LIGHTS, MATERIALS, DIFFUSE SHADING

getting colorful

BASED ON MIT 6.837

slides adapted & project started code translated to Swift by Dion Larson

adapted course materials available for free [here]

original course materials available for free [here]

Mathematical toolbox

Lights

Materials

Diffuse shading

Next week

Recap of ray-sphere intersections

Mob programming (spheres, normals)

VECTORS

Vectors have magnitude and direction

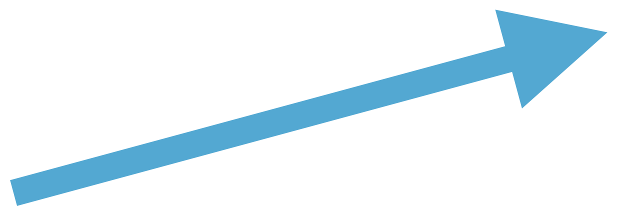
Used throughout computer graphics

$$\vec{\boldsymbol{A}} = \; < \boldsymbol{A}_x , \boldsymbol{A}_y , \boldsymbol{A}_z >$$

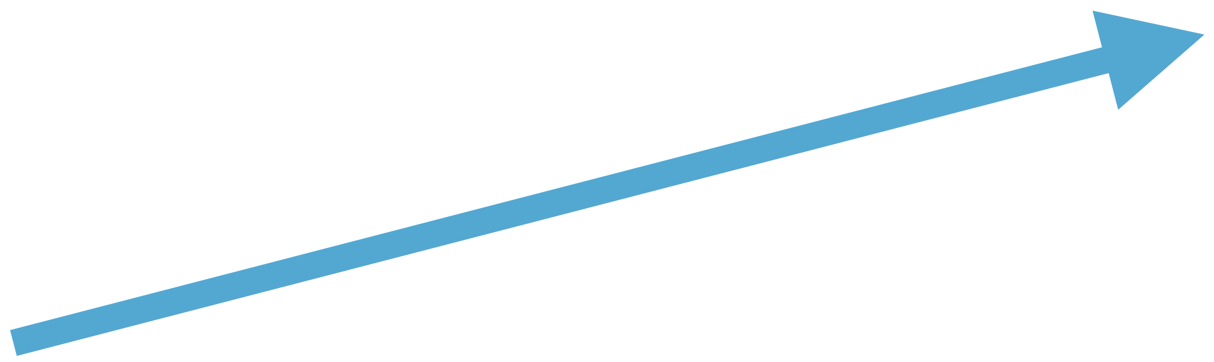
SCALING VECTORS

Multiply each element by the scalar s

$$As = \ < A_x s, A_y s, A_z s >$$



$\times \ s =$



NORMALIZING VECTORS

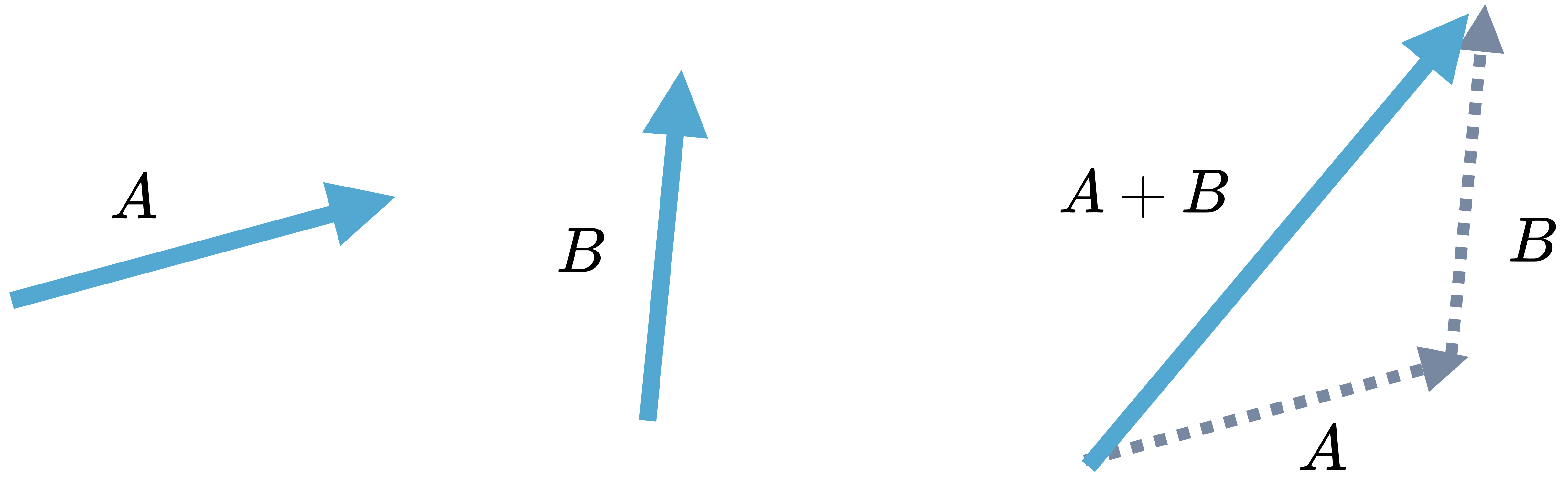
Create a vector with same direction but length of one

$$normalize(A) = \frac{A}{||A||}$$

VECTOR ADDITION

Add each element to the corresponding element of other vector

$$\mathbf{A} + \mathbf{B} = \ < A_x + B_x , \ A_y + B_y , \ A_z + B_z >$$



DOT PRODUCT

Extremely useful for light calculations in shading step

$$\mathbf{A} \cdot \mathbf{B} = \sum_{i=1}^{3} A_i B_i = A_x B_x + A_y B_y + A_z B_z$$

$$\mathbf{A} \cdot \mathbf{B} = \|\mathbf{A}\| \, \|\mathbf{B}\| \cos \theta$$

When perpendicular

$$\mathbf{A} \cdot \mathbf{B} = 0$$

When parallel

$$\mathbf{A} \cdot \mathbf{B} = \|\mathbf{A}\| \, \|\mathbf{B}\|$$

Mathematical toolbox

Lights

Materials

Diffuse shading

Next week

Recap of ray-sphere intersections

Mob programming (spheres, normals)

TYPES OF LIGHTS

Ambient light: simplified indirect illumination

Point light: light with position, emits equally in all directions

Directional light: point light infinitely far away, all light comes from same direction

