
Computer Graphics

8 - Lighting

Yoonsang Lee
Hanyang University

Spring 2026

(Updated) Midterm Exam Announcement

- Date & time: **May 4 (Mon), 10:00 - 11:30 AM**
- Place: IT.BT **508, 502**
- Scope: Lecture, Lab, Lab Check 2~7, Project 1
- **Seating / room assignment:**
 - Please check the LMS announcement for your assigned room
 - https://learning.hanyang.ac.kr/courses/201420/discussion_topics/488402
- All other policies remain the same as announced last week

(Updated) Project 2 Announcement

- Project 2
 - Due: 23:59, May 17, 2026 (NO SCORE for late submissions!)
- Report:
 - You must use the **report template** provided in the LMS announcement
 - The template will be uploaded later today

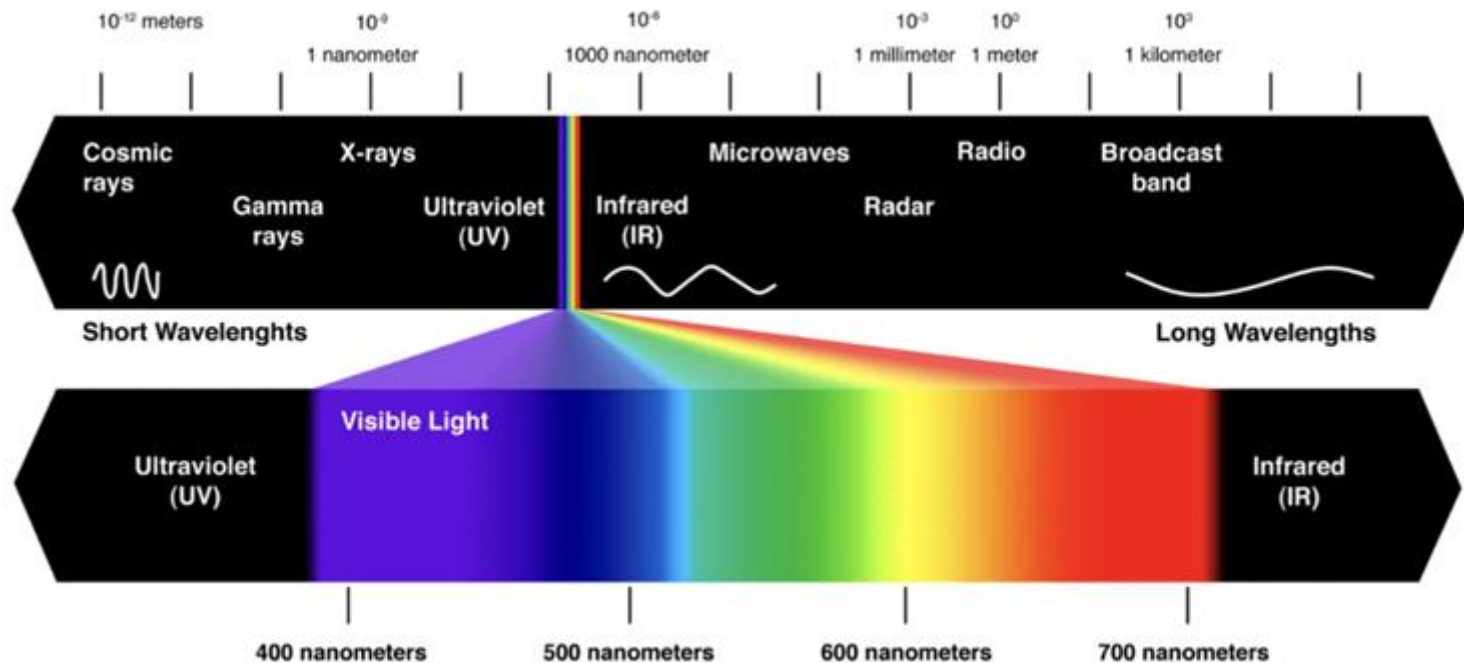
Outline

- Perceived Color of Objects
- Computing Object Color in Computer Graphics
- Reflection of Light
- Phong Illumination Model
- Polygon Shading

Perceived Color of Objects

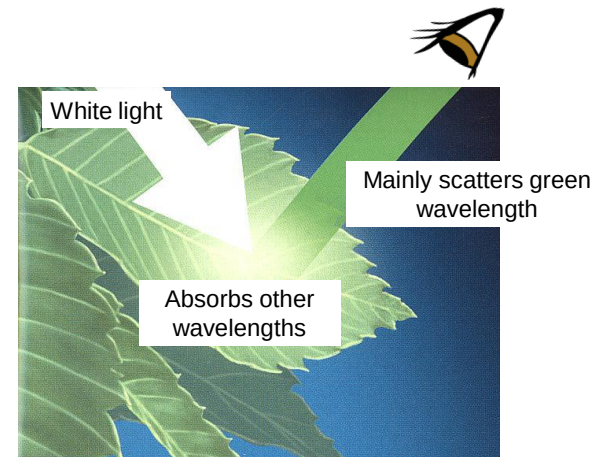
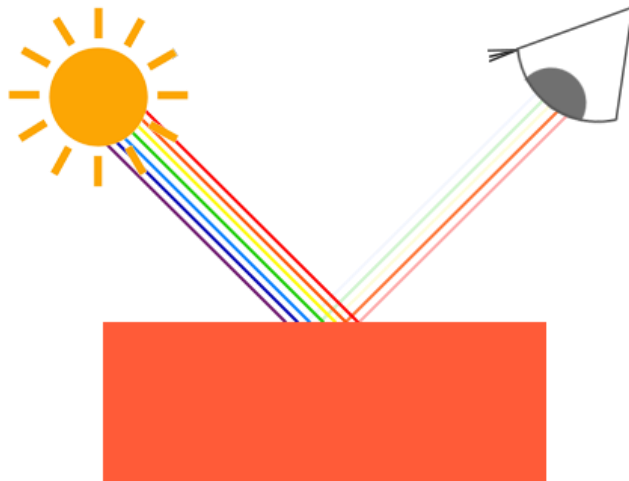
What Is Light?

- Light = electromagnetic waves (visible range only)



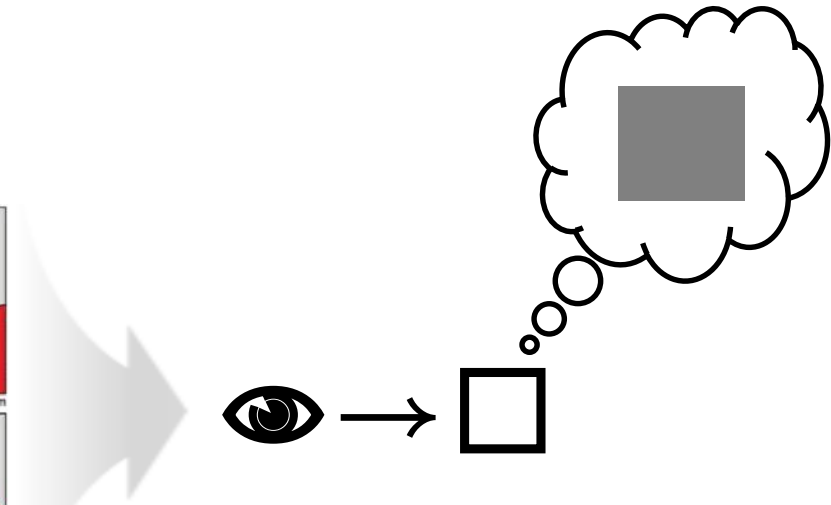
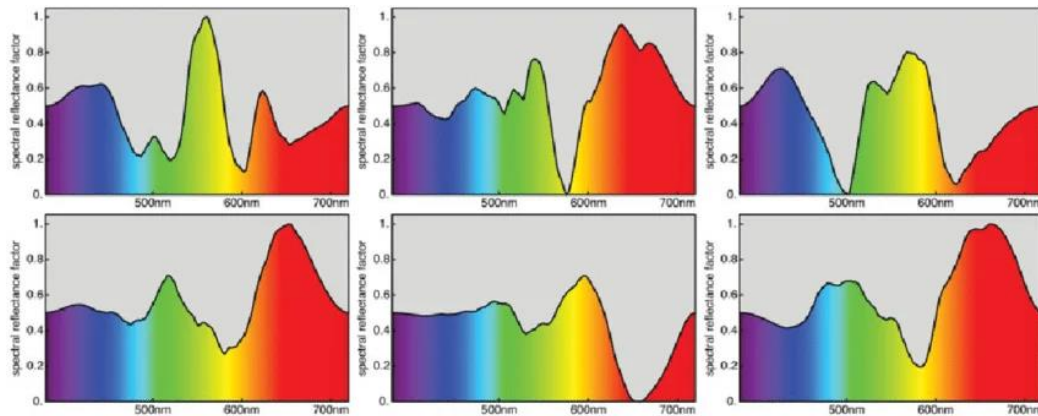
How Do Objects Get Their Color?

- When light hits an object: some is **absorbed**, some is **reflected**.
- **Color** of an object — determined by the **wavelengths of reflected light**
 - Example: a red object appears red because it reflects red and absorbs others
- The **surface properties** determine which wavelengths are reflected or absorbed.



What Are the Wavelengths of “Gray”?

- These different spectra → same perceived color (gray)



- This phenomenon is called *metamerism*

Where Does Light Come From?

