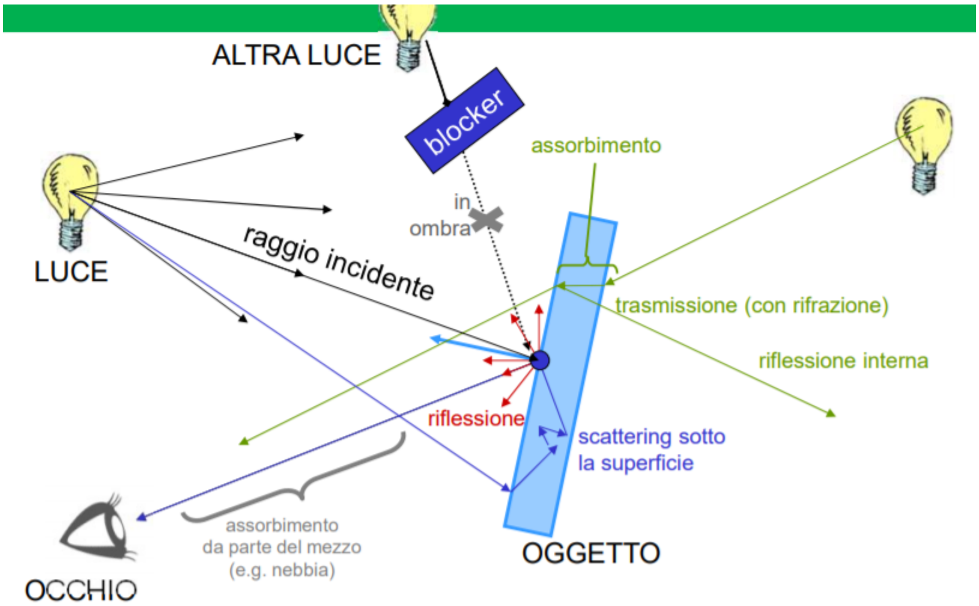
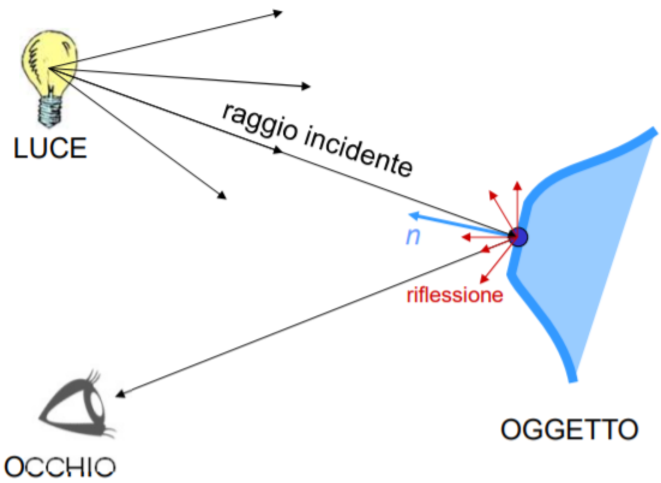
Lighting

Introduzione

Una fase molto importante nel determinare il colore di un frammento durante il fragment process è quello di calcolare il **lighting**, cioè il colore che quell'oggetto (o meglio quel determinato frammento di quell'oggetto) dovrà avere in funzione dell'illuminazione della scena. L'illuminazione influenza moltissimo il colore degli oggetti in scena e da' informazioni sul colore dell'oggetto, sul materiale, sulla sua forma e molto altro.
Il problema è assai complesso, e soprattutto se tentassimo di simulare tutti gli effetti di luce che avvengono nella realtà, di cui alcuni di essi sono mostrati nel seguente esempio:



Un sistema di illuminazione che copre questi e molti altri fenomeni è detto **lighting globale**. Noi ci limiteremo invece al **lighting locale**, che si occupa esclusivamente di simulare la luce riflessa su un oggetto opaco verso il nostro occhio:

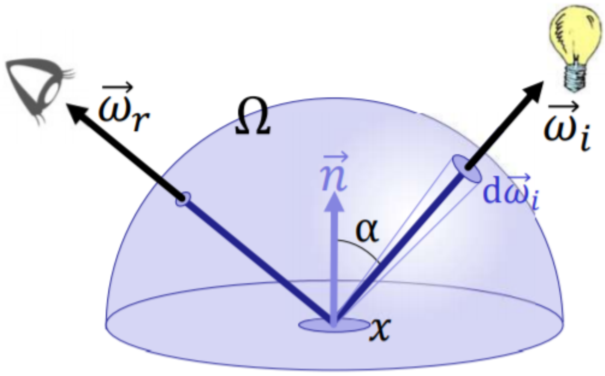


Illuminazione locale

L'illuminazione locale viene calcolata attraverso il fenomeno fisico della **radianza**.

