POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

D E
I B

2024

Dipartimento di Elettronica, Informazione e Bioingegneria

Computer Graphics

Milano, 2024

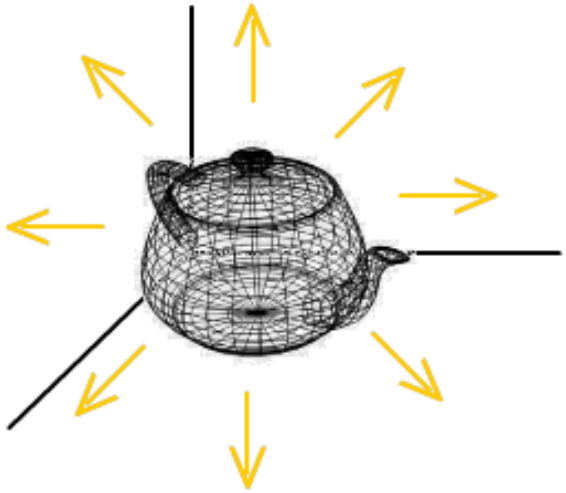
Computer Graphics

- Emission and Indirect Lighting Approximation

Material emission

The *emission* term of a material accounts for the small amount of light emitted directly by an object.

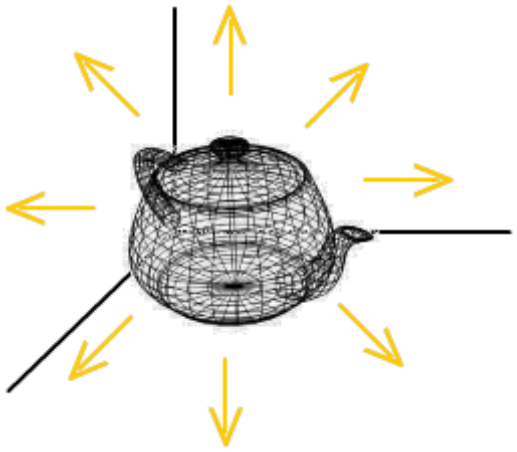
$$\boldsymbol{m}_E = (m_{ER}, m_{EG}, m_{EB})$$



Material emission

It corresponds to the emissive part of the rendering equation.

$$\boldsymbol{m}_E = (m_{ER}, m_{EG}, m_{EB})$$



$$L(x, \omega_r) = L_e(x, \omega_r) + \sum_l L_e\left(l, \overrightarrow{lx}\right) f_r\left(x, \overrightarrow{lx}, \omega_r\right)$$

Material emission

The material emission term is summed to the other parts of the rendering equation. It does not depend on the environment, but only on the considered object.

For example, if we consider a direct light source, Phong specular, Lambert diffuse reflection, and emission, we have:

$$\boldsymbol{r}_x = 2\boldsymbol{n}_x \cdot (\boldsymbol{d} \cdot \boldsymbol{n}_x) - \boldsymbol{d}$$

$$L(x, \omega_r) = clamp(\boldsymbol{l}_D * (\boldsymbol{m}_D \cdot clamp(\boldsymbol{d} \cdot \boldsymbol{n}_x) + \boldsymbol{m}_S \cdot clamp(\omega_r \cdot \boldsymbol{r}_x)^\gamma) + \boldsymbol{m}_E)$$

