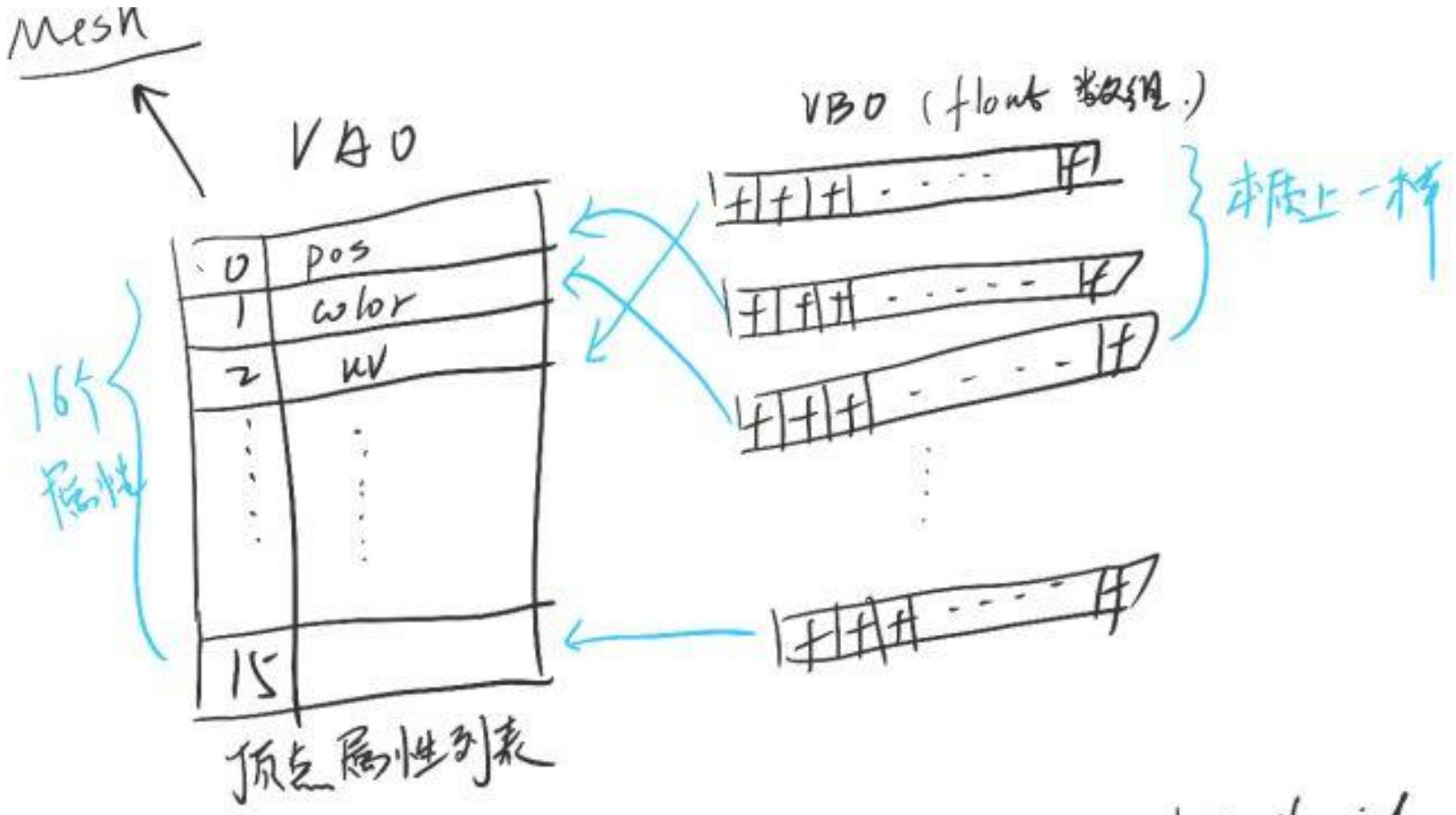
Describe vertex attribute

- For example if the attribute is vec2:
- glBindBuffer(**GL_ARRAY_BUFFER**, id)
- glEnableVertexAttribArray(1);
- glVertexAttribPointer(1, 2, GL_FLOAT, normalize?, stride, offset);

