CMPM 163 Notes

Malcolm Riley

Winter 2019 — April 16, 2019

Shader Concepts

- A **Texel** is conceptually equivalent to a pixel, but refers to the use case of sampling from a texture.
- When sampling from a texture, it's possible to query the neighboring pixels
- In Unity, if there exists a texture called `_MainTex`, there will exist a special `uniform float4` called `_MainTex_TexelSize` with the following members: `x` and `y` are the width and height of the texture in pixels, respectively

Rendering to Other Targets

- A powerful technique in shader programming is to render to an intermediate state (other than the screen), typically referred to as a **Offscreen Buffer**, **FBO**, or **Frame Buffer Object**. In Unity, this construct is called a **RenderTexture**.
- Instead of rendering directly to the screen, the shader renders to a special memory construct - the aforementioned buffer.
- This buffer object exists in the GPU's memory.
- This technique can be used for any number of fancy effects, such as a "security-camera" effect, a "game-of-life" effect, or an exotic "reaction-diffusion" effect.

Ping-Pong Render Technique

- Also referred to as **Double-Buffered Rendering**
- In this technique, the output of one pass in a frame will be used as the input of another pass in the same frame, or the output of one frame will be used as the input of another frame
- Since texture data on the GPU is read-only, it will be necessary to swap inputs between two buffer objects; hence the name of "Ping-Pong"

- This method can be used to perform iterative computation on the GPU.
- Every sophisticated iterative technique in which it is necessary to know the output of the previous timestep will use this or a similar technique (smoke, water, clouds, fire, etc.)
- It is not uncommon to have multiple shaders performing different operations between each "Ping-Pong": Either chained per-frame, or alternating their execution

Game of Life

- The universe is an infinite two-dimensional orthogonal grid of square cells, each of which has two possible states: alive, and dead
- Each cell interacts with its eight neighbors, which are the cells that are horizontally, vertically and diagonally adjacent
- The behavior of these cells is governed by simple rules

Unity Texture Manipulation

- Unity provides a number of texture-manipulation methods

Method	Type	Relative Speed	Explanation
`SetPixels`, `GetPixels`	CPU	Fast	Inserts or extracts data from the texture. The texture must be available on the CPU.
`CopyTexture`	GPU	Fast	Moves data from one texture to another
`Apply`	CPU to GPU	Slow	Copy data from the CPU to the GPU
`ReadPixels`	GPU to CPU	Slow	Copeis texture data from GPU to the CPU
`Blit`	GPU	Fast	Triggers rendering into an offscreen buffer (the `RenderTexture`)

Texture Formats

- There are quite a few
- This class will concern itself primarily with `RGBA32`. Each **channel** (Red, Green, Blue, Alpha) is an 8 bit floating-point value
- See docs.unity3d.com/ScriptReference/TextureFormat.html for an itemized list of what Unity supports
- For certain computations, one may need more precision, and for that one can use `RGBAFloat` which offers full floating-point precision per channel