Graphics 2008/2009, period 1

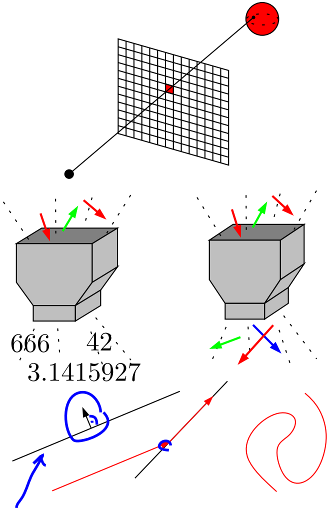
Lecture 3

Surfaces, normals, and reflections

Recapitulation: Ray tracing

Seen so far:

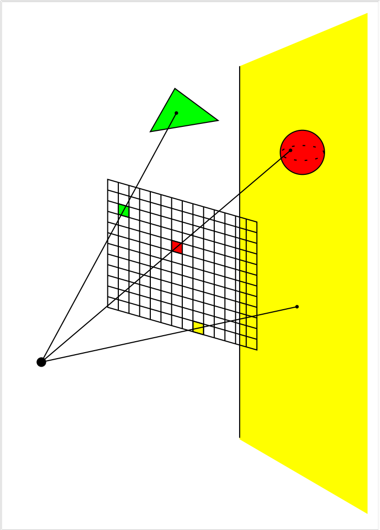
- ray → pixel center
- sphere representation
- ray/sphere intersection
- dot product: angles $\cos\theta$
- cross product: vector orthogonal to input vectors.
- implicit and parametric equations for 2D lines and curves



To-do list

The objects in our ray tracer are spheres (already covered), planes, and triangles, so we need to discuss

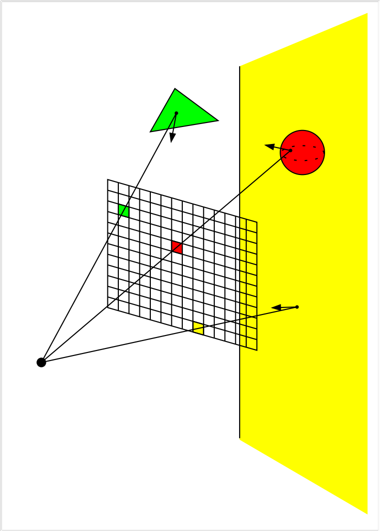
- representations of planes and triangles.
- ray/plane and ray/triangle intersection tests.



To-do list

Coloring objects with a fixed color makes them look flat. To get a sense of depth, we need to do shading (which has nothing to do with *shadows*)

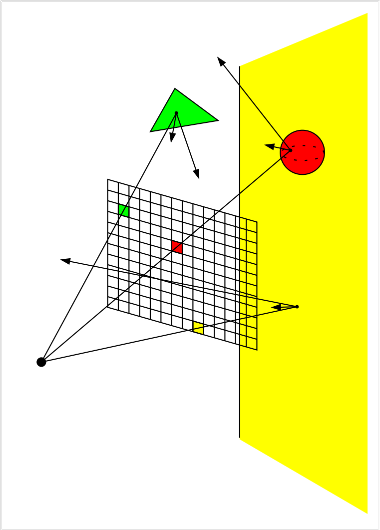
It turns out that we need surface normals to do shading using local illumination.



To-do list

Global illumination deals with light that falls indirectly (i.e., via reflections in other objects) on the objects to be shaded.

How do we determine reflected rays?



From 2D to 3D

Different ways to represent objects in 2D:

- Implicit representation: $f(x, y) = 0$
- Parametric representation: Controlled by one parameter

In addition:

- Vector representation

Objects covered so far:

- (general curves), circles, lines $\rightarrow$ all in 2D

Now: Generalization from 2D to 3D (spheres, planes, ...)