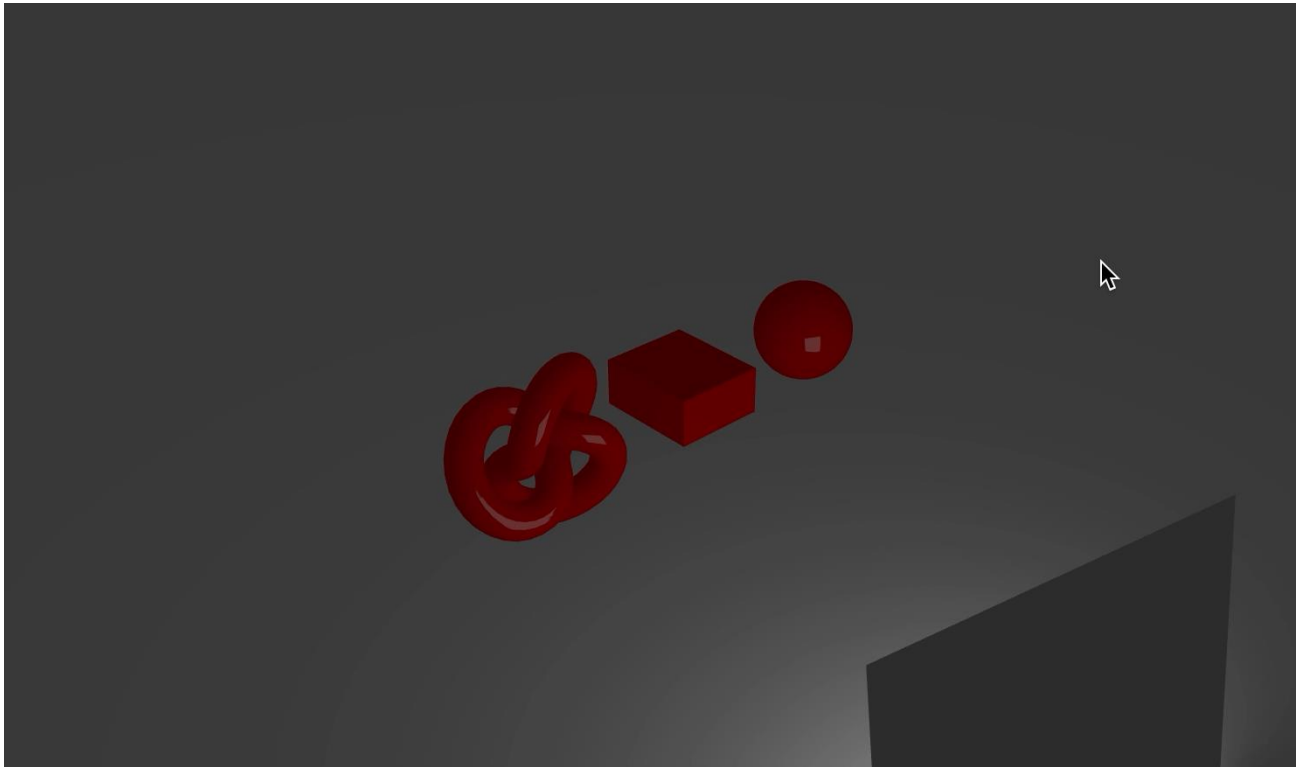
Ambient lighting

Considering only direct light sources (such as directional, point or spotlight) can produce very dark images.

Realistic rendering techniques, tries to consider also *indirect lighting*: illumination caused by lights that bounces from other objects.

Ambient lighting

Ambient lighting is the simplest approximation for indirect illumination: it is a factor which is constant for the entire scene, and that it accounts for the light reflected by all the objects in all the directions.

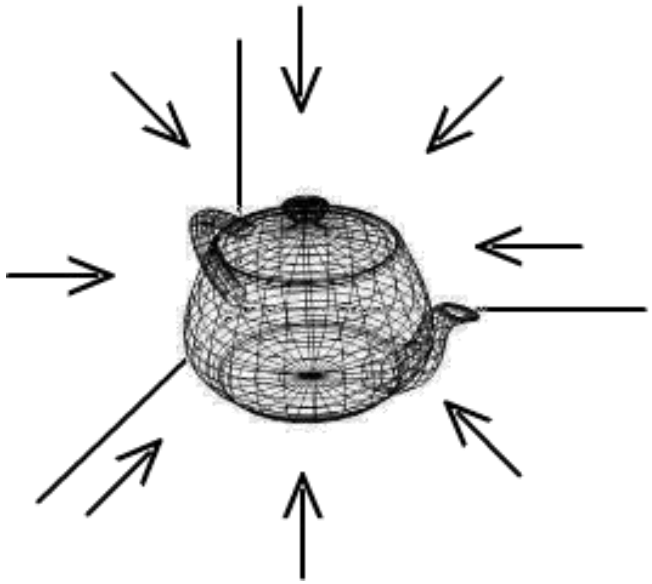


The *Ambient light* term is first quickly reduced to zero, and then it is gradually increased.

Ambient lighting

The ambient light emission is specified by a constant RGB color value $\boldsymbol{l}_A$.

$$\boldsymbol{l}_A = (l_{AR},\ l_{AG},\ l_{AB})$$



Ambient lighting

The BRDF of the object, is then extended by adding another component $f_A(x, \omega_r)$ that specifically considers ambient lighting.

Such component does not depend on the light direction.

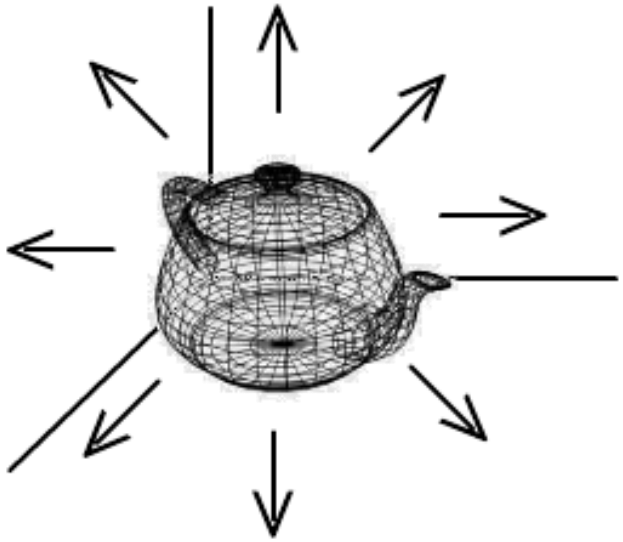
$$L(x, \omega_r) = \sum_l L\left(l, \overrightarrow{lx}\right) f_r\left(x, \overrightarrow{lx}, \omega_r\right) + \boldsymbol{l}_A f_A(x, \omega_r)$$

Ambient lighting

In most of the cases the BRDF for the ambient term $f_A(x, \omega_r)$ is a constant known as the *ambient light reflection color* m_A.

$$\boldsymbol{m}_A = (m_{AR}, m_{AG}, m_{AB})$$

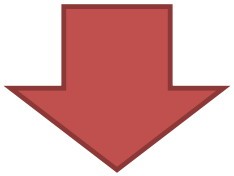
Generally, m_A corresponds to the main color of the object (i.e. $m_A = m_D$), but it can be tuned to obtain special lighting for particular objects.



