GPU & OpenGL Basics

GAME 300

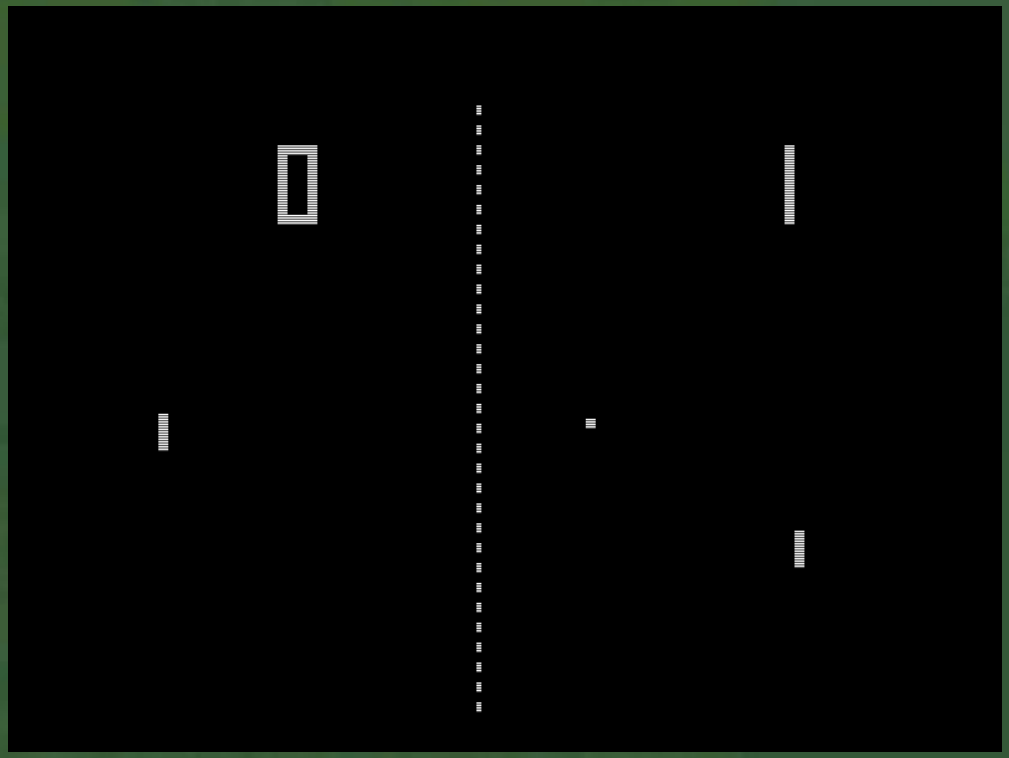
James Dupuis

Objectives

- Learn About:
  - History of Graphics
  - Graphical hardware basics
  - Basic of OpenGL / Computer Graphics

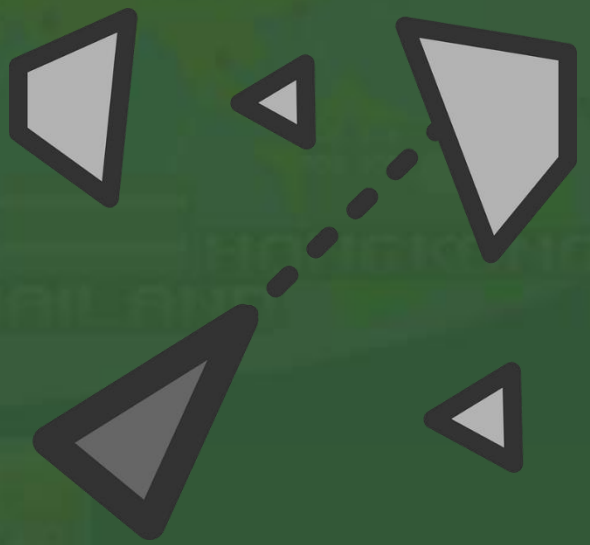
The Beginning

- **Pong** (1972):
  - Simple virtual ping pong game.
  - Black background
  - Primitive shapes rendered in white to convey users paddles, the ball and score.



