The Graphics Pipeline

Prashant Gupta

January 15, 2017

The idea of Computer Graphics came in 1950's post World War II after the invention of Cathode Ray Tube (CRT) when research started in the field for military application like flight simulators, but the term 'Computer Graphics' itself was coined in 1960's. In modern computing world the GPU's (Graphics Processing Unit) are becoming extremely popular, what started from stunning visuals and high performance games has now also extended its applications to Machine Learning & Deep Learning applications. Before understanding Graphics Pipeline and its various stages it is important to understand what does the word 'pipeline' mean. *Pipeline* means the various steps that needs to be performed on some entity to take it to its final form.
Graphics Pipeline or (rendering pipeline) means the steps that are performed to create a 2-Dimensional representation of a 3-Dimensional scene. Lets suppose we have a soccer ball lying on top of a table inside a closed room. This article describes how a computer with help of a GPU creates this scene on the computer screen and explains along the way what happens at each stage of the pipeline. The graphics pipeline has the following main steps -

1 Application

The function of the application stage is that it sets up the graphics pipeline. Now taking the example of the football placed on a table in the room, the entire room has two objects- table and football. The user specifies information like what are the dimensions of the objects, where are these objects placed in the room, where is the light source and from which angle the user looking at these objects.

Application Stage takes the data that is provided by the programmer and parallel processes it so that it can be understood by the GPU. For a computer system to understand the data, all the information entered by the programmer has to be converted into binary form (i.e. 0's and 1's).

2 Geometry/Vertex

To model the objects present in the room we need to understand how the computer system interprets the data. The Geometry or the Vertex stage takes the binary data and converts it into points in the 3-Dimensional world. These points can later in the pipeline are connected to form *vertices* which form triangles. Mesh of triangles form more complex shapes(3D Models) like football or table. Now the question arises why use triangles when a football in made of hexagons? As it turns out, Hexagons can be divided into combination of multiple triangles. The triangle is the most basic shape and can be connected to form any shape like squares, cubes, rectangles etc. Triangles also have mathematical properties that makes processing the models quicker.
Another important thing to note is that each of these 3D models are defined in its own coordinate space called model space and thus, no model has any information about any other models present in the room. Geometry stage combines all these models into a common world and then things like lightning and user's viewpoint can be taken into consideration by the GPU.
The user viewpoint is called camera, much like when we take a picture from a camera it converts a 3D space into a 2D photo, similarly the Geometry Stage takes the 3D space and converts it to a 2D image. The area of the room that are not covered in the photo are *clipped* or simply removed from the scene. At this point, the distance of various 2D models are calculated from the camera. This distance information is later used to calculate if something is overlapping, giving a sense of depth, lightning among other things.

3 Rasterization

In this stage the various points (specified in the Geometry stage) are joined to form triangles. The overlapping information is used to calculate what parts of what models are displayed in the picture. The depth data is also used to make objects in the front look bigger compared to the objects present in the back. Mesh of triangles form 2D shapes of football and table. A picture consists of 2D grid of very small square blocks called pixels. The process of converting mesh of 2D triangles into pixels is called *rasterization.* Each of the small block inside the rasterized model is called a *fragment.* The GPU also utilizes a *programmable shader* called Vertex shader so that the blocky edges of model look smooth in the photo.

4 Texture

After Rasterization, textures are applied to the models. Textures are wrapped around the surface of a model to give a feeling of 3D in the photo. Textures are also a good way to show imperfections in the model, like if you want to show some marks on ball from playing. At this point GPU can utilize another *programmable shader* called pixel shader to calculate how light reacts with each pixel making the photo look much more detailed and realistic. Thus, they define the final color that a pixel is going to get.

5 Composite

After Clipping, Rasterization, Coloring and Texturing the final image is created. Frame Buffer is used by the GPU to hold each pixel data, that is going to be displayed on screen. Each model data is copied to the frame buffer one by one until frame buffer has data for all the pixels.

6 Display

To finally display the image to the user, the frame buffer is read and each pixel data is sent to the screen starting from top left corner moving from left to right until bottom right corner of the screen is reached. This entire process starting from processing the code till the final image is displayed on the screen is called *rendering*. For videos multiple images are rendered per second to give an illusion of motion.