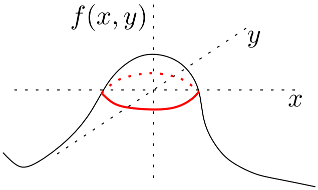
Implicit surfaces: from 2D to 3D

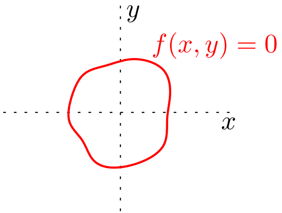
Recall that an implicit curve has the form

$f(x, y) = 0$

The 3D generalization is an implicit surface with a similar form

$f(x, y, z) = 0$





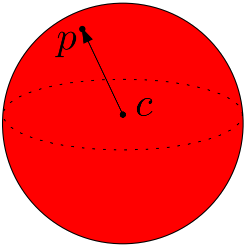
Implicit spheres

We have already seen the sphere equation:

$$(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 - r^2 = 0$$

Just as in the circle case, this can be written in dot product form for any point p on the sphere:

$(p - c) \cdot (p - c) - r^2 = 0$, or
$||p - c||^2 - r^2 = 0$, or
$||p - c|| = r$.



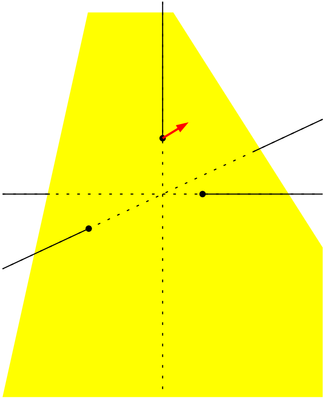
Implicit planes

The implicit equation for a plane in 3D looks a lot like the equation for a line in 2D:

$ax + by + cz - d = 0$

Here, (a, b, c) is a normal vector of the plane.

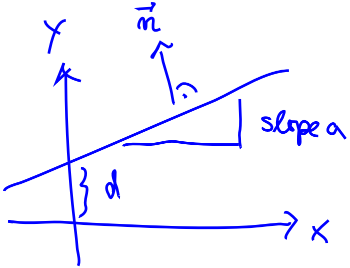
Q: what is the meaning of d?



Implicit planes

Q: what is the meaning of d?
(remember lines in 2d)

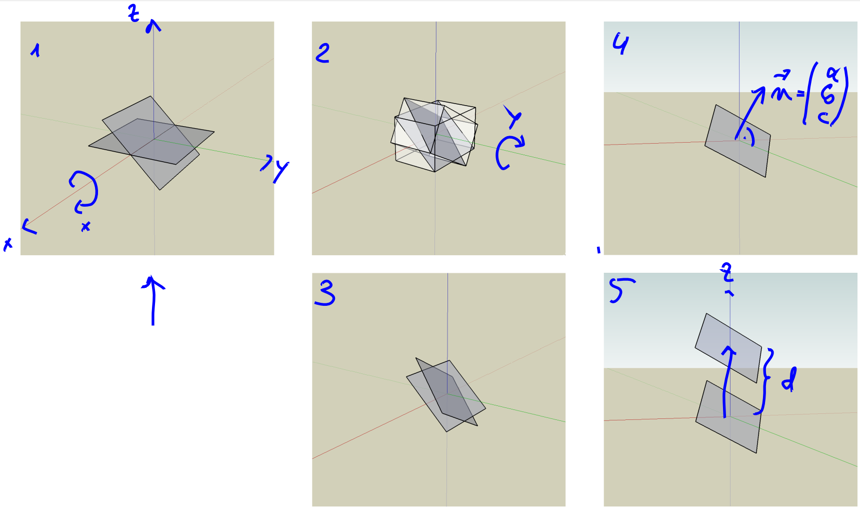
$-ax + y - d = 0$



Vector repr.

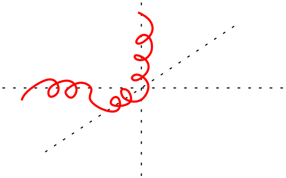
$\vec{n} \cdot (\vec{p} - \vec{p_0}) = 0, \quad \vec{n} = \begin{pmatrix} -a \\ 1 \end{pmatrix}, \quad \vec{p_0} = \begin{pmatrix} 0 \\ d \end{pmatrix}$

Implicit planes



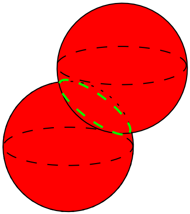
Implicit one-dimensional curves in 3D?

Cooking up an implicit function for a one-dimensional thingy in 3D is in general not possible; such thingies are degenerate surfaces.



