Tutorial 6
CS3241 Computer Graphics (AY23/24)

October 16, 2023

Wong Pei Xian

✉ e0389023@u.nus.edu

Question 1

In which pipeline stage and in which coordinate space is the OpenGL **built-in lighting computation** performed?

> OpenGL's built-in lighting computation is performed when `GL_LIGHTING` is enabled.

Question 1

- vertex processing: We do calculations per vertex
- camera space: We need the view direction, it's defined after camera space

The Mathematics of Lighting

Advanced

This section presents the equations used by OpenGL to perform lighting calculations to determine colors when in RGBA mode. (See "The Mathematics of Color-Index Mode Lighting" for corresponding calculations for color-index mode.) You don't need to read this section if you're willing to experiment to obtain the lighting conditions you want. Even after reading this section, you'll probably have to experiment, but you'll have a better idea of how the values of parameters affect a vertex's color. Remember that if lighting is not enabled, the color of a vertex is simply the current color; if it is enabled, the lighting computations described here are carried out in eye coordinates.

Reference at RED BOOK

QUESTION 1
○○
QUESTION 2
●○○
QUESTION 3
○○
QUESTION 4
○○
QUESTION 5
○○○○○
QUESTION 6
○○○○○○○
QUESTION 7
○○○○○○○○

Question 2

We are using Phong Illumination Equation

$$I_{\text{Phong}} = I_a k_a + I_p k_d (N \cdot L) + I_p k_s (R \cdot V)^n$$

to compute a color for vertex *v*. Given that the material at *v* is **totally diffuse**, which of the following will result in a change in computed color at *v*?

Which will result in change of color of a diffuse material?

$$I_{\text{Phong}} = I_a k_a + I_p k_d (N \cdot L) + I_p k_s (R \cdot V)^n$$

1. Moving the light source further away from v, while maintaining the direction L.
2. Moving the light source to a different direction from v, while maintaining its distance from v.
3. Moving the viewpoint to a different direction from v.
4. Rotating the normal vector N.

QUESTION 1
OO

QUESTION 2
OOO

QUESTION 3
OO

QUESTION 4
OO

QUESTION 5
OOOOO

QUESTION 6
OOOOOOO

QUESTION 7
OOOOOOOO

Which will result in change of color of a diffuse material?

$$I_{\text{Phong}} = I_a k_a + I_p k_d (N \cdot L) + I_p k_s (R \cdot V)^n$$

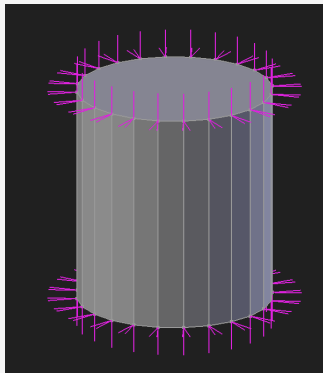1. Moving the light source further away from v, while maintaining the direction L. False. No values changed

2. Moving the light source to a different direction from v, while maintaining its distance from v. True. Changes L

3. Moving the viewpoint to a different direction from v. False. Changes V

4. Rotating the normal vector N. True (assuming rotating about an axis that's not itself)

Question 3

A closed cylinder (with top and bottom caps) is approximated by 32 rectangles on its smooth curved surface. Each cap is represented as a fan of 32 triangles, all sharing a vertex at the center of the cap.

How many **distinct** vertex normal altogether are needed to model the cylinder?

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○●

QUESTION 4
○○

QUESTION 5
○○○○○

QUESTION 6
○○○○○○○

QUESTION 7
○○○○○○○○

Vertex Normals of a Cylinder

Question 4
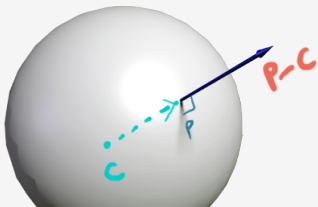
A sphere, centered at the location C, is approximated by a polygon mesh. A vertex v is on the surface of the sphere, and has the location V. Suppose vertex v is shared by exactly 3 polygons whose polygon normals are N1, N2, and N3.

Show two ways to compute the *vertex normal* at v. Normalize your results to unit vectors.