- Light sources



direct light
(from sources)



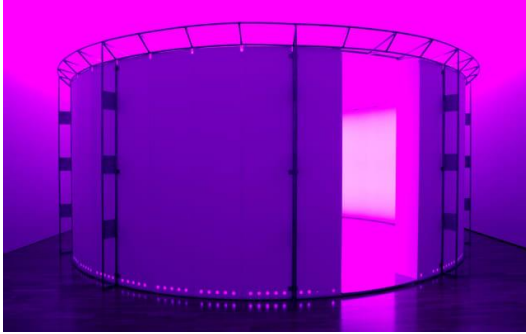
reflected /
scattered light
(from surfaces)

- Object surfaces



Effect of Light Source on Object Color

- An object's perceived color depends on the light source
 - Because objects reflect **only the wavelengths present in the light**
- Same object under different light→ different colors



Room for all colours, Olafur Eliasson

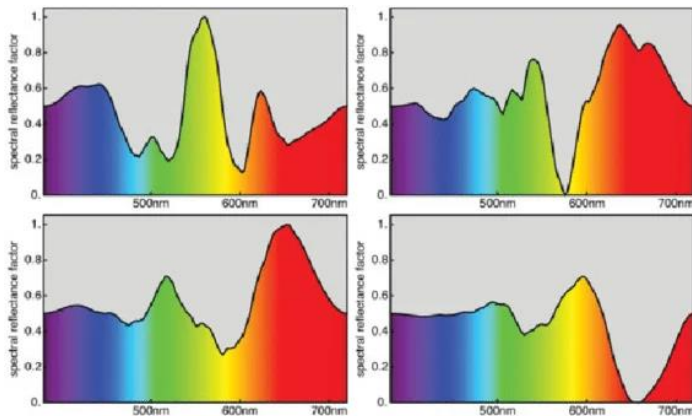


Computing Object Color in Computer Graphics

Light: From Physics → Computer Graphics

- In computer graphics, we represent color using RGB values.

- From spectrum → **RGB values**



(128, 128, 128)

or

(0.5, 0.5, 0.5)

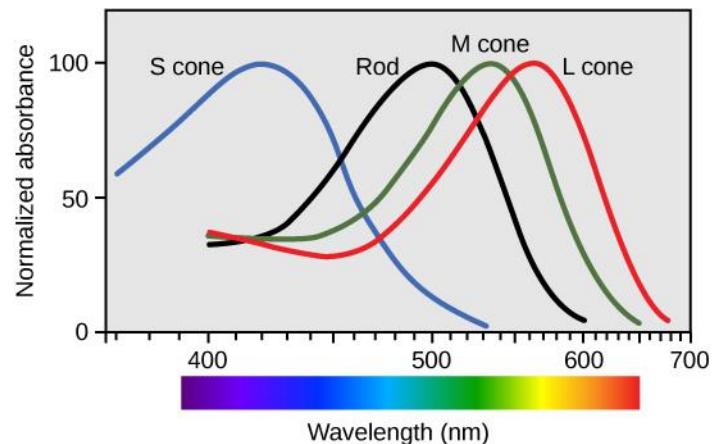
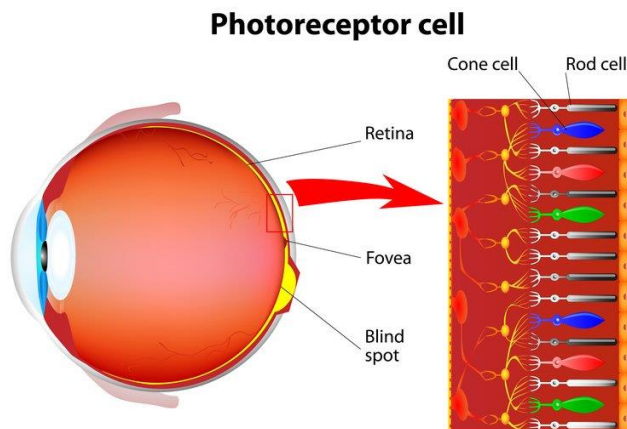
8-bit integer [0–255]



normalized float [0.0–1.0]

Why Can We Use RGB?

- Human eyes use **three types of cone cells**
- Each responds to different wavelengths
- → **Light spectrum is reduced to three signals**
- RGB is a convenient approximation of these responses.



Computing Object Color

- Final color = Light color $\times$ Material reflectance

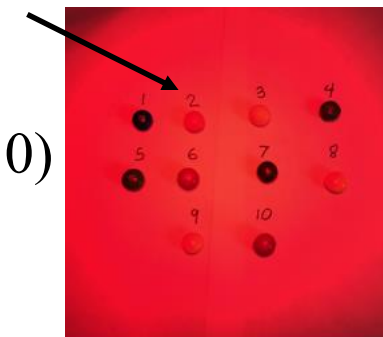
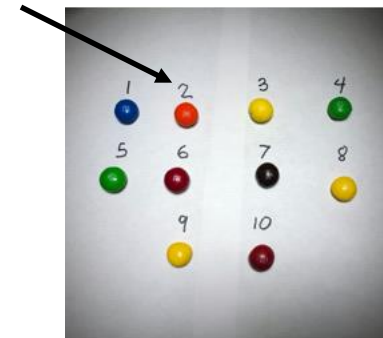
$$\mathbf{C} = \mathbf{L} \odot \mathbf{M}$$

- $\odot$: element-wise multiplication (R, G, B channels independently)

- Light color (L): RGB intensity from the light source
- Material reflectance (M): RGB reflectance of the object surface $\rightarrow$ intrinsic property of the surface
- Each channel is computed independently:
(light amount) $\times$ (how much is reflected)

Computing Object Color: Example

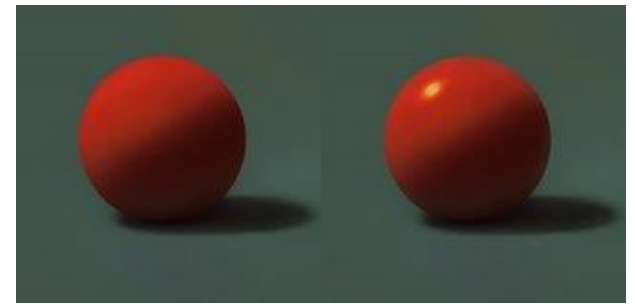
- $C = L \odot M$
- Each channel is computed as: (light) $\times$ (reflectance)
- Material reflectance: (0.92, 0.26, 0.0)
 - Reflects 92% of red, 26% of green, and no blue from the incident light.
- Under white light (1.0, 1.0, 1.0),
→ Object color: (0.92, 0.26, 0.0)
- Under red light (1.0, 0.0, 0.0),
→ Object color: (0.92, 0.0, 0.0)
(only red light is present → green & blue are 0)



Reflection of Light

Reflection of Light

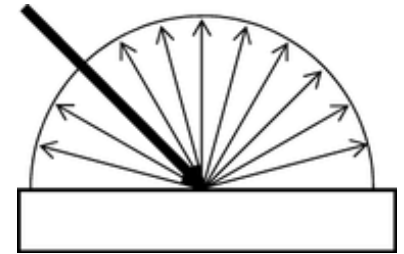
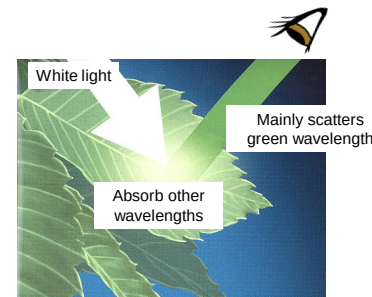
- Light interacts with objects in different ways:
 - absorption, reflection, scattering, refraction
- For opaque objects:
 - reflection (including scattering) is most important
- Types of reflection:
 - Diffuse → determines color
 - Specular → determines highlights



* In computer graphics, both scattering and reflection are often referred to as "reflection"

Diffuse Reflection

- Scatters specific wavelengths in **all directions**
- → **Determines surface color**
- **View-independent**



reflects magenta
wavelengths



reflects all wavelengths
equally



absorbs most wavelengths
(reflects very little)

Diffuse Reflection - Lambert's Cosine Law

- **Reflected intensity $\propto \cos \theta$**
 - θ : angle between the light direction and the surface normal

$$\begin{aligned} I_{reflected} &= I_{incident} \cos \theta \\ &= I_{incident} (\hat{\mathbf{N}} \cdot \hat{\mathbf{L}}) \end{aligned}$$

$\hat{\mathbf{N}}$

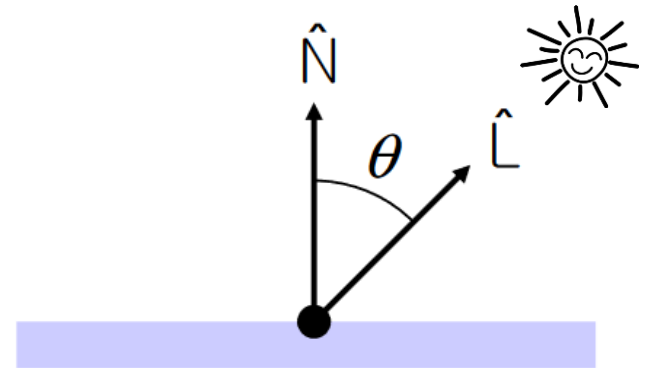
surface normal (unit vector)