The Beginning

- **Asteroids** (1979):
  - **Vector graphics** to display faster moving asteroids and enemies.
    - Resulted in sharp crisp lines instead of blocky pixel graphics all prior games had previously been programmed to use.
    - Cornerstone of graphics rendering discovering which paved the way for future 3d graphics.
  - One of first destructible environments in a game.



Galaxian (1979):
First game with multiple **colour**.

Console Graphics History

- Graphics of consoles had a few characteristics early on which defined how developers could create their games.
  - Restrictions based on hardware
- **Resolution**:
  - how many pixel rows by columns could the processor render to the screen
- **Colours Palette**:
  - How many colours the user could choose from at one time.
  - Sometimes a console would have hundreds of colours, but each palette would be limited to a much smaller number

LAYERS

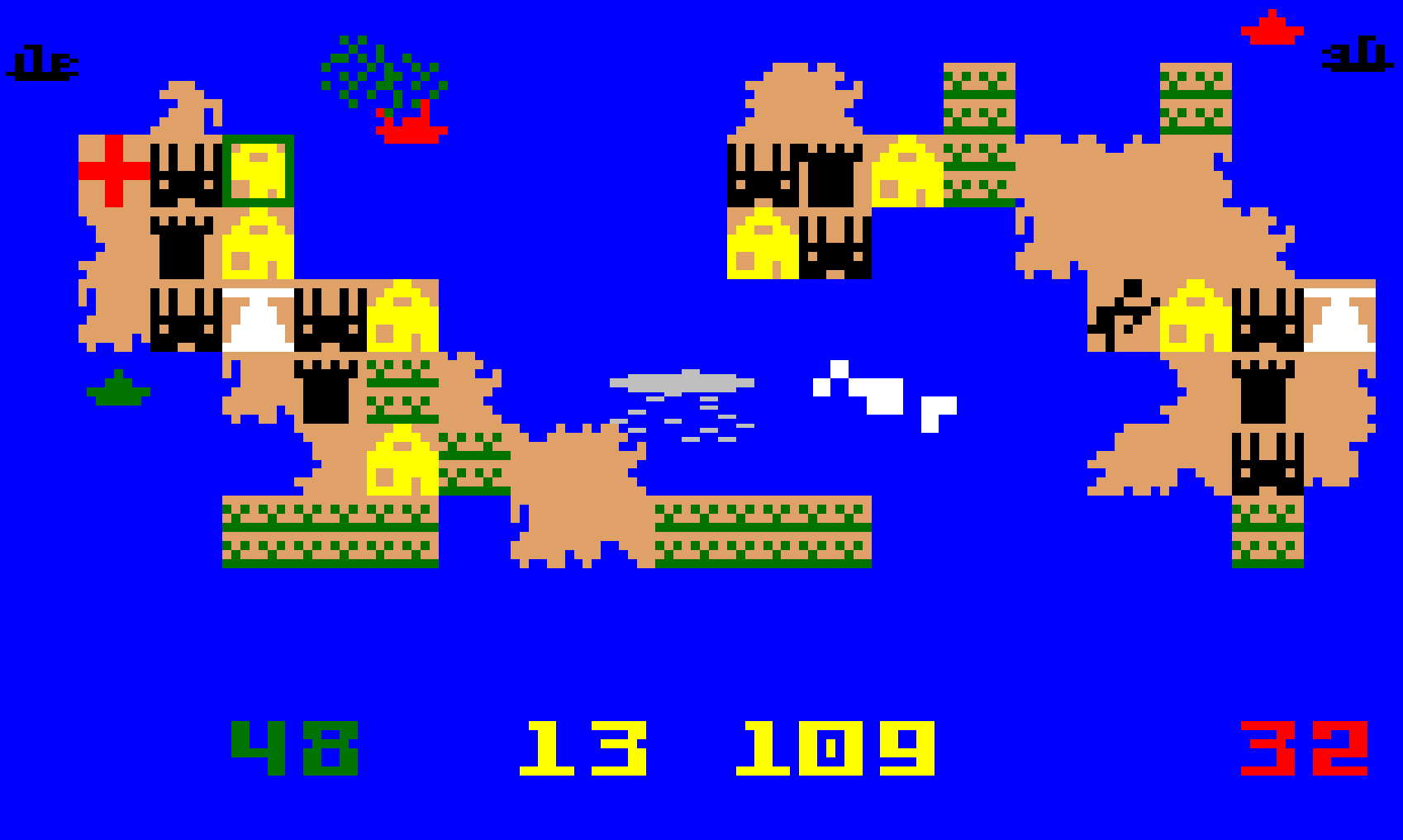
History

- **Tiles:**
  - background layer of sections of pixels.
  - Similar to a chess board.
  - Often used as an object which includes data to determine if Sprites can be placed on them or not.
  - Predefined limits for pixel sizes

- **Sprites:**
  - Rendered after the Tiles (over top of)
  - Used for characters or things the character might interact with.
  - Predefined limits for pixel sizes

UTOPIA SPRITE

History



Notable Console Games

History

- Vector based rendering was extremely rare in the early days of console games, with many developers choosing the safer tile and sprite based approach to rendering.
- Some developers opted to push the boundaries of the consoles setting individual pixel colours based off 3D math. (Vector Graphics)
- **Nintendo Entertainment System** (1985)
  Elite: https://www.youtube.com/watch?v=zoBIOi00sEI
- **SEGA Master System** (1985J/86/87) [GEN 3]
  F-22 Interceptor: https://www.youtube.com/watch?v=4m6pYzzCnwc
- **SEGA Genesis** (1989) [GEN 3]
  Virtual Racing: https://www.youtube.com/watch?v=jfGQT_pzfs0
