TRANSFORMATIONS

it's over NINE THOUSAAAAAAAND!

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

Transforms for intersections

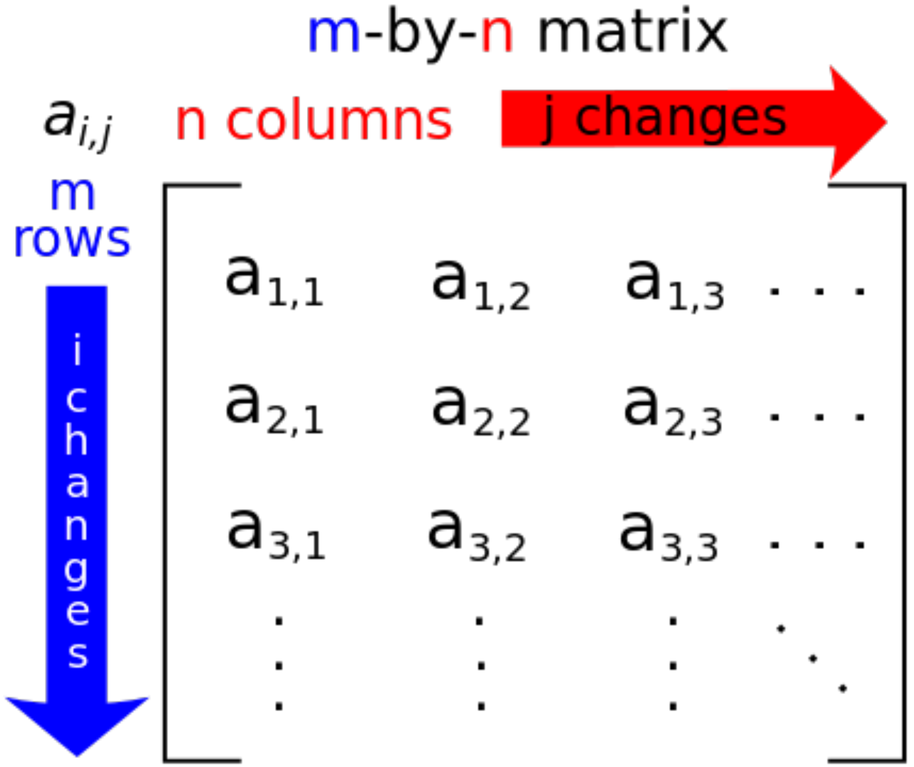
Transforms for normals

Next week

Recap of specular shading

Mob programming (specular shading)

MATRICES



https://en.wikipedia.org/wiki/Matrix_(mathematics)

https://www.khanacademy.org/math/precalculus/precalc-matrices

VOCABULARY

Identity - the result of a matrix multiplied by identity matrix is the original matrix

Transpose - a matrix that has the columns and rows swapped when compared to original

Inverse - a matrix that when multiplied by the original, outputs the identity matrix

WHY DO WE CARE?

We'll use matrices to store and apply our transforms

A vector multiplied by a transformation matrix will output a transformed vector!

We'll use 4D matrices for the sake of translation (impossible in 3D)

http://metalbyexample.com/linear-algebra/

READ THROUGH MATHHELPERS.SWIFT

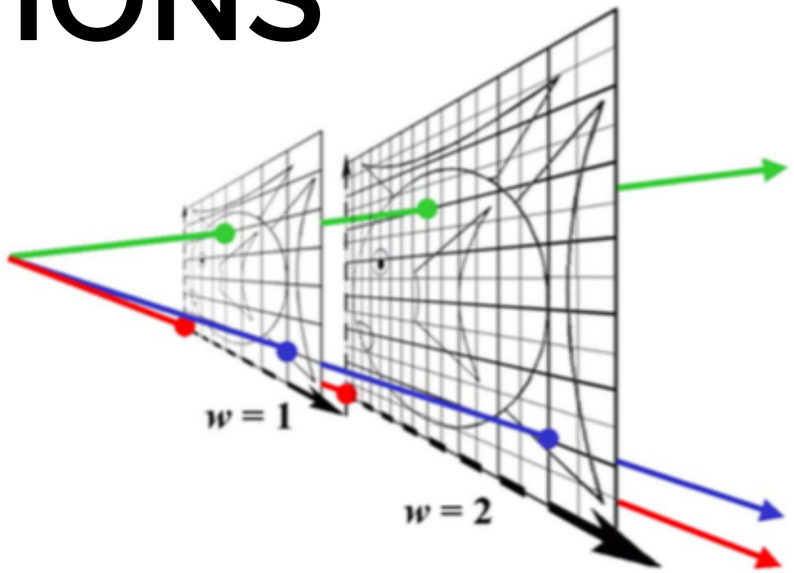
TRANSFORMS ON POINTS VS DIRECTIONS

Transform point

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} ax+by+cz+d \\ ex+fy+gz+h \\ ix+jy+kz+l \\ 1 \end{bmatrix}$$

Transform direction

$$\begin{bmatrix} x' \\ y' \\ z' \\ 0 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} = \begin{bmatrix} ax+by+cz \\ ex+fy+gz \\ ix+jy+kz \\ 0 \end{bmatrix}$$



Homogeneous coordinates (x, y, z, w)
w = 0 is a point at infinity (direction)

We'll apply all transforms in 4D, logic for point and direction is already written for you. See MathHelper.swift!

