Oregon State University

CS 450/550 -- Fall Quarter 2025

Project #7

100 Points

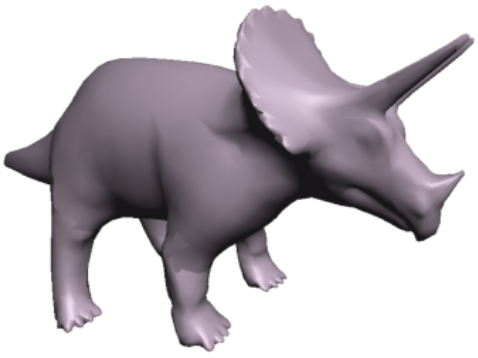
Due: December 8 -- No Bonus Days

Shaders, II

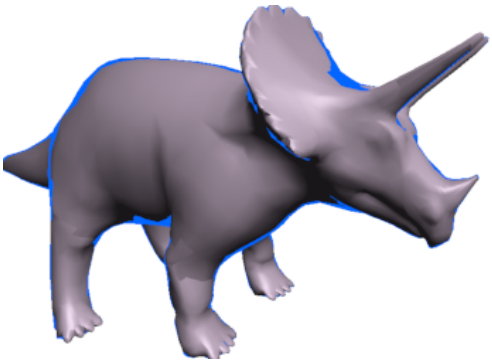
This page was last updated: September 12, 2025

I created a summary document describing the steps in using Shaders. It is called Shader Steps. Check it out. It might help you!

Introduction:



Surface, no Silhouette



Surface, Silhouette

No Surface, Silhouette

The goal of this project is to use a GLSL fragment shader to show a shape's silhouette edges. You need to be able to toggle the silhouette on and off. You need to be able to toggle the solid surface of the object on and off. You also need to be able to adjust the amount of tolerance the shader uses when discerning the silhouette edge. Your fragment shader needs to implement per-fragment lighting on the surface, but not on the silhouette edges.

Learning Objective:

When you are done with this assignment, you will understand how to create a GPU-program, known as a *shader*. Your fragment shader will let you display silhouette edges and use per-fragment lighting on the surface.

Instructions:

1. You can use any 3D object you want that has surface normals. Pick something with a lot of surface geometry variation in it. Computing silhouette edges on a sphere is pretty uninteresting. The dino.obj file is good for this. So is cow.obj.

File	Triangles
dino.obj	5,660
cow.obj	5,804

   Left-click to look at the OBJ file. Right-click to download it.

2. Use our GLSLProgram C++ class to create a shader program from your vert and frag files.
3. Write a vertex shader that that sets your object up for per-fragment lighting.
4. Write a fragment shader that:
   1. Is able to perform per-fragment lighting on the surface of the object
   2. Toggles that lit surface on and off under program control
   3. Is able to draw a solid color whenever it decides that a fragment is on the silhouette.
   4. Toggles that silhouette on and off under program control
5. Uniform variables:

Variable	Type	Purpose
uSurface	bool	Whether or not to display the solid surface
uSilh	bool	Whether or not to display the silhouette
uTol	float	How close to 0.0 the dot product must be to see the silhouette

6. **uSurface** and **uSilh** are bools, that is, true or false. That means that your C++ program can use keyboard keys to toggle those global variables on and off using the *not* operator (the exclamation point).

7. **uTol** is a float. It can be set by animating its value. It can also be set by using keyboard keys to increase or decrease its value.

8. How do you know if a fragment is on the silhouette edge? You check the angle between its interpolated surface normal and a vector from the eye to its interpolated position. The closer that angle is to 90°, the more likely it is to need to be drawn in the silhouette solid color. See **what this situation looks like.**

9. You need to be able to change the tolerance under program control. This does could be continuous, such as animating its value based on *Time*. But it could also have discrete values, such as toggling between a low value and a high value.

10. The fragment shader also needs to have the per-fragment lighting information: Color, SpecularColor, Ka, Kd, Ks, and Shininess. These can be hard-coded. They can also be animated if you so choose.

11. To help you get started, here are some program skeletons to work from:
    - silh.vert
    - silh.frag

The GLSLProgram C++ class

The glslprogram.h and glslprogram.cpp files are already in your Sample folder. You just need to un-comment their #include's.

See the class Shader notes for how to use the GLSLProgram C++ class.

Per-Fragment Lighting

You need to do per-fragment lighting. The sample vert and frag programs are already setup for per-fragment lighting.

Turn-in:

Use Canvas to turn in:

1. Your .cpp, .vert, and .frag files
2. A short PDF report containing:
    - Your name
    - Your email address
    - Project number and title
    - A description of what you did to get the display you got
    - A couple of cool-looking screen shots from your program.
    - Tell us what convinces you that your animation is indeed doing what you set it up to do.
    - The link to the Kaltura video demonstrating that your project does what the requirements ask for. If you can, we'd appreciate it if you'd narrate your video so that you can tell us what it is doing.

3. To see how to turn these files in to Canvas, go to our Project Notes noteset, and go the the slide labeled **How to Turn In a Project on Canvas.**

4. **Be sure that your video's permissions are set to *unlisted.***. The best place to set this is on the OSU Media Server.

5. A good way to test your video's permissions is to ask a friend to try to open the same video link that you are giving us.

6. The video doesn't have to be made with Kaltura. Any similar tool will do.

Grading:

Item	Points
Able to make some sort of "edge" appear	10
Able to make a silhouette edge appear in the correct places	30
Able to toggle the silhouette edge on and off	10
Able to change the tolerance on finding the silhouette edge	20
Able to toggle the surface on and off	10
Per-fragment lighting works on the surface	20
Potential Total	100