E.g., $x^2 + y^2 = 0$ is a cylinder with radius 0: the Z-axis.

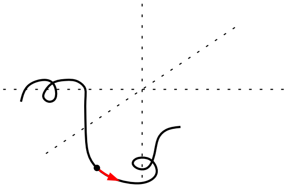
More complex curves can be described as the intersection of two or more implicit surfaces.



Parametric curves and surfaces

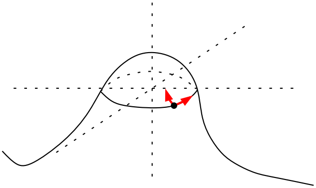
As opposed to implicit curves, it is possible to specify parametric curves in 3D:

$$x = f(t),$$
$$y = g(t),$$
$$z = h(t).$$



Parametric surfaces depend on two parameters:

$$x = f(u, v),$$
$$y = g(u, v),$$
$$z = h(u, v).$$



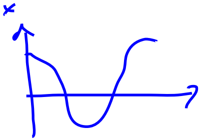
Parametric curves and surfaces

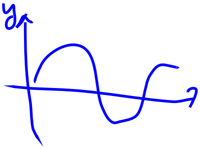
Example:

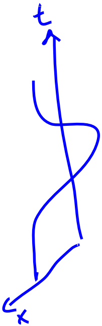
$x = \cos t,$

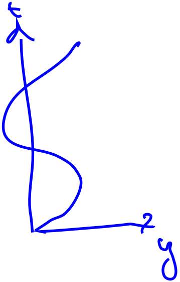
$y = \sin t,$

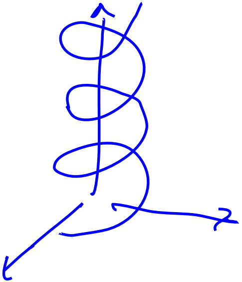
$z = t.$







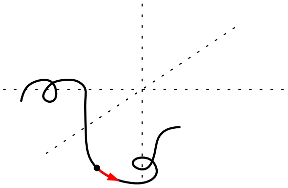




Parametric curves and surfaces

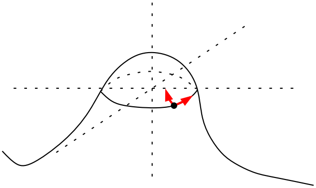
As opposed to implicit curves, it is possible to specify parametric curves in 3D:

$$x = f(t),$$
$$y = g(t),$$
$$z = h(t).$$



Parametric surfaces depend on two parameters:

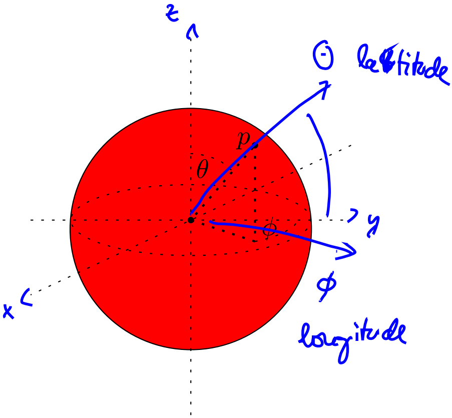
$$x = f(u, v),$$
$$y = g(u, v),$$
$$z = h(u, v).$$



Parametric spheres

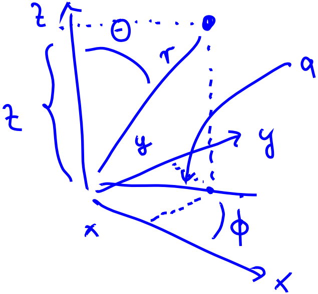
Spheres can also be represented parametrically. For instance, a sphere with radius r centered at the origin has the equation:

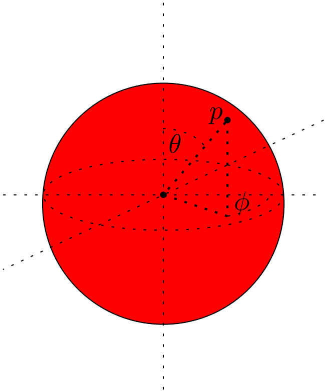
$$x = r\cos\phi\sin\theta,$$
$$y = r\sin\phi\sin\theta,$$
$$z = r\cos\theta$$



