Introduction to Computer Graphics

Computer Graphics: Concerned with all aspects of producing pictures or images using a computer.

- Began some 50 years ago with display of a few lines on a cathode-ray tube (CRT).
- Now, can generate images that are indistinguishable from photographs of real objects.
- Create feature length movies, entirely rendered by computers.
- Seamlessly integrate computer generated imagery into live action, also indistinguishable from live images.

Application Areas

Display of Information

This is just about anything you can think of. No specific example, because really, this incorporates any kind of graphics display.

Design

Think of things like the ability to create and display architectural diagrams, and consumer level visualizations (CAD). Also consider engineering tools such as printed circuit board layout, to provide a tool that allows an engineering to interactively create and visualize a design before sending to manufacturing.

Simulation, Animation, & Entertainment

Flight simulators used to train pilots is the typical example, but simulation is used in plenty of other places. With respect to animation, think of the special effects in movies, and even fully rendered movies (e.g., Toy Story, and everything Pixar produces).

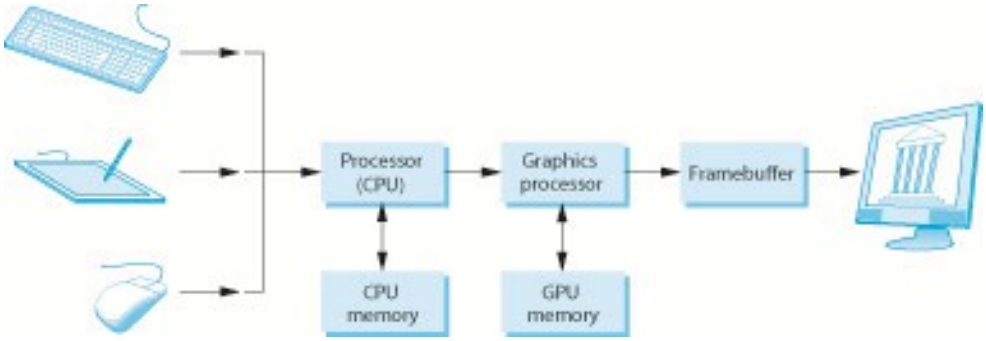
Of course, computer games and entertainment are dominated by the need to produce computer generated imagery. With the advent of VR, a host of new rendering techniques and technologies are being researched and developed.

User Interfaces

Your computer, phone, and now even watch all have a computer generated display that provides an interactive user interface for accessing its features and functions.