Ambient lighting

In case of a single direct light, plus the ambient term (which is assumed to be constant), the rendering equations reduces to:

$$L(x,\omega_r) = \sum_l L\left(l,\overrightarrow{lx}\right) f_r\left(x,\overrightarrow{lx},\omega_r\right) + \boldsymbol{l}_A f_A(x,\omega_r)$$



$$L(x,\omega_r) = \boldsymbol{l} * f_r(x,\boldsymbol{d},\omega_r) + \boldsymbol{l}_A * \boldsymbol{m}_A$$

Hemispheric lighting

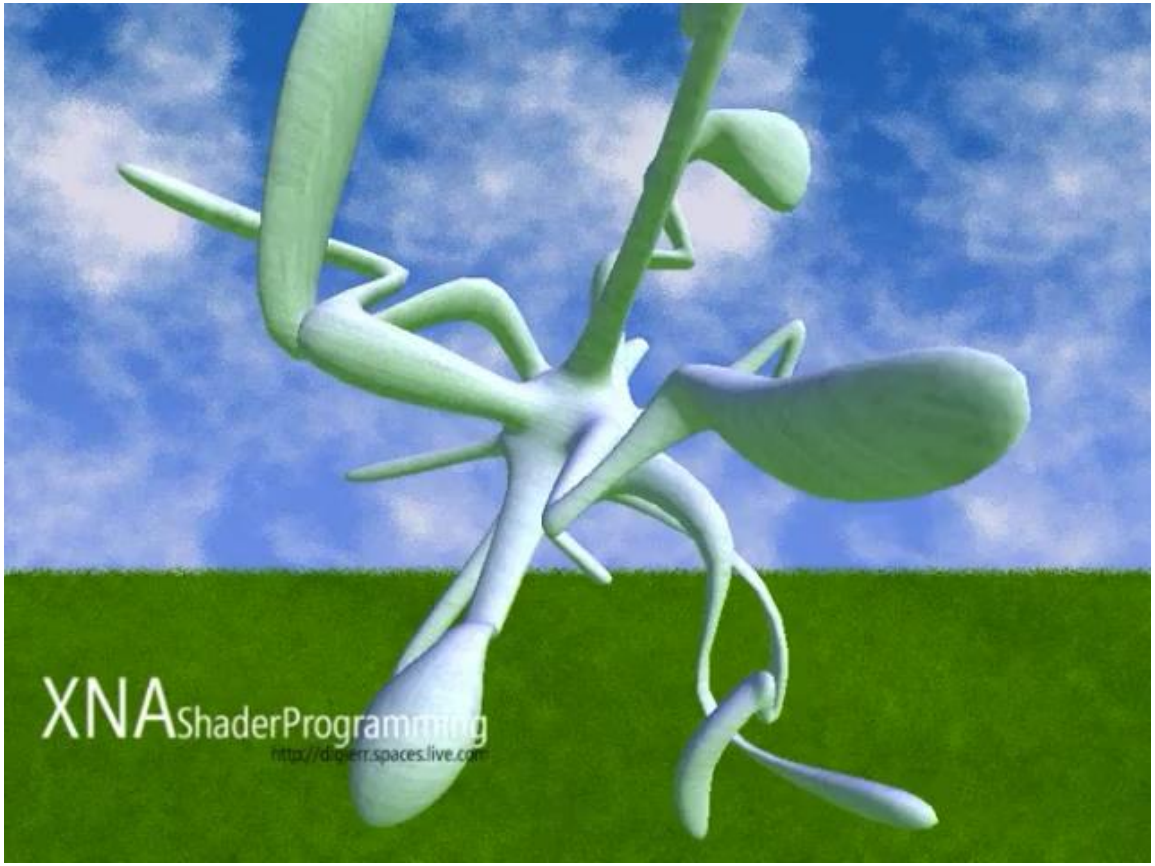
A slight extension of ambient lighting is the *hemispheric lighting*.

In this case, there are two ambient light colors (the “upper” or “sky” color, and the “lower” or “ground” color) and a direction vector.

This model simulates the fact that both the color of the sky and the one of the ground have a big impact on the indirect light component for the considered object.

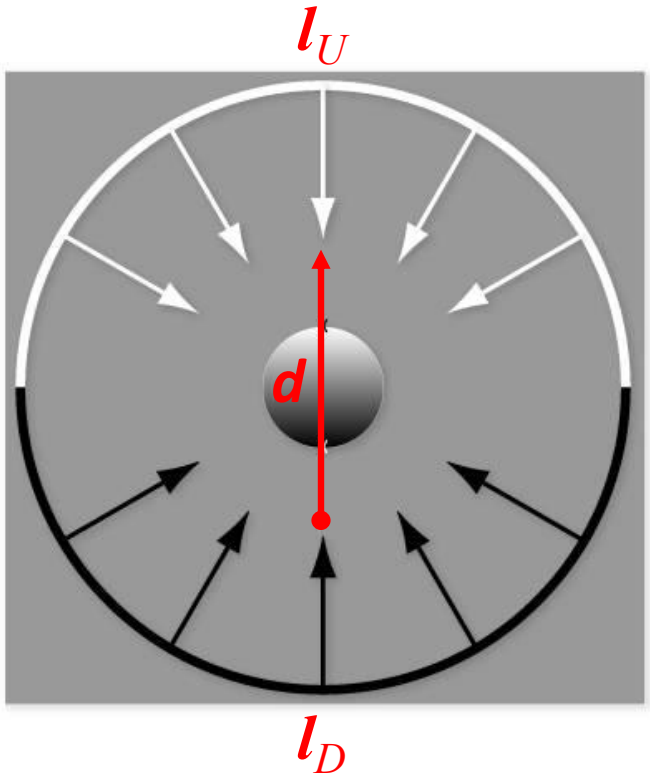
Hemispheric lighting

The technique creates an ambient light color factor by “blending” the two colors, with respect to the orientation of the object.



