The Graphics Pipeline

Lecture 2

Robb T. Koether

Hampden-Sydney College

Wed, Aug 28, 2019

Outline

Outline

Vertices

- Objects consist basically of a set of vertices that define points in the plane (2D) or in space (3D).
- The vertices are grouped to create triangles and triangles are grouped to make more complex surfaces (meshes).
- For example, a 3D box might consist of 6 rectangles, i.e., 12 triangles.
- To make a sphere, we would typically use about 6400 triangles (80 points around the equator, 40 points from pole to pole, each region split into 2 triangles.)

Vertices

- Vertices are described by specifying their attributes.
  - Position
  - Color
  - Normal vector (orientation)
  - Texture coordinates
- These attributes are fed into the graphics pipeline.
- Each stage of the pipeline performs an operation on the object.

Outline

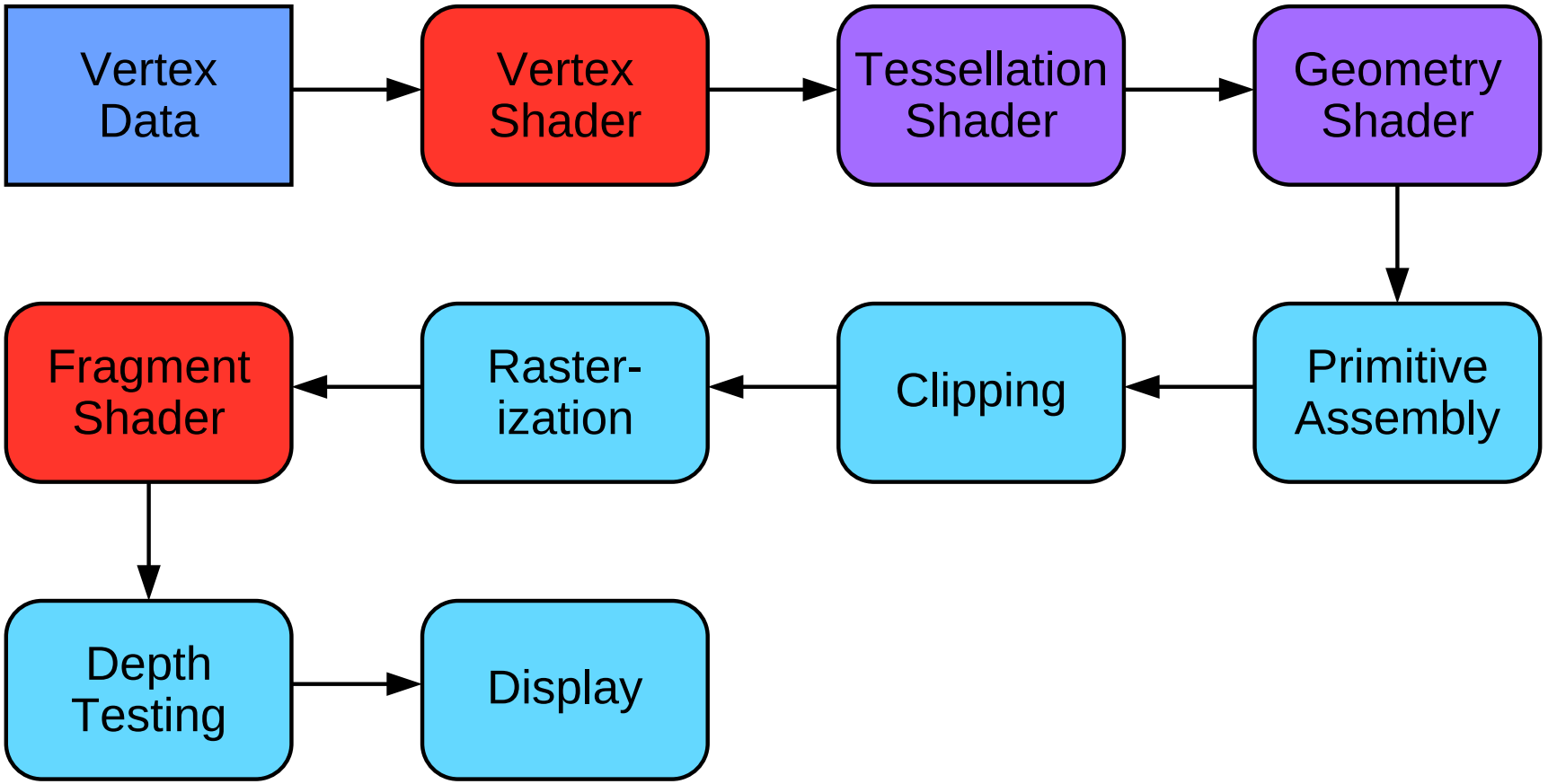
Stages of the Pipeline

- The stages of the pipeline:
  - Vertex shader
  - Tessellation shader
  - Geometry shader
  - Primitive assembly
  - Clipping
  - Rasterization
  - Fragment shader
  - Per-fragment processing

Shaders

- The programmer is responsible for the shaders
  - Vertex shader (mandatory)
  - Tessellation shader (optional)
  - Geometry shader (optional)
  - Fragment shader (mandatory)
- The other stages are “fixed.” The programmer has no control over them.

The Graphics Pipeline



Outline

The Vertex Shader

- At a minimum, the vertex shader assigns a value to the built-in shader variable `gl_Position`.
- If that is all that the shader does, then it is called a pass-through shader.
- Vertex shaders typically do much more than that.

A Pass-Through Vertex Shader

A Pass-Through Vertex Shader

```
#version 450 core

layout (location = 0) in vec3 vPosition;

void main()
{
    gl_Position = vec4(vPosition, 1.0f);
}
```

A Pass-Through Vertex Shader

A Pass-Through Vertex Shader (with Color)

```
#version 450 core

layout (location = 0) in vec3 vPosition;
layout (location = 1) in vec3 vColor;

out vec4 color;

void main()
{
    gl_Position = vec4(vPosition, 1.0f);
    color = vec4(vColor, 1.0f);
}
```

Outline

The Fragment Shader

- A fragment is typically a pixel, although it may be a part of a pixel.
- At a minimum, the fragment shader outputs the color of the pixel.
- The name of the output variable does not matter because all fragment shaders output only the color.
- A pass-through shader receives the color from the fragment shader and outputs it unchanged.
- Fragment shaders typically do much more than that.

A Pass-Through Fragment Shader

A Pass-Through Fragment Shader (with Fixed Color)

```
#version 450 core

out vec4 fragColor;

void main()
{
    fragColor = vec4(1.0f, 0.0f, 0.0f, 1.0f);
}
```

A Pass-Through Fragment Shader

A Pass-Through Fragment Shader (with Color Input)

```
#version 450 core

in vec4 color;
out vec4 fragColor;

void main()
{
    fragColor = color;
}
```

Outline

Assignment

Assignment

- Read pp. 10 - 14 in The Red Book.