Parametric spheres

$$x = r\cos\phi\sin\theta, = r \cdot \frac{x}{a} \cdot \frac{a}{r} = x$$
$$y = r\sin\phi\sin\theta, = r \cdot \frac{y}{a} \cdot \frac{a}{r} = y$$
$$z = r\cos\theta \checkmark$$





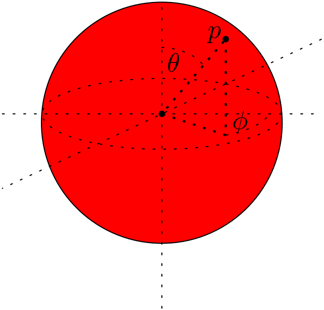
Parametric spheres

$$x = r\cos\phi\sin\theta,$$
$$y = r\sin\phi\sin\theta,$$
$$z = r\cos\theta$$

$x = c_x + r\cos\phi$

$y = c_y + r\sin\phi$

2d



Q: What would the equation for a sphere with radius r centered at $c = (c_x, c_y, c_z)$ be?

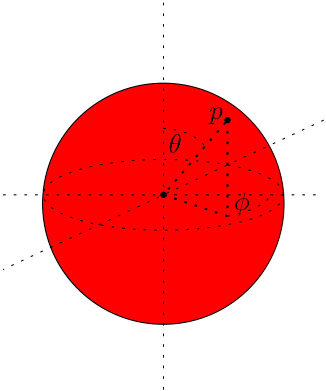
...

Parametric spheres

$$x = r\cos\phi\sin\theta,$$
$$y = r\sin\phi\sin\theta,$$
$$z = r\cos\theta$$

The parametric representation of a sphere looks much more inconvenient than the implicit equation.

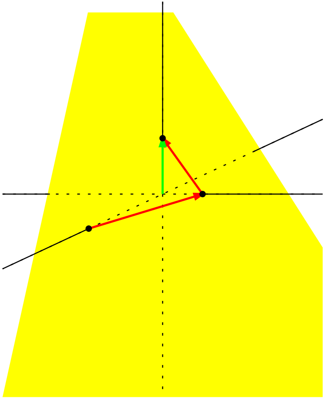
However, when we have to do texture mapping, the parametric representation turns out to be quite convenient.



Parametric planes

Planes can also be described parametrically. Instead of one direction vector (as for lines), we need two:

$$(x, y, z) = (x_p, y_p, z_p) + s(x_v, y_v, z_v) + t(x_w, y_w, z_w).$$

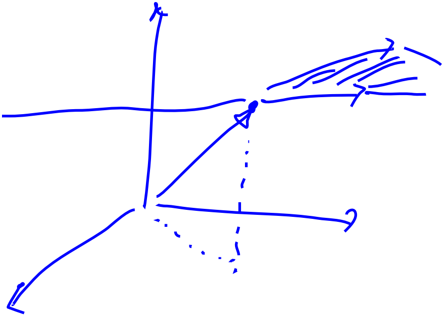


Parametric planes

(cf. lines in 2D and 3D)

Planes can also be described parametrically. Instead of one direction vector (as for lines), we need two:

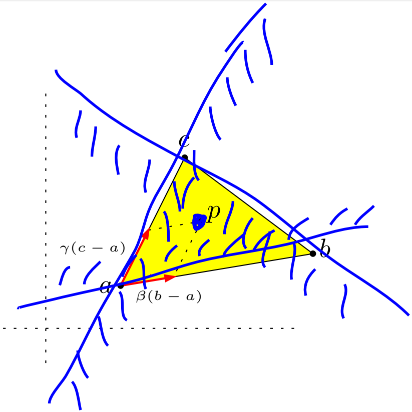
$$(x, y, z) = (x_p, y_p, z_p) + s(x_v, y_v, z_v) + t(x_w, y_w, z_w).$$



Triangles: barycentric coordinates

Triangles can be specified by their three vertices a, b, and c. Points in the triangle lie in the plane induced by these three points, but there are additional constraints.