$\hat{\mathbf{L}}$

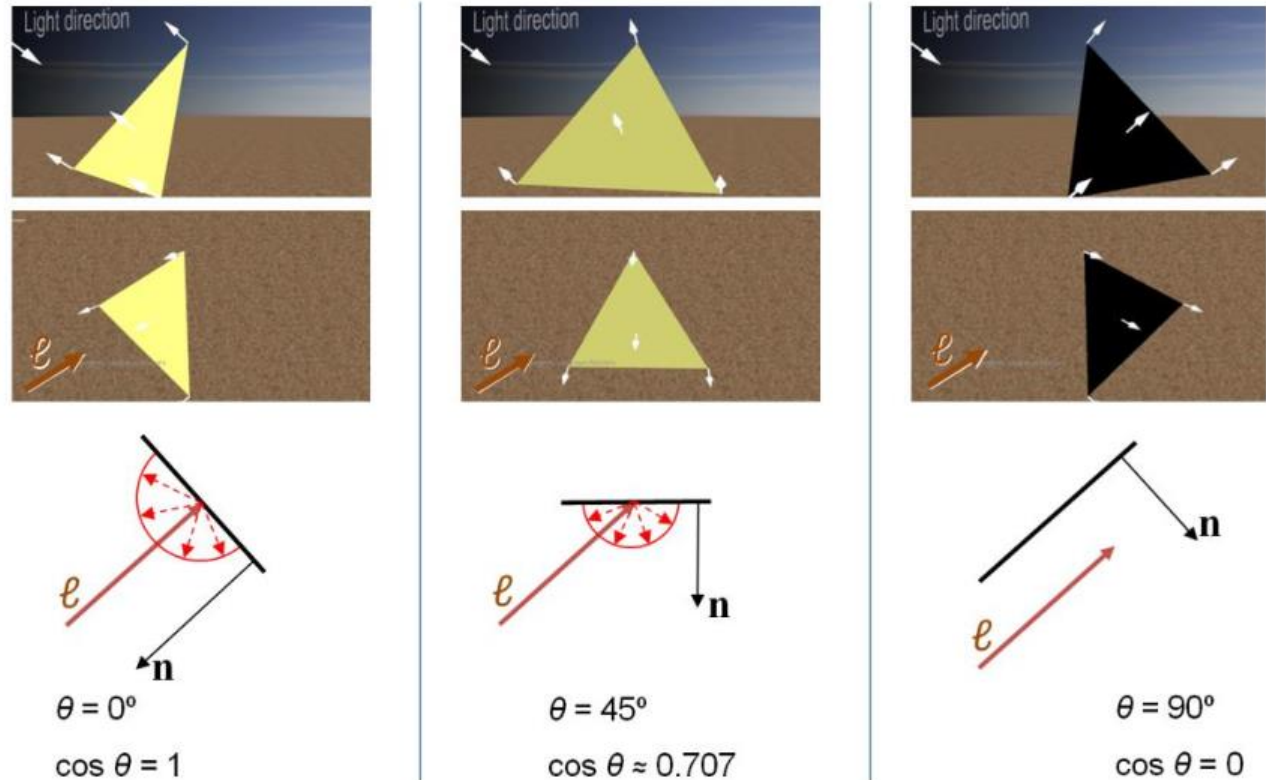
light direction (unit vector) $\hat{\mathbf{N}} \cdot \hat{\mathbf{L}} = \cos \theta$

$I_{reflected}$ intensity of reflected ray

$I_{incident}$ intensity of incident ray



Diffuse Reflection - Lambert's Cosine Law



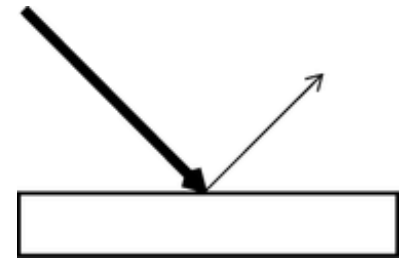
- More perpendicular lighting $\rightarrow$ brighter surface

* This slide is from the slides of Prof. Andy van Dam (Brown Univ.)

<http://cs.brown.edu/courses/csci1230/lectures.shtml>

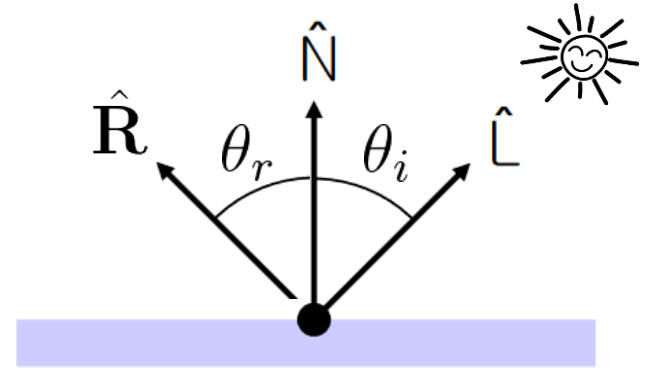
Ideal Specular Reflection (Mirror)

- Mirror-like reflection from **smooth, polished surfaces**
- → **Generate mirror images**
- Light reflects in a **single direction**
- **View-dependent**



Ideal Specular Reflection - Laws of Reflection

- The **angle of reflection equals the angle of incidence**: $\theta_r = \theta_i$
- Light reflects **symmetrically** around the **surface normal**
- $\hat{\mathbf{N}}, \hat{\mathbf{L}}, \hat{\mathbf{R}}$ lie in the same plane



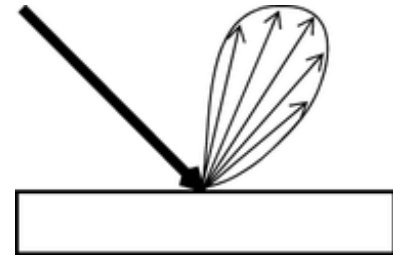
$\hat{\mathbf{N}}$ surface normal (unit vector)

$\hat{\mathbf{L}}$ light direction (unit vector)

$\hat{\mathbf{R}}$ reflected direction (unit vector)

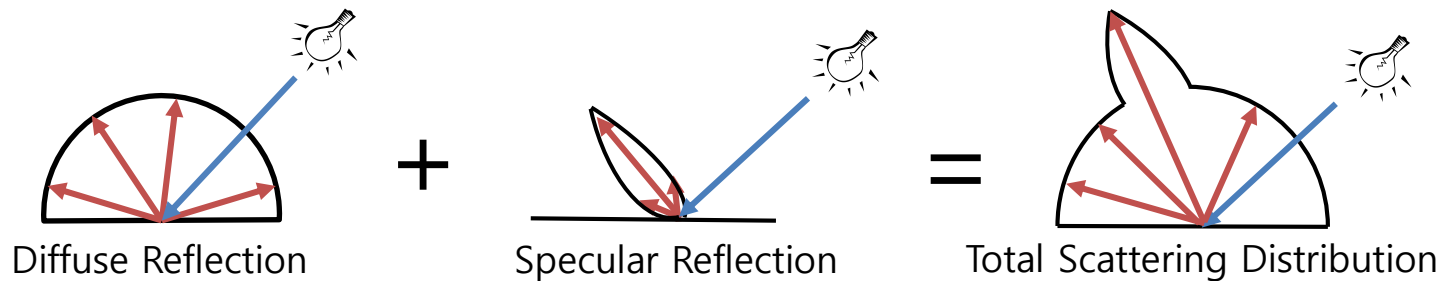
Non-Ideal Specular Reflection (Glossy)

- Reflection from **glossy surfaces** (not perfectly smooth like a mirror)
- Reflected rays are **spread around the ideal reflection direction**
- → Creates bright **highlights** near that direction
- Rougher surfaces → wider highlights
- **View-dependent**



Reflection of General Materials

- Most materials have both diffuse and (non-ideal) specular reflection.



- → Their relative strength determines the final appearance

* This image is from the slides of Prof. Andy van Dam (Brown Univ.)

<http://cs.brown.edu/courses/csci1230/lectures.shtml>

Quiz 1

- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"
- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. **2024123456: 4.0**
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.

Phong Illumination Model

Lighting (or Illumination)

- In computer graphics, **lighting** (or **illumination**) refers to the process of computing the effect of lights on surfaces.
- → Determines surface color (diffuse) and highlights (specular)

Phong Illumination Model

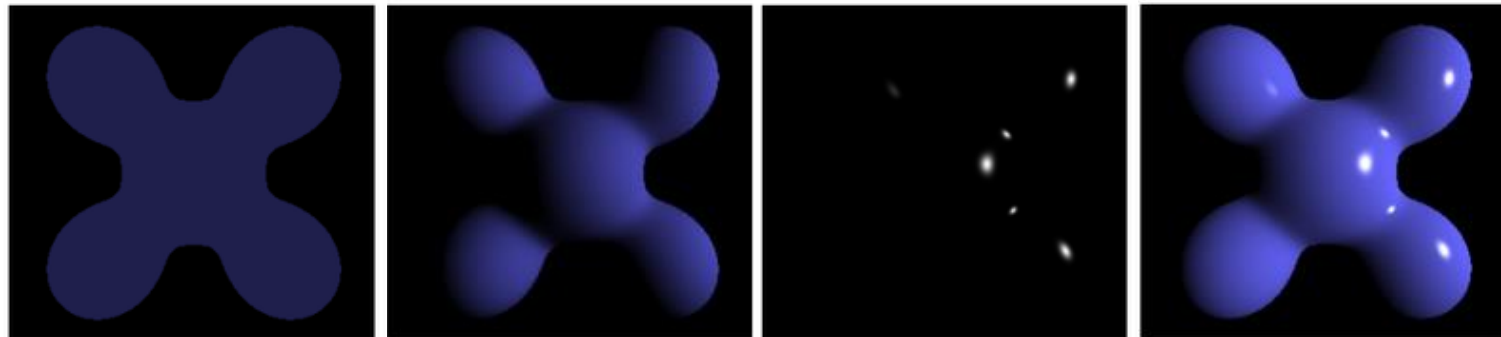
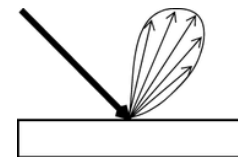
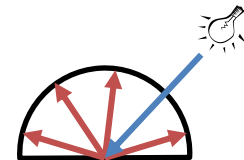
- One of the most widely used classical illumination models
- Empirical model — not physically based, but simple and efficient



Bùi Tường Phong
(1942 – 1975)

Phong Illumination Model

- **Ambient**
 - Constant light added everywhere (approximation of indirect light).
- **Diffuse**
 - Models diffuse reflection using Lambert's law.
 - Determines the surface color.
- **Specular**
 - Approximation for glossy reflection using $\cos^n(\alpha)$.
 - α : angle between view direction and reflection direction
 - Computes highlights on shiny objects.