Hemispheric lighting

The two colors, $\boldsymbol{l}_U$ and $\boldsymbol{l}_D$, represent the values of the ambient light at the two extremes, and the direction vector ***d*** orients the blending of the two colors.



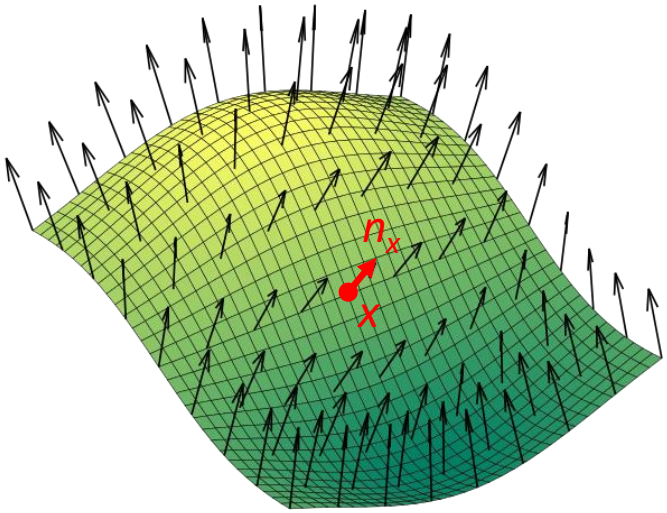
Hemispheric lighting

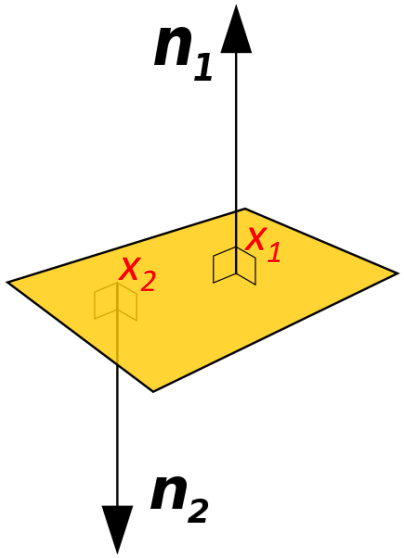
The rendering equation is then modified to have the ambient light term $\boldsymbol{l}_A(x)$ that depends on the orientation of the surface of the object in the considered point *x*.

$$L(x,\omega_r) = \sum_l L\left(l,\overrightarrow{lx}\right) f_r\left(x,\overrightarrow{lx},\omega_r\right) + \boldsymbol{l}_A(x) f_A(x,\omega_r)$$

Hemispheric lighting

The orientation of the object is characterized by n_x the normal vector to the surface in point x.