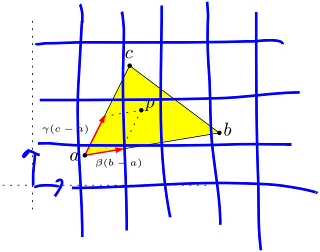
Expressing the points in the plane in a different basis (namely, in barycentric coordinates) turns out to be helpful.



Triangles: barycentric coordinates

Triangles can be specified by their three vertices a, b, and c. Points in the triangle lie in the plane induced by these three points, but there are additional constraints.

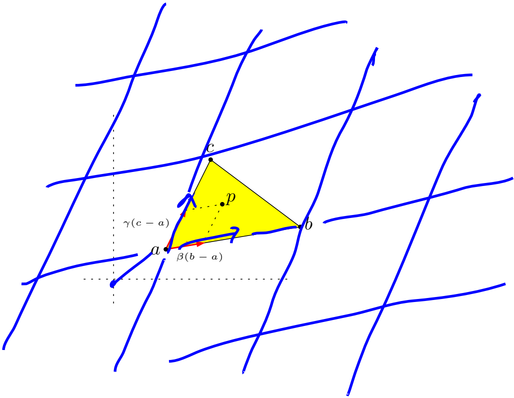
Expressing the points in the plane in a different basis (namely, in barycentric coordinates) turns out to be helpful.



Carthesian Coord. System: $\vec{x} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $\vec{y} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

Triangles: barycentric coordinates

Triangles can be specified by their three vertices a, b, and c. Points in the triangle lie in the plane induced by these three points, but there are additional constraints.



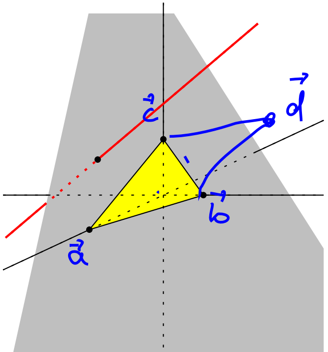
Expressing the points in the plane in a different basis (namely, in barycentric coordinates) turns out to be helpful.

Base vectors: $\vec{c}-\vec{a}$, $\vec{b}-\vec{a}$ $\quad\|\quad$ $\vec{p}=\vec{a}+\beta(\vec{b}-\vec{a})+\gamma(\vec{c}-\vec{a})$

Ray/triangle intersections

We can compute the intersection of the line and the plane induced by the triangle, and then test if the point lies in the triangle by changing to barycentric coordinates.

If we define the plane in barycentric coordinates, we can combine the two steps into one.



$\beta = \gamma = 0 \rightarrow \vec{a}$

$\beta = 1, \gamma = 0 \rightarrow \vec{b}$

$\beta = 0, \gamma = 1 \rightarrow \vec{c}$

$\beta = \gamma = 1 \rightarrow \vec{d}$

$0 < \beta, \gamma < 1$ inside parallelogr.

inside $\triangle$?

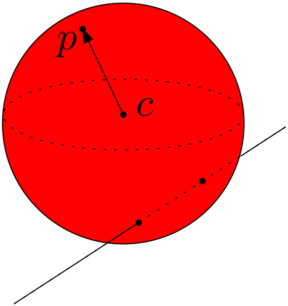
$\gamma, \beta \geq 0$

$\beta + \gamma \leq 1$

Ray/sphere intersections

We have already seen how to do ray/sphere intersection tests:

The intersection points have to satisfy both the sphere equation and the line equation.

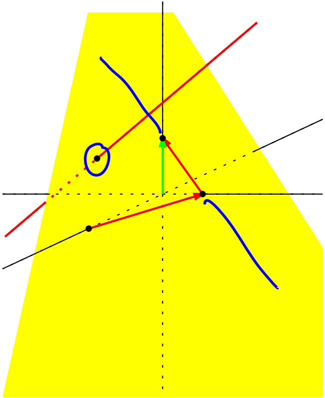


Ray/plane intersections

Ray/plane intersection tests are similar to ray/sphere intersection tests:

The intersection point has to satisfy both the plane equation and the line equation.

solutions: 0, 1, ∞



Recap and outlook

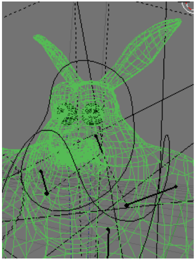
Now we know the mathematics to

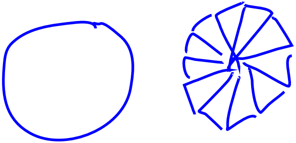
- create (very simple) models
- get them on the screen with ray tracing