Ambient + Diffuse + Specular = Phong Reflection

Phong Illumination Model

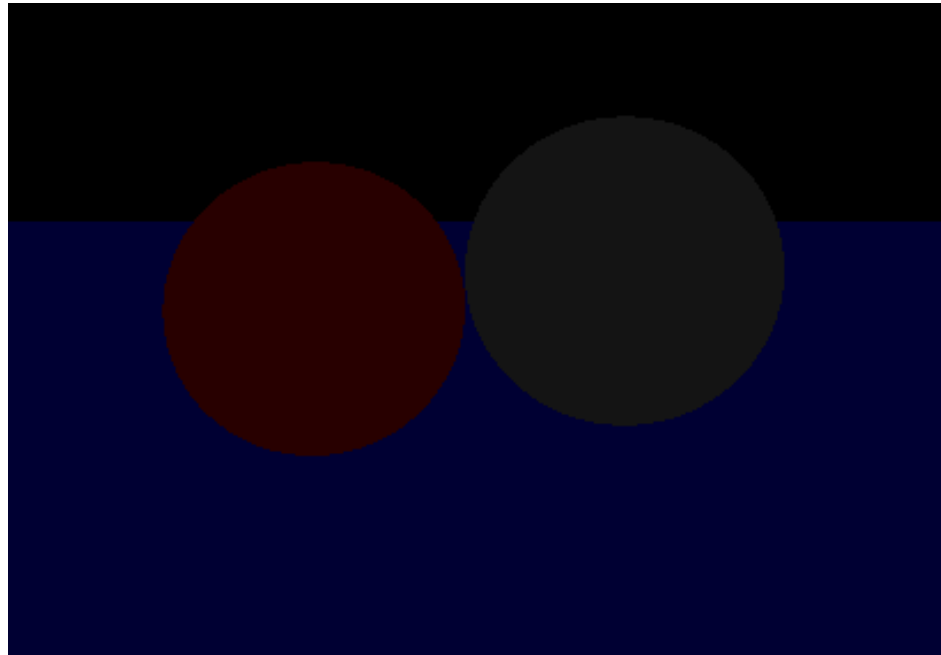
- Compute each component of the Phong model:
 - Ambient, Diffuse, Specular
- At a specific point on the surface
- The point can be:
 - a vertex
 - a pixel (inside a polygon)

Ambient Component

- $\mathbf{I}_a = \mathbf{l}_a * \mathbf{m}_a$
- Constant light applied everywhere
- $\mathbf{l}_a$: light ambient color
- $\mathbf{m}_a$: material ambient reflectance
- $\mathbf{I}_a$: final ambient color
- $*$: element-wise multiplication

Result

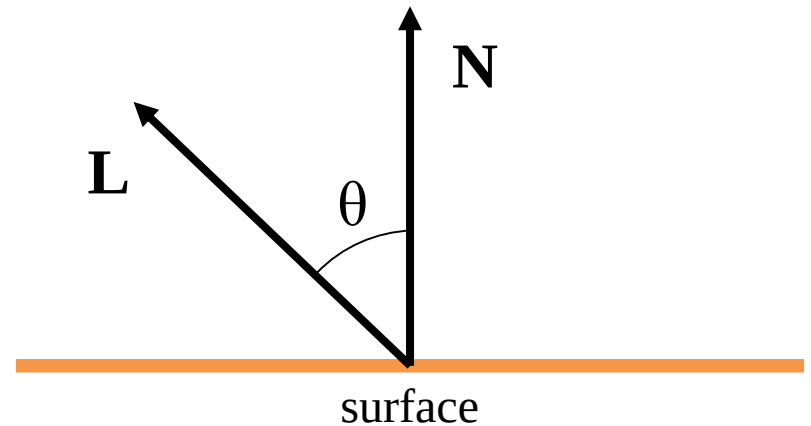
- $\mathbf{I} = \mathbf{l}_a * \mathbf{m}_a$



* The images are from the slides of Prof. Jinxiang Chai (Texas A&M University):
http://faculty.cs.tamu.edu/jchai/csce441_2016spring/lectures.html

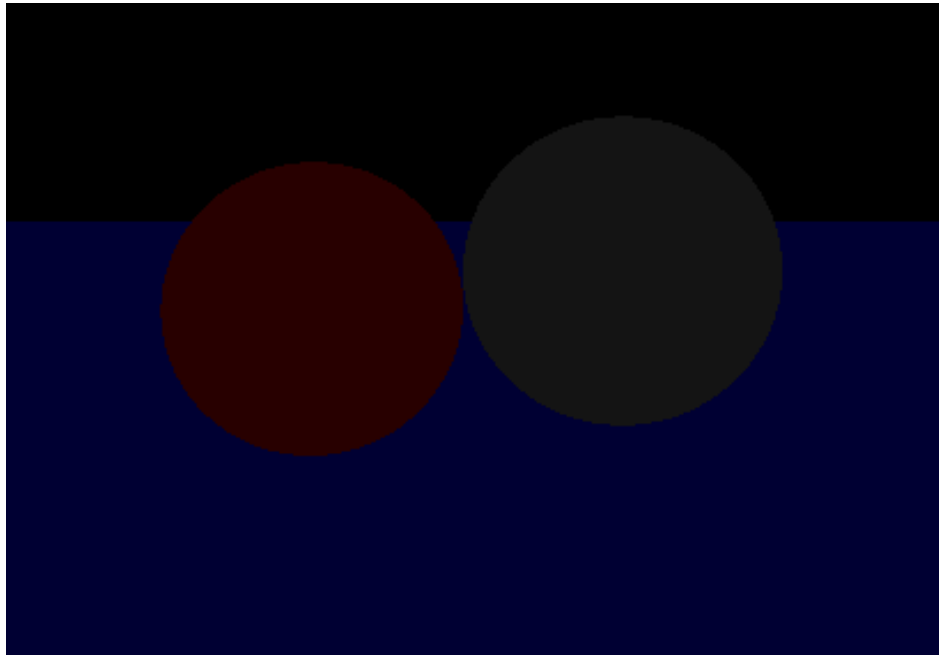
Diffuse Component

- $\mathbf{I}_d = \mathbf{l}_d * \mathbf{m}_d \max(0, \cos(\theta)) = \mathbf{l}_d * \mathbf{m}_d \max(0, \mathbf{L} \cdot \mathbf{N})$
- More perpendicular light $\rightarrow$ brighter surface
- $\mathbf{L} \cdot \mathbf{N} = \cos \theta$ (unit vectors)
- $\max()$: Only when light hits the front side ($\mathbf{L} \cdot \mathbf{N} > 0$)
- $\mathbf{L}$: light direction (unit vector)
- $\mathbf{N}$: surface normal (unit vector)
- $\mathbf{l}_d$: light diffuse color
- $\mathbf{m}_d$: material diffuse reflectance
- $\mathbf{I}_d$: final diffuse color



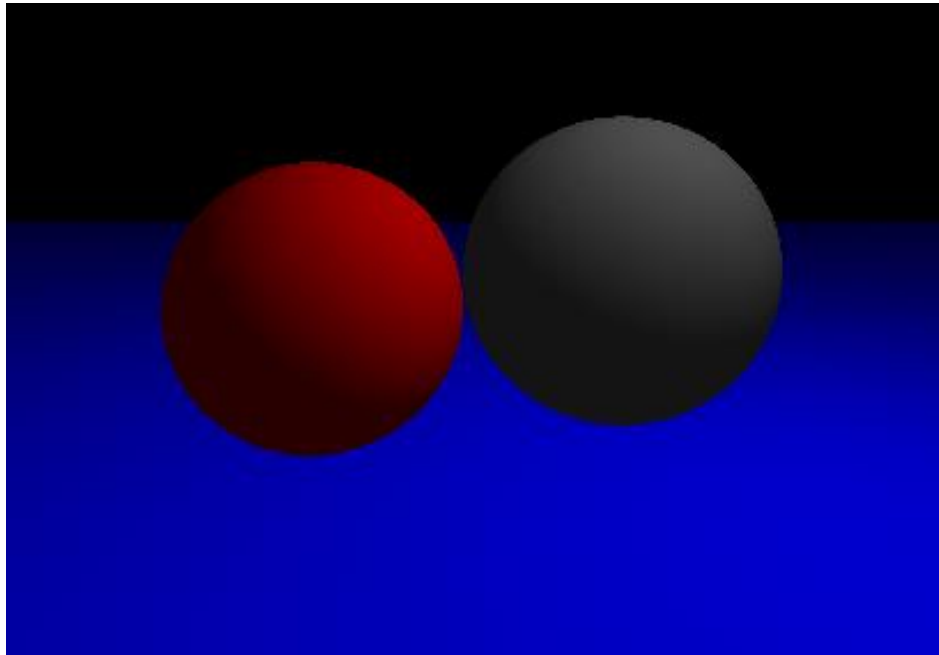
Result

- $\mathbf{I} = \mathbf{l}_a * \mathbf{m}_a$



Result

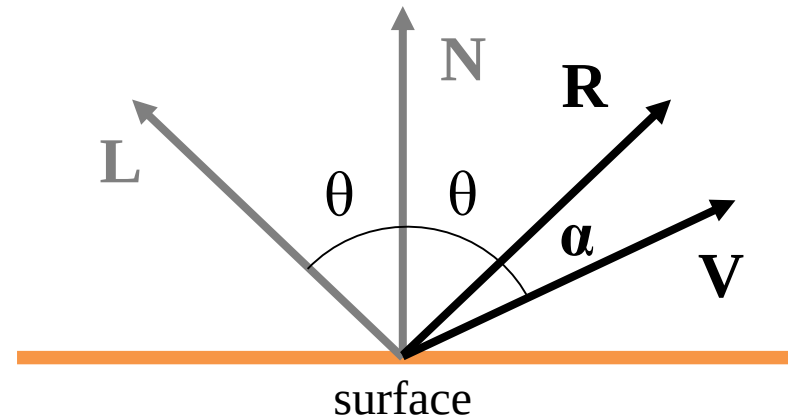
- $I = l_a * m_a + l_d * m_d \max(0, L \cdot N)$



