GAM250: Advanced Games Programming

4: Graphics Programming

FALMOUTH
UNIVERSITY

Learning outcomes

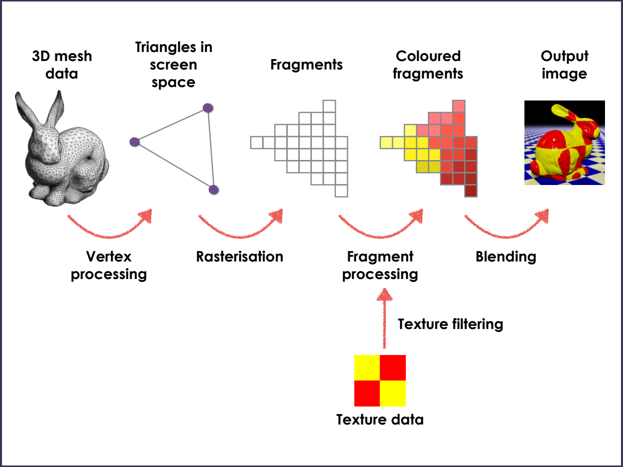
- **Understand** the modern Programmable Graphics Pipeline
- **Understand** Unity's Material System
- **Write** Subsurface and Image Processing Shaders in Unity

FALMOUTH
UNIVERSITY

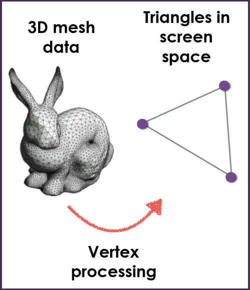
The Graphics Pipeline

FALMOUTH UNIVERSITY

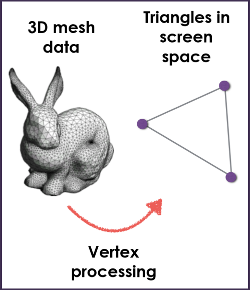
The 3D graphics pipeline



Vertex processing

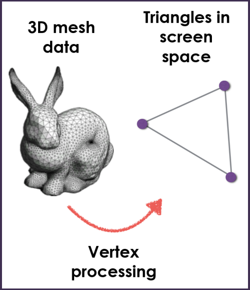


Vertex processing



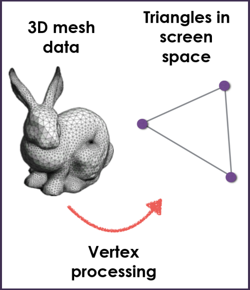
- Geometry is provided to the GPU as a **mesh** of **triangles**

Vertex processing



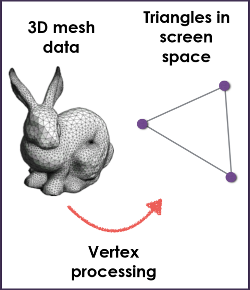
- Geometry is provided to the GPU as a **mesh** of **triangles**
- Each triangle has three **vertices** specified in 3D space (x, y, z)

Vertex processing



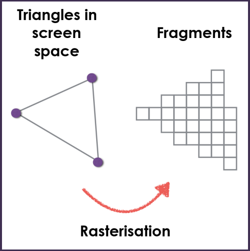
- Geometry is provided to the GPU as a **mesh** of **triangles**
- Each triangle has three **vertices** specified in 3D space (x, y, z)
- Vertex processor **transforms** (rotates, moves, scales) vertices and **projects** them into 2D screen space (x, y)

Vertex processing

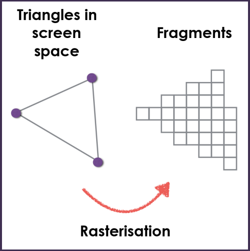


- Geometry is provided to the GPU as a **mesh** of **triangles**
- Each triangle has three **vertices** specified in 3D space (x, y, z)
- Vertex processor **transforms** (rotates, moves, scales) vertices and **projects** them into 2D screen space (x, y)
- May also apply particle simulations, skeletal animations or deformations, etc.