Mathematical Toolbox

Transforms for intersections

Transforms for normals

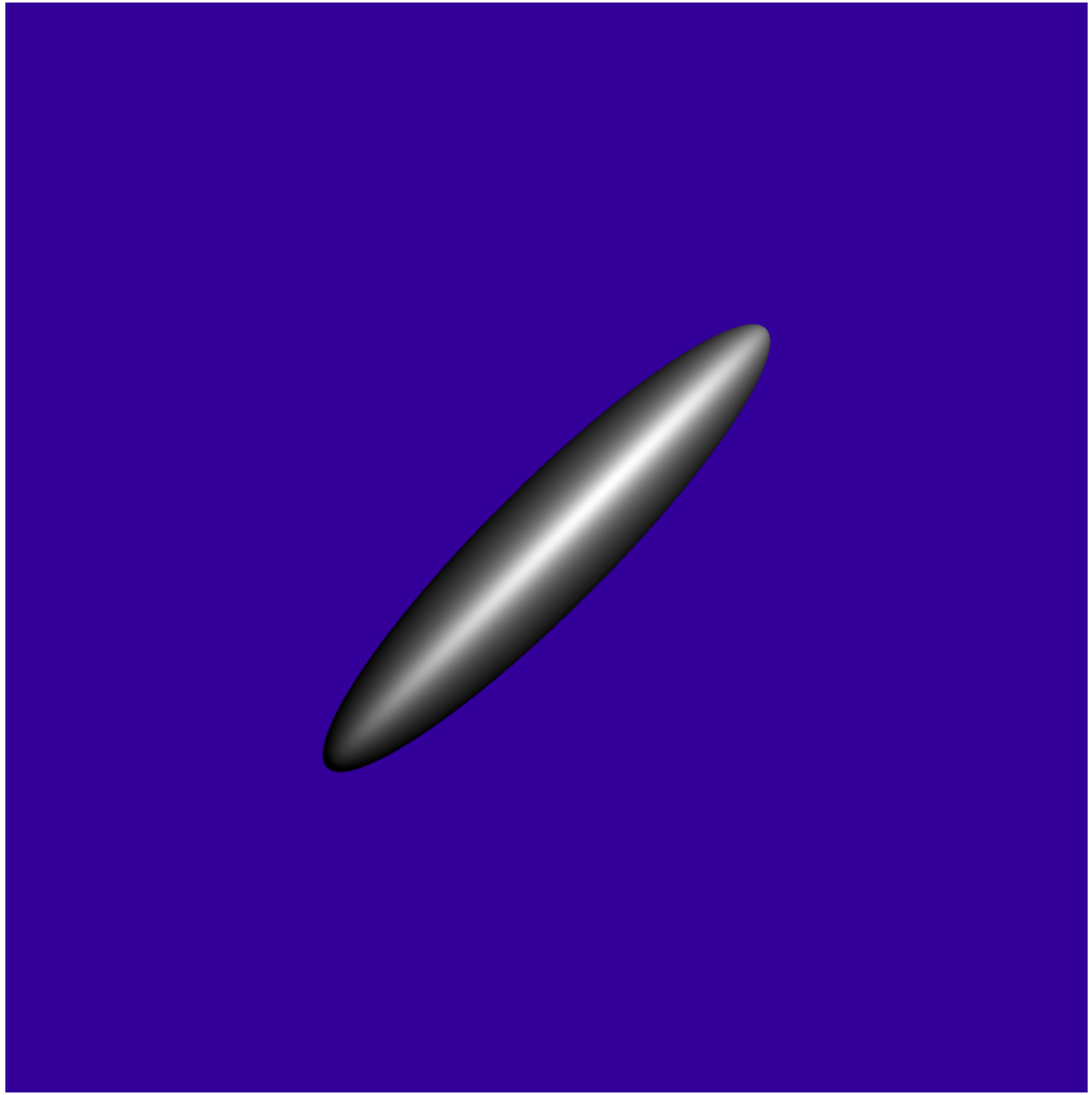
Next week

Recap of specular shading

Mob programming (specular shading)

TRANSFORMS

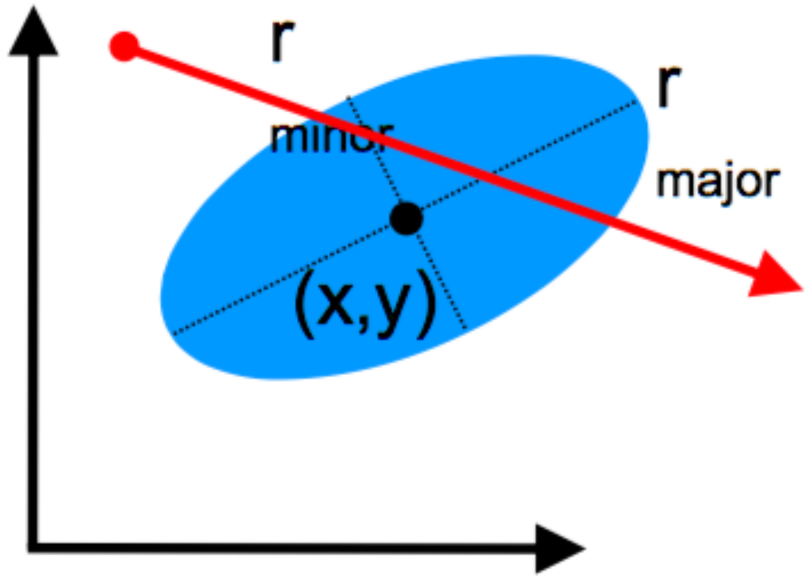
Manipulate primitives with transformation matrices



BRUTE FORCE APPROACH

Make each primitive handle transforms

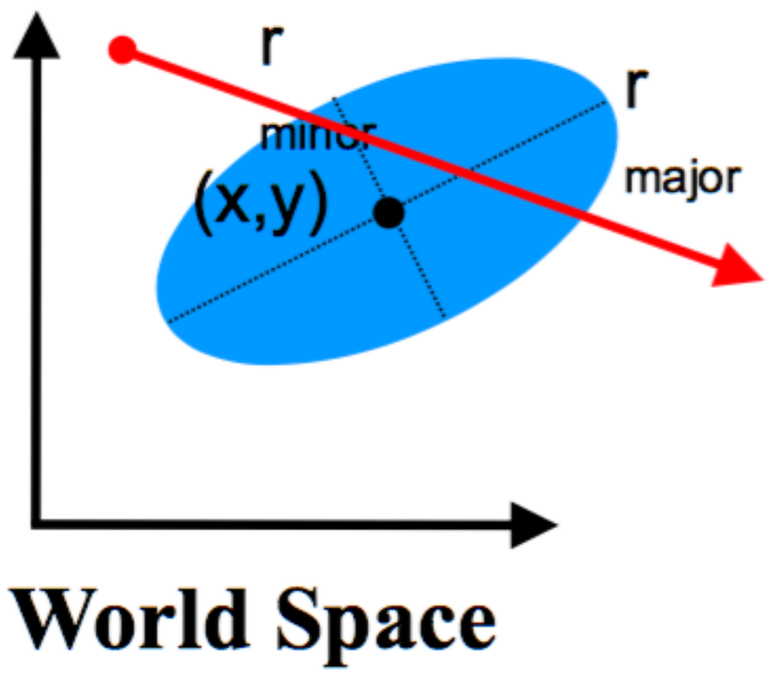
```
Sphere {
     center 3 2 0
     z_rotation 30
     r_major 2
     r_minor 1
}
```