Specular Component

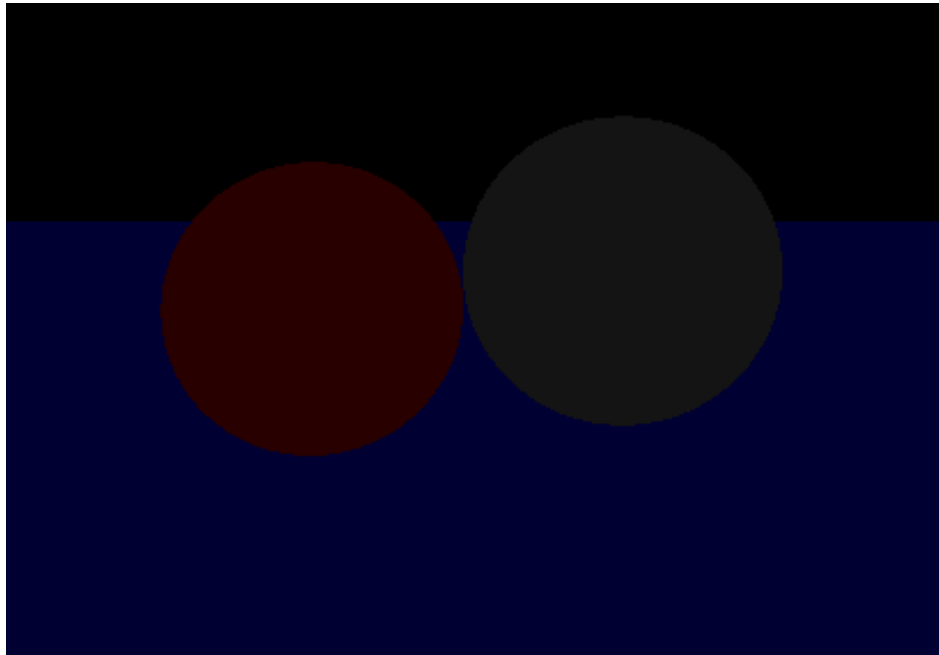
- $\mathbf{I}_s = \mathbf{l}_s * \mathbf{m}_s \max(0, \cos(\alpha))^n = \mathbf{l}_s * \mathbf{m}_s \max(0, (\mathbf{V} \cdot \mathbf{R}))^n$
- Highlight appears when viewing direction $\approx$ reflection direction
- $\max()$: Only when looking near the mirror reflection direction ($\mathbf{V} \cdot \mathbf{R} > 0$)

- $\mathbf{V}$: view direction (unit vector)
- $\mathbf{R}$: reflection direction (unit vector)
- n : **shininess coefficient**
- $\mathbf{l}_s$: light specular color
- $\mathbf{m}_s$: material specular reflectance
- $\mathbf{I}_s$: final specular color



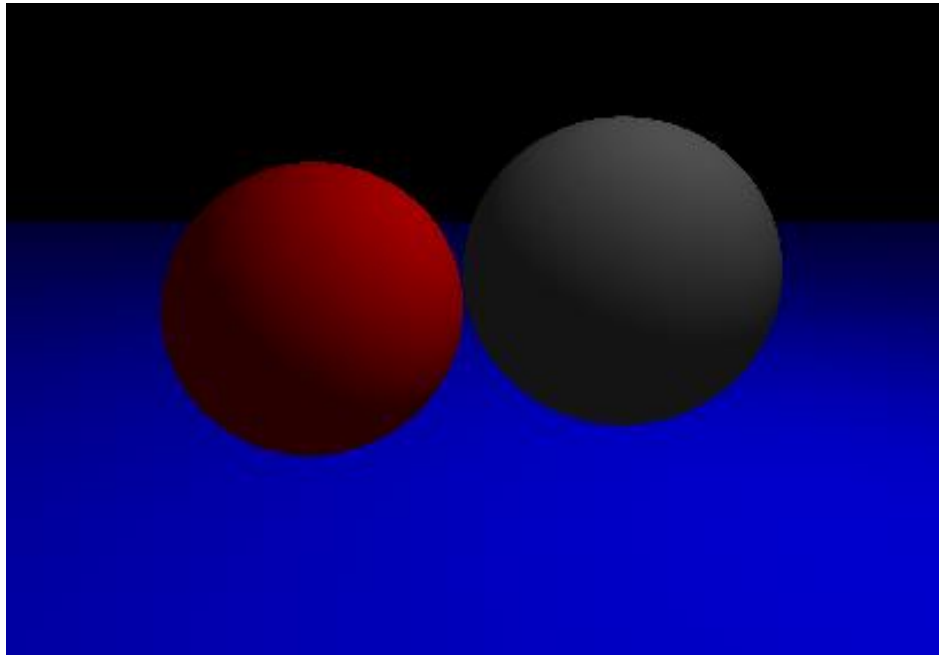
Result

- $\mathbf{I} = \mathbf{l}_a * \mathbf{m}_a$



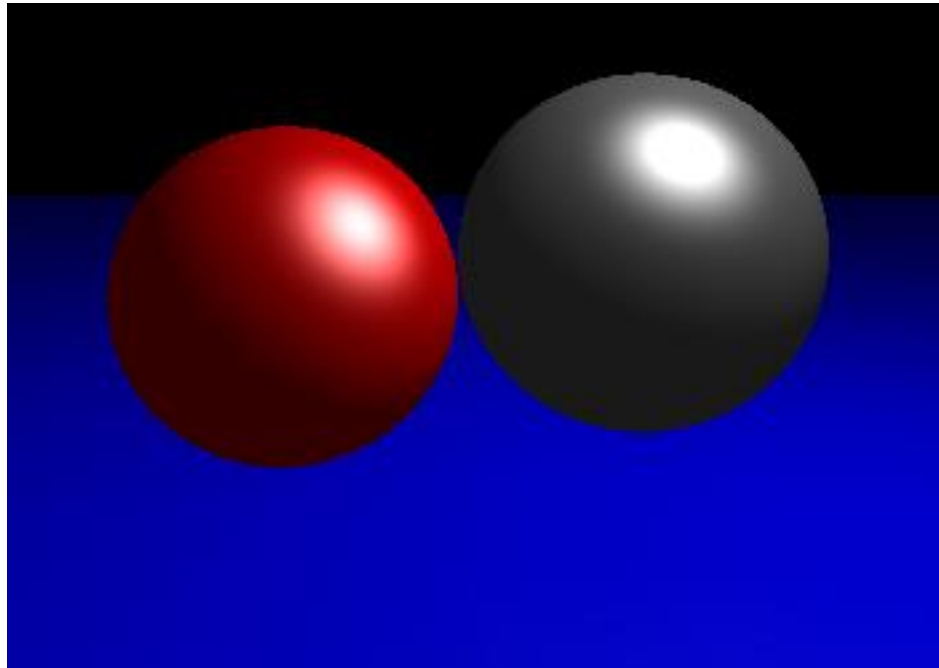
Result

- $I = l_a * m_a + l_d * m_d \max(0, L \cdot N)$



Result

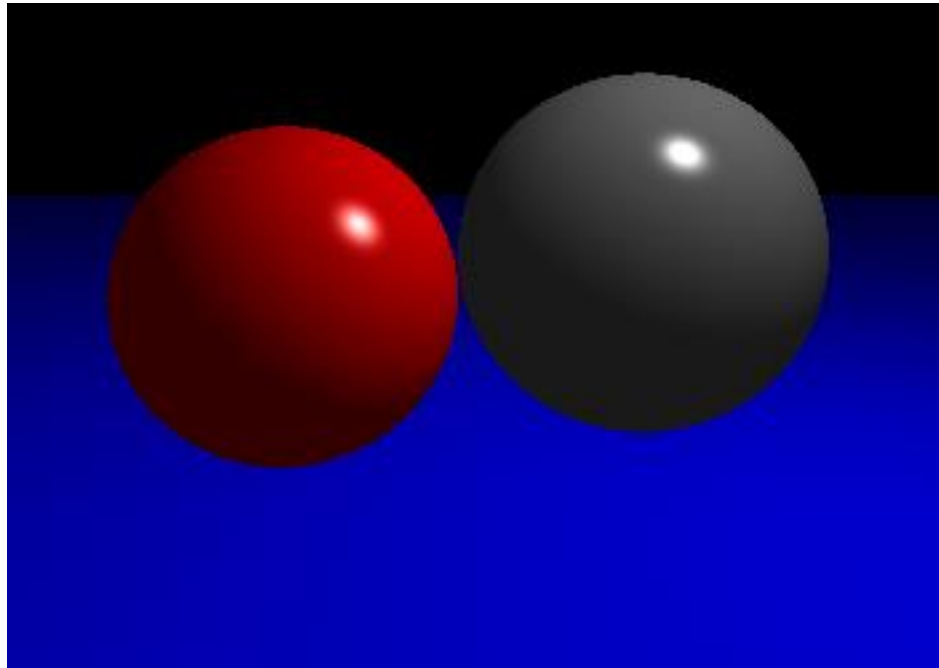
$$\mathbf{I} = \mathbf{l}_a * \mathbf{m}_a + \mathbf{l}_d * \mathbf{m}_d \max(0, \mathbf{L} \cdot \mathbf{N}) + \mathbf{l}_s * \mathbf{m}_s \max(0, \mathbf{V} \cdot \mathbf{R})^n$$