Vertex Array Object

- BindBuffer and specify vertex attribute before each draw call
  - time-consuming.
- Vertex Array Object hold this information for us.
- Struct VAO {
- bool enabled[N];
- struct Attrib attrib[N]
- };
- Struct Attrib {
- int buffer_id;
- int offset, stride;
- };

Relation between VAO and VBO



Draw Call

- glUseProgram(program)
- glBindVertexArray(vao)
- glDrawElements(…) / glDrawArrays() / glDrawArraysInstanced…

Vertex Shader

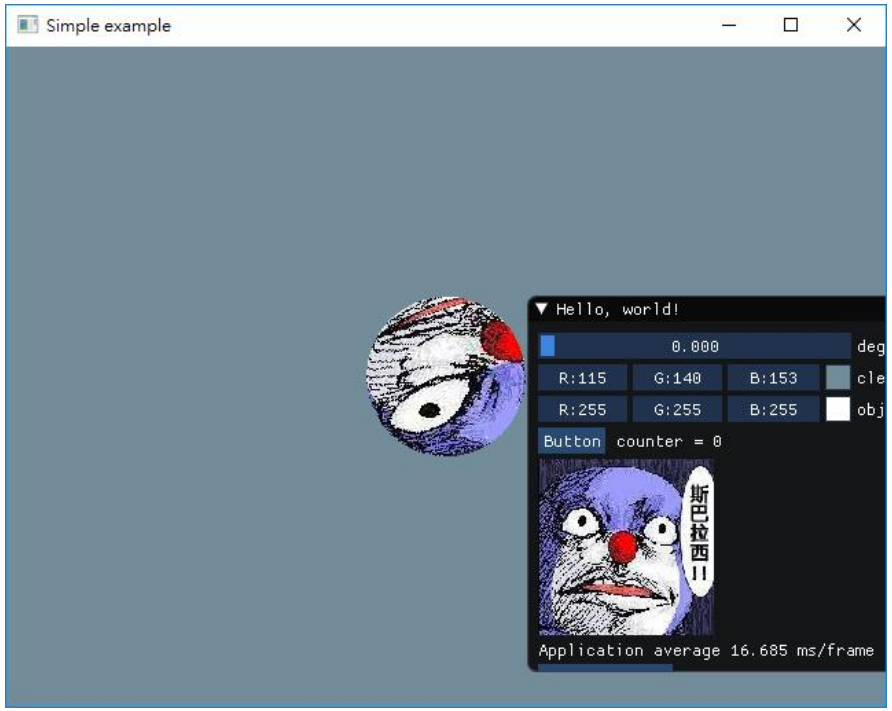
```
#version 330
layout(location = 0) in vec3 position;
layout(location = 1) in vec2 texcoord;
layout(location = 2) in vec3 normal;

uniform mat4 model;
uniform mat4 vp;

out vec2 uv;

void main()
{
    uv = texcoord;
    gl_Position = vp*model*vec4(position, 1.0);
}
```

Fragment Shader

```
#version 330
layout(location=0) out vec4 color;

uniform sampler2D text;
uniform vec3 object_color;

in vec2 uv;

void main()
{
    color = vec4(object_color, 1.0)*texture(text, uv);
}
```

Homework Description

- In this homework, we have finished most of the codes for you.
  All you need to do is to complete the requirements in the next page.
- https://github.com/tim37021/CGHW1-2019



Requirements

- First, you need to add an Earth which can rotate (自轉), revolve (公轉) around the Sun, and rotation about an arbitrary axis (任意軸旋轉).

- Second, you have to add a moon which can rotate (自轉), revolve (公轉) around the earth, and rotation about an arbitrary axis (任意軸旋轉).

Scoring

- Sun (20%)
- Earth (30%)
- Moon (30%)
- Demo (10%)
- Hackmd (10%): Fill your own Hackmd link in https://docs.google.com/spreadsheets/d/1V2EbTwFRCjbRYQYDOpKCJgi98IciBTFpsCvgRD9HVyU/edit#gid=0

- **Due: 4/1 10:00 pm**