- **SNES** (1993)
  starfox: https://www.youtube.com/watch?v=ZnmnpWKoUS0

3D History

- With the turn of Gen 5 of consoles 3D graphics emerged.
- Became apparent that 3D was the way of the future and Sprite based games were becoming less appealing.
- Early versions of shaders
  - 3d concepts possible
  - Graphics and colours were starting to become less of a concern with millions of colours available through the size of memory available at low costs.
  - Instead GPU's, CPU's and transfer rates became the bottleneck of advancing games graphically.
  - Lack of knowledge from the industry and investment in sprite based design and workflow required an adapting period.
  - Many programmers at the time were not familiar with many of the concepts possible with 3D calculations.
    - (Most still aren't)

PS1

History

- Sony Playstation (1994 /1995) [GEN 5]
  - Resolution: 640 x 480 pixels
  - Colours: 1.7 million colours
  - Sprites: 4,000 max, 256 x 256
  - Geometry Transform Engine – included in CPU
    - Allowed 3d math calculations built into the console.
    - Lighting, 3d geometry, matrix math for graphics programming and transformation calculations possible
  - Included a GPU: 32 bit
  - Large frame buffer for storing images before being supplied to the screen.
  - Capable of:
    - Fog
    - Rotation
    - Scaling
    - Alpha Blending
    - Clipping
    - Shading
    - Coloured lighting
    - More…

More Gen 5

History

SEGA Saturn (1994/95) [GEN 5]

32-bit processor
Resolution: 320×224 to 704×224 pixels
Colours: 16.77 million colours
Sprites: no limits / restraints
No 3D math engine
heavily sprite based like prior consoles
Couldn't compete with the PS1's true 3D games

- Nintendo 64 (1996/97) [GEN 5]
  - 64-bit processor
  - Resolution: 320 x 240 -> 720 x 576 pixels
  - Colours: 16,777,216 colours
    - Pallette: 2,097,152 max on screen
  - GPU: 64 bit
    - Similar capabilities to list from PS1

History

Big 3 War (Awkward Teenage Phase)