$n = 5$

Result

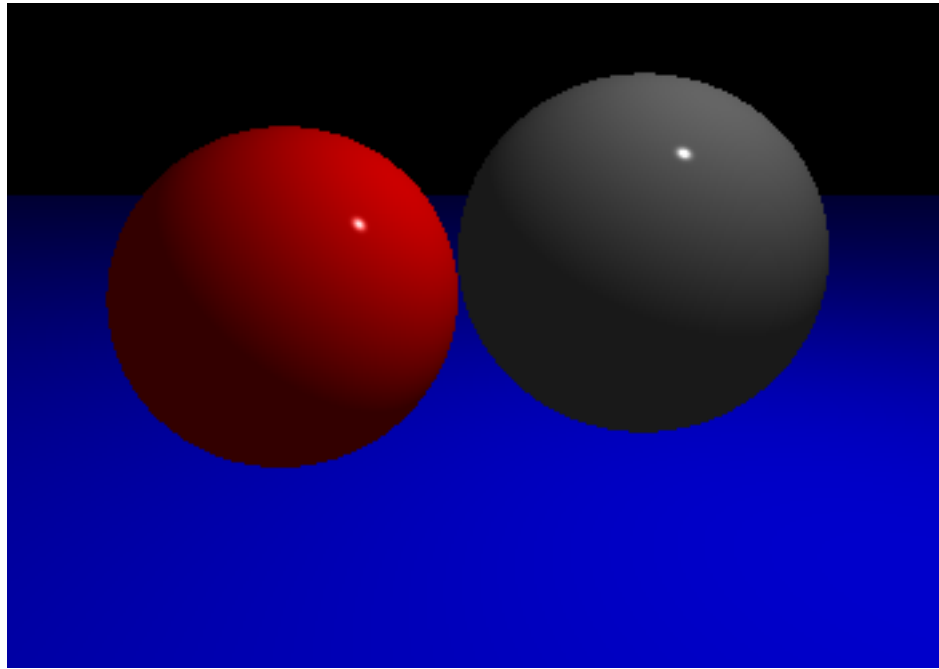
$$\mathbf{I} = \mathbf{l}_a * \mathbf{m}_a + \mathbf{l}_d * \mathbf{m}_d \max(0, \mathbf{L} \cdot \mathbf{N}) + \mathbf{l}_s * \mathbf{m}_s \max(0, \mathbf{V} \cdot \mathbf{R})^n$$



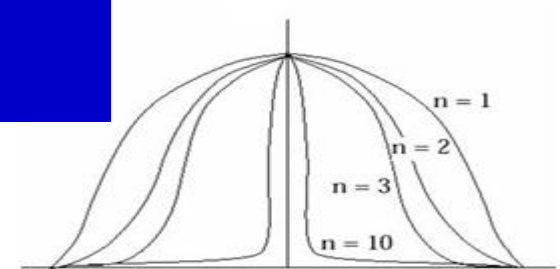
$n = 50$

Result

$$\mathbf{I} = \mathbf{l}_a * \mathbf{m}_a + \mathbf{l}_d * \mathbf{m}_d \max(0, \mathbf{L} \cdot \mathbf{N}) + \mathbf{l}_s * \mathbf{m}_s \max(0, \mathbf{V} \cdot \mathbf{R})^n$$

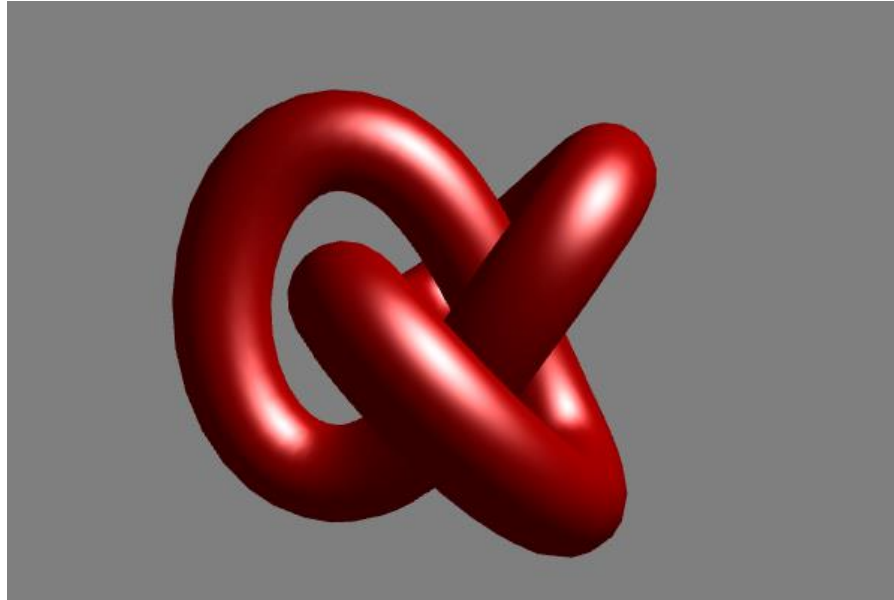


$n = 500$



Specular falloff of $(\cos \alpha)^n$

[Demo] Phong Illumination



<http://www.cs.toronto.edu/~jacobson/phong-demo/>

- Set the value of the first drop down box to “Phong Shading”
- Try changing
 - ambient, diffuse, and specular reflection coefficients and colors
 - specular shininess
 - you can also change object type, light position and background color

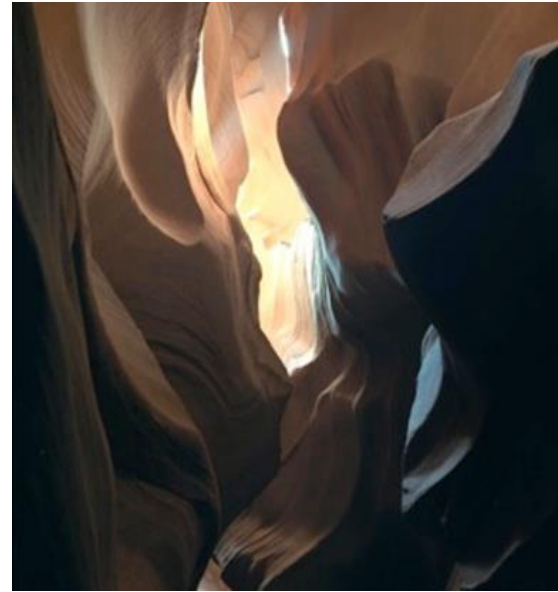
Quiz 2

- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"
- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. **2024123456: 4.0**
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.f

Polygon Shading

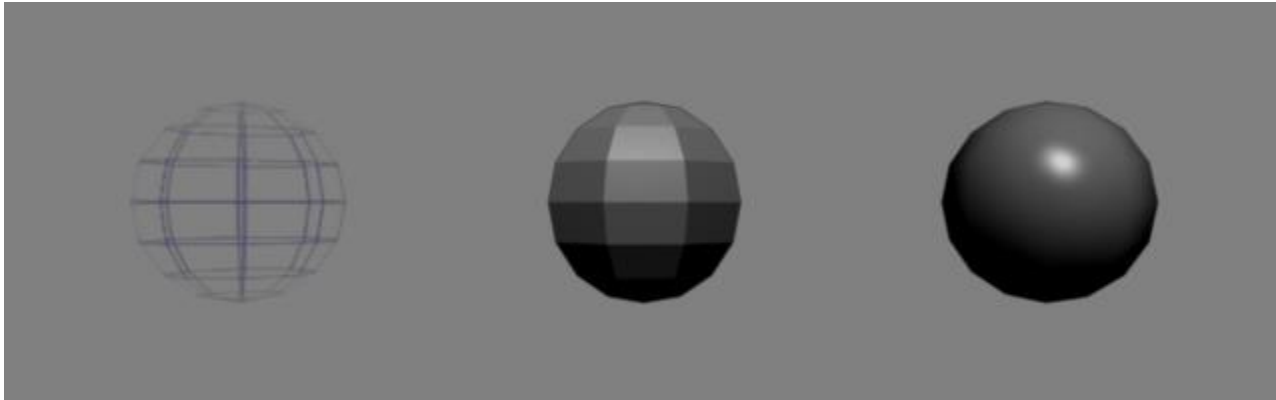
Shading

- Variation in observed color across a surface
 - Caused by lighting and geometry