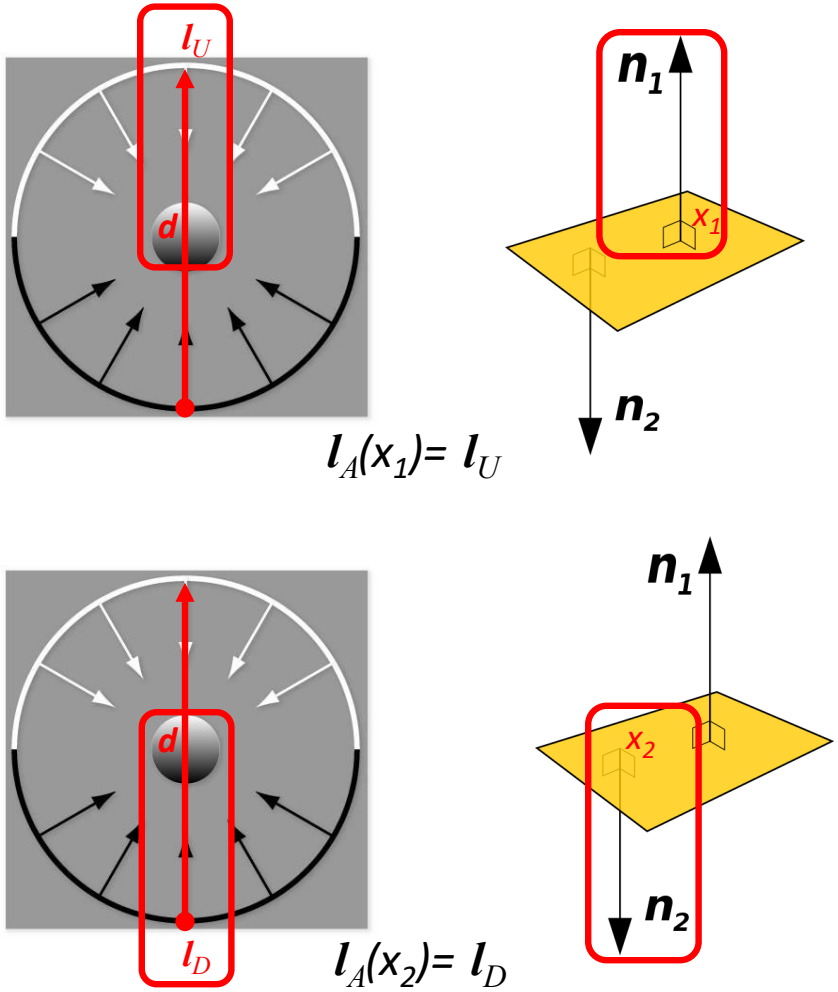




Hemispheric lighting

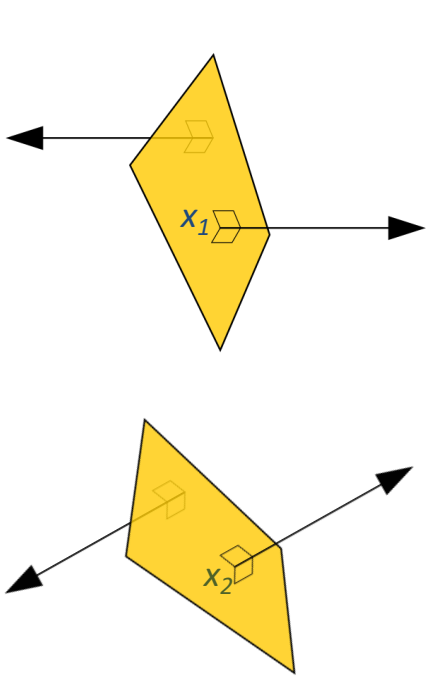
If the normal vector is aligned and in the same direction as ***d***, ambient color $\boldsymbol{l}_U$ is used.

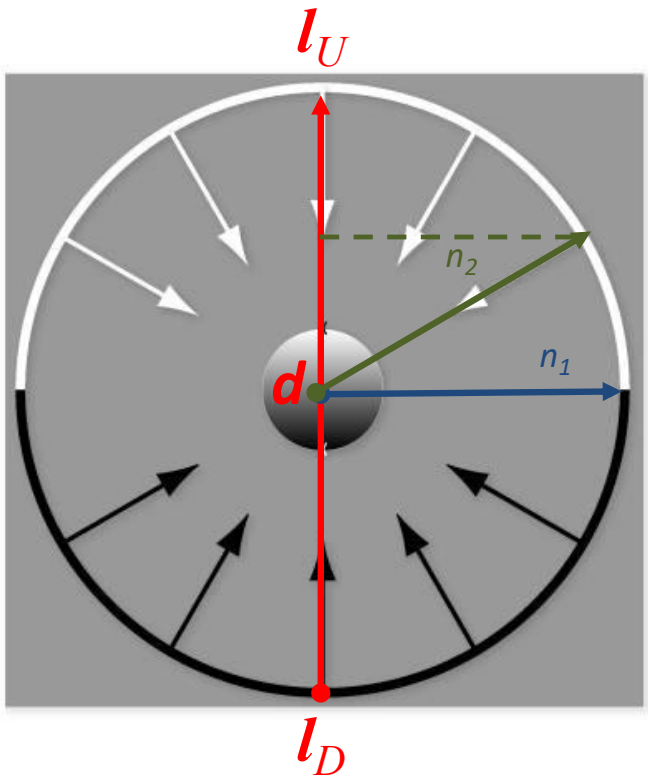
If the normal vector is instead aligned, but in the opposite direction of ***d***, ambient color $\boldsymbol{l}_D$ is used.



Hemispheric lighting

For the normal vectors oriented in other directions, the two colors are blended proportionally to the cosine of their angle with vector ***d***.





$l_A(x_2) = \frac{3}{4}\, l_U + \frac{1}{4}\, l_D$

$l_A(x_1) = \frac{1}{2}\, l_U + \frac{1}{2}\, l_D$

Hemispheric lighting

In particular, $\boldsymbol{l}_A(x)$ is defined as follows:

$$\boldsymbol{l}_A(x) = \frac{n_x \cdot d + 1}{2}\boldsymbol{l}_U + \frac{1 - n_x \cdot d}{2}\boldsymbol{l}_D$$

In case of a single direct light plus the hemispheric ambient term, the rendering equations reduces to:

$$L(x, \omega_r) = \boldsymbol{l} * f_r(x, \boldsymbol{d}, \omega_r) + \left(\frac{\boldsymbol{n}_x \cdot \boldsymbol{d} + 1}{2}\boldsymbol{l}_U + \frac{1 - \boldsymbol{n}_x \cdot \boldsymbol{d}}{2}\boldsymbol{l}_D\right) * \boldsymbol{m}_A$$

The road to Image Based Lighting

More accurate reproduction of light sources and more advanced approximations to the rendering equations are the key to the photo-realistic effects we are now used to see in high-end 3D applications.

Although a complete description of such techniques is outside the scope of this course, we can briefly highlight some ideas that drives some of them.

Image based lighting

The hemispheric lighting we have just introduced, computes the light received by one object as function of the direction in which the points on its surface are oriented, as specified by the direction of the corresponding normal vector.

In this case, however, the function is very simple: it just interpolates two colors according to relative orientation with respect to a given direction.