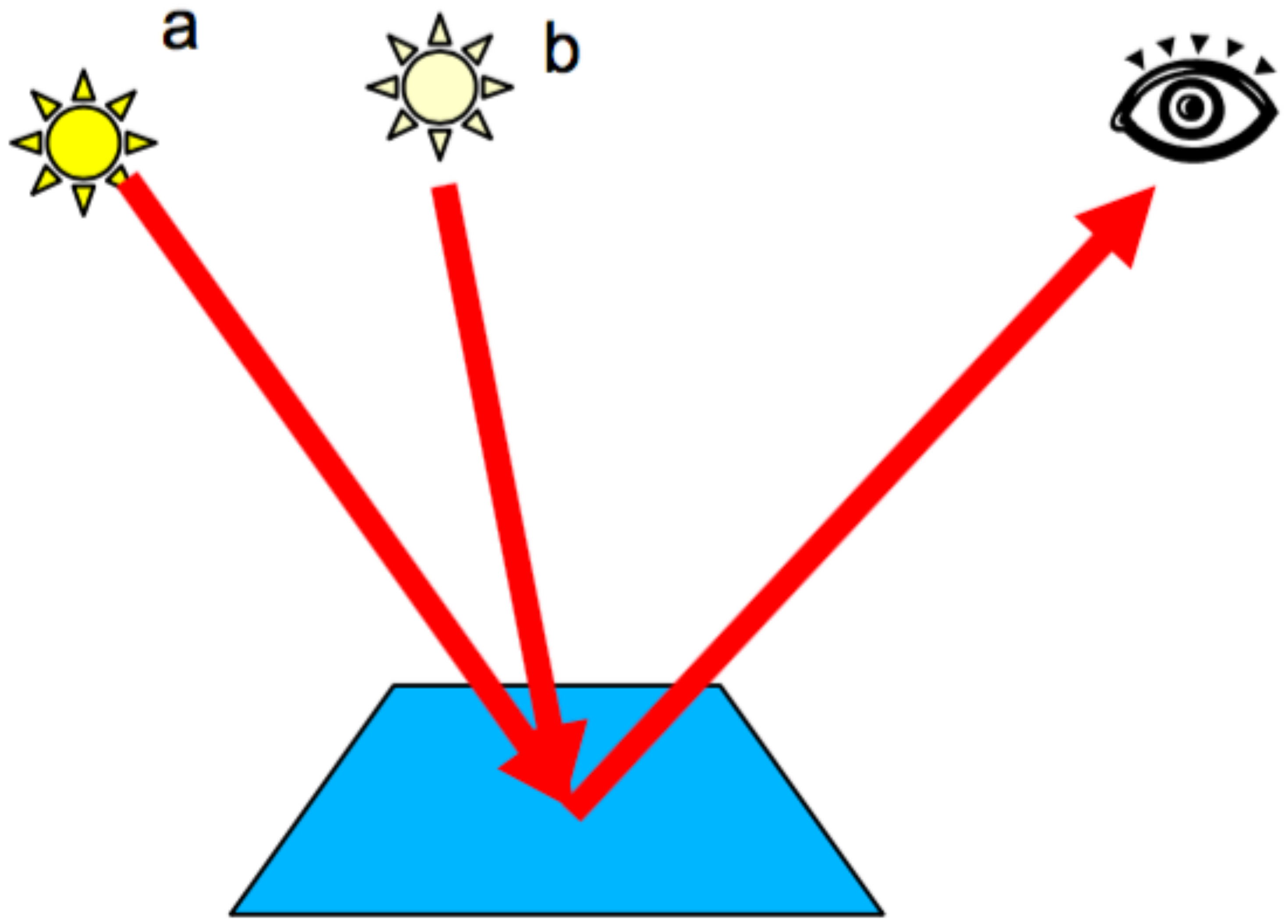
LINEARITY

Light intensity/color is additive

We'll process each light source separately and sum the results

$$I(a+b) = I(a) + I(b)$$

$$I(sa) = sI(a)$$

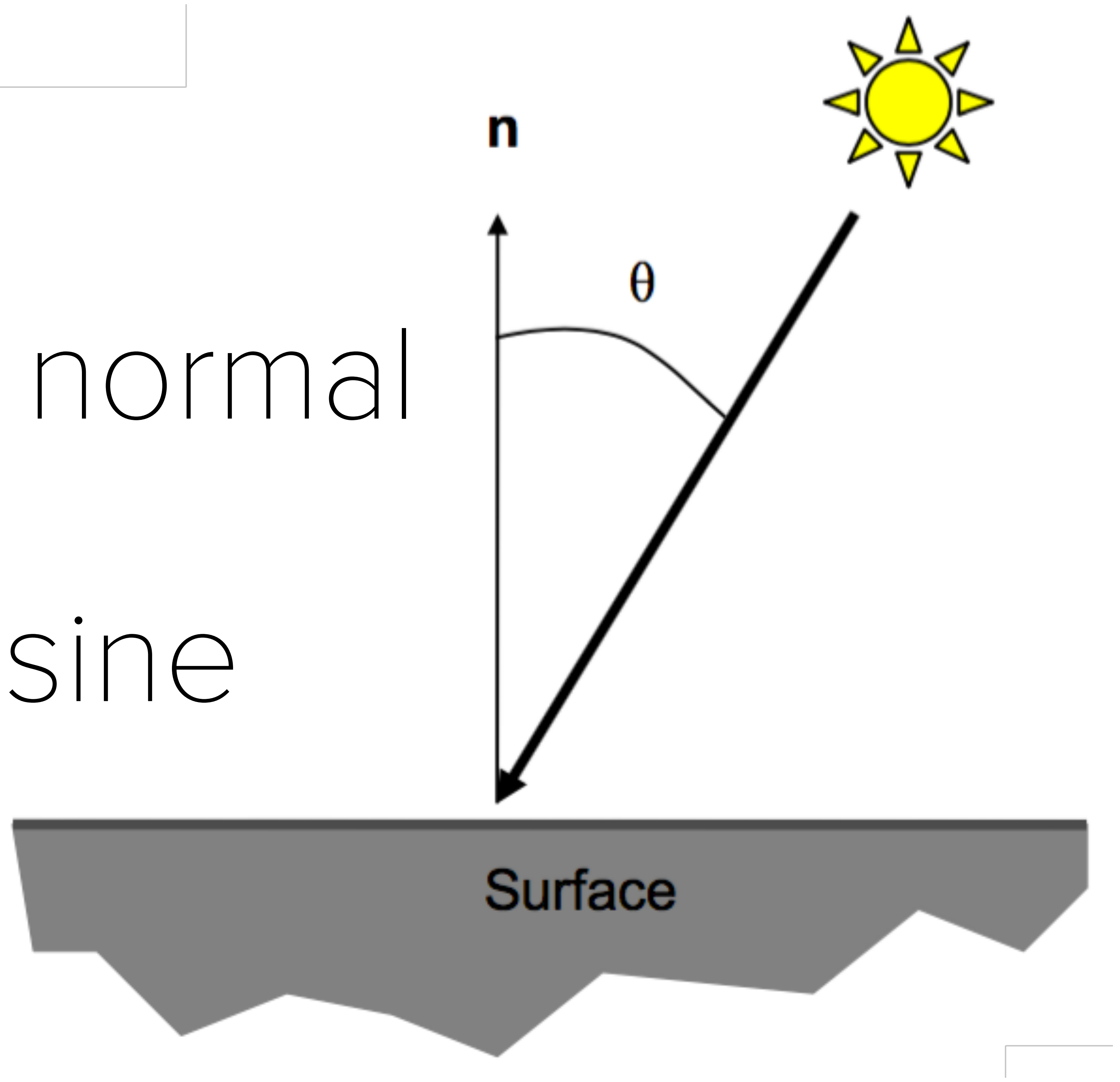


INCOMING IRRADIANCE

Amount of light energy received by surface depends on incoming angle

Largest when light is lined up with normal

Why? Dot product depends on cosine



CALCULATING IRRADIANCE

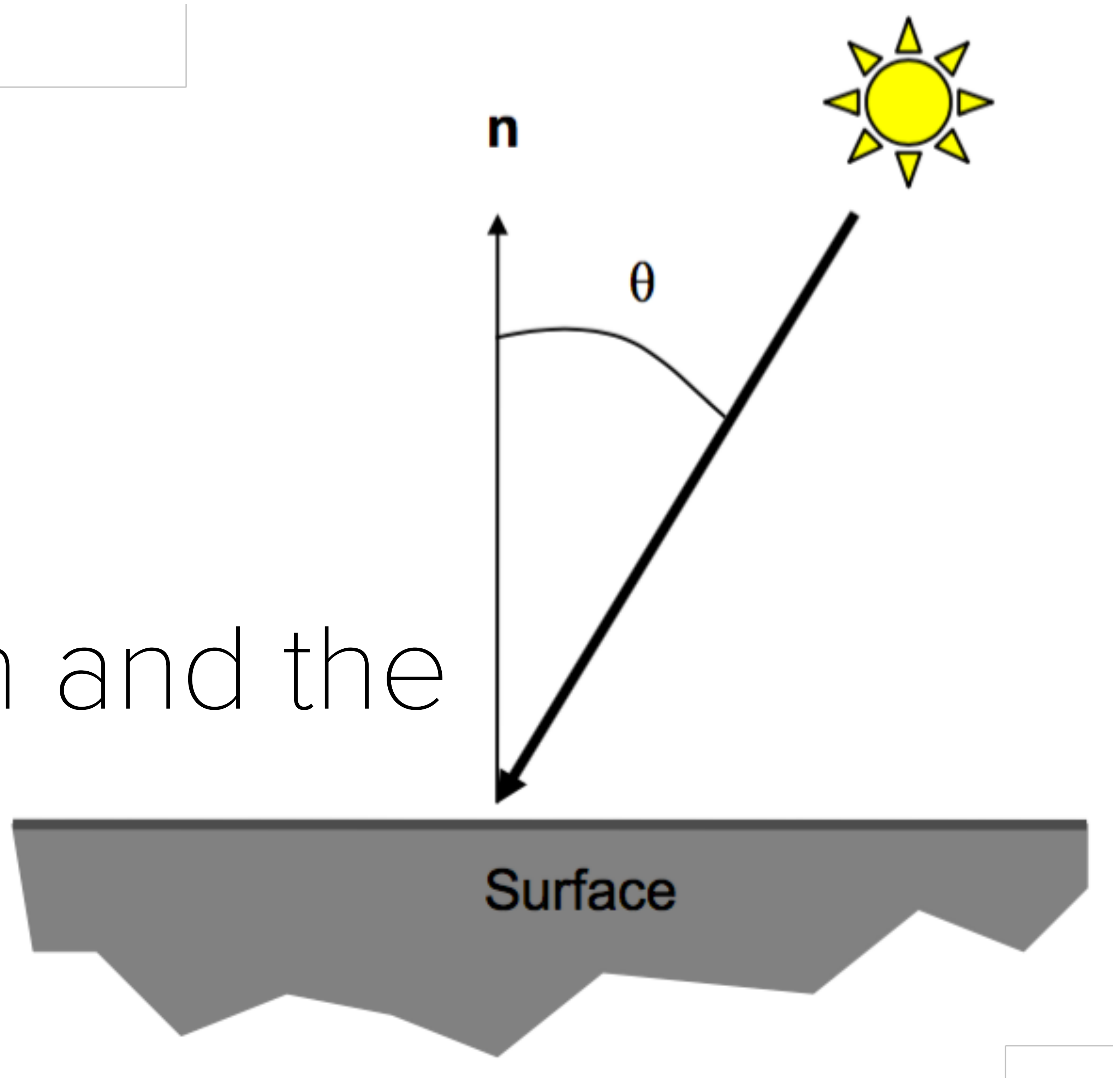
For each light, the irradiance at a point *x* is:

$$I_{in} = I_{light} \cos \theta$$

I_{in} is the irradiance at surface point

I_{light} is the intensity/color of the light

θ is the angle between light direction and the surface normal



Mathematical toolbox

Lights

Materials

Diffuse shading

Next week

Recap of ray-sphere intersections

Mob programming (spheres, normals)

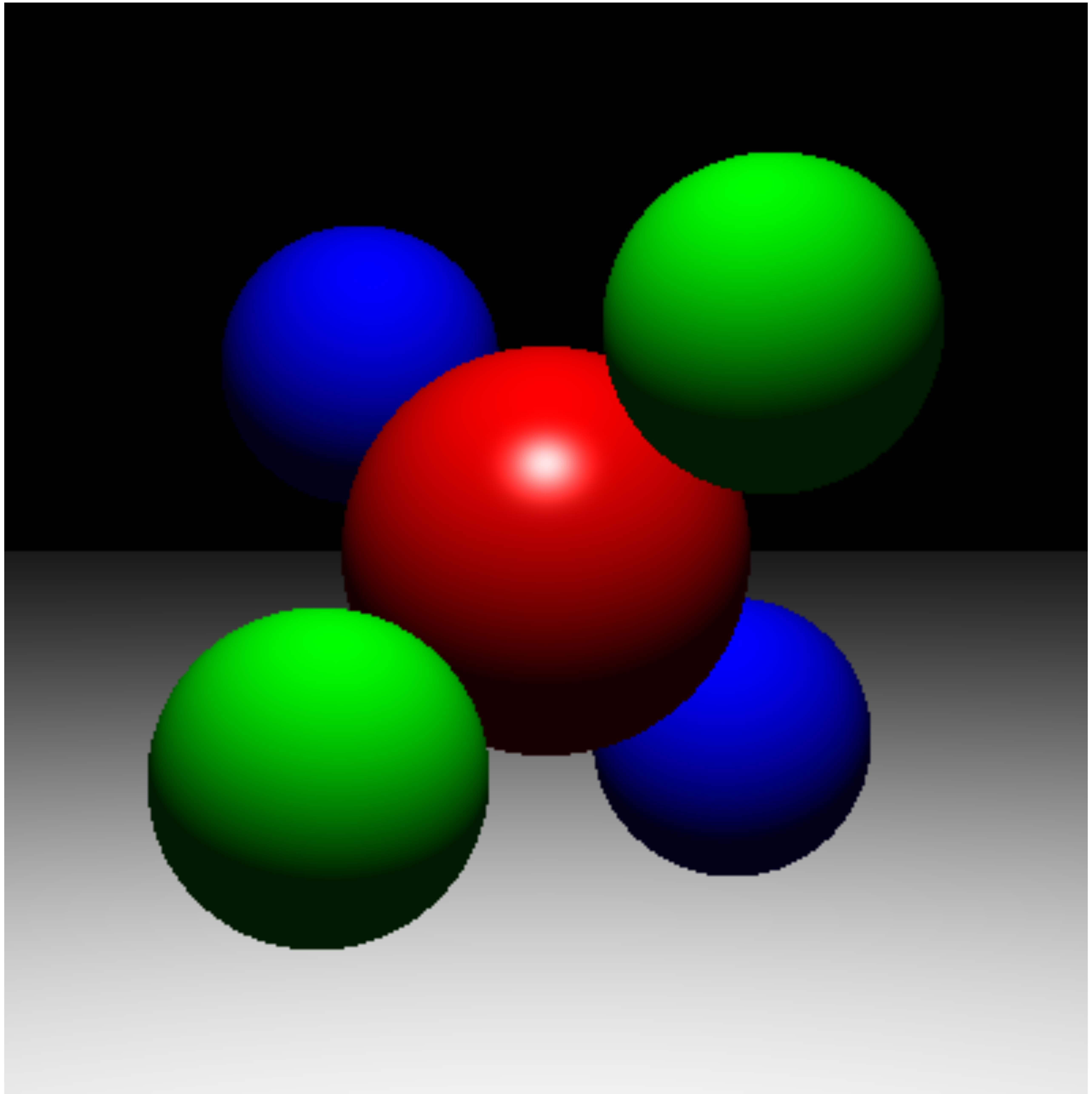
BRDF

Bidirectional Reflectance
Distribution Function

Ratio of light coming from one direction that gets reflected in another direction