Still missing:

- lighting, reflection
- colors, etc.

i.e. some shading





Recap and outlook

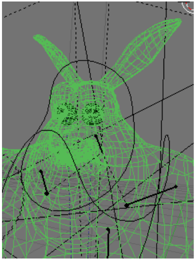
Now we know the mathematics to

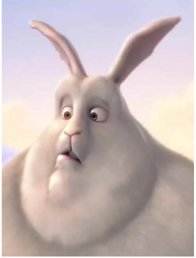
- create (very simple) models
- get them on the screen with ray tracing

Still missing:

- lighting, reflection
- colors, etc.

i.e. some shading





Recap and outlook

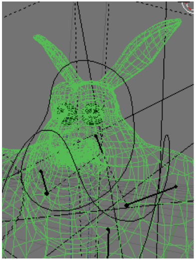
Now we know the mathematics to

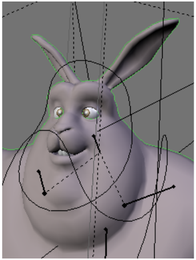
- create (very simple) models
- get them on the screen with ray tracing

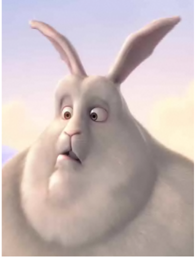
Still missing:

- lighting, reflection
- colors, etc.

i.e. some shading



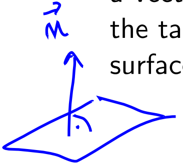


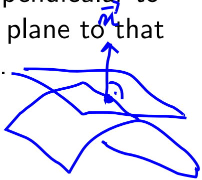


Surface normals

Generally, a surface normal on

- a *flat surface* is a vector which is perpendicular to that surface
- a *non-flat surface* at a point p on that surface is a vector perpendicular to the tangent plane to that surface at p.





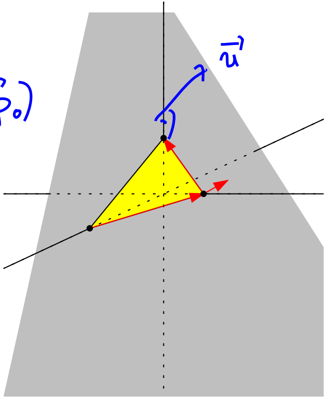
Comments:

- Normals vectors of a point on a surface are given by their gradient
- If a surface does not have a tangent plane at p, it does not have a normal vector at p either
- Normals are needed for lighting calculations

Normals of planes and triangles

A plane normal is trivially determined if the plane is specified with an implicit equation.

$\vec{n} \cdot (\vec{p} - \vec{p}_0)$



Q: how do we compute a normal if the plane is specified with a parametric equation?

Q: does the direction of the normal matter?

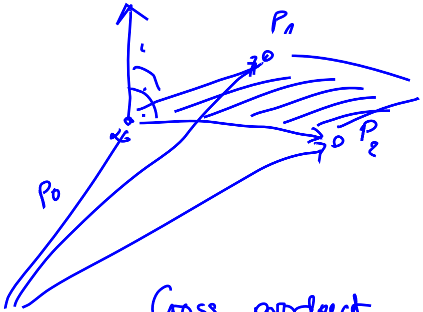
Normals of planes and triangles

A plane normal is trivially determined if the plane is specified with an implicit equation.

Q: how do we compute a normal if the plane is specified with a parametric equation?

Q: does the direction of the normal matter?

$$\vec{p}(s,t) = \vec{p}_0 + s(\underline{\vec{p}_1 - \vec{p}_0}) + t(\underline{\vec{p}_2 - \vec{p}_0})$$



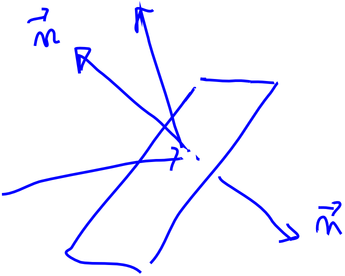
Cross product

Normals of planes and triangles

A plane normal is trivially determined if the plane is specified with an implicit equation.

