Thomas Fiocchi

CS 147

Professor Urness

The Graphics Pipeline

The graphics pipeline is a conceptual model that is used to help visualize and understand the path that a graphics system must go down in order to fully render and type of image. The pipeline takes a three dimensional image and is able to transfer it onto a two dimensional screen. This process is used in developing video games, creating animation films and projects, and has many other practical applications within the world of computer science. Application programming interfaces are used to help streamline this process for anyone who is working with graphics code by providing graphics libraries for the user to implement in their code. This also helps prevent the occurrence of errors within the code that could potentially disrupt the entire rendering process. The pipeline itself does not have a predetermined number of stages because this number can vary depending on the type of program being created or implemented by the user. The process is similar to that of an assembly line that works in conjunction with the graphics processing unit (GPU) that is able to run millions of tiny calculations in a matter of mere seconds, down to even the millisecond, in order to process and generate the image desired. This is done by creating lines between different points that are specified by the user, and filling in the pixels on the screen that are encompassed by these lines. For example, creating a triangle would require 3 different lines that share two points between each line, and would then fill in the pixels within those lines, almost like a coloring book in a sense that you don’t want to color outside the lines. These color values are created by what's called a frame buffer that stores color information that is read by the GPU and then translated into different colors during the graphics pipeline process. In going down this process or assembly line, if you will, the GPU goes through many different stages, such as the tessellator and rasterization stages, that allow for the creation of the image. If there is a problem with any of these stages within the code, it can result in a poorly generated image, or no image being generated at all. WIthout the use of these graphics pipeline visualizations, it would be far more difficult to imagine how such a complicated process would work, especially as a student experiencing something like this for the first time. Thanks to the continued development of more complex computing machines and graphics programs there are limitless possibilities for developers and casual users alike to create incredible works of art, whether it be a new video game with revolutionary water shaders, or a simple animation project for a class. New doors are being opened every day by ordinary users and professionals alike to express their creativity through graphical development, but none of this would be possible without the concept of the graphics pipeline that takes this complex process and makes it easier to understand. As time progresses, the process will become more and more streamlined allowing increased user friendliness and more creative minds to flourish.

Sources:

<http://www.ntu.edu.sg/home/ehchua/programming/opengl/cg_basicstheory.html>

<https://www.pcmag.com/encyclopedia/term/43933/graphics-pipeline>

<https://stanford.edu/class/ee267/lectures/lecture2.pdf>

<https://docs.microsoft.com/en-us/windows/desktop/direct3d11/overviews-direct3d-11-graphics-pipeline>