- Giant noticeable leap in graphics.
- Limitations become less about graphics for colour and resolution and more dependent on the players TV's
  - CRT screens are starting to be replaced by LCD/ Plasma TV's but are still the commonly used TV's in households.
  - Because of this, the following consoles still targeted CRT tv's for resolutions.
    - When a game with low resolution was later played on LCD, Plasma, LED Tv's, they looked pixelated and blurry because of stretching.
    - Playstation 2 (2000)
    - Microsoft Xbox (2001)
    - Nintendo Gamecube (2002)

- Example of changes from PS2->PS4
  - Note the drastic changes occurred from PS2-PS3.
  - https://www.youtube.com/watch?v=ix3z3xc7szU

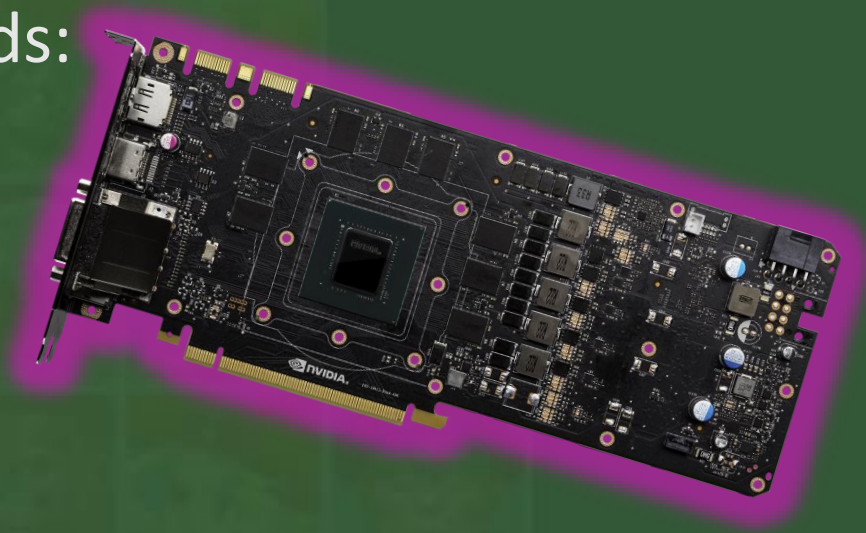

×00
×00
LIFE
8
WATER

History

GPU

- A GPU is a **Graphics Processing Unit**.
  - Similar to a CPU but generally dedicated to graphics.
  - Typically found on a graphics card (dedicated)
    - can also be built into a motherboard. (integrated)
  - The term GPU was first created by **NVidia** in 1999 with the GeForce 256 despite graphics processors existing for decades before without a common name.

- Popular Brands of graphics cards:
  - NVidia:
    - GeForce RTX 2080
  - AMD:
    - Radeon HD
    - Radeon RX (Vega)