Graphics System Components

1. Input devices
2. CPU
3. GPU
4. Memory
5. Framebuffer
6. Output devices



Pixels & the Framebuffer

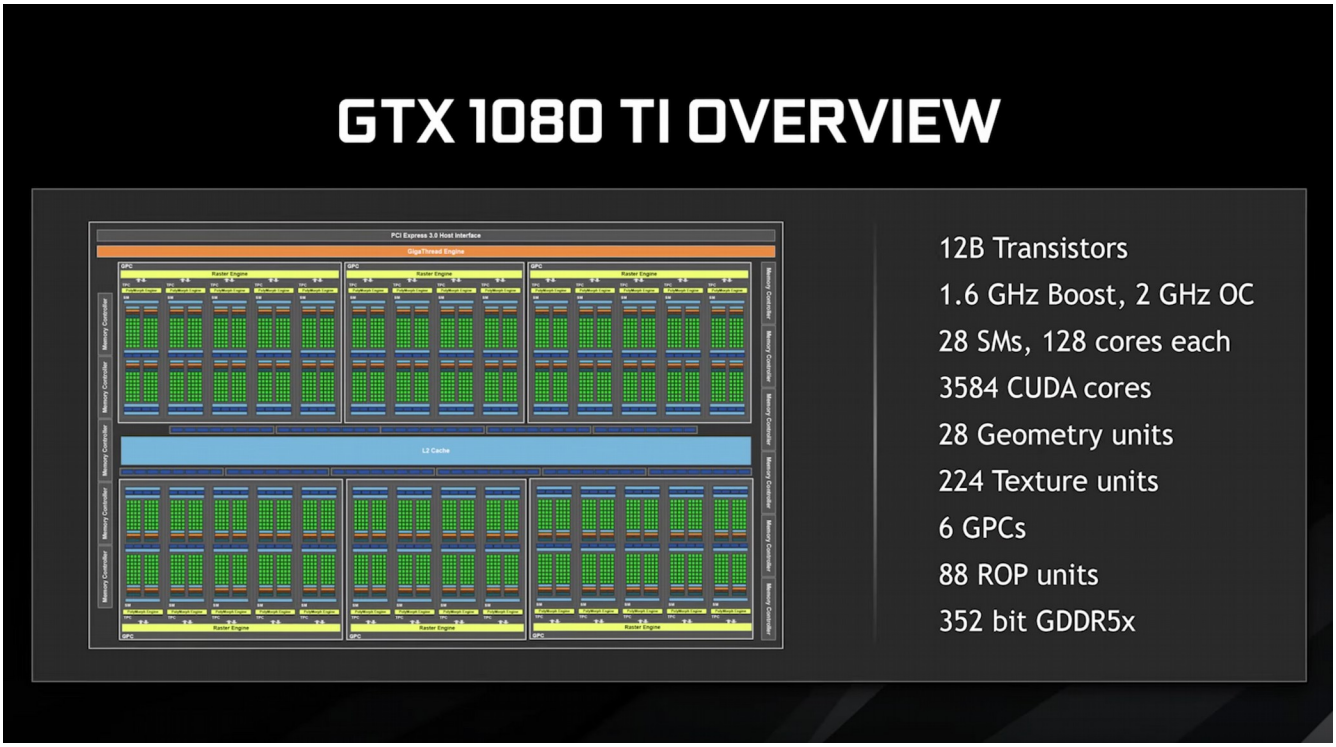
Modern graphics systems are **raster** based; meaning the image displayed is an array – the raster – of picture elements (**pixels**). Collectively, this array of pixels is called the **framebuffer**.

- The **resolution** – number of pixels in the framebuffer – determines the display detail of the image.
- The **depth** or **precision** of the framebuffer is the number of bits used for each pixel. This determines how many colors can be represented in the image. A **full-color** system has 24, or more, bits per pixel; with 8 bits for each of Red, Green, Blue (**RGB**) elements of the pixel. **High Dynamic Range** (**HDR**) is usually defined as having 12, or more, bits per color element, or 36 bits per pixel.
    - **Dynamic Range** : The ratio between the maximum and minimum intensities in the image/display. For example, the interior of a room with a sunlit view outside the window has a dynamic range on the order of 100,000 : 1. The max dynamic range of an image with 12 bits per color element is 4096 : 1; but most 12-bit camera sensors only have a dynamic range of 1000 : 1. Experiments indicate the human eye has a dynamic range of 10,000 : 1.
    - Talk about framebuffers using 8 bits per pixel or 16 bits per pixel...that the colors are represented using a color palette, and the value of the pixel in the framebuffer is a lookup index into the palette.
- A 1080p display is 1920 x 1080, or 2,138,400 pixels. At 24 bits per pixel, that is 2MB of memory.
- A 4K (Ultra HD) display is 3840 x 2160, or 8,294,400 pixels.
- For an HDR 4K at 12 bits per color channel, that works out to be 8,294,400 * 36 / 8 ~= 9MB of memory.
    - Note that “Cinema” 4K really is 4096 horizontal resolution, but vertical resolution isn’t specified.

Graphics Processing Unit (GPU)

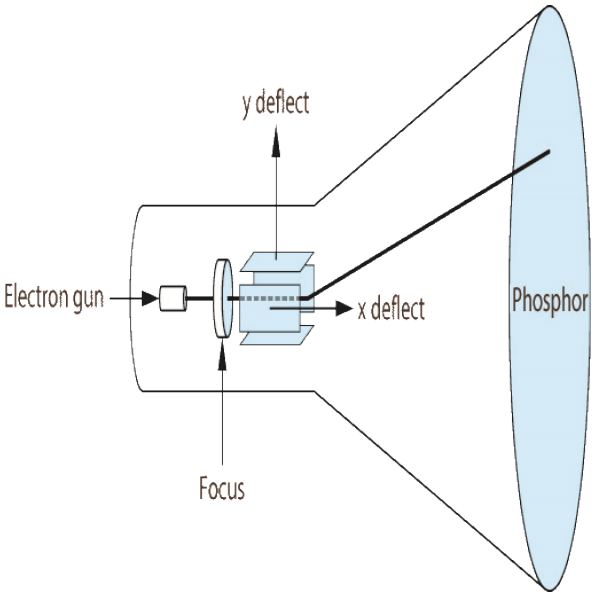
Specialized processing chip that performs graphics operations. Started life by doing **fixed function** graphics operations, but today are highly **programmable** chips that provide generalized vector compute capabilities.