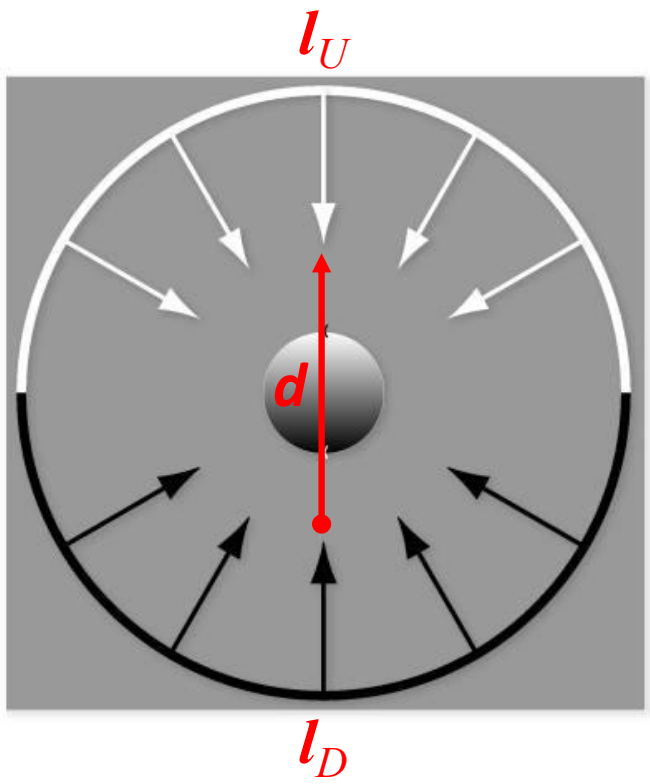


Image based lighting

The next idea is then to have generic functions $\boldsymbol{l}_A(x_i)$ that return the color received from the outside environment by a point x_i on a surface oriented in the direction described by its normal vector $\boldsymbol{n}_i$.

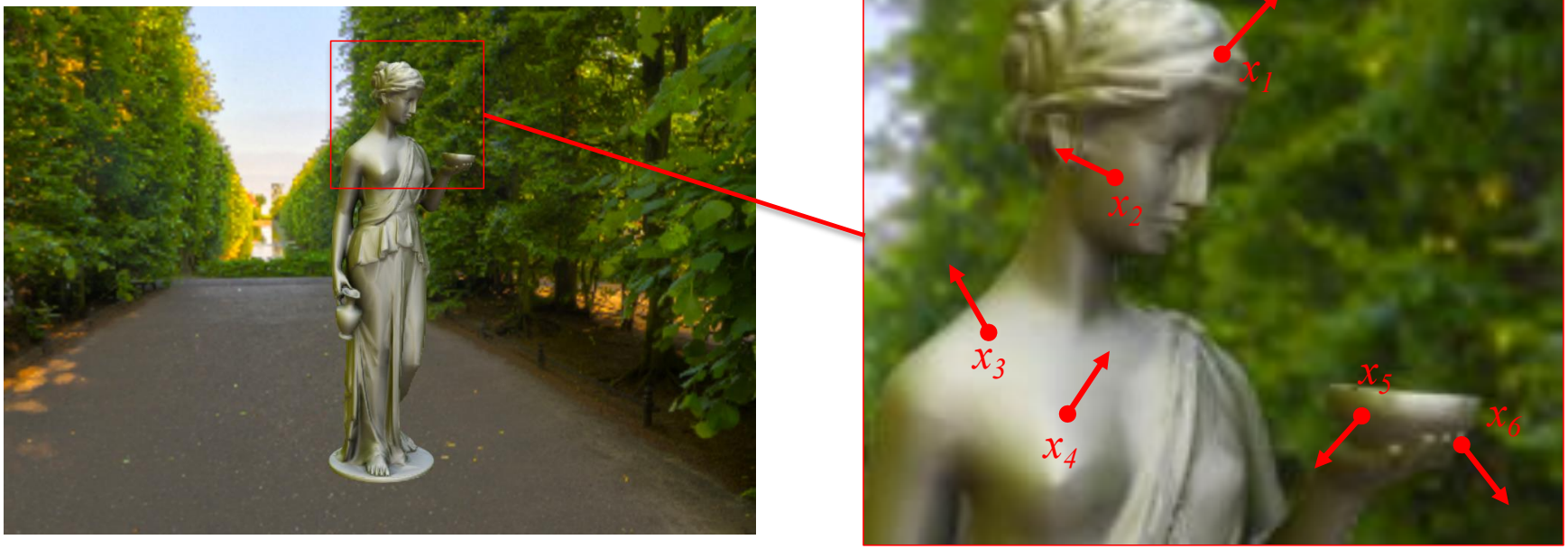


Image based lighting

Each point is thus illuminated according to $\boldsymbol{l}_A(x_i)$.

Image based lighting

These functions $l_A(x_i)$ can be computed either from specially taken pictures or from high quality off-line rendering of the environment, and encoded in a way that can be used in real time.