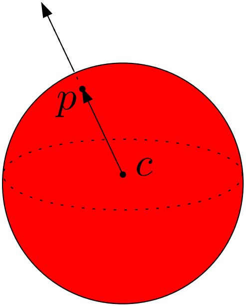
Q: how do we compute a normal if the plane is specified with a parametric equation?

Q: does the direction of the normal matter?



Normals of spheres

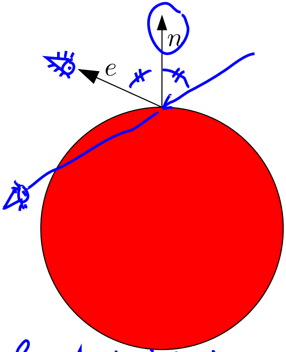
Computing the sphere normal in a point p on a sphere is not tremendously complicated, but mind the direction!



Reflection vectors

Given a point p on an object, with surface normal n, and a ray hitting p coming from direction e, how do we compute the specularly reflected ray r?

Q: what does physics tells us about the relation between e, n, and r?



mirror ⇒ angle of reflection = angle of incident

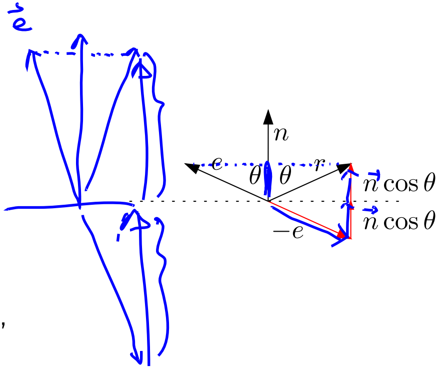
Reflection vectors

The problem is essentially two-dimensional. For simplicity, let's assume that e and n are normalized.

In the image, we see that r is the sum of the red vectors, i.e.,

$$\vec{r} = -\vec{e} + 2(e \cdot n)\vec{n}$$

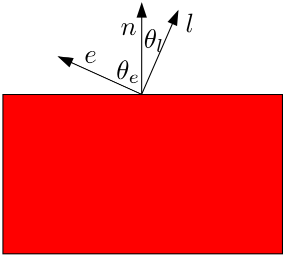
$$p = \frac{\vec{e} \cdot \vec{n}}{|\vec{n}|} = \vec{e} \cdot \vec{n} = \cos\theta$$



Shading parameters: diffuse reflection

How does the diffuse reflection of light depend on

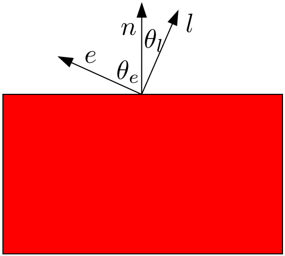
- the viewing angle?
- the angle of the incident light?
- properties of the material and of the incident light?
- the distances from the light source and the viewer to the object?



Light and viewing direction

With diffuse reflection, the viewing direction doesn't matter for the intensity of the reflected light.

$\vec{e}$, Θ_e



Light and viewing direction

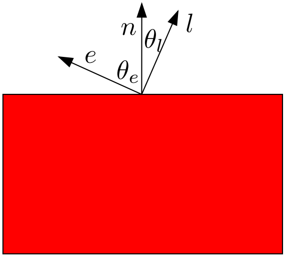
The direction of the incident light does matter: the intensity of the reflected light is proportional to the cosine of θ_l.

Θ_l

Lambert's cosine law

$c \sim \cos \Theta_l$

$\sim \vec{n} \cdot \vec{l}$



Shading parameters: light and material properties

In the (very simplistic) RGB model, colors are given by a triple (r, g, b).

