

Lighting, Culling, Color, & Mapping – Assignment 5

Due: November 21, 2007

Fall 2007 Computer Science 484
Principles of Computer Graphics

Instructions: Please submit your response to this assignment on **November 21, 2007** at the beginning of class. Your submission must, at a minimum, be cleanly typeset, easy to read and properly presented. Assignments spanning multiple pages should be stapled. All pages must include a header identifying the student, the assignment and the student's contact information. A footer must be included showing the page number. Should you need to diagram your response, you may insert a diagram drawn freehand. A machine assisted diagram is preferred.

Plagiarism and academic dishonesty will not be tolerated. Correctly and properly attribute all 3rd party material and references.

Lighting

Scene

Consider the tetrahedron with the vertices $a = (-10, 0, -10)$, $b = (0, 0, -20)$, $c = (0, 0, 0)$, $d = (-5, 10, -5)$. The faces of the tetrahedron are abc , dba , dcb , dac . Note that all the faces are order in a counter-clockwise direction denoting a right-hand coordinate system.

Model Transformation

First the tetrahedron is rotated about the Y axis in the counter-clockwise direction by 90° , R . Then the tetrahedron is translated in the positive X direction by 20 units, T . The model transformation is called M_m .

View Transformation

The view transformation, M_v , is the identity matrix I .

$$M_v = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Projection Transformation

The perspective projection transformation, M_p , is the following matrix.

$$M_p = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & -2 & -4 & 0 \\ 0 & 0 & -3 & -8 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Material Properties

Tetrahedron's material properties are the following values.

- $O_a = (0.8, 0.7, 1.0)$
- $O_d = (0.2, 0.3, 0.4)$
- $O_s = (0.5, 0.5, 0.5)$

- $k_a = 0.8$
- $k_d = 0.25$
- $k_s = 1.0$
- specular exponent $n = 3$.

Light

There is one light in the scene at $(15, -20, 5)$ with color $I_d = (1.0, 1.0, 0.0)$ and the specular component $I_s = (1.0, 1.0, 1.0)$. The ambient light in the scene is black, $I_a = (0.0, 0.0, 0.0)$.

As a reminder, the lighting equation is

$$I_\lambda = I_{a\lambda} k_a O_{d\lambda} + \sum_{1 \leq i \leq m} f_{att_i} I_{p\lambda_i} (k_d O_{d\lambda} (N \cdot L_i) + k_s O_{s\lambda} (R_i \cdot V)^n)$$

Note

Feel free to use Matlab, Octave, Mathematica or other computational tools to aide you in completing this assignment. Please provide listings of any code you use to assist you in developing an answer. Please provide the steps taken to come upon your answer regardless of whether a computational tool was used or not.

1. How do you compute the normal at face abc ? Compute and show the normal vectors at all the four faces and all the four vertices. In what situations would you use each type of normal (face normal/vertex normal)? What are the effects on the shading of the tetrahedron depending on the normal used?
2. Choose the correct answer: If P is a vertex (one of a, b, c , or d in homogenous coordinates), then after model transformation, the transformed vertex is computed as (a) $R \times T \times P$ (b) $T \times R \times P$.
3. Compute and show the composite model transformation M as described under the heading *Model Transformation* above. Compute and show the transformed vertices A', B', C' and D' , after applying the model transformation (in world coordinates).
4. Compute the normal vector of face abc after model transformation, using the coordinates of the transformed vertices a', b' , and c' .
5. The homogenous coordinate of a point $P(x, y, z)$ is $(x, y, z, 1)$ and the homogenous coordinate of a vector $V(x, y, z)$ is $(x, y, z, 0)$. Transform P and V using the model transformation matrix computed in question 3. Are the transformed coordinates of P and V the same? Perform the same operation, but use only the rotation transformation described under the heading *Model Transformation*. Are the results the same now? Interpret your results.
6. Using insights from the previous question, how would you compute the normal at each vertex after the model transformation (in world coordinates)? Compute the vertex normal for vertices a, b and c using what you have observed.
7. In what stage of the pipeline is the lighting computation performed? Knowing that the coordinates of the light source are given in world coordinates, what transformations should be applied to the position of the light in order to correctly calculate the lighting at that stage? Why doesn't the light's position necessarily go through the same transformations as the vertices of the tetrahedron?
8. Even though OpenGL computes it at a different stage of the pipeline, you can know the lighting of a particular vertex earlier, as long as the position of the light, the position of the vertex and the normal are in the same coordinate system. Take a vertex $v = (1, 2, 3)$ with normal $n_v = (0, 1, 0)$ in world coordinates. Assume its material properties are those of point a in the tetrahedron, and the light source is also as described above, in world coordinates. Compute the resulting color of this vertex at world coordinates (without further transforming any coordinates) using the illumination model which adds ambient, diffused and specular components to calculate the total illumination.
9. Compute the illumination of the vertex in the previous question, but we want to do it after the view and the projection transformations have taken place (in projected coordinates). Transform the necessary coordinates from the previous