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○●

QUESTION 5
○○○○○

QUESTION 6
○○○○○○○

QUESTION 7
○○○○○○○○
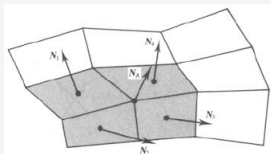
Question 4
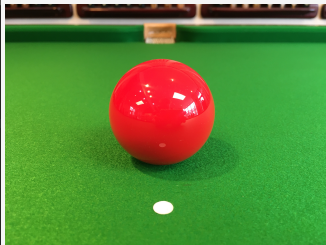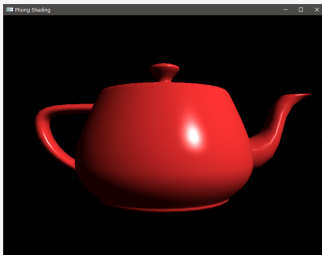

$$N_v = \frac{N_1 + N_2 + N_3}{|N_1 + N_2 + N_3|}$$

$$N_v = \frac{v - C}{|v - C|}$$

Question 5

Find the values for k_d, k_s, n to model the following:

1. Red shiny plastic (billiard ball)
2. Dry white clay
3. Brushed bronze

QUESTION 1
OO

QUESTION 2
OOO

QUESTION 3
OO

QUESTION 4
OO

QUESTION 5
O●OOO

QUESTION 6
OOOOOOO

QUESTION 7
OOOOOOOO

Red shiny plastic

1. $k_d = [0.9, 0.2, 0.2]$
2. $k_s = [1.0, 1.0, 1.0]$
3. $n = 64.0$

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○○

QUESTION 5
○○●○○

QUESTION 6
○○○○○○○

QUESTION 7
○○○○○○○○

Dry white clay
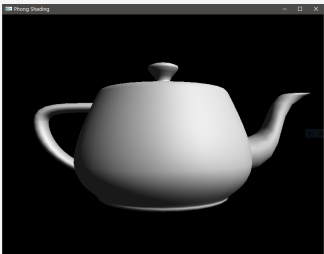
1. $k_d = [0.8, 0.8, 0.8]$
2. $k_s = [0.1, 0.1, 0.1]$
3. $n = 1.0$

QUESTION 1
OO

QUESTION 2
OOO

QUESTION 3
OO

QUESTION 4
OO

QUESTION 5
OOO●O

QUESTION 6
OOOOOOO

QUESTION 7
OOOOOOOOO

Brushed Bronze
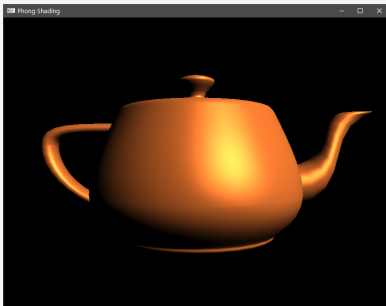
1. $k_d = [1.0, 0.5, 0.2]$
2. $k_s = [1.0, 0.5, 0.2]$
3. $n = 8.0$

Points to note

- If object is completely diffuse: $k_s = [0, 0, 0]$
- If object is mostly diffuse: Lower value of k_s
- Metallic: $k_d = k_s$
- Shininess (smaller highlights): Higher n

Question 6

Consider a variant of the Phong model, called the Blinn-Phong model, whose computation is

$$I_{\text{Blinn-Phong}} = I_a k_a + I_p k_d (N \cdot L) + I_p k_s (N \cdot H)^n$$

H is called the Half vector and it is the vector exactly in the middle of L and V.

Blinn Phong *H* vector

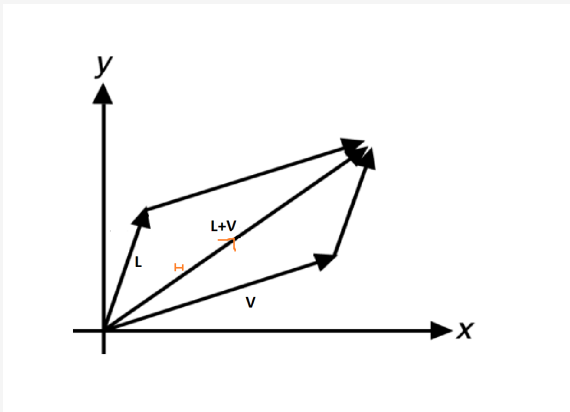Write an expression to compute *H*. Note that *H* must be a unit vector.

QUESTION 1
OO

QUESTION 2
OOO

QUESTION 3
OO

QUESTION 4
OO

QUESTION 5
OOOOO
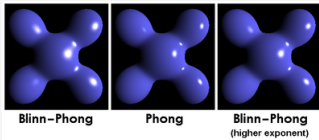
QUESTION 6
OOO●OOO

QUESTION 7
OOOOOOOO

Blinn Phong H Vector

$$H = \frac{L + V}{|L + V|}$$

Blinn Phong vs Phong

What is the difference in appearance between a surface rendered using
Phong model and the same surface rendered using Blinn-Phong
model?

The specular highlights from Blinn-Phong model will be relatively larger.