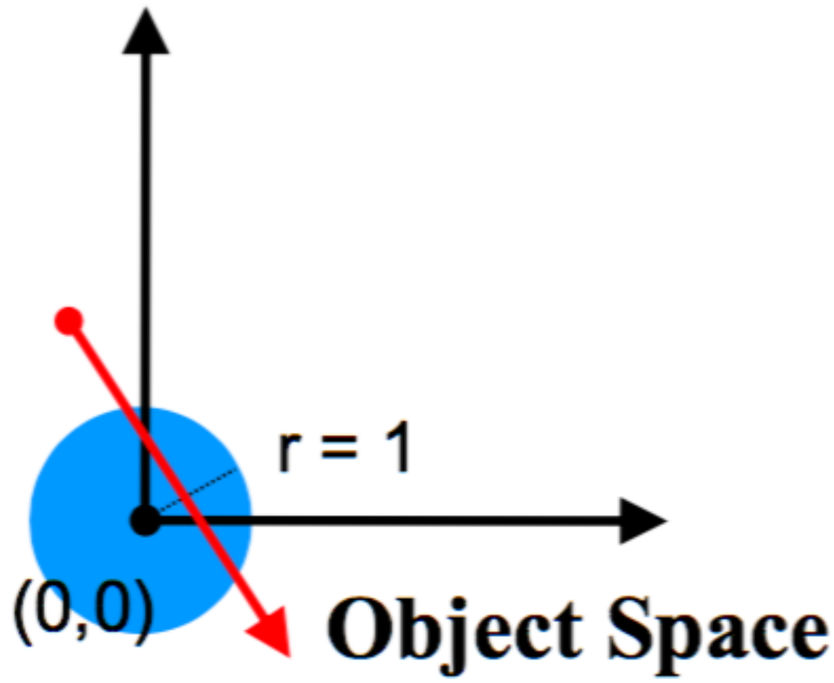


This gets complicated really fast…

A CLEVER HACK

Move the ray from *World Space* to *Object Space*





TRANSFORMING RAYS

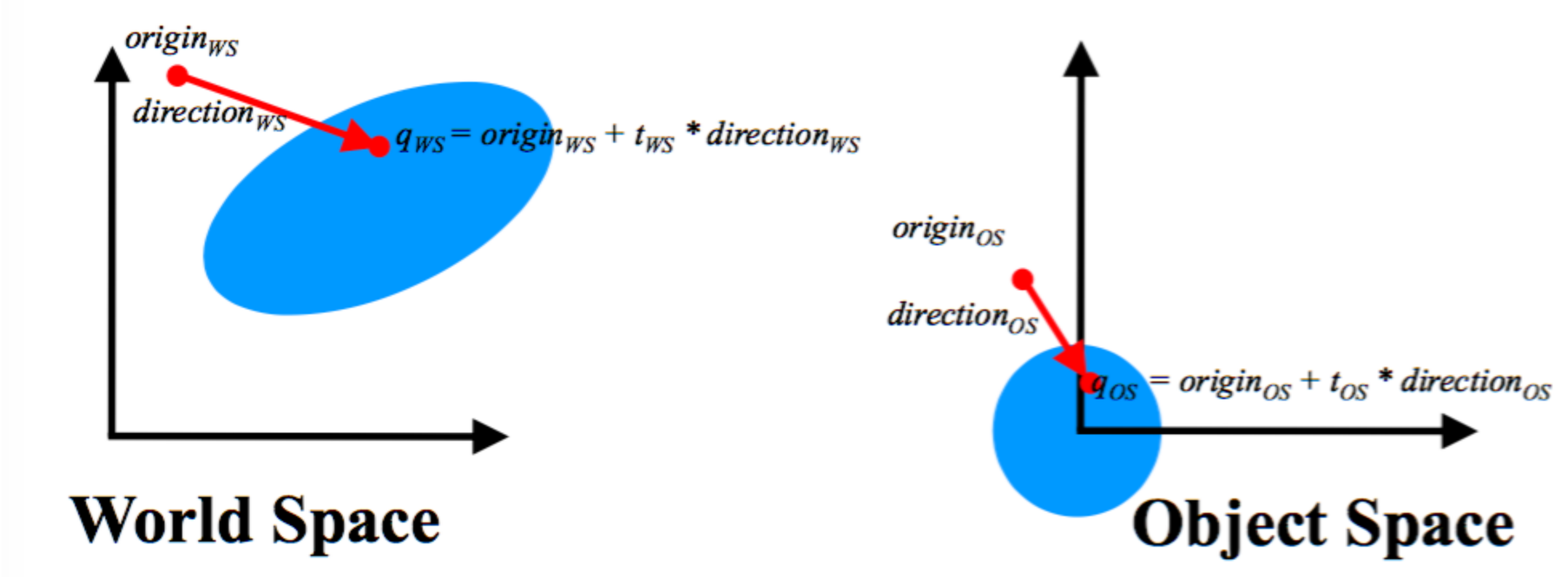
Transform origin

$$origin_{OS} = \mathbf{M}^{-1}\, origin_{WS}$$

Transform direction

$$direction_{OS} = \mathbf{M}^{-1}\,(origin_{WS} + 1 * direction_{WS}) \;-\; \mathbf{M}^{-1}\, origin_{WS}$$

$$direction_{OS} = \mathbf{M}^{-1}\; direction_{WS}$$



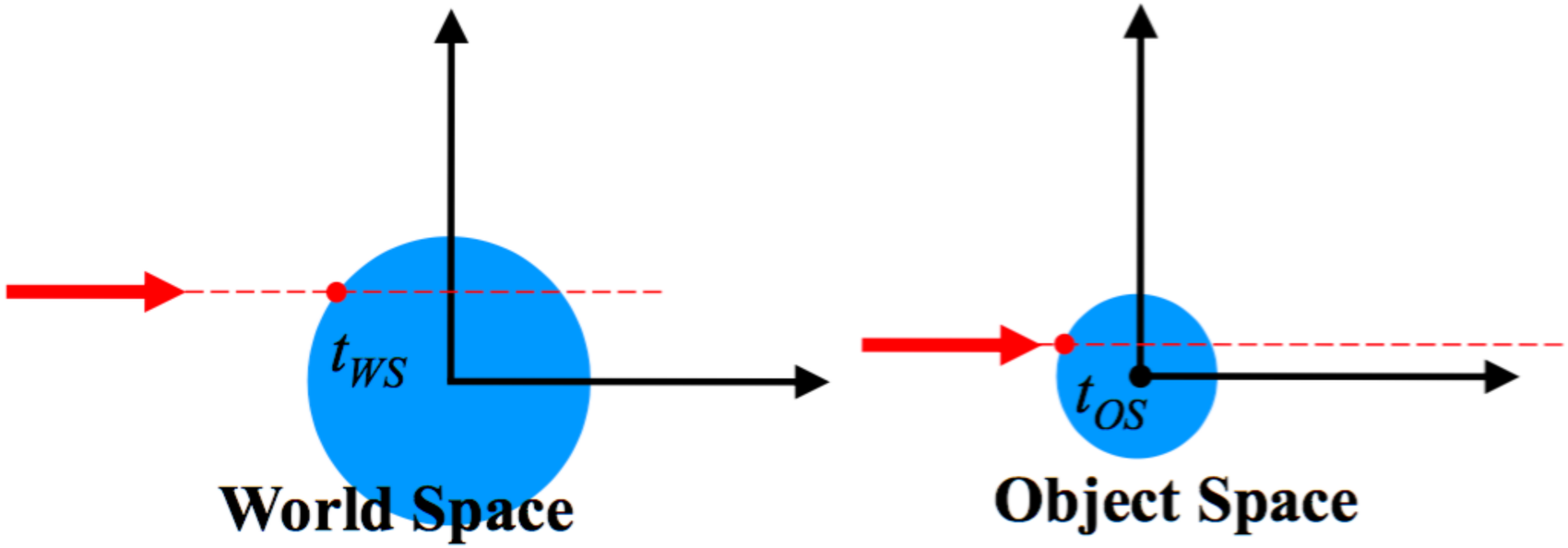
WHAT ABOUT T?

If your transform matrix includes scaling, the transformed direction will not be normalized

Do you normalize the transformed direction or not?