Used to approximate real materials

See original slides for more info



Mathematical toolbox

Lights

Materials

Diffuse shading

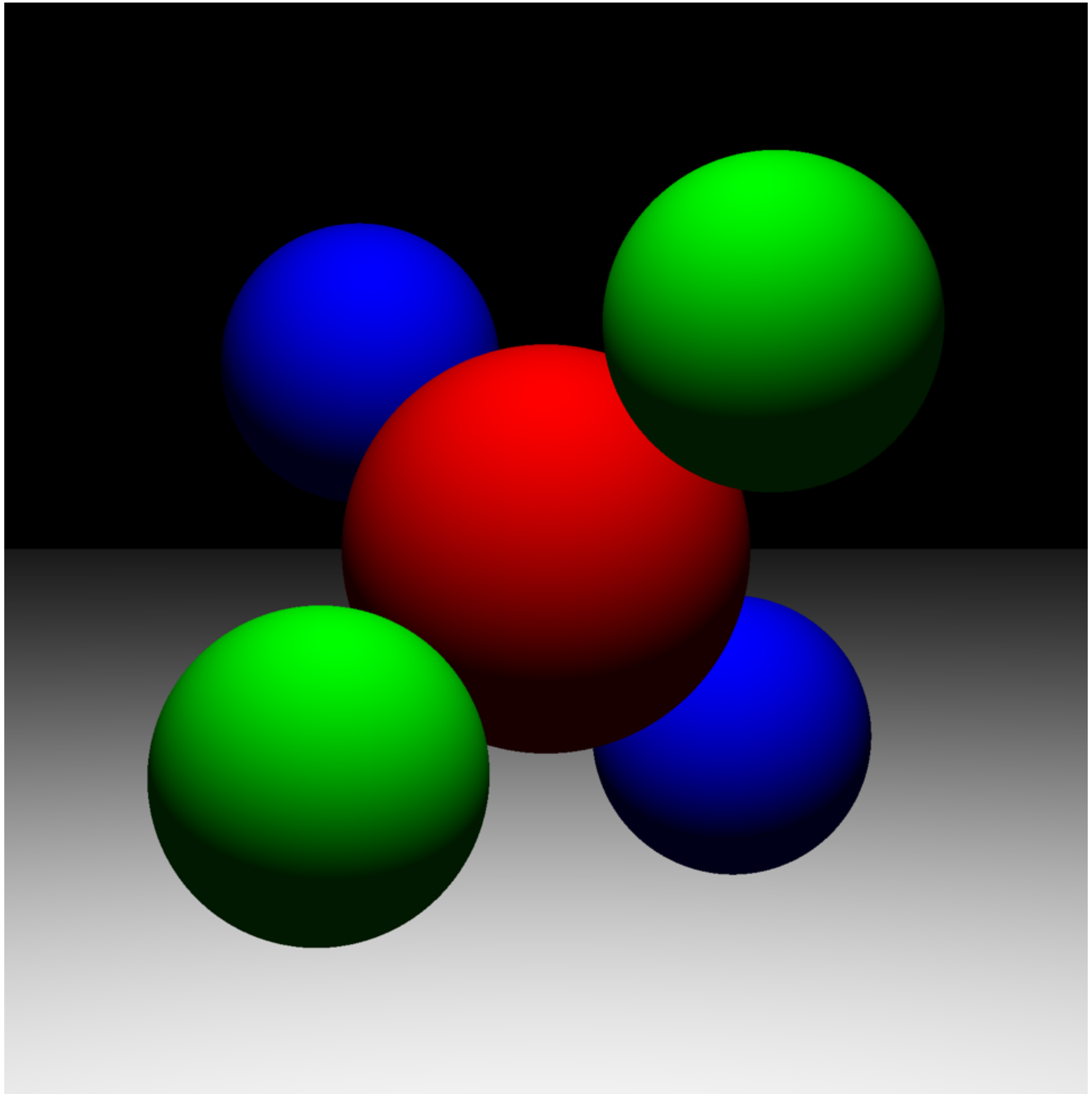
Next week

Recap of ray-sphere intersections

Mob programming (spheres, normals)

DIFFUSE SHADING

Diffuse surfaces reflect light equally in all directions

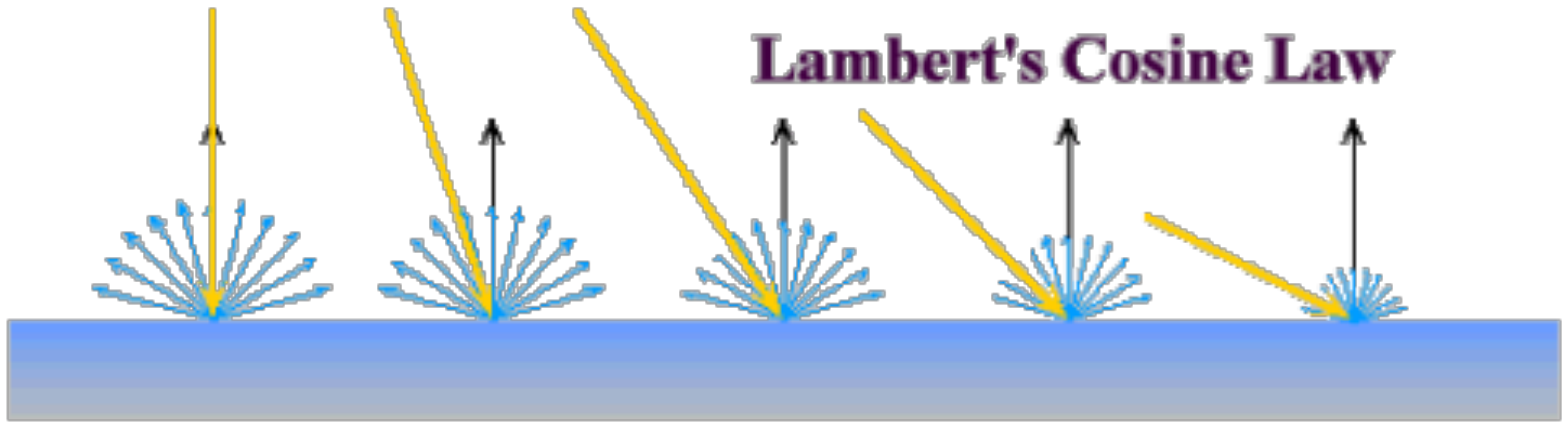


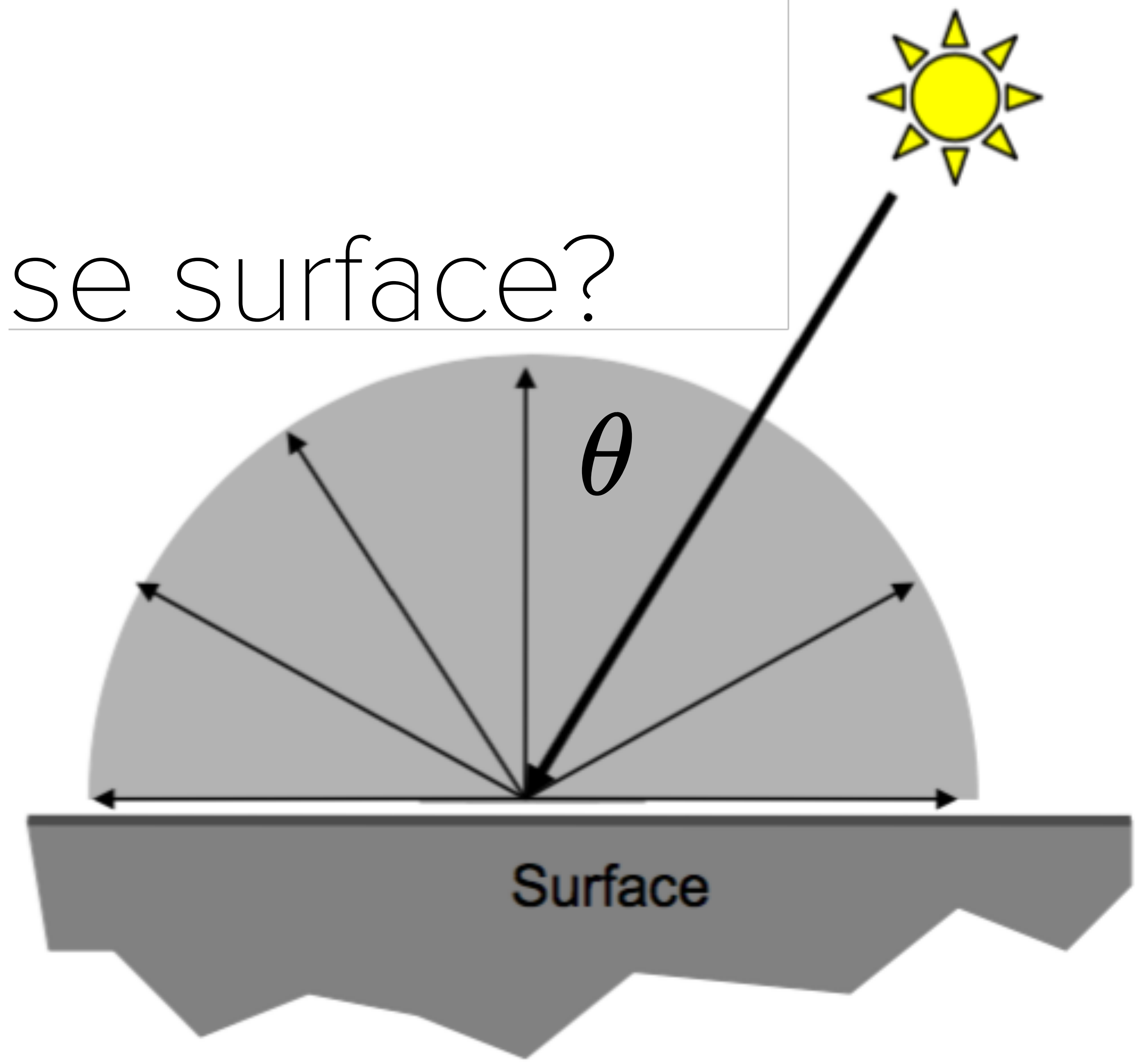
DIFFUSE SURFACES

Perfect/ideal diffuse surface is very rough at the microscopic level (chalk, clay, some paints)

Reflected light varies with cosine (Lambert's cosine law) but not viewing angle

What do we know about BRDF of ideally diffuse surface?



