
CPU Code (C++)
vertex position in local space
Vertex Data[]
A collection of data per 3D coordinate (could be position, tex. coord., normals, color, ...)
viewM * modelM * vec4(local_space_pos, 1.0)
vertex position in view space
projectionM * view_space_pos
vertex position in clip (homogeneous) space
clip space is intermediate space before NDC, since it sets up positions to undergo clipping and perspective division
Vertex Shader
Transforms vertices to clip space;
Outputs per-vertex data (normals, UVs, ...)
Tessellation?
Yes
Tessellation
No
Geometry shader?
Yes
Primitive Assembly
Additional shape assembly step to form shape input for the geometry shader
Geometry Shader
Can transform primitives or add additional ones
No
Primitive Assembly
Assembles output for the rasterizer
Which primitive we chose (Line, Triangle) in the draw call command
Primitives outside the clip volume, which was specified by the projection matrix, are discarded
Clipping
X, Y, Z components of the clip space position are divided by the W component to produce that perspective feel and convert to NDC space
Perspective Division
Face culling?
Yes
Face culling for closed objects
Additional step to omit back-facing primitives of closed objects
Done to prevent unnecessary processing of primitives' fragments that will be then omitted during depth testing either way
No
Now that coordinates are converted to [-1, 1] range, they are mapped to screen coordinates (screen space) according to the size of the window
Viewport Transform
Rasterization
Fragments are generated - fragmentation
MSAA Enabled?
Yes
Multi-Sample Anti-Aliasing
Generates a given number of samples for a fragment
Only one sample is available per fragment
No
Early Depth/Stencil Testing?
Yes
Early Depth/Stencil Testing
Possible under some conditions;
Done to increase performance by running less fragment shaders
Done per sample
No
Done only if any samples pass
OpenGL interpolates between the given primitive's vertices output data (pos, normal, UVs, ...) to produce the correct values to be then used as input for a given fragment in the fragment shader
Fragment Shader
Per-Sample Operations
Early Depth/Stencil Testing Skipped?
Early Depth/Stencil Testing was not possible
Yes
Stencil Testing Enabled?
Yes
(Late) Stencil Testing
Compares a reference value against the current content of the stencil buffer at the same position
Used for masking to determine if a fragment should pass on or not;
One application is outlining
No
Stencil Test Failed?
Yes
Fragment is discarded
No
Early Depth/Stencil Testing was possible
No
Depth Testing Enabled?
No
Yes
(Late) Depth Testing
Compares the depth value of the current fragment against the content of the depth buffer at the same position
Depth Test Failed?
Yes
Fragment is discarded
No
If the test is successful, the fragment passes on, the frame buffer is updated with the given fragment's data and the depth buffer is updated with the new depth value from the fragment
Blending Enabled?
Yes
Blending
Performs the blending equation by taking input from the current fragment shader values and the values stored in the framebuffer at the same position
No
MSAA Enabled beforehand?
Yes
Multi-Sample Anti-Aliasing Resolve
Depending on how many samples out of all generated are within the assembled primitive, that is how much influence will the fragment color have over the final output color for the pixel
No
Additional Framebuffer?
Yes
Write to Framebuffer
Do Post-Processing
No
Final Output (Swap Buffers)