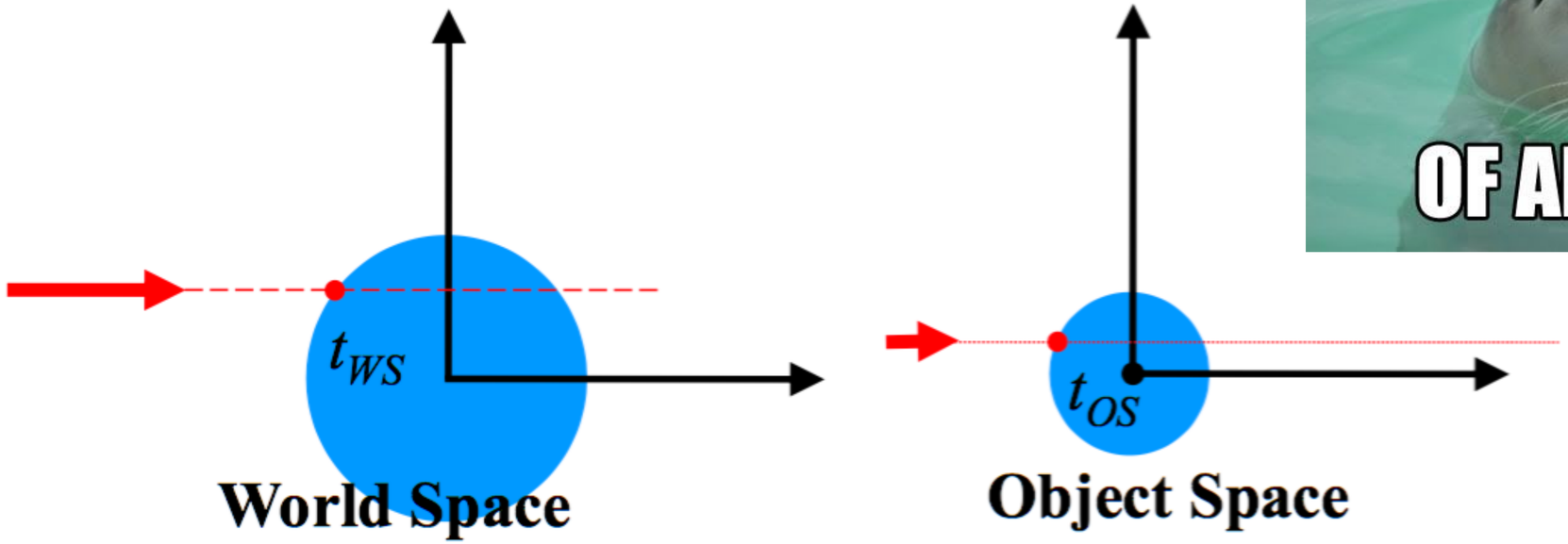
NORMALIZE DIRECTION?

$t_{os} \neq t_{ws}$ and must be rescaled after intersection



DO NOT NORMALIZE DIRECTION

$t_{os} = t_{ws}$ but you cannot rely on t_{os} being the true distance in your intersection equations



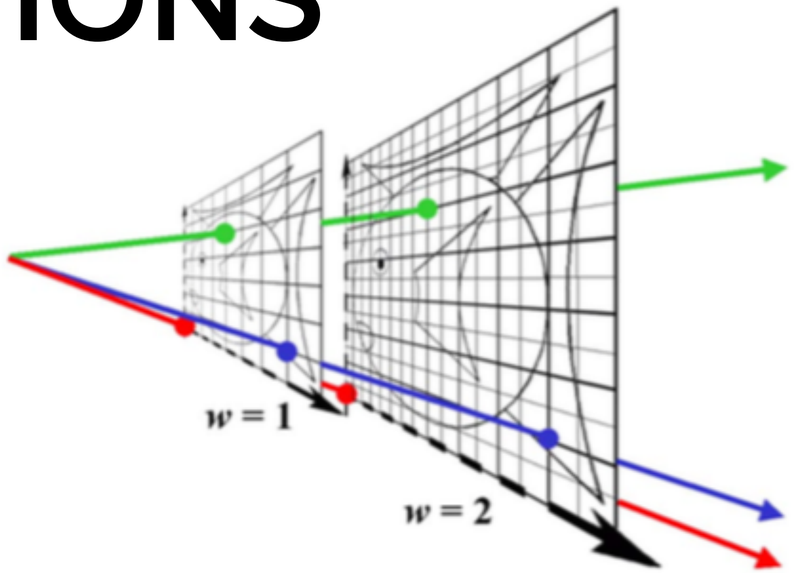
TRANSFORMS ON POINTS VS DIRECTIONS

Transform point

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} ax+by+cz+d \\ ex+fy+gz+h \\ ix+jy+kz+l \\ 1 \end{bmatrix}$$

Transform direction

$$\begin{bmatrix} x' \\ y' \\ z' \\ 0 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} = \begin{bmatrix} ax+by+cz \\ ex+fy+gz \\ ix+jy+kz \\ 0 \end{bmatrix}$$



Homogeneous coordinates (x, y, z, w)
w = 0 is a point at infinity (direction)

We'll apply all transforms in 4D, logic for point and direction is already written for you. See MathHelper.swift!

