Jacob Colby

Computer Graphics

02/14/2021

Assignment 2: The Graphics Pipeline

The graphics pipeline is the process and steps that need to be taken to render 3D graphics. The pipeline has seven main steps. The first step starts with the data setup. Setting up the data starts by creating data for the required geometry which will then be passed to shaders as buffers (a region of memory to temporarily store/transfer data). The second step is the execution of a vertex shader on each vertex of the geometric model. This means that each vertex’s position of a primitive polygon is calculated and stored using **gl\_Position.** The vertex shader also calculates vertex color, texture coordinates and vertices. Step three is to clip away any data that is outside the field of view of the actual camera. The fourth step is to map the 3D object coordinates into pixel coordinates for the final rasterized image. Fifth, we rasterize a geometric primitive by determining which pixels in the raster image are in the boundaries. Culling is one of the aspects of rasterization where the orientation of the polygon is determined and any triangles facing away or are not visible are discarded. This is done to prevent any unnecessary work on the GPU since anything outside the FOV or facing away from the camera won’t be seen. The sixth step executes the fragment shader on each pixel which outputs a color value for each pixel. The fragment shader gets some important information: data from each vertex in the form of varying variables, primitives from the rasterization stage, and finally calculates the color values for each pixel between the vertices. The color values that are stored with the fragment shader will be used by the fragment operations in the next step. The seventh and final step before the output image combines the fragment shader pixel color with the draw buffer pixel color. This step is carried out after the primitive pixel color is determined for each pixel. This operation can include depth, color buffer blend, and dithering. Once all the steps are completely processed, a 2D image will be displayed on the screen, but the frame buffer is the final destination in the pipeline. The frame buffer holds scene data in memory such as height and width of the surface, color of each pixel, and depth/stencil buffers. Overall the pipeline is a very detailed process that is able to use JavaScript to write WebGL applications in order to have more control of the process.

Sources:

<https://www.tutorialspoint.com/webgl/webgl_graphics_pipeline.htm>

<https://runestone.academy/runestone/books/published/learnwebgl2/01_the_big_picture/3_3d_graphics_pipeline.html>