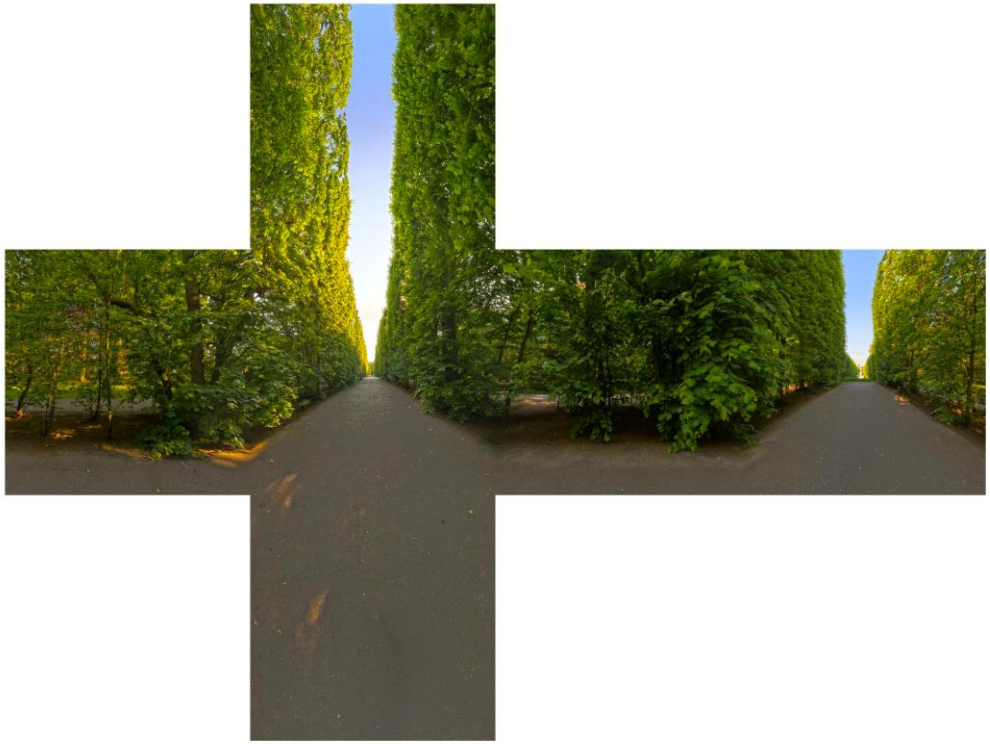


Image based lighting

Starting from an image (this motivates the name Image based lighting), and applying a step called *filtering*, the actual light received from each point in any direction can be determined.

Standard cubemap

Irradiance cubemap

Image based lighting

The approximation of the rendering equation remains the same as the one seen for the hemispheric lighting: only the light function changes to return the values computed in the filtering step.

In this case, the approximation includes both the ambient and diffuse components of the light.

$$L(x,\omega_r) = \sum_l L\left(l,\overrightarrow{lx}\right) f_r\left(x,\overrightarrow{lx},\omega_r\right) + \boldsymbol{l}_A(x) f_A(x,\omega_r)$$

Image based lighting

Encoding function $\boldsymbol{l}_A(x)$ in an efficient way is not a simple task. The most common approaches are:

- Interpolating values stored in a table (*Cubic Mapping*)
- Spectral expansion (*Spherical Harmonics*)
- Other approximations

We will present Cubic Mapping in a following lesson. Next we briefly introduce the Spherical Harmonics, and a custom technique

Image based lighting

With the *Spherical Harmonics* expansion, the color received from a given direction $\boldsymbol{n}_x$ can be expressed as a sum of contributions that multiply a set of basic functions.

Conventionally, these basic function are indexed by two numbers *l* and *m*, with $-l \leq m \leq l$: and $l \geq 0$.

$$\boldsymbol{l}_A(x) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \boldsymbol{c}_{l,m} \cdot y_l^m (\boldsymbol{n_x})$$

Image based lighting

If we call respectively $(\boldsymbol{n}_x).x$, $(\boldsymbol{n}_x).y$ and $(\boldsymbol{n}_x).z$ the components of the (unitary) normal vector direction, and we limit ourselves to $l \leq 1$, we have a particularly simple expression for the expansion:

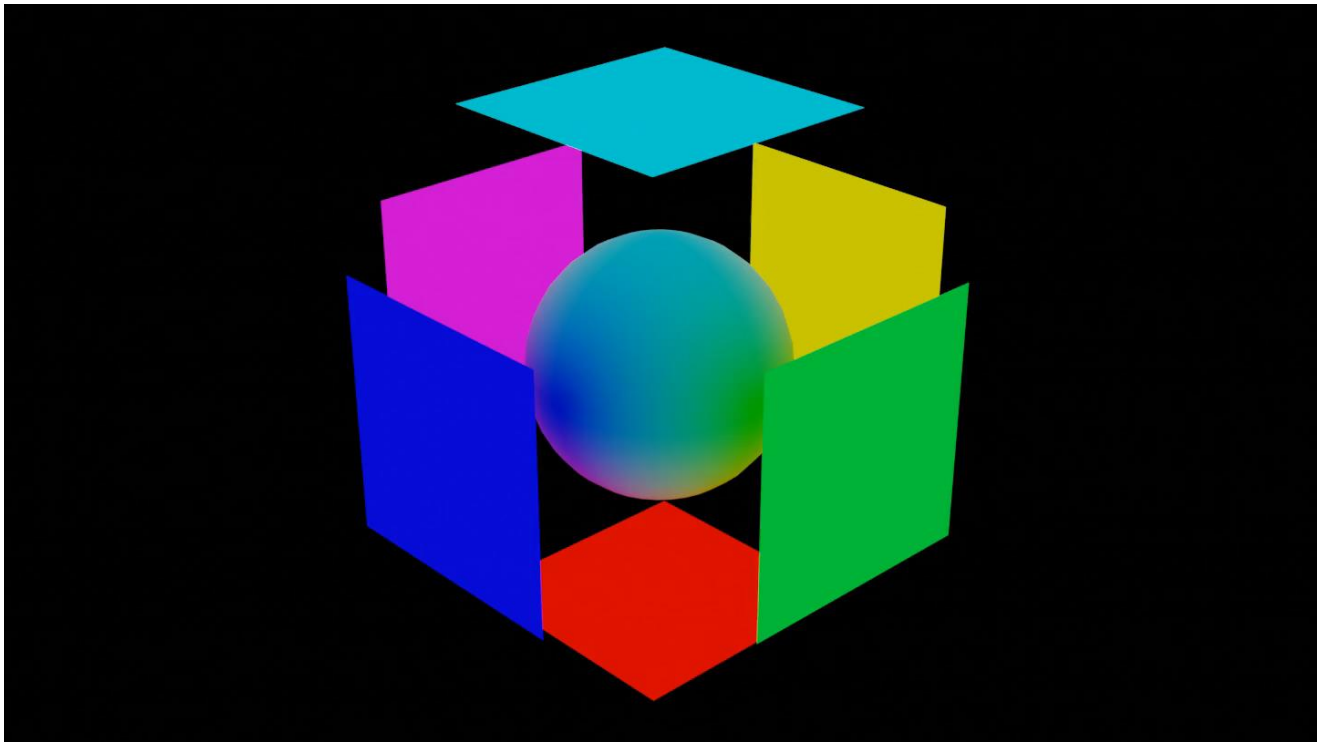
$$\boldsymbol{l}_A(x) = \boldsymbol{c}_{0,0} + (n_x).x \cdot \boldsymbol{c}_{1,1} + (n_x).y \cdot \boldsymbol{c}_{1,-1} + (n_x).z \cdot \boldsymbol{c}_{1,0}$$

With $l \leq 2$, the expression becomes just slightly more complex, but it can capture a greater set of illumination conditions with only 9 coefficients:

$$\begin{aligned}\boldsymbol{l}_A(x) = \boldsymbol{c}_{0,0} &+ (n_x).x \cdot \boldsymbol{c}_{1,1} + (n_x).y \cdot \boldsymbol{c}_{1,-1} + (n_x).z \cdot \boldsymbol{c}_{1,0} + \\ &+\big((n_x).x \cdot (n_x).y\big) \cdot \boldsymbol{c}_{2,-2} + \big((n_x).y \cdot (n_x).z\big) \cdot \boldsymbol{c}_{2,-1} + \big((n_x).z \cdot (n_x).x\big) \cdot \boldsymbol{c}_{2,1} + \\ &+\big((n_x).x^2 - (n_x).y^2\big) \cdot \boldsymbol{c}_{2,2} + (3 \cdot (n_x).z^2 - 1) \cdot \boldsymbol{c}_{2,0}\end{aligned}$$

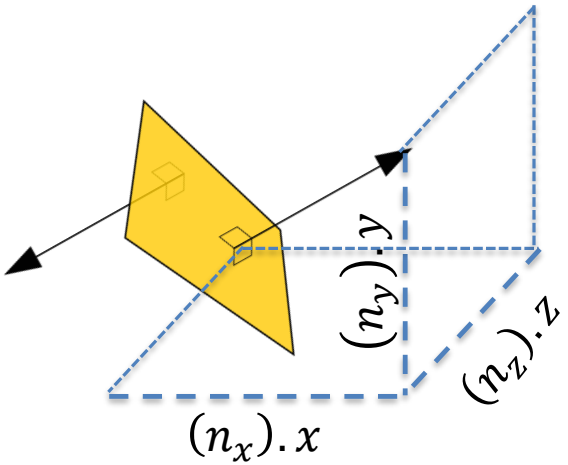
The “six direction ambient” lighting

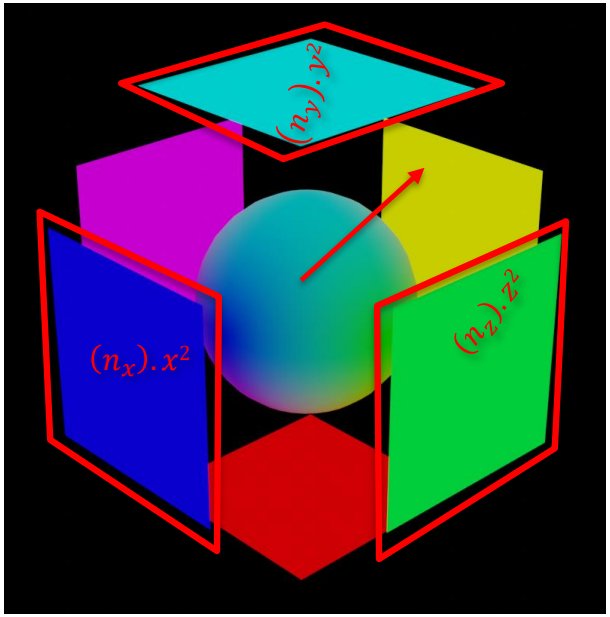
In the assignments, I have always used a custom made ambient light term, which depends on six colors, one per axis direction.



The “six direction ambient” lighting

The shader always looks for the three of the six colors that are closer to the normal vector directions, and interpolates them according to the square of the its *x*, *y* and *z* components.





The “six direction ambient” lighting

The closest colors are selected according to the sign of the components of the normal vector. Since the normal vector is of unitary length, interpolation coefficient will surely sum up to one.

```
// A special type of non-uniform ambient color, invented for this course
const vec3 cxp = vec3(1.0,0.5,0.5);
const vec3 cxn = vec3(0.9,0.6,0.4);
const vec3 cyp = vec3(0.3,1.0,1.0);
const vec3 cyn = vec3(0.5,0.5,0.5);
const vec3 czp = vec3(0.8,0.2,0.4);
const vec3 czn = vec3(0.3,0.6,0.7);

vec3 Ambient = ((Norm.x > 0 ? cxp : cxn) * (Norm.x * Norm.x) +
                (Norm.y > 0 ? cyp : cyn) * (Norm.y * Norm.y) +
                (Norm.z > 0 ? czp : czn) * (Norm.z * Norm.z));
```

The six colors, for the positive and negative axis directions

`Norm` is the normal vector direction

Marco Gribaudo

Associate Professor

CONTACTS

Tel. +39 02 2399 3568

marco.gribaudo@polimi.it

https://www.deib.polimi.it/eng/home-page

(Remember to use the phone, since mails might require a lot of time to be answered. Microsoft Teams messages might also be faster than regular mails)