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○○

QUESTION 5
○○○○○

QUESTION 6
○○○○●○○

QUESTION 7
○○○○○○○○

Computation of specular term

Is there an advantage in using $N \cdot H$ over using $R \cdot V$ to compute the specular term? If yes, what is that?

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○○

QUESTION 5
○○○○○

QUESTION 6
○○○○○●○○

QUESTION 7
○○○○○○○○
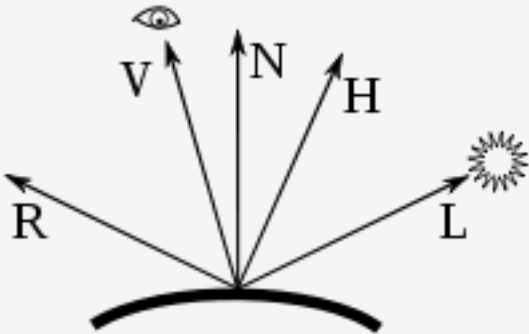
Faster computation of specular term

What happens when **light source** and **viewer** are very far away?
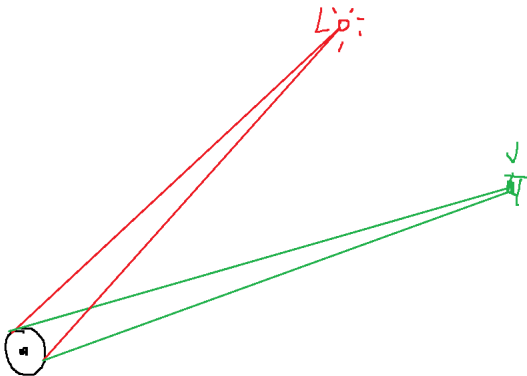
very far i.e. very small

Can observe that as L and V get further, the more L and V are going to be constant for all the surface points of the object

QUESTION 1
oo

QUESTION 2
ooo

QUESTION 3
oo

QUESTION 4
oo

QUESTION 5
ooooo

QUESTION 6
ooooooo●

QUESTION 7
oooooooo

H vector

- If L, V are far, then L, V are mostly constant across surface points
- Value of H can be cached
- Unlike for R, which must be recomputed per surface point (different N per surface point).
- Hence computation of $N \cdot H$ is faster than $R \cdot V$.

QUESTION 1
oo

QUESTION 2
ooo

QUESTION 3
oo

QUESTION 4
oo

QUESTION 5
ooooo

QUESTION 6
ooooooo

QUESTION 7
●oooooooo

Question 7

A retroreflector reflects most of the light back in the incident direction. For example, most road signs are retroreflectors.

Suppose we have a retroreflective material that behaves very similarly to a Phong material, with a diffuse and specular component, except that the specular reflection is **centered about the direction of the incident light**.

Question 7

> Suppose we have a retroreflective material that behaves very
> similarly to a Phong material, with a diffuse and specular com-
> ponent, except that the specular reflection is **centered about the
> direction of the incident light**.

Modify the Phong Illumination Equation to model this retroreflective
reflection.

$$I_{\text{Phong}} = I_a k_a + I_p k_d (N \cdot L) + I_p k_s (R \cdot V)^n$$

Question 7

We modify only the specular term of the original PIE to $I_p k_s (L \cdot V)^n$.

$$I_{\text{Retro-PIE}} = I_a k_a + I_p k_d (N \cdot L) + I_p k_s (L \cdot V)^n$$

centered about the direction of the incident light.

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○○

QUESTION 5
○○○○○

QUESTION 6
○○○○○○○

QUESTION 7
○○○●○○○○

Retroreflectors in real life

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○○

QUESTION 5
○○○○○

QUESTION 6
○○○○○○○

QUESTION 7
○○○○○●○○○

Retroreflectors in real life

QUESTION 1
○○

QUESTION 2
○○○

QUESTION 3
○○

QUESTION 4
○○

QUESTION 5
○○○○○

QUESTION 6
○○○○○○○

QUESTION 7
○○○○○●○○

Question 8

How is Phong Shading computationally more expensive than Gouraud Shading in general?

QUESTION 1
OO

QUESTION 2
OOO

QUESTION 3
OO

QUESTION 4
OO

QUESTION 5
OOOOO

QUESTION 6
OOOOOOO

QUESTION 7
OOOOOO●O

Phong vs Gouraud shading

Phong Shading: requires lighting computation at *every fragment*
Gouraud Shading: requires lighting computation at *every vertex*, which resulting color is interpolated

> # Vertices $<<$ # Fragments

Attendance taking

Thanks! Get the slides here after the tutorial.

https://trxe.github.io/cs3241-notes