Rasterisation

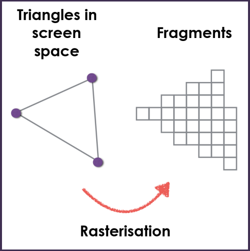


Rasterisation



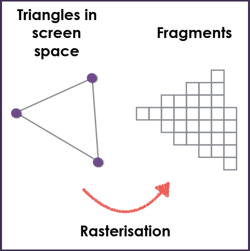
- Determine **which fragments** are covered by the triangle

Rasterisation



- Determine **which fragments** are covered by the triangle
- In practical terms, “fragment” = “pixel”

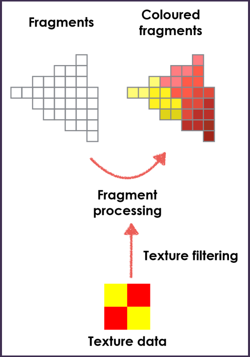
Rasterisation



- Determine **which fragments** are covered by the triangle
- In practical terms, “fragment” = “pixel”
- Vertex processor can associate **data** with each vertex; this is **interpolated** across the fragments

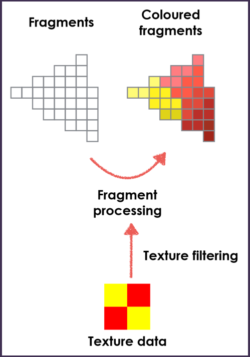
FALMOUTH UNIVERSITY

Fragment processing



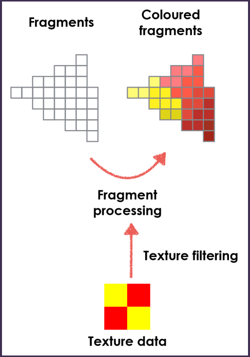
FALMOUTH UNIVERSITY

Fragment processing



- Determine the **colour** of each fragment covered by the triangle

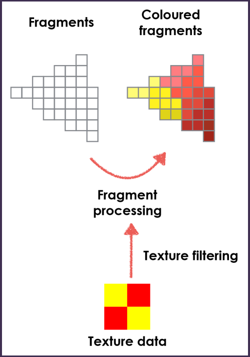
Fragment processing



- Determine the **colour** of each fragment covered by the triangle
- **Textures** are 2D images that can be **wrapped** onto a 3D object

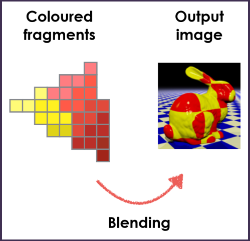
FALMOUTH UNIVERSITY

Fragment processing

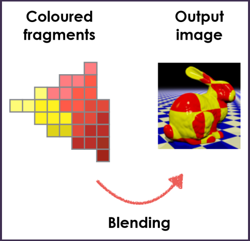


- Determine the **colour** of each fragment covered by the triangle
- **Textures** are 2D images that can be **wrapped** onto a 3D object
- Colour is calculated based on **texture**, **lighting** and other properties of the surface being rendered (e.g. shininess, roughness)

Blending

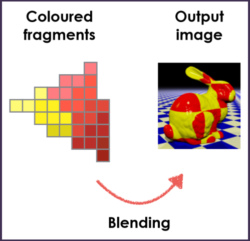


Blending



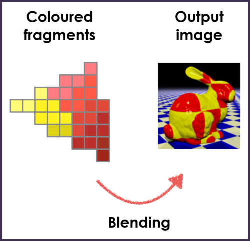
- Combine these fragments with the existing content of the image buffer

Blending



- Combine these fragments with the existing content of the image buffer
- **Depth testing**: if the new fragment is "in front" of the old one, replace it; if it is "behind", discard it

Blending



- Combine these fragments with the existing content of the image buffer
- **Depth testing**: if the new fragment is "in front" of the old one, replace it; if it is "behind", discard it
- **Alpha blending**: combine the old and new colours for a semi-transparent appearance

Shaders

Shaders

- The vertex processor and fragment processor are **programmable**

Shaders

- The vertex processor and fragment processor are **programmable**
- Programs for these units are called **shaders**

Shaders

- The vertex processor and fragment processor are **programmable**
- Programs for these units are called **shaders**
- **Vertex shader**: responsible for geometric transformations, deformations, and projection

Shaders

- The vertex processor and fragment processor are **programmable**
- Programs for these units are called **shaders**
- **Vertex shader**: responsible for geometric transformations, deformations, and projection
- **Fragment shader**: responsible for the visual appearance of the surface

Shaders

- The vertex processor and fragment processor are **programmable**
- Programs for these units are called **shaders**
- **Vertex shader**: responsible for geometric transformations, deformations, and projection
- **Fragment shader**: responsible for the visual appearance of the surface
- Vertex shader and fragment shader are separate programs, but the vertex shader can pass arbitrary values through to the fragment shader

Further Reading

- Game Programming Patterns - `http://gameprogrammingpatterns.com/contents.html`
- Game Programming Patterns in Unity - `http://www.habrador.com/tutorials/programming-patterns/`
- Unity Design Patterns - `https://github.com/Naphier/unity-design-patterns`