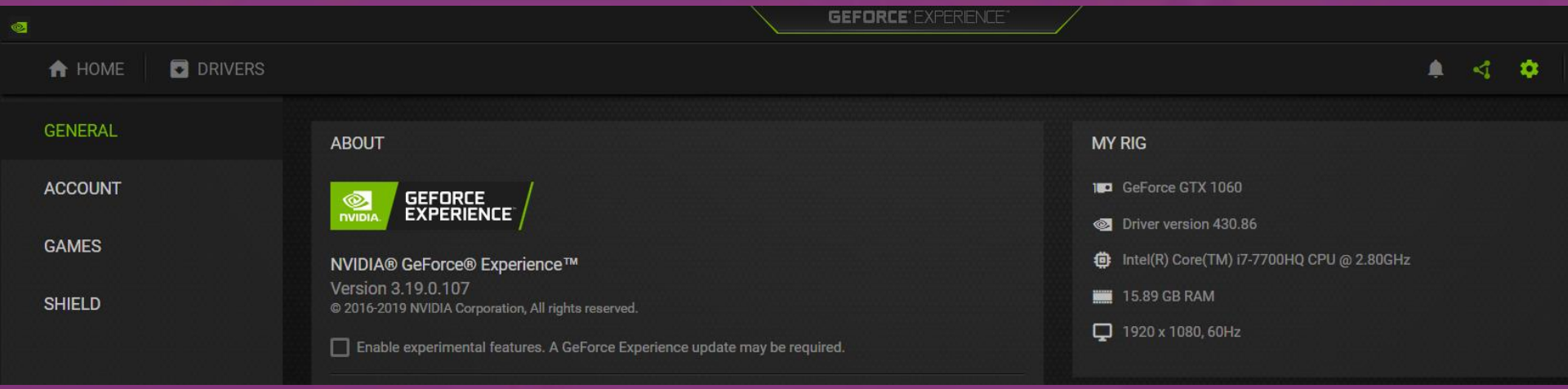
OpenGL

OpenGL

- What is Open GL?
  - A Graphics Library – library with API **(application programming interface)**
    - Exposed functions which allows programmers to render 3D worlds as 2D images on the screen.
- Usually packaged with Windows located in something like:
  - C:\Program Files (x86)\Microsoft SDKs\Windows\v7.1A\Lib**OpenGL32.Lib**
  - Open GL has access to manage hardware resources that control graphics processing.
    - handles multiple different types of hardware from console to pc to mobile
    - handles different resolutions and hardware properties to ensure a consistent product.

OpenGL (cont)

- Requires graphics drivers to be updated to handle new versions



- Most graphics drivers (last 5 year) support at least 4.1
- Contrary to the name it is **not** Open Source, however it is free to use and an Open API.
  - Encourages contributions from many graphics leaders.
- Works on both **Windows** and **Unix** based systems.
- Available in **C/C++, C#, Java, Python & Lua**.

OpenGL History

- Versions:
  - 1.0 (1992) ->2.0->3.0->**3.3**->3.5->4.0->4.6 (2017)
  - Typically supports backwards compatibility to older openGL versions.
    - Some changes have been force upgraded throughout time **(3.3)**
- **3.3** was a major revision which changed the way many things operate inside the library.

Level of Abstraction OpenGL

- Many game engines use OpenGL and Direct X (either)
  - The engine is interfacing with OpenGL to render the images you choose to create in the worlds.
- This is the layer OpenGL works on, the low->mid ground.
  - You could theoretically work on a lower level, interfacing directly with the hardware.
    - it would require you to know all the different hardware available and drivers interfaces for them.
  - This is what the OpenGL library handles for you.

GPU / CPU? & Shaders OpenGL

- OpenGL works on both the CPU and the GPU.
- Parts of the API are processed through the CPU and then it also interfaces and passes off data to the GPU.

- GPU's contain a bunch of tiny processors called shader processors
  - 100's -> 1000's
  - Run small applications (Shaders, more on them later)
  - GPU manages communication of processing from all these shader processors.
  - The CPU compiles programmable shader files through the CPU and runs them on the GPU.

Data Pipeline

Pipeline

- Data flows in a lateral movement / flow from one pipe or processor to another.

- Components of the pipeline include:
  - Processors:
    - Functions
      - api functions which modify or set the states for the OpenGL framework
      - Fixed function states
    - Shaders
      - Compiled programs which run on the GPU's shader processors.
  - Data Storage:
    - Textures
      - Images applied in 3d space
    - Frame buffers
      - Image still of data to be manipulated and applied to the screen.

OpenGL Program

- Runs on the CPU
- Has active states by default (more on this soon)
- Manages communicating data through the rendering Pipeline.
  - A series of Shaders which produces the end results of graphics displayed.
- Compiles shader code at runtime.
- Loads and applies shaders and executes draw calls

Summary

- The core functionality of a GPU is to make calculations on a smaller scale.
  - This allows us to process many pixels on screen at one time, asynchronously.
- OpenGL is an API which runs on the CPU and handles communicating data to the GPU for calculations and returning data back to our program for output.
  - This process uses things called shaders.

- Tomorrow:
  - Review the SDL Project
    - What is SDL?
    - How does it work?