Radianza

Data la seguente situazione:



dove:

- x è il punto sulla superficie illuminato
- $\vec{\omega_r}$ è la direzione che va da x verso l'osservatore
- $\vec{\omega_i}$ è la direzione che va da una fonte luminosa a x
- $\vec{n}$ è la normale della superficie dove giace il punto illuminato
- α è l'angolo formato da $\vec{n}$ e $\vec{\omega_i}$
- Ω è l'insieme di tutte le direzioni possibili, cioè l'insieme dei vettori unitari appartenenti alla semisfera illustrata

L'equazione della radianza è la seguente:

$$L_o = \underbrace{L_e(x, \vec{\omega}_r)}_{\text{emessa}} + \underbrace{L_r(x, \vec{\omega}_r)}_{\text{riflessa}}$$

dove:

- L_o è la quantità di luce ricevuta dall'osservatore che guarda un punto illuminato x
- L_e è la quantità di luce *emessa* da x
- L_r è la quantità di luce *riflessa* da x

La luce emessa viene calcolata solo per quei materiali che emettono luce propria.

La luce riflessa è invece molto più comune, e viene calcolata nel seguente modo:

$$L_r(x, \vec{\omega}_r) = \int\limits_{\vec{\omega_i} \in \Omega} \underbrace{f_r(x, \vec{\omega_i}, \vec{\omega_r})}_{\text{BRDF}} \underbrace{L_i(x, \vec{\omega_i})}_{\text{luce incidente}} \underbrace{(\vec{\omega_i} \cdot \vec{n})}_{\text{legge del } cos} d\vec{\omega_i}$$

Si tratta di un integrale (somma nel continuo è la giusta interpretazione) che "scorre" tutti i raggi $\vec{\omega_i}$ che colpiscono x nel dominio Ω. Per il singolo raggio $\vec{\omega_i}$ vengono quindi calcolati le tre componenti evidenziate, e le analizzeremo singolarmente.

Luce incidente

$$L_i(x, \vec{\omega_i})$$

È la quantità di luce che viaggia verso x dalla direzione $\vec{\omega_i}$, cioè quella emessa dalla fonte luminosa. In pratica modella l'ambiente di illuminazione, in quanto modella sia l'intensità di luce che arriva a x, ma anche il *colore* della luce, se espressa in RGB.

Bidirectional Reflectance Distribution Function - BRDF

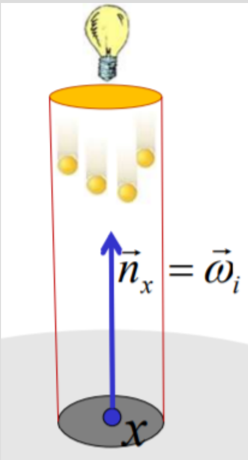
$$f_r(x, \vec{\omega_i}, \vec{\omega_r})$$

Tale funzione determina quanta parte della luce incidente che arriva in x da $\vec{\omega_i}$ viene riflessa in direzione $\vec{\omega_i}$. Questa funzione modella quindi le caratteristiche dell'oggetto su cui giace x, come il materiale.
In generale $f_r(\vec{\omega_i}, \vec{\omega_r})$ è detta BDRF del materiale, e ogni materiale avrà quindi una sua BRDF.