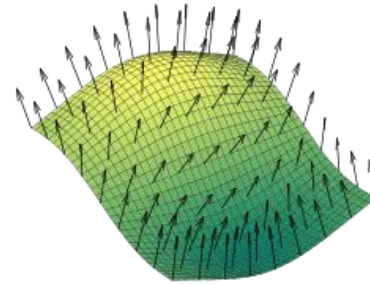
Polygon Shading

- In computer graphics, shading can mean:
 - Variation in surface color due to lighting
 - How pixel colors are computed inside a polygon
- Here, we focus on the second meaning.
- **Polygon shading:** Computing the **color of each pixel** in a polygon using an illumination model



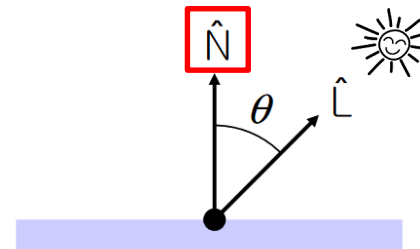
Surface Normal

- A vector perpendicular to the surface at a point
 - Typically normalized (unit length)



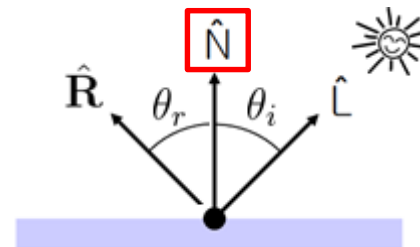
- Plays a key role in lighting & shading

- Diffuse reflection
 - Lambert's Cosine Law



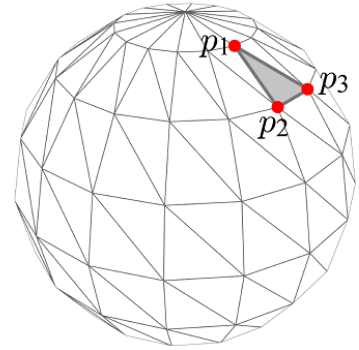
$$I_{reflected} = I_{incident} \cos \theta$$

- Specular reflection
 - Laws of Reflection



$$\theta_r = \theta_i$$

Face Normal



- Computing the face normal of a polygon

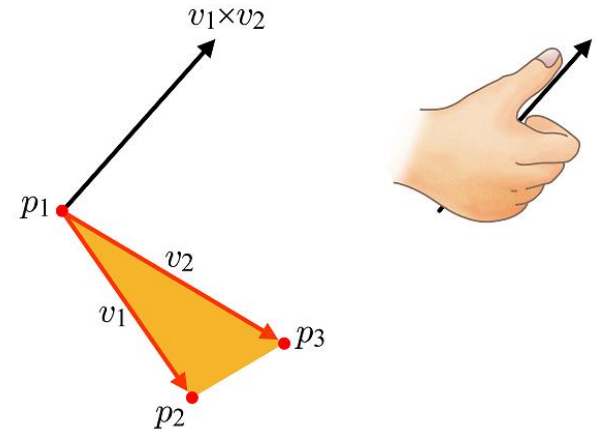
- Given triangle (p1, p2, p3):

$$v_1 = p_2 - p_1$$

$$v_2 = p_3 - p_1$$

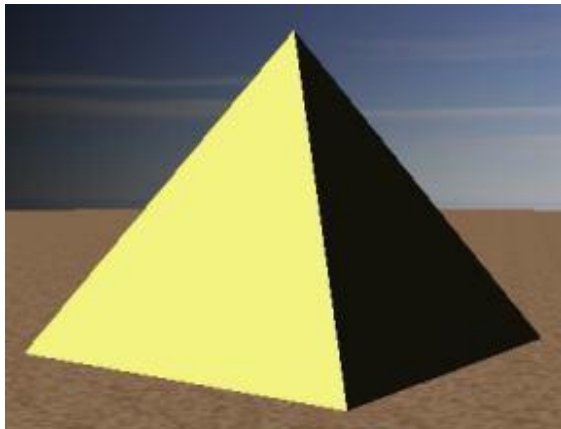
$$\text{face normal} = \frac{v_1 \times v_2}{\|v_1 \times v_2\|}$$

- Vertex order determines the normal direction
 - This defines the orientation of the face
 - which determines the front/back face (Lec 7)



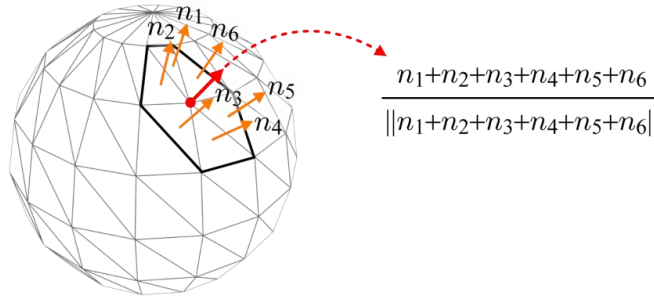
Flat Shading

- Compute one normal per polygon
- Compute one color per polygon (using that normal)
- Fast, but not very desirable for curved shapes
 - Even with more polygons, the surface still appears faceted



Smooth Shading

- Use an "averaged" normal per vertex



- Produces smooth color transitions across polygons
- Two methods:
 - Gouraud shading (interpolate colors)
 - Phong shading (interpolate normals)

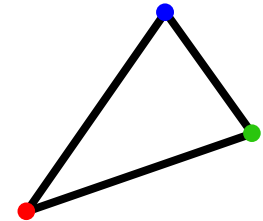
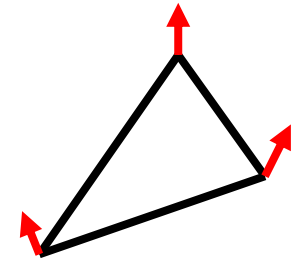


Gouraud Shading

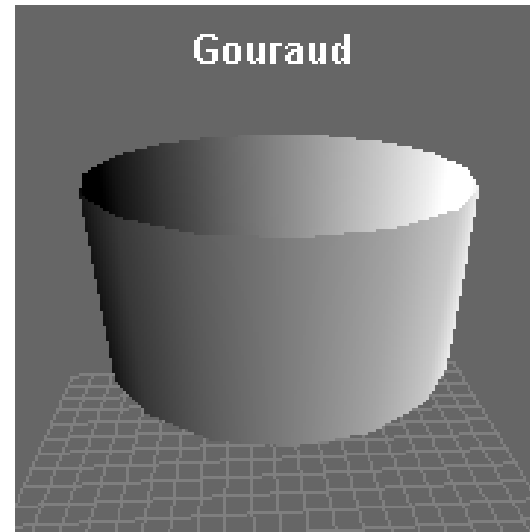
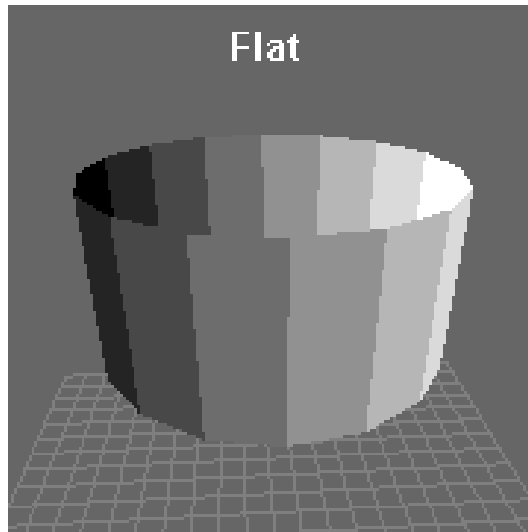


Henri Gouraud
(1944~)

- Use one normal per vertex
- Compute lighting **at each vertex**
- **Interpolate vertex colors** across the polygon
 - using barycentric interpolation

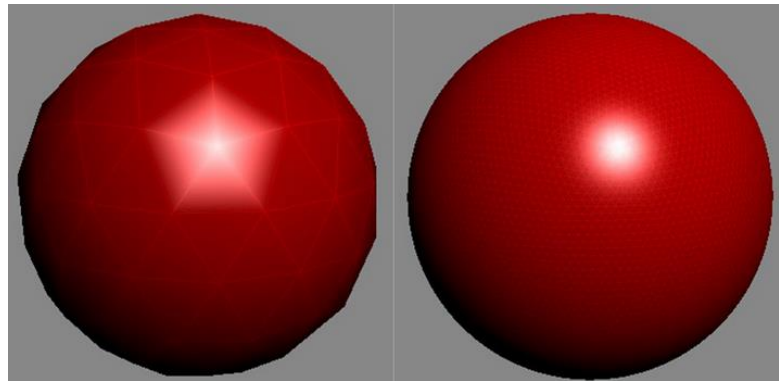
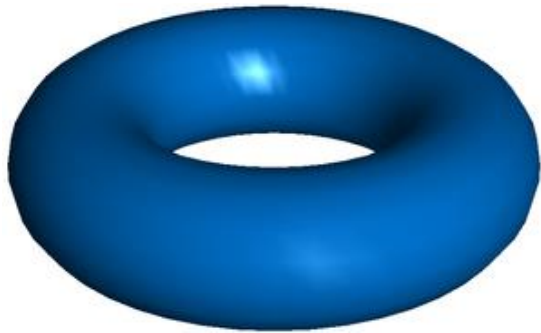


Gouraud Shading



Gouraud Shading

- Problem: poor specular highlight
 - Specular highlights may be distorted or averaged away altogether