Mathematical Toolbox

Transforms for intersections

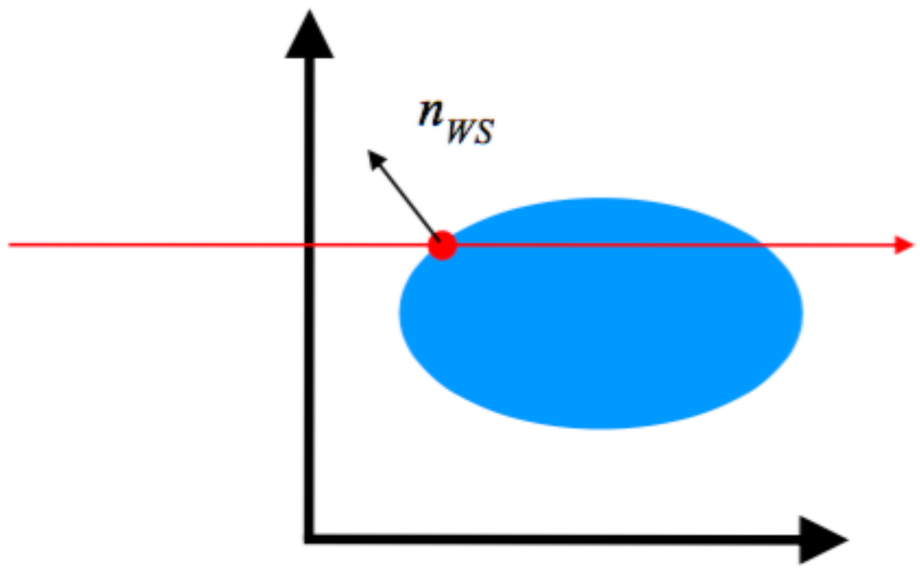
Transforms for normals

Next week

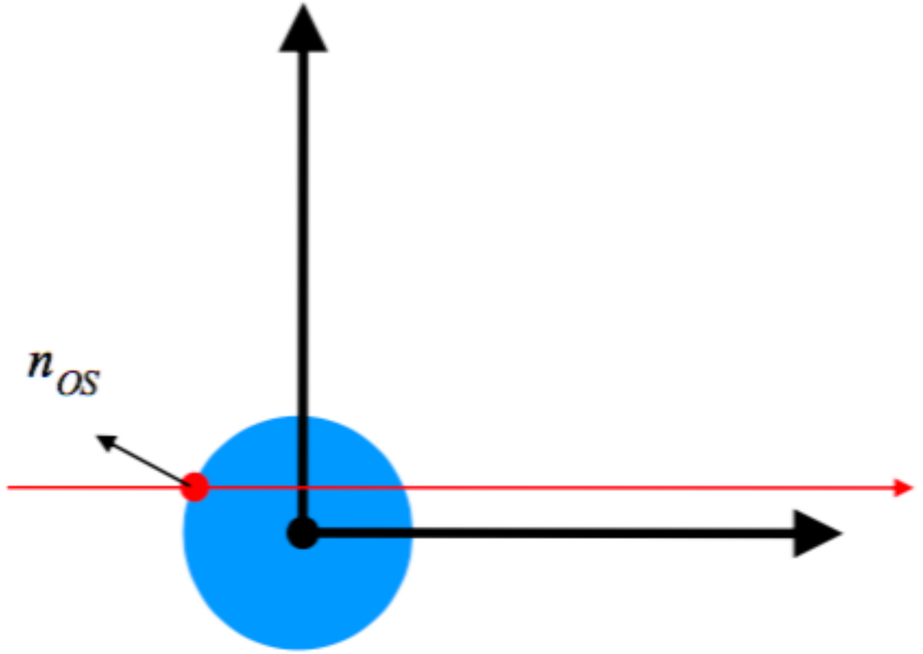
Recap of specular shading

Mob programming (specular shading)

WHAT ABOUT NORMALS?



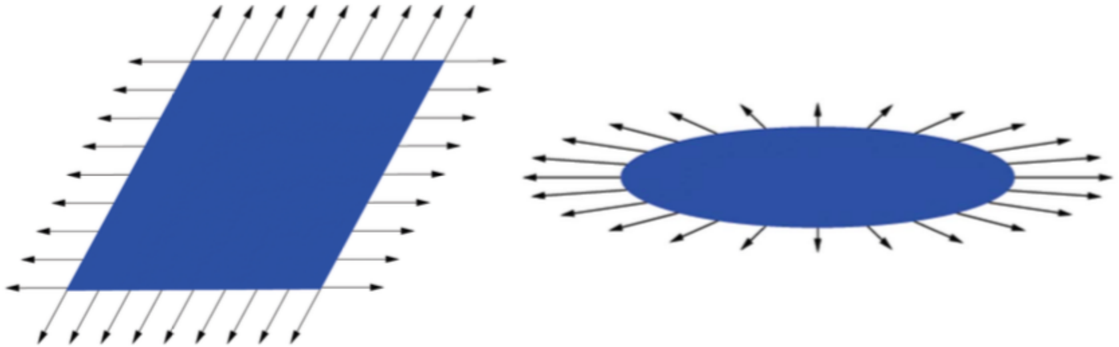
World Space



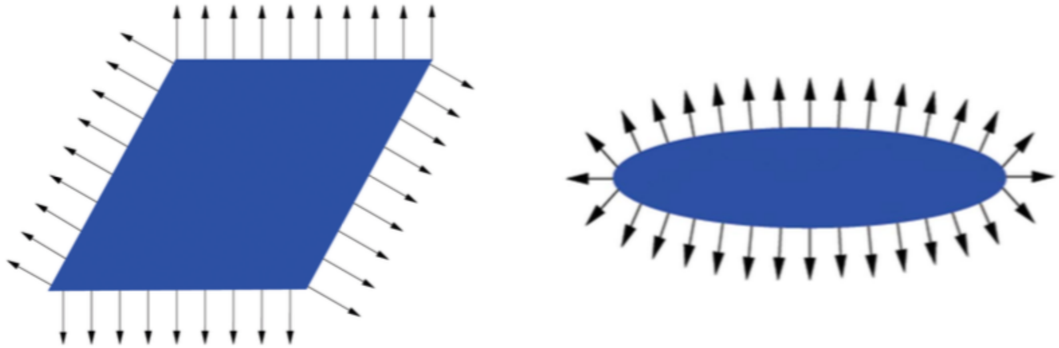
Object Space

FAILS WITH SCALE AND SHEAR

Incorrect :(

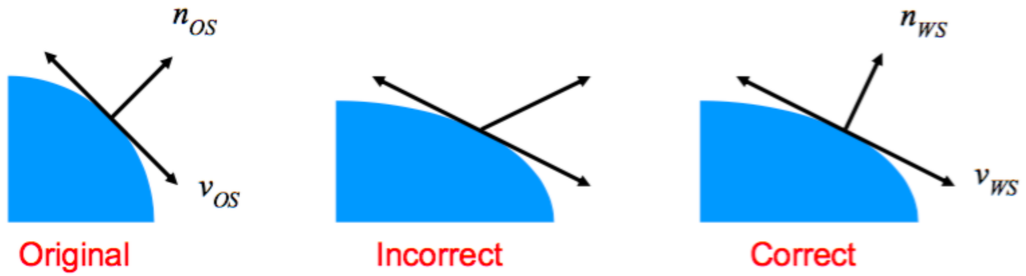


Correct! :)



SO HOW DO WE DO IT RIGHT?

Think of about transforming the tangent plane to the normal, not the normal vector



Pick any vector in the tangent plane, how is it transformed by a matrix?

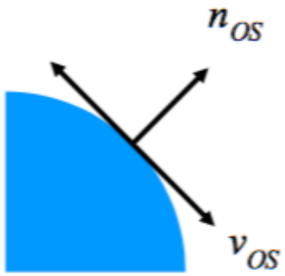
$$v_{WS} = \mathbf{M}\ v_{OS}$$

TRANSFORM TANGENT VECTOR

v is perpendicular to normal n:

Dot product

$$n_{OS}^{T} v_{OS} = 0$$



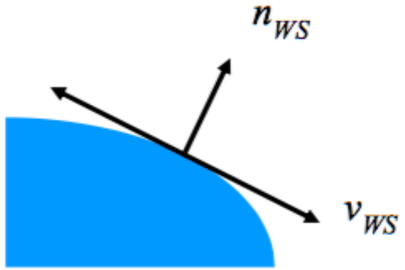
$$n_{OS}^{T} (\mathbf{M}^{-1} \mathbf{M}) v_{OS} = 0$$

$$(n_{OS}^{T} \mathbf{M}^{-1}) (\mathbf{M} v_{OS}) = 0$$

$$(n_{OS}^{T} \mathbf{M}^{-1}) v_{WS} = 0$$

v_{WS} is perpendicular to normal n_{WS}:

$$n_{WS}^{T} v_{WS} = 0$$



$$n_{WS}^{T} = n_{OS}^{T} (\mathbf{M}^{-1})$$

$$n_{WS} = (\mathbf{M}^{-1})^{T} n_{OS}$$

Mathematical Toolbox

Transforms for intersections

Transforms for normals

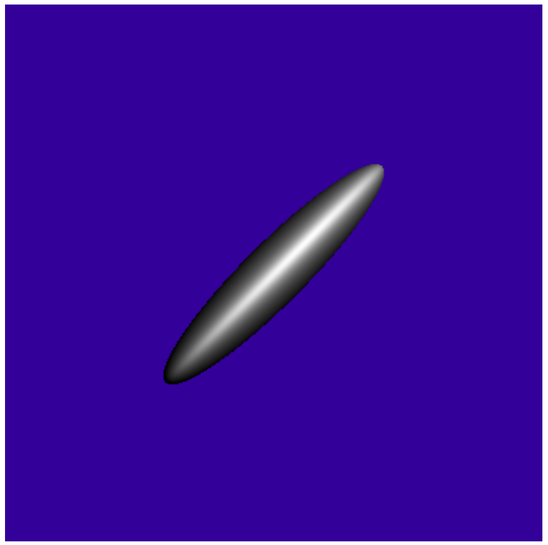
Next week

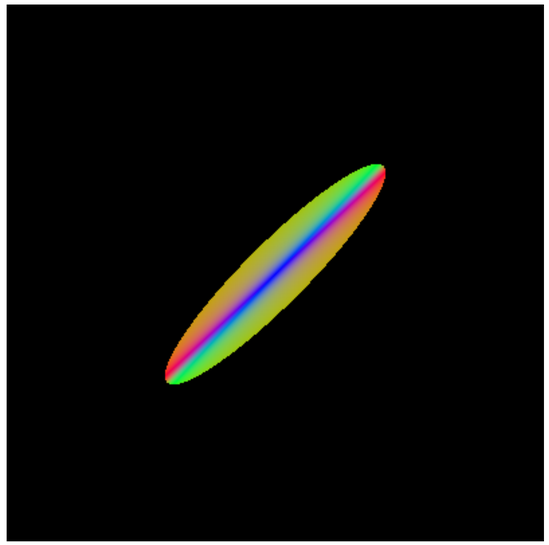
Recap of specular shading

Mob programming (specular shading)

TRIANGLES

Ray-triangle intersection
Meshes of triangles to represent complex geometry





DUE NEXT SESSION

transformations
reference the guide

Mathematical Toolbox

Transforms for intersections

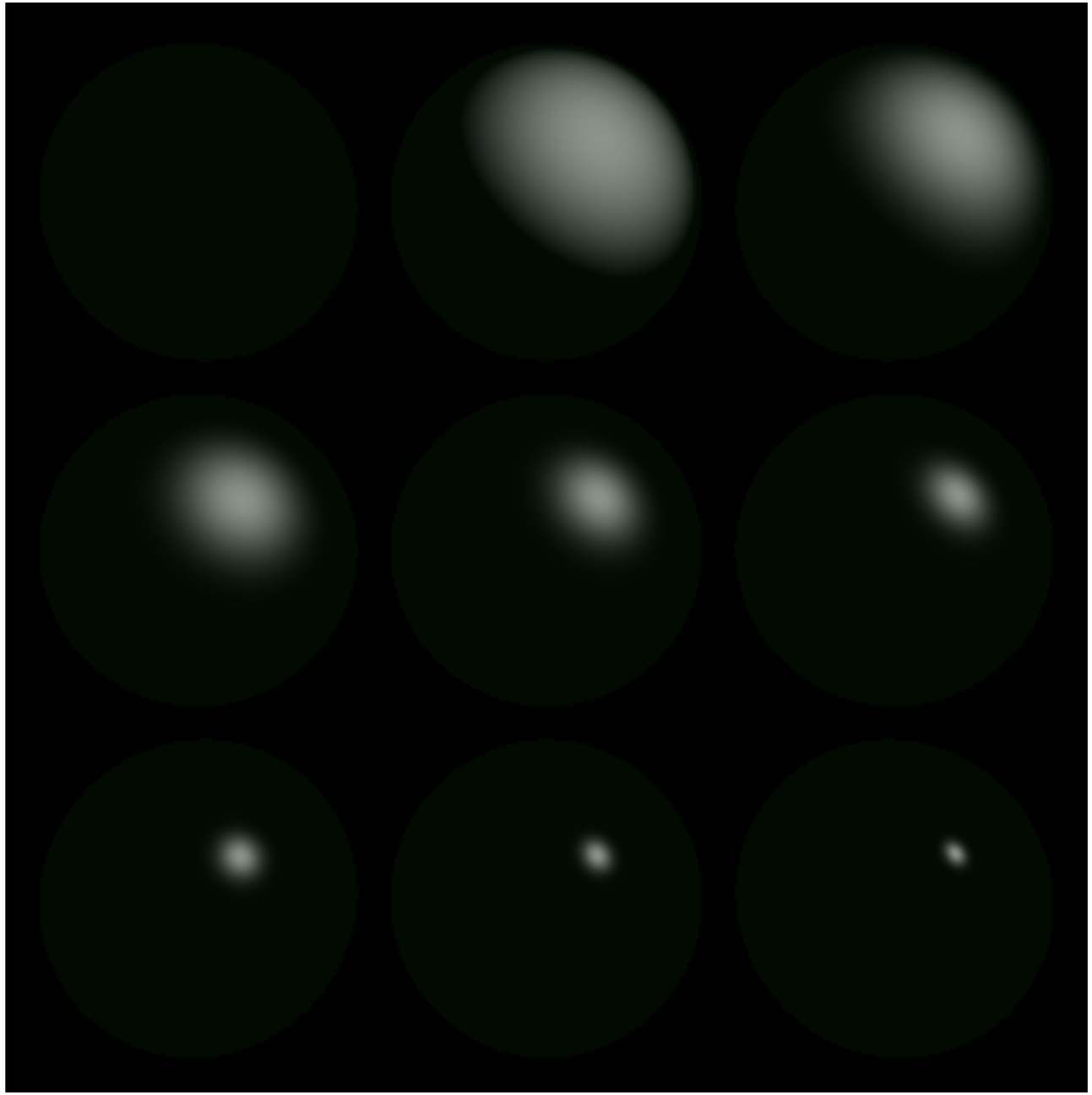
Transforms for normals

Next week

Recap of specular shading

Mob programming (specular shading)