- High-degree of parallelism, thousands of vector processing units inside of them.
- SIMD vector and matrix operations.
- Have their own dedicated memory, including holding the framebuffer(s) from which the display is generated.
- Performs some specialized graphics operations
  - Rasterization of triangles/polygons into pixels, which are then processed by the programmable pipeline.
  - Interpolate values from vertex of triangles, pass those values along with the rasterized pixels.



Output Devices

Historically displays (or monitors) were cathode-ray-tubes (CRT); see image on the right. Two ways images were drawn, and resolutions discussed: 1) Non-Interlaced and 2) Interlaced.



Non-Interlaced

Interlaced

Today almost everything uses some kind of flat-screen technology, where each pixel is independently controlled. In these displays, all pixel values are changed at the same time, there is no *scanning* as there is with a traditional CRT.

Refresh rate is how often the image is replaced on the display. LCD displays/monitors typically have a refresh rate of 60 Hz. TVs and higher end monitors have much higher refresh rates, up to 240 Hz.

Coordinate Representations

Need to understand the terminology used when speaking of different coordinate systems. The following are some to get started with, we'll learn a few more during the semester.

Device/Screen/Image/Pixel Coordinates

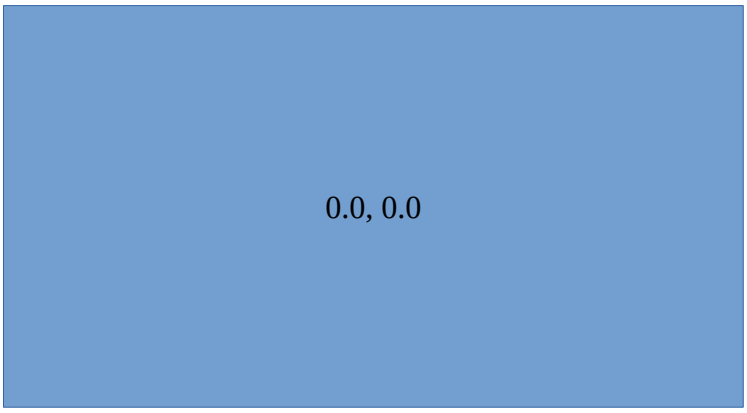
The underlying or fundamental resolution of the thing being rendered to. Usually we render to a framebuffer and then that framebuffer is drawn to a device. The framebuffer has some resolution and it may or may not match the resolution of the display.

World Coordinates

A unit coordinate system that allows for device independent representation of the data before it is rasterized for display.

Another way to represent world coordinates…

1.0, 1.0



-1.0, -1.0

Need the following mappings…

- World → Device (for rendering)
  - `X = DeviceSizeX / 2 + WorldX * DeviceSizeX / 2`
  - `Y = DeviceSizeY / 2 + WorldY * DeviceSizeY / 2`
- Device → World (for things like **picking**)
  - `WorldX = (DeviceX - DeviceSizeX / 2) / (DeviceSizeX / 2)`
  - `WorldY = (DeviceY - DeviceSizeY / 2) / (DeviceSizeY / 2)`

DeviceSizeX	1000
DeviceSizeY	1000

World to Device

WorldX	WorldY	DeviceX	DeviceY
-1	-1	0	0
0	0	500	500
0.5	0.5	750	750
1	1	1000	1000

Device to World

DeviceX	DeviceY	WorldX	WorldY
0	0	-1	-1
500	500	0	0
750	750	0.5	0.5
1000	1000	1	1