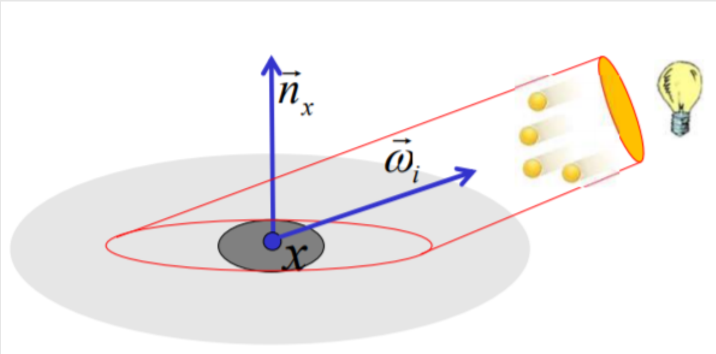
Legge del coseno

$$(\vec{\omega_i} \cdot \vec{n})$$

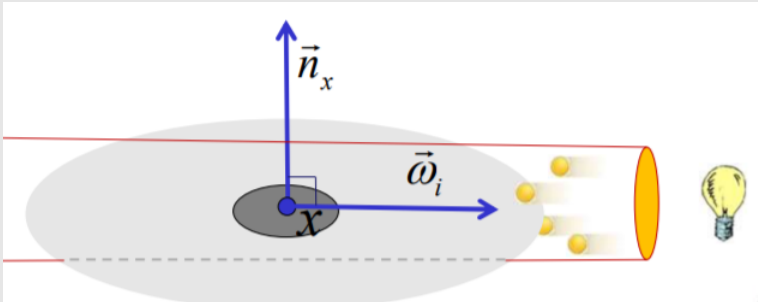
Si tratta del **coseno** di α, dove α è l'angolo compreso tra $\vec{\omega_i}$ e la normale $\vec{n}$. È quindi uguale al prodotto dot fra i due vettori, come mostra la formula. Questa legge quantifica quanta luce viene ricevuta da x in funzione della direzione da cui arriva la luce. Infatti deriva dall'osservazione delle seguenti situazioni:



Il raggio luminoso è ortogonale al piano e quindi ha la stessa direzione della normale. La quantità di luce ricevuta da x è quella massima



Il raggio luminoso forma un certo angolo con la normale, quindi la quantità di luce ricevuta da x è inferiore



Il raggio luminoso è parallelo al piano e quindi ortogonale alla normale, la quantità di luce ricevuta da x è pari a 0

Una relazione che modella questo comportamento è proprio il coseno, in quanto all'aumentare dell'angolo il coseno diminuisce il suo valore, mentre se l'angolo si riduce il valore del coseno aumenta.

Radianza - Efficienza

Il rendering dell'illuminazione locale basato sulla radianza è molto accurato ed è ottima per il rendering offline, ma spesso troppo oneroso per le applicazioni real-time.
Per permetterci il rendering real-time, dovremmo applicare delle semplificazioni.

Radianza - versione semplificata

Introduciamo alcune semplificazioni.

Ambiente di illuminazione discreto e semplificato

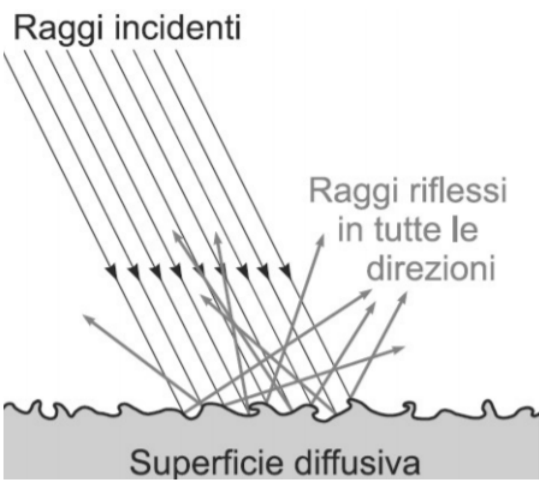
- Assumiamo che ogni punto della superficie riceva la stessa luce, quindi l'argomento x sparisce da L_i
- Il numero di fonti luminose sarà un numero N discreto e possibilmente piccolo, mentre dalle altre direzioni non arriva alcuna luce
- Il colore RGB della luce e l'intensità della luce saranno costanti per ogni fonte luminosa

  Quindi avremo:

$$L_r(x, \overrightarrow{\omega_r}) = \sum_{i=0}^{N-1} \underbrace{f_r(x, \overrightarrow{\omega_i}, \overrightarrow{\omega_r})}_{\text{BRDF}} \underbrace{\begin{pmatrix} L_i^R \\ L_i^G \\ L_i^B \end{pmatrix}}_{\text{luce incidente}} \underbrace{(\overrightarrow{\omega_i} \cdot \overrightarrow{n})}_{\text{legge del } cos}$$

Materiale semplificato

Non possiamo permetterci di assumere un materiale la cui BRDF varia punto per punto, bensì dobbiamo assumere un materiale con BRDF costante, che non varia né in base alla posizione della fonte luminosa né alla posizione dell'osservatore. I materiali di questo tipo esistono, e sono i materiali la cui superficie è molto irregolare e che quindi possiamo approssimare rifletta allo stesso modo in tutte le direzioni:



Queste superfici sono dette **superfici puramente diffusive** (o **puramente Lambertiane**).

Quindi assegniamo come BRDF un valore costante, detto **base color** (o **diffusive color**), descritto come RGB nel punto x.

Quindi avremo:

$$L_r = \sum_{i=0}^{N-1} \underbrace{\begin{pmatrix} D_x^R \\ D_x^G \\ D_x^B \end{pmatrix}}_{\text{base color}} \otimes \underbrace{\begin{pmatrix} L_i^R \\ L_i^G \\ L_i^B \end{pmatrix}}_{\text{luce incid.}} \cdot \underbrace{(\vec{\omega_i} \cdot \vec{n})}_{\text{legge del } cos}$$

Radianza - Considerazioni sulla versione semplificata