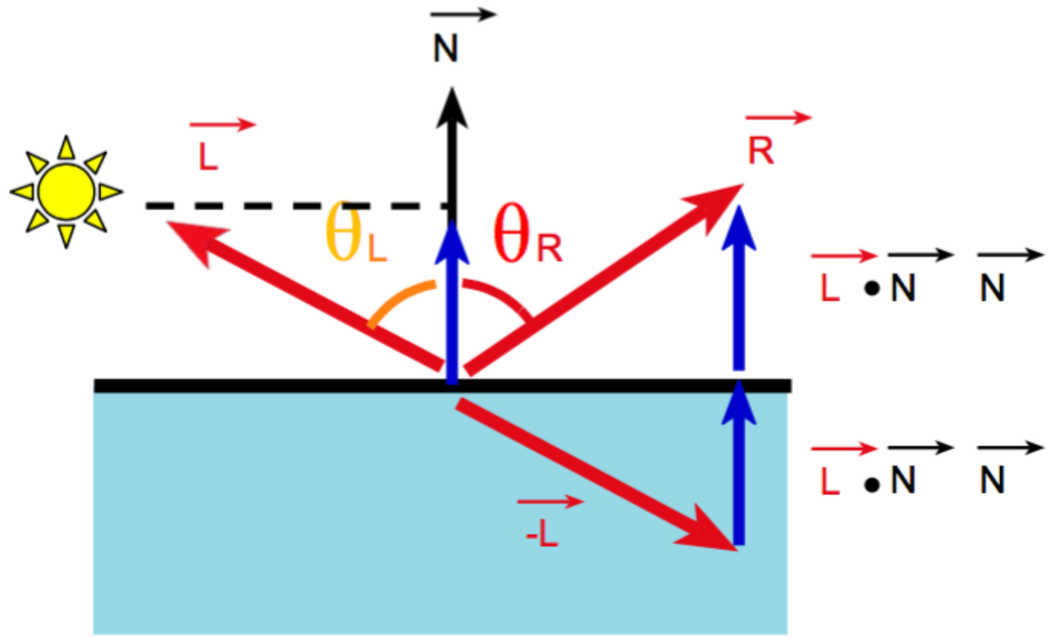
SPECULAR SHADING



HOW TO GET MIRROR DIRECTION

Reflection angle = light angle

$$R = -L + 2(L \cdot N)N$$

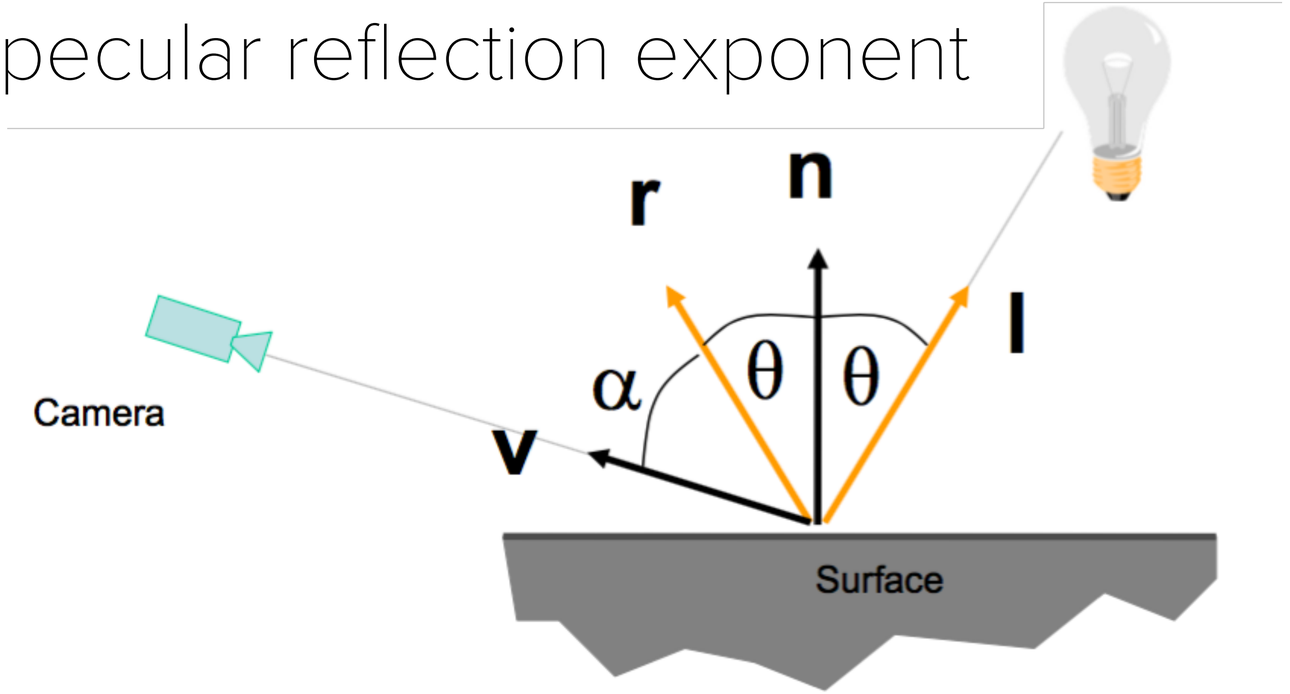


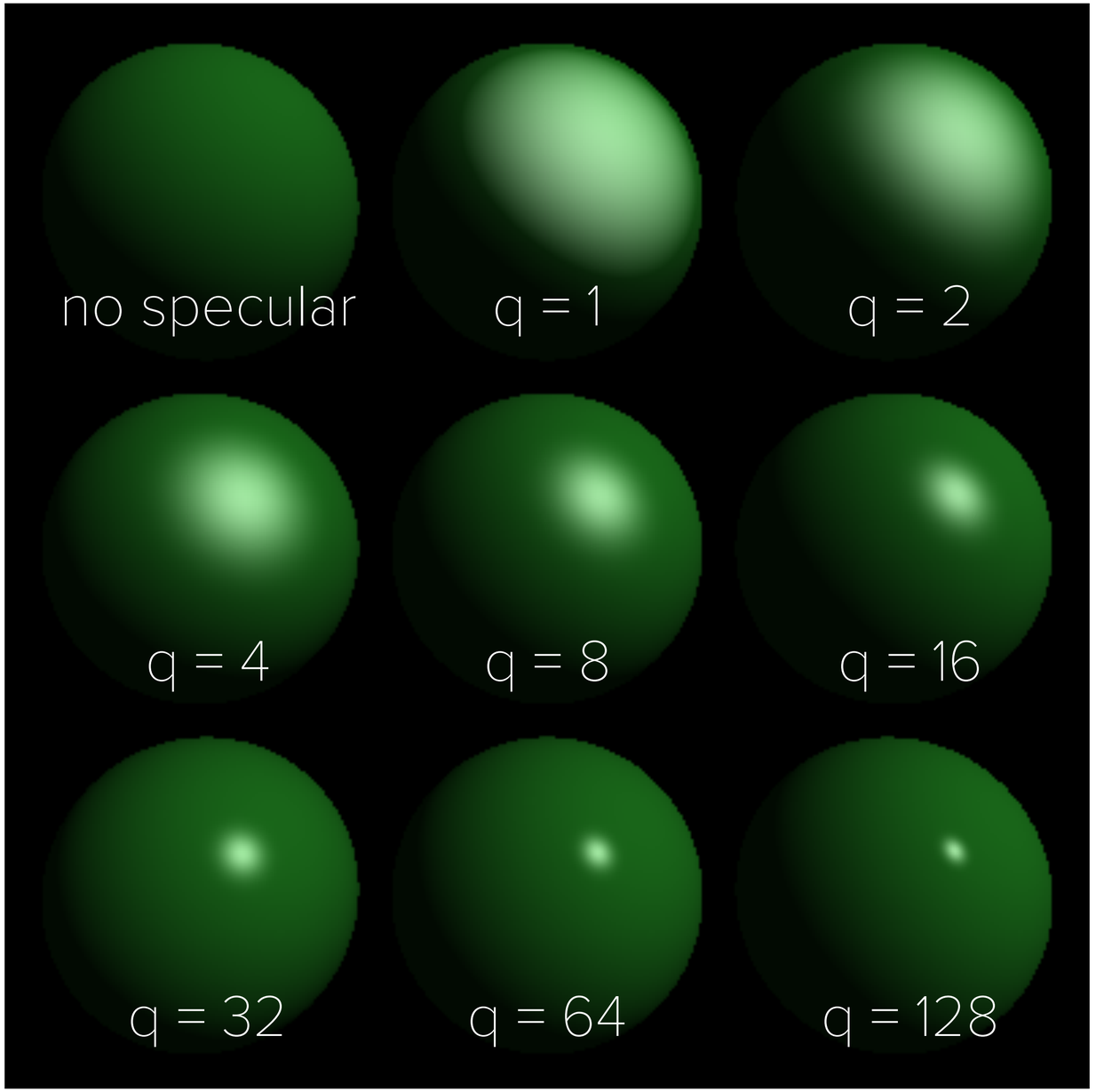
SPECULAR SHADING

$$L_o = k_s(\cos\alpha)^q L_i = k_s(v \cdot r)^q L_i$$

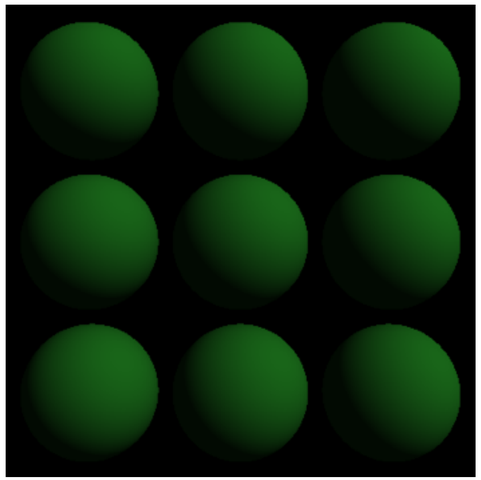
k_s specular reflection coefficient

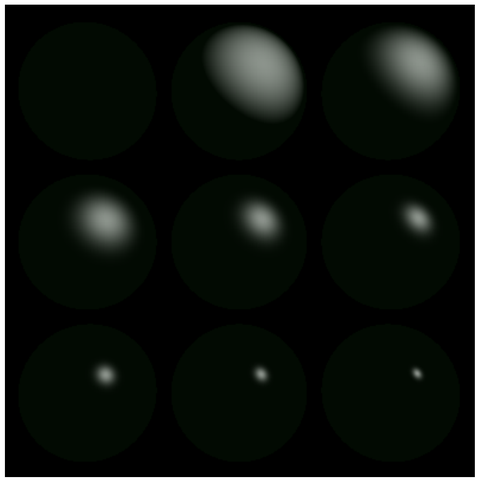
q specular reflection exponent



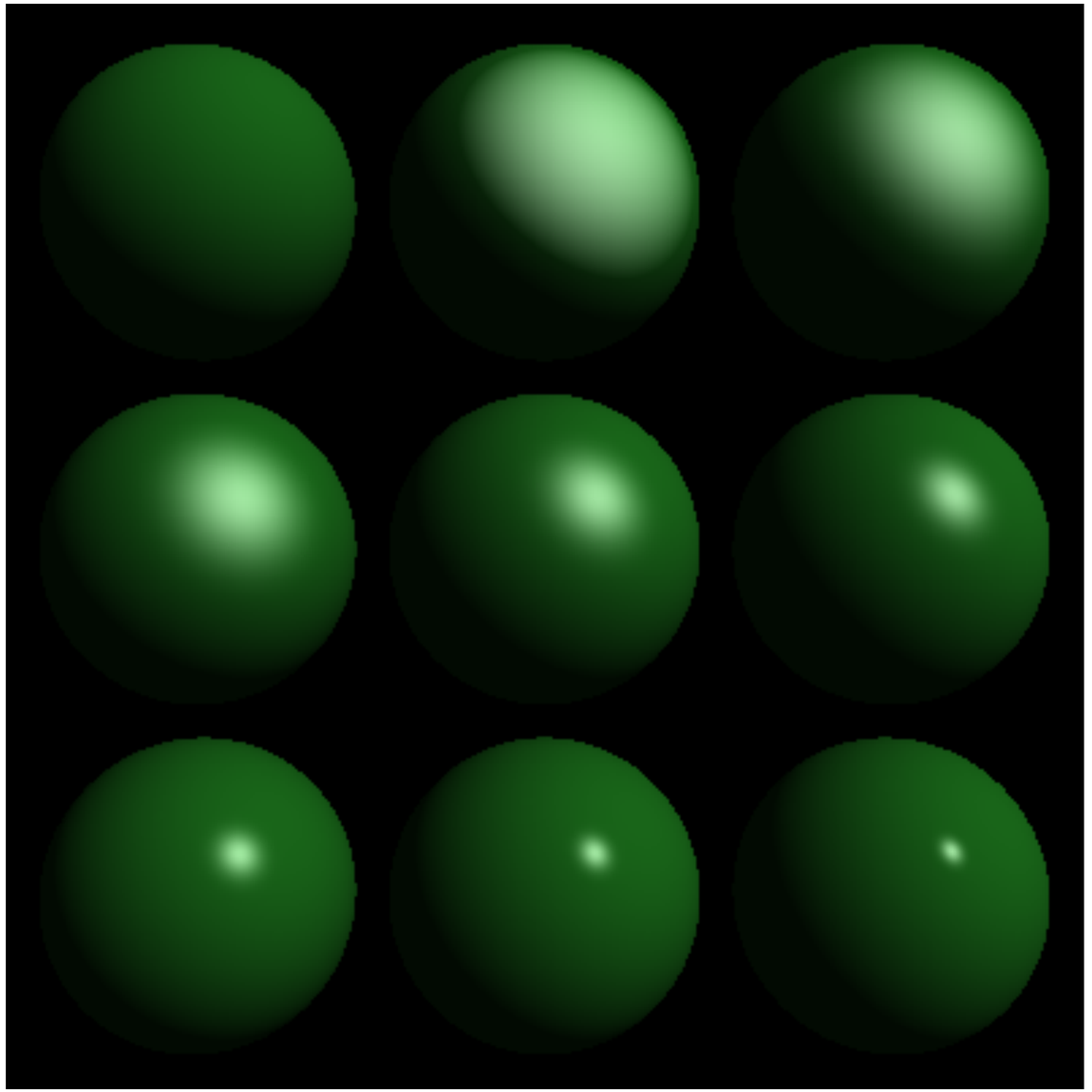

no specular
q = 1
q = 2
q = 4
q = 8
q = 16
q = 32
q = 64
q = 128

PHONG SHADING MODEL





diffuse + specular = Phong shading



COMPLETE PHONG MODEL

$$L_o = k_a + \sum_{i=1}^{n} L_i \left[k_d (n \cdot l) + k_s (v \cdot r)^q \right]$$

Not physically based

Does not conserve energy, may reflect more energy than goes in

Specular portion does not completely conform to BRDF

Ambient illumination is a total hack

Mathematical Toolbox

Transforms for intersections

Transforms for normals

Next week

Recap of specular shading

Mob programming (specular shading)