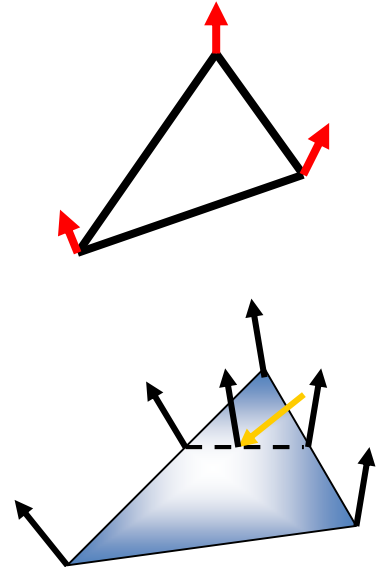
Higher polygon count
reduces this artifact

Phong Shading

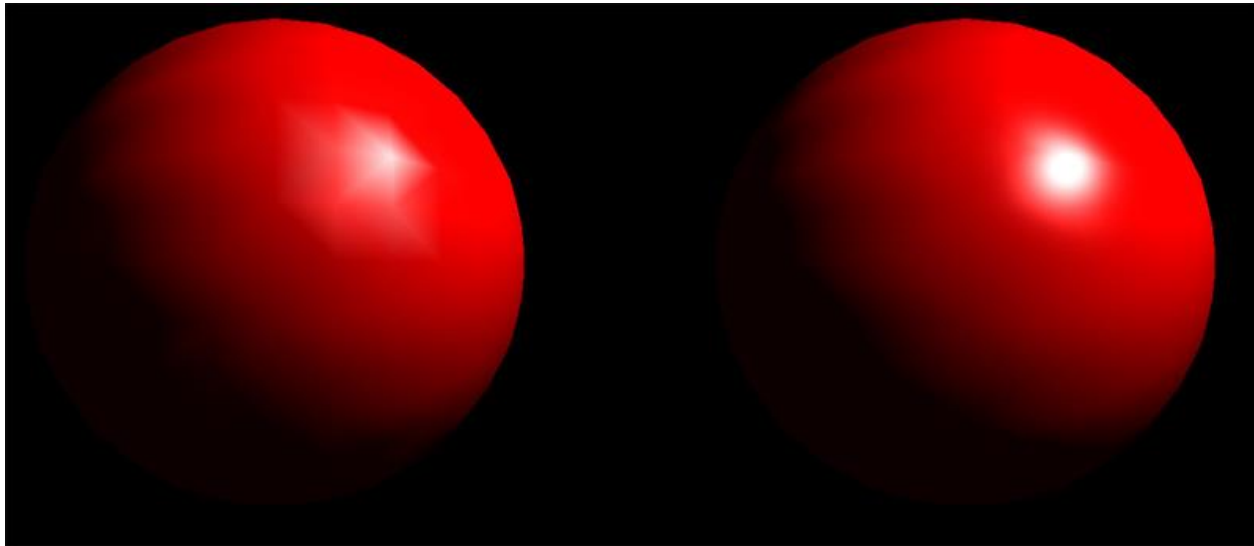


Bùi Tường Phong
(1942 – 1975)

- Use one normal per vertex
- **Interpolate vertex normals** across the polygon
- Compute lighting **at each pixel**
 - using the interpolated normal



Phong Shading



Gouraud shading

Phong shading

Phong Shading

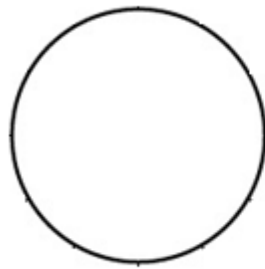
- Captures highlights much better
 - Interpolated normals better approximate the true surface normals
 - Higher quality, but needs more computation
- Not to be confused with Phong's illumination model (developed by the same person)
 - Phong shading $\neq$ Phong illumination model

[Demo] Polygon Shading

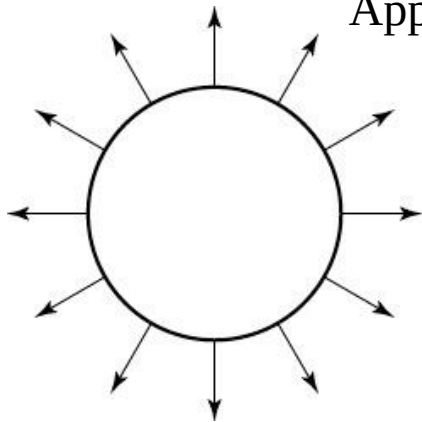
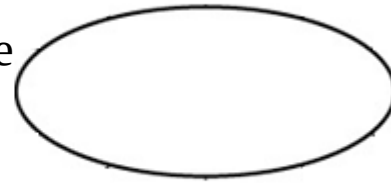
- Flat & Gouraud shading
 - <http://math.hws.edu/graphicsbook/demos/c4/smooth-vs-flat.html>
- Gouraud & Phong shading
 - <http://www.cs.toronto.edu/~jacobson/phong-demo/>

Normal Vector Transformation

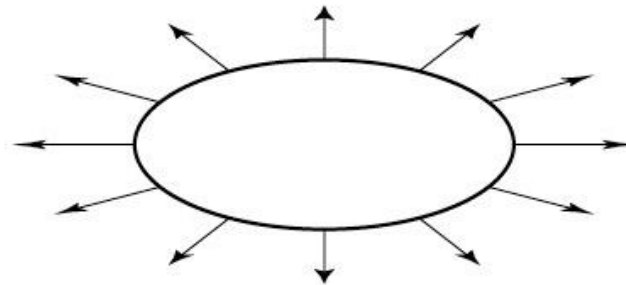
- If a set of points on a surface is transformed by an affine transformation M ,
- Tangents are transformed by M .
 - Because the differences of points are transformed by M .
- However, normals should NOT be transformed directly by M .
 - Because normals should be perpendicular to tangents.



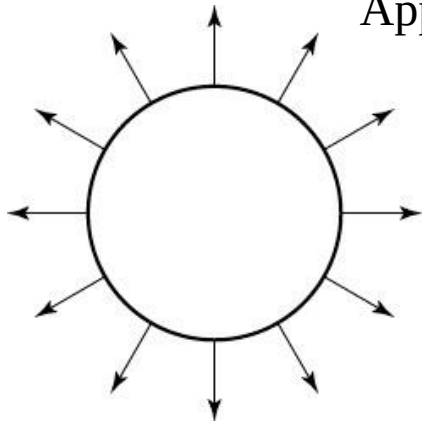
Apply M to shape



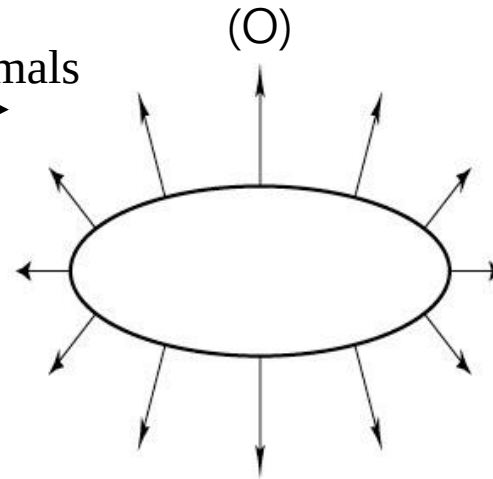
Apply M to normals



Normals must remain
(X) perpendicular to the surface

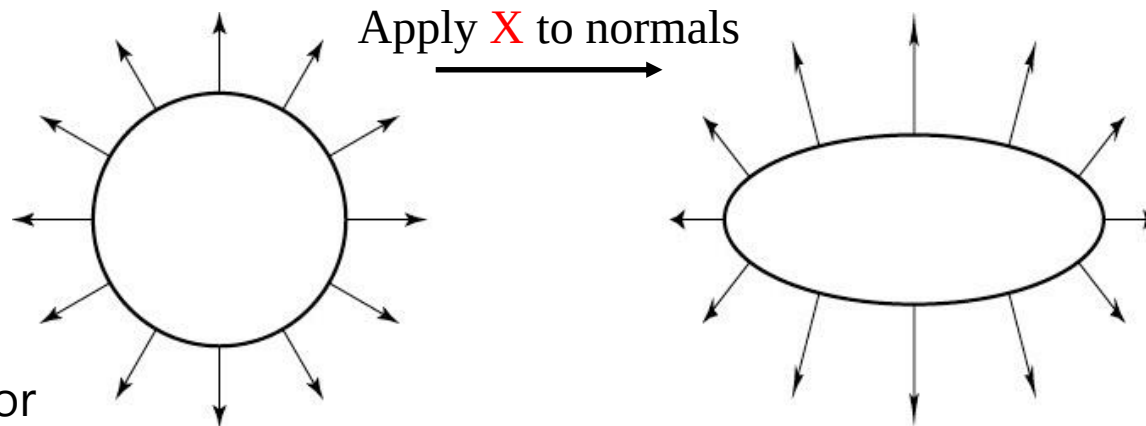


Apply ? to normals



(O)

Normal Vector Transformation



t: tangent vector
n: normal vector

have: $\mathbf{t} \cdot \mathbf{n} = \mathbf{t}^T \mathbf{n} = 0$ $\longrightarrow$ normal is perpendicular to tangent

want: $M\mathbf{t} \cdot X\mathbf{n} = \mathbf{t}^T M^T X\mathbf{n} = 0 \rightarrow$ normal must stay perpendicular to tangent after transformation

so set $X = (M^T)^{-1}$

then: $M\mathbf{t} \cdot X\mathbf{n} = \mathbf{t}^T M^T (M^T)^{-1} \mathbf{n} = \mathbf{t}^T \mathbf{n} = 0$

Solution: $X = (M^T)^{-1}$ $\longrightarrow$ So normals must be transformed by $(M^{-1})^T$

Lab Session

- Now, let's start the lab today.