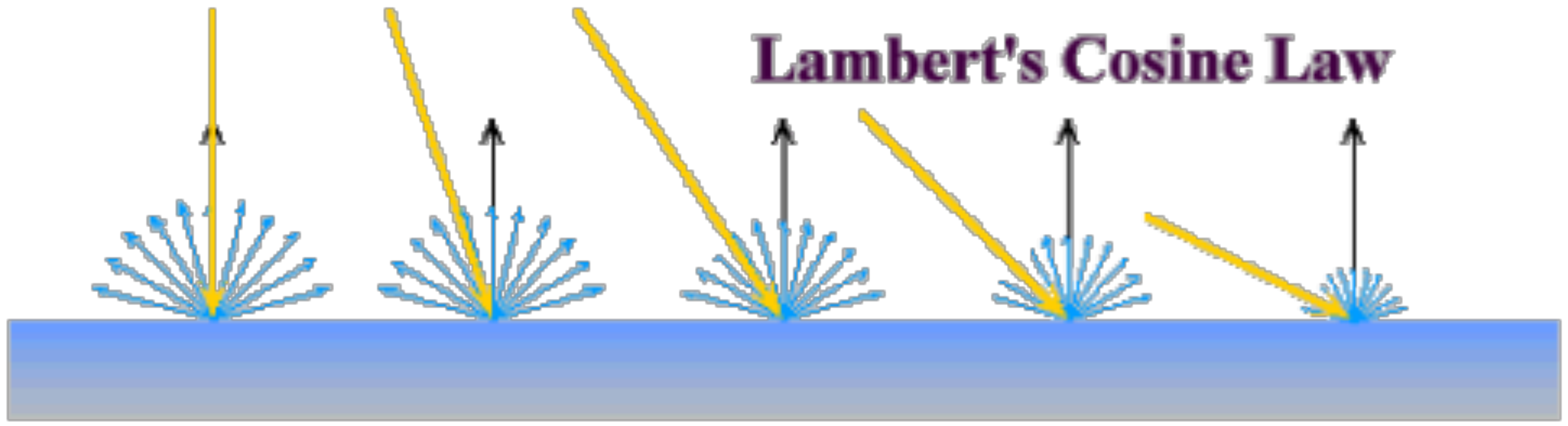


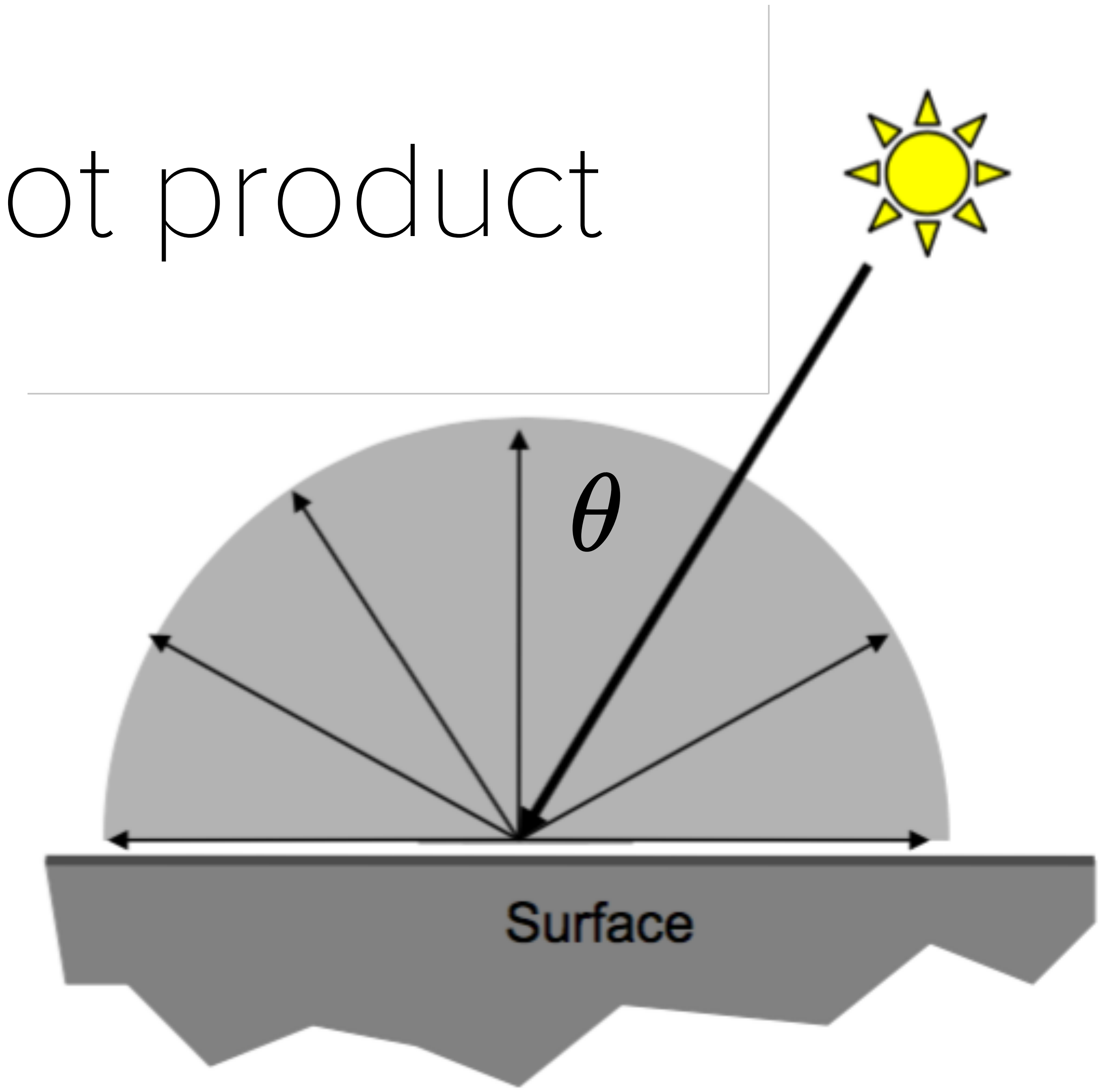
IDEAL DIFFUSE REFLECTANCE

BRDF of an ideal diffuse surface is constant

We usually refer to it as the "diffuse color", $\boldsymbol{k_d}$

Calculation across RGB is easy using dot product





DIFFUSE CALCULATION

$$L_o = k_d \, max(0, n \cdot l) \, L_i$$

For a single light source

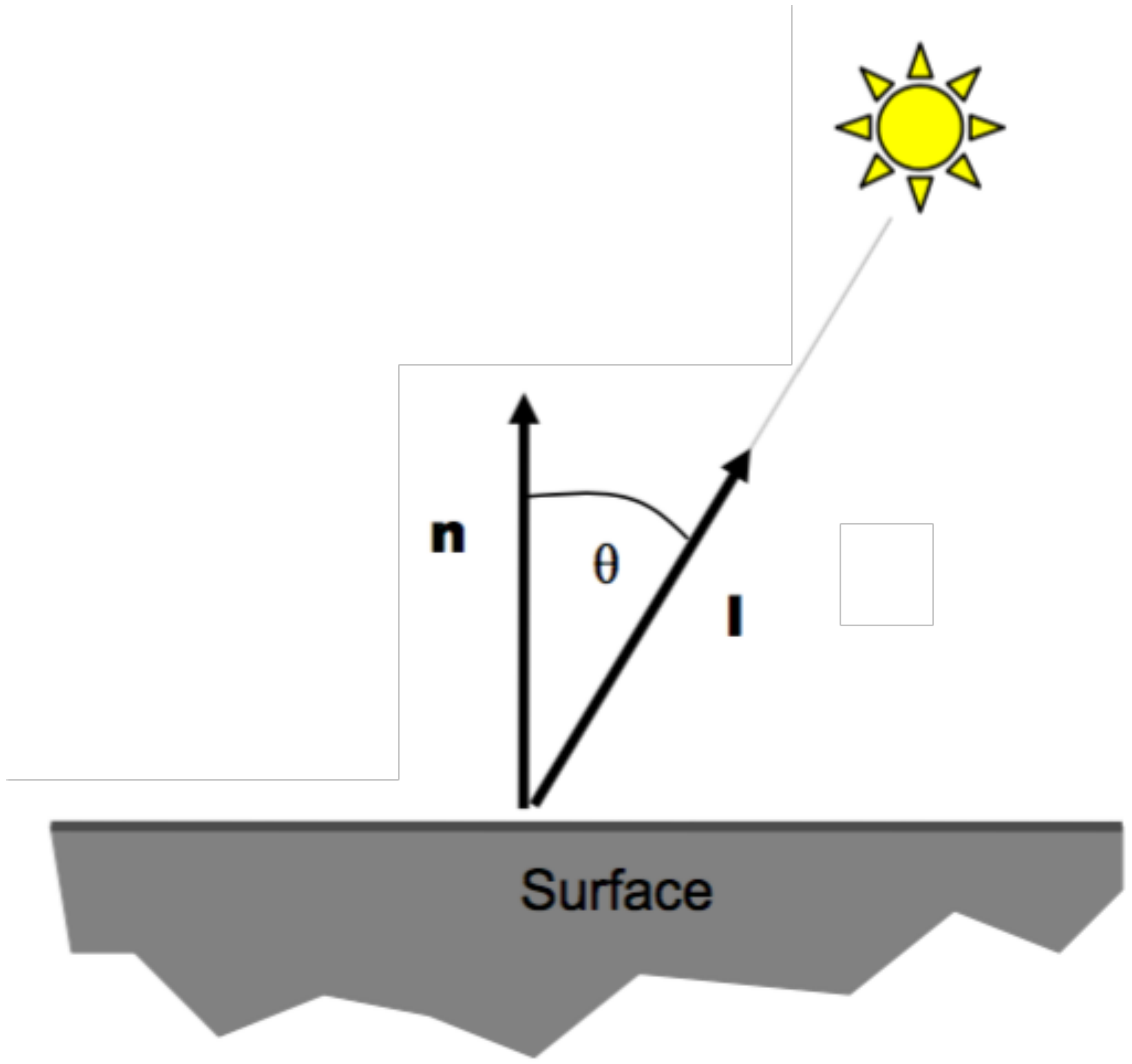
k_d diffuse coefficient (color)