question (from world to projected coordinates), and compute the illumination of vertex v . Do you get the same results? Interpret your results.

10. Compute the illumination in world coordinates at the centroid of face abc by (a) Gouraud shading (interpolating the color at the vertices). (b) Phong shading (applying the lighting equation at the centroids using the normal vectors computed in question 6). (These are not exactly OpenGL Gouraud or Phong shading, as these shadings are actually done in OpenGL in 2D after perspective transformation of the triangles. In this assignment, you are doing this interpolation in 3D. The concepts are same.)
The centroid can be computed by averaging the vertices of the triangle. For example, in Matlab or Octave you can use a function such as the one listed below.

```
# a, b, c are three component vectors, it returns the centroid p
function p = centroid( a, b, c )
    p = (a + b + c) / 3
endfunction
```

11. Using view transformation matrix, M_v , perspective transformation matrix, M_p , and the model transformation M_m that you computed in question 3, you can compute the projected coordinates of the vertices of the tetrahedron. Compute these projected coordinates for all 4 vertices, a, b, c, d . Which of the points will be visible, and which ones will be clipped? How do you know?
12. You are in an empty room (just 4 walls, floor and ceiling - no windows), with a single light bulb. You carefully measure the dimensions of the room, the color of the paint on each wall, and the radiant intensity of the light bulb. You use these measurements to construct an OpenGL model of the room, and render the scene. Nevertheless, the rendering will not look like a photograph of the room. Why not?

Color

1. You connect your computer to a portable LCD projector. When you switch on the LCD projector you observe that it is projecting predominantly blacks and purples. You make an educated guess that one of the wires connecting to the primaries R, G and B may be malfunctioning. Which one is it and why?
2. Define additive-color and subtractive-color. Provide an example of a color model for each one and a real world instance of its use. In OpenGL, the color model is RGB. Is this an additive-color or subtractive-color model?

Culling

1. How does one find back facing triangles?
2. Silhouette edges are the edges in the manifold that have one back-facing polygon and one front facing polygon incident on it. How do you compute the silhouette edges of a manifold?
3. In OpenGL you can draw only back-facing polygons, only front facing polygons or both. If you render the manifold, then clear the frame-buffer but not the depth buffer, then again render only the back facing polygons. What do expect to see? Feel free to provide a screen capture from a demonstration program to support your claim.

Mapping

1. List and describe at least three types of mapping that you can do in OpenGL. Define *forward-mapping* and *inverse-mapping*.
2. You change the normal vectors of an OpenGL triangle, but do not change the position of the vertices. Which components of the color seen by the viewer (ambient, diffuse, and specular) might change? Why?
3. You are rendering a black and white checkered tiled floor using a single texture mapped polygon. The view is simulating a person standing on the floor and looking down on the floor at a long distance away from him. Do you expect to see any artifacts at the distant end of the floor and why? If so, how would you remove these artifacts?

4. What is an approach to avoid aliasing while texture mapping a surface?
5. You want to map a texture to an arbitrary surface. The surface is has a right hand orientation and is a manifold. Describe two ways in which you can map the texture to this surface. Provide diagrams if necessary.
6. Environment mapping is computationally less expensive than ray tracing. Describe why environment mapping is a useful stand-in for ray tracing. Describe the short comings of environment mapping with regard to ray tracing.