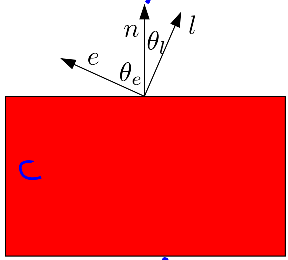
If the color of the incident light is (r_l, g_l, b_l) and the color of the object is (r_o, g_o, b_o), then the color of the reflected light (assuming flat shading) is

$$(r_o r_l,\ g_o g_l,\ b_o b_l)$$

(assuming that color values lie in $[0..1]$)

factor c_r

RGB intensity term c_l



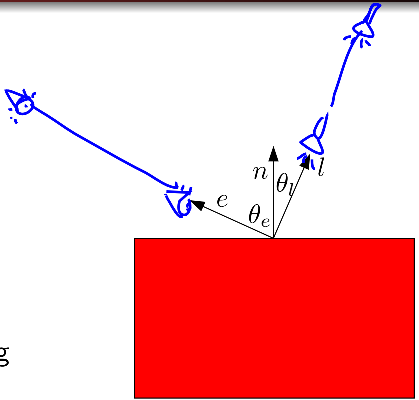
$c = c_l \cdot c_r \cdot \vec{n} \cdot \vec{l}$

$\max\{0, \vec{n} \cdot \vec{l}\}$

Light and viewing distance

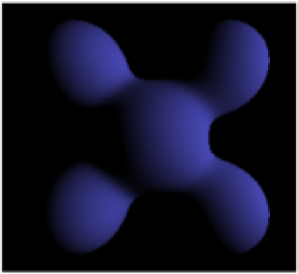
Viewing distance doesn't matter for percieved intensity of the lit objects.

In practice, the distance from the light source to the object does matter, but in ray tracing this is often conveniently ignored.

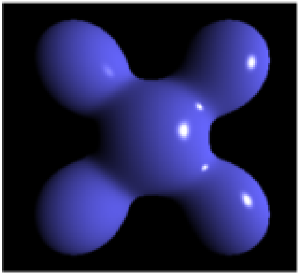


Glossy reflection

Glossy reflection / Phong shading 92



Diffuse

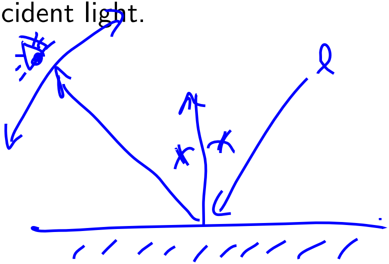


Phong Reflection

(source: http://en.wikipedia.org/wiki/Phong_shading)

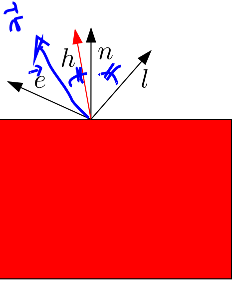
Glossy reflection

For glossy reflection, the viewing direction matters. Reflection is maximized when the angle of the reflected light equals the angle of the incident light.



$\vec{h} \cdot \vec{n}$

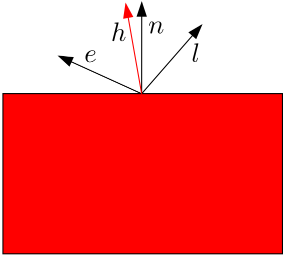
$\vec{r} \cdot \vec{e}$
$\sim \cos\theta$



Glossy reflection

In the Phong model, we use the vector h that lies "halfway" between e and l:

$$c = c_l c_p (\vec{h} \cdot \vec{n})^p$$; c_p = highlight color



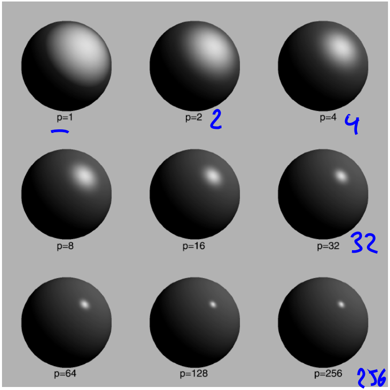
Diffuse shading:

$$c = c_r \cdot c_l \cdot (\vec{l} \cdot \vec{n})$$

c_l = RGB intensity

c_r = reflectance factor

Glossy reflection



(source: textbook, p. shirley, fig. 9.6, page 195)

Programming assignment P1

- Vectors
- Windowing transformations
- Color clamping   Ch. 3
- Spheres
- Windowing transformations revisited
- Local illumination model (diffuse reflection)
- Shadow feelers   Ch. 10.5
- Gamma correction   Ch. 3
- Planes
- Glossy (phong) reflection
- Global illumination