n surface normal

l light direction

L_i light intensity (color)

L_o shaded color



DIFFUSE CALCULATION

$$L_o = k_d \, max(0, n \cdot l) \, L_i$$

For a single light source

k_d diffuse coefficient (color)

n surface normal

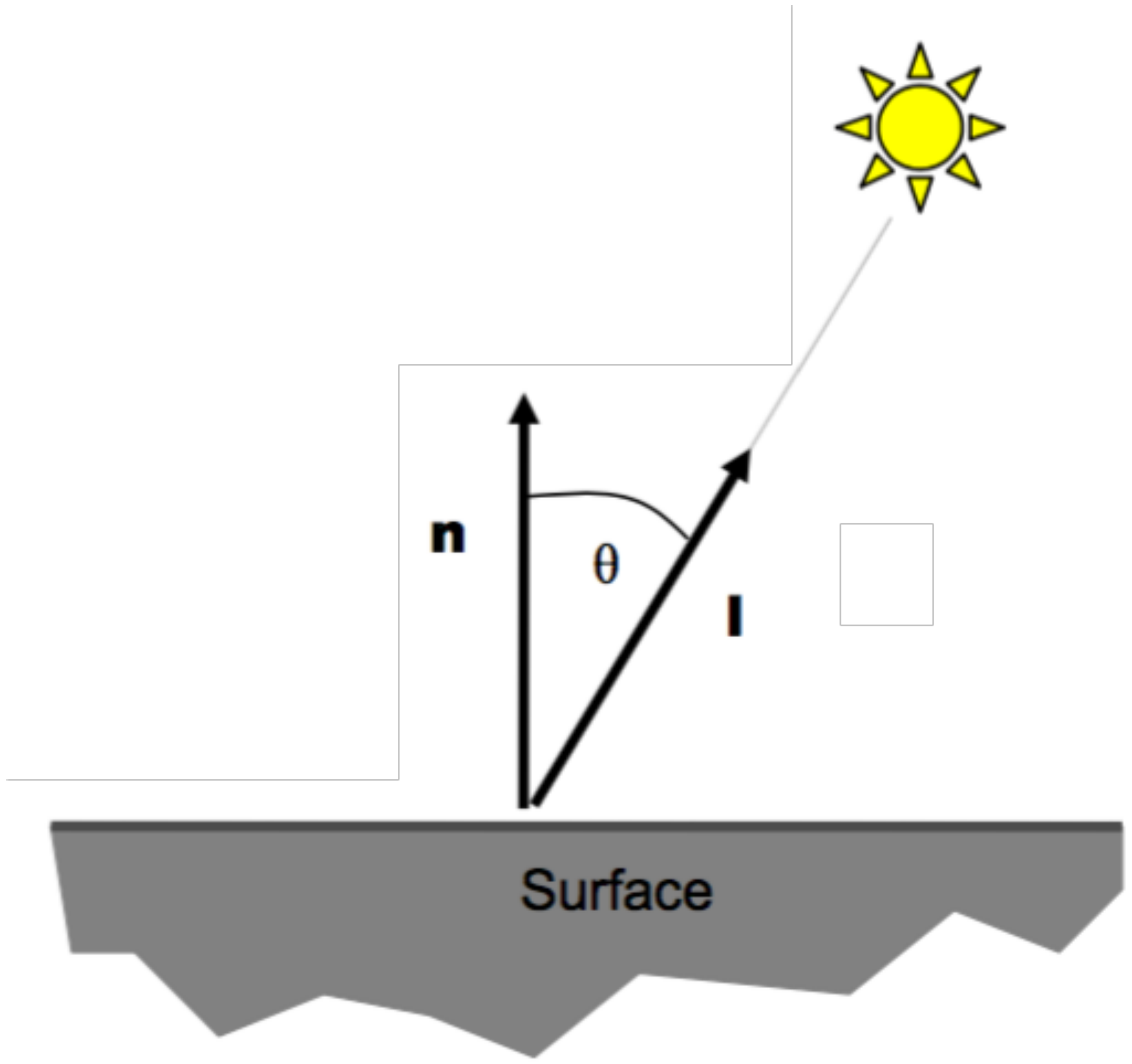
l light direction

L_i light intensity (color)

L_o shaded color

Why clamp the dot product?

Don't forget to normalize $\boldsymbol{n}$ and $\boldsymbol{l}$



Mathematical toolbox

Lights

Materials

Diffuse shading

Next week

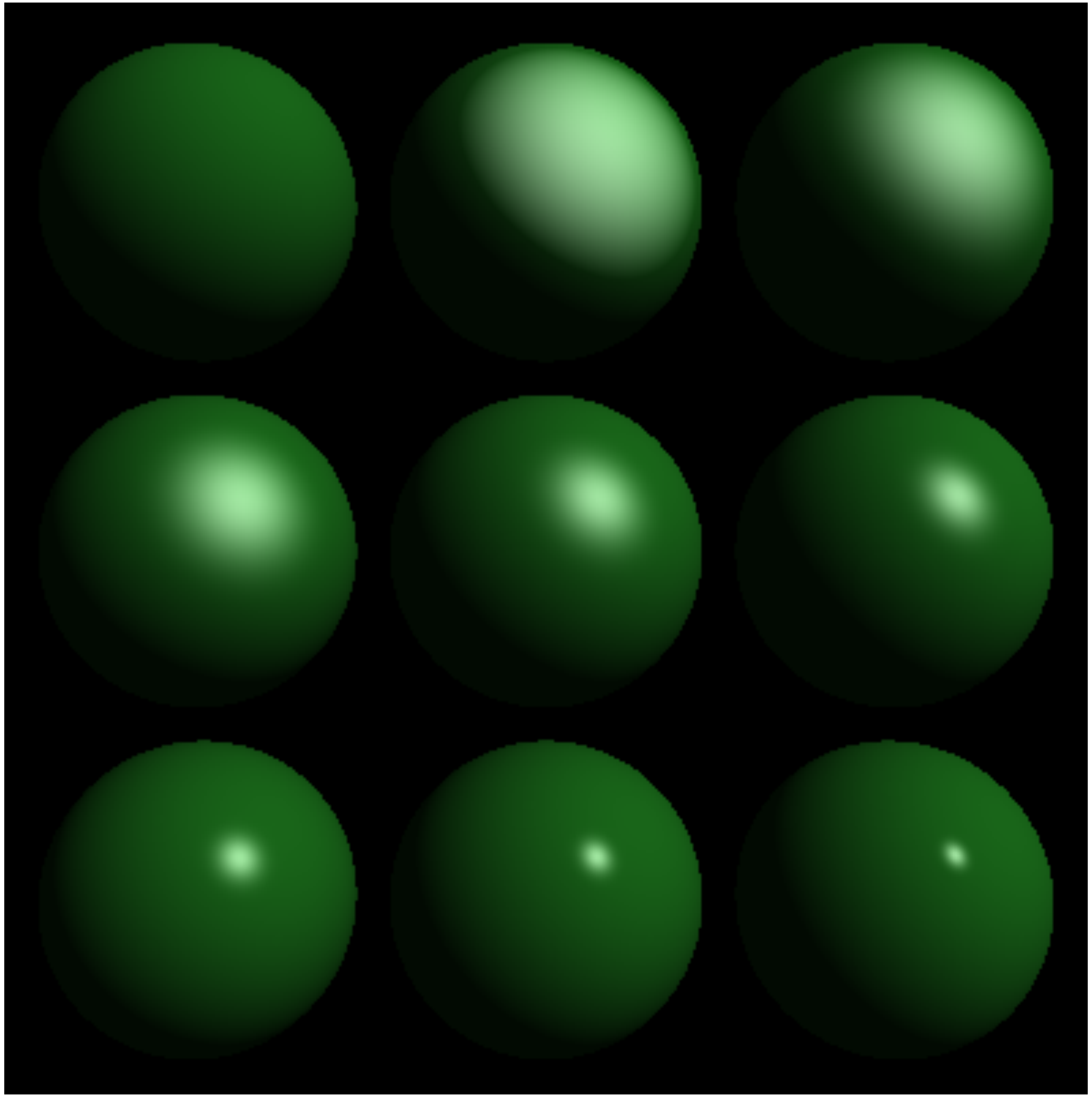
Recap of ray-sphere intersections

Mob programming (spheres, normals)

PHONG SHADING MODEL

Specular shading

Phong shading model



DUE NEXT SESSION

lights, materials, and diffuse shading

reference the guide

Mathematical toolbox

Lights

Materials

Diffuse shading

Next week

Recap of ray-sphere intersections

Mob programming (spheres, normals)

SPHERE EQUATION

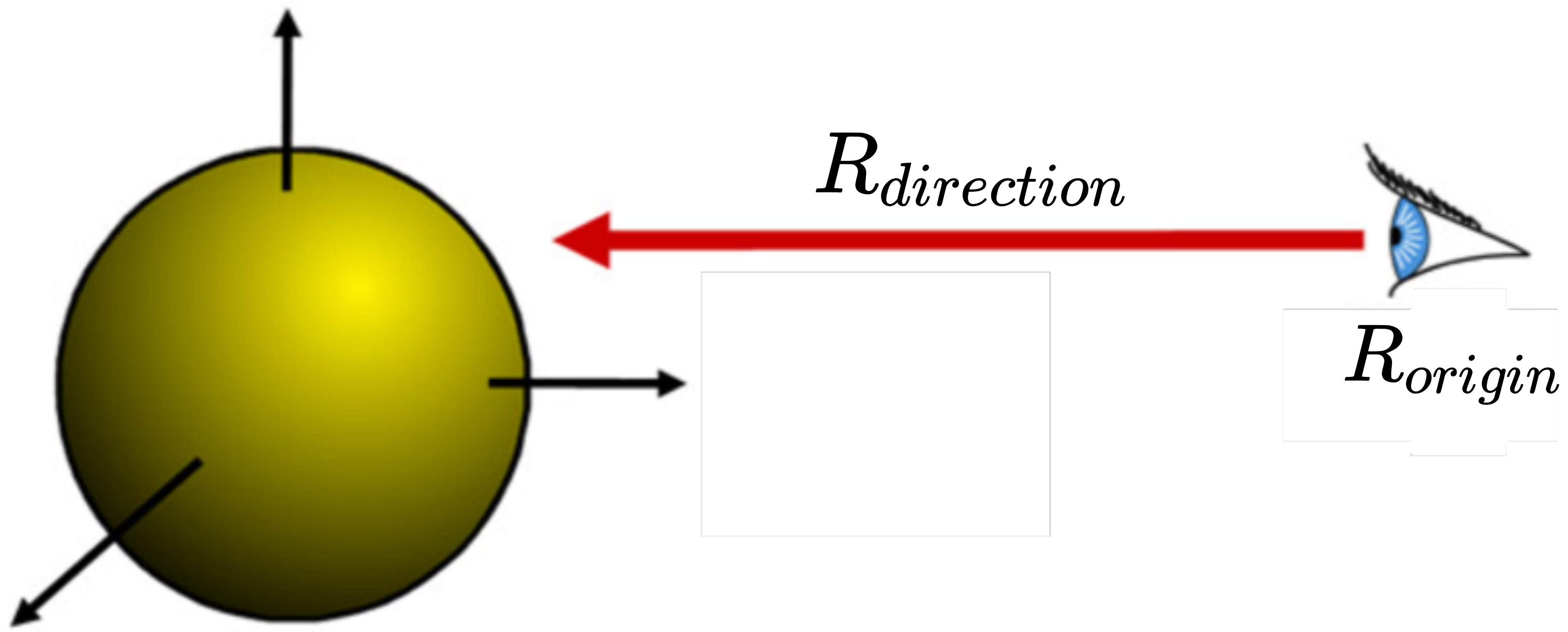
Implicit sphere equation

$$H(P) = ||P||^2 - r^2 = P \cdot P - r^2 = 0$$

Assume sphere is centered at origin

Move the ray's origin instead!

$$R_{origin} = R_{real\ origin} - H_{center}$$

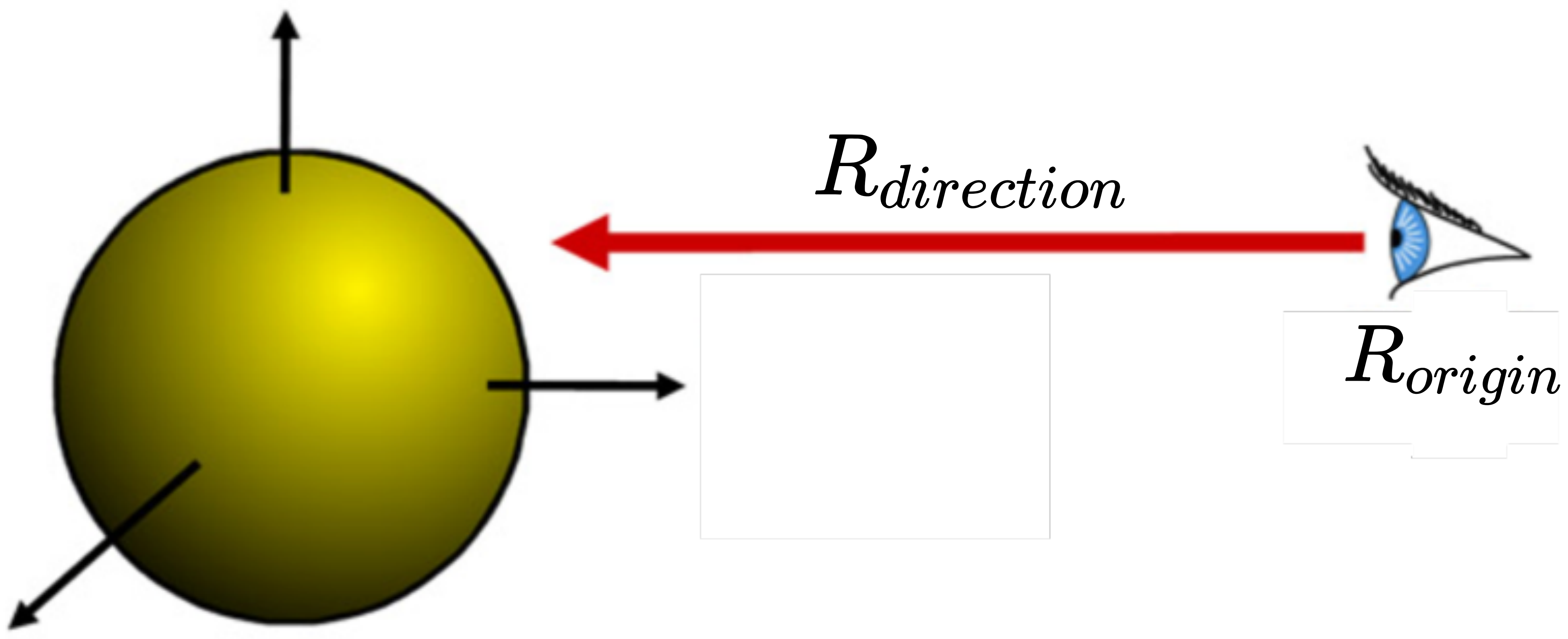


RAY-SPHERE INTERSECTION

Insert explicit ray equation into implicit plane equation and solve for *t*

$$P(t) = R_o + tR_d$$

$$H(P) = P \cdot P - r^2 = 0$$



$$(R_o + tR_d) \cdot (R_o + tR_d) - r^2 = 0$$

$$R_d \cdot R_d t^2 + 2R_d \cdot R_o t + R_o \cdot R_o - r^2 = 0$$

IT'S QUADRATIC!

Quadratic $$at^2 + bt + c = 0$$

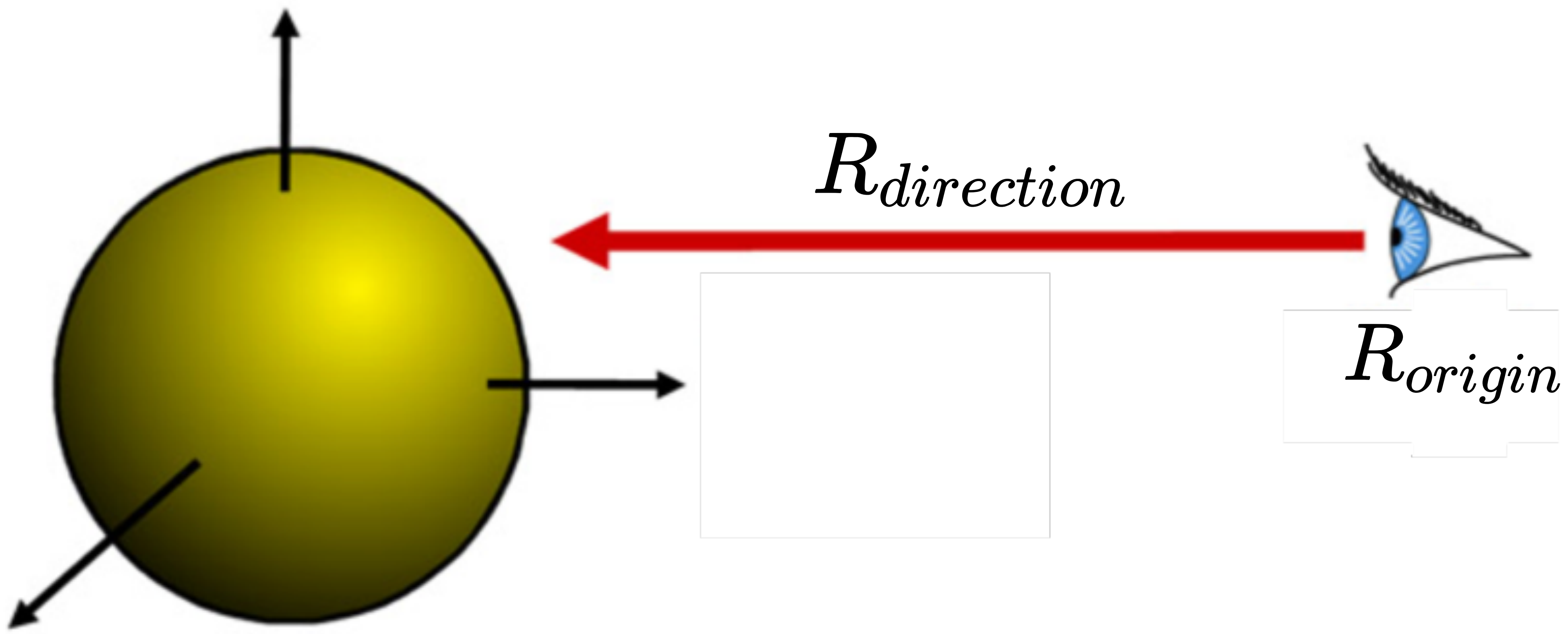
$$a = ||R_d||^2$$

$$b = 2R_d \cdot R_o$$

$$c = R_o \cdot R_o - r^2$$

Discriminant $$d = \sqrt{b^2 - 4ac}$$

Solutions $$t_{\pm} = \frac{-b \pm d}{2a}$$



SPHERE NORMALS

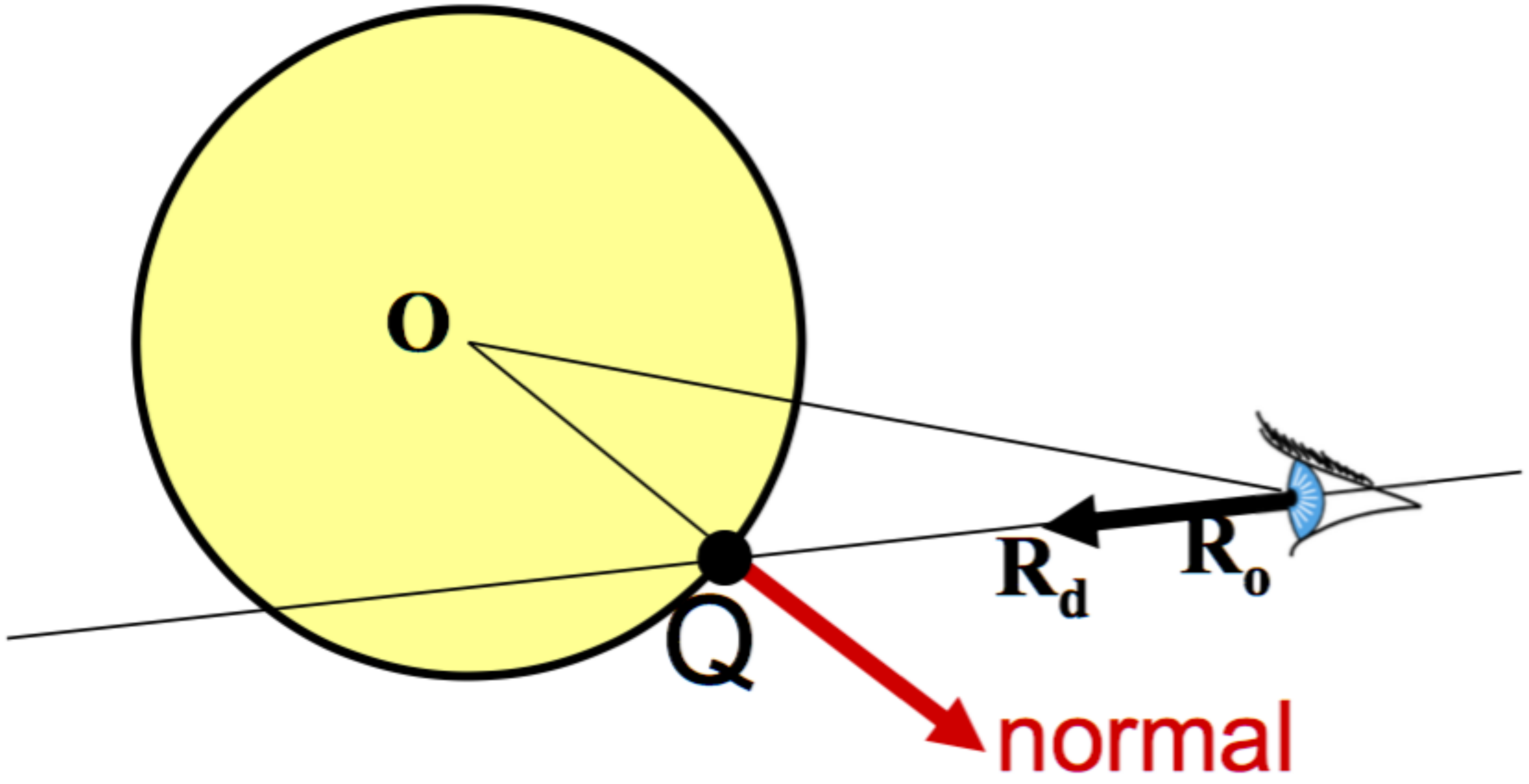
Simply

$$normalize(Q)$$

Where

$$Q = P(t)$$

or the intersection point
(for spheres centered at origin)



Mathematical toolbox

Lights

Materials

Diffuse shading

Next week

Recap of ray-sphere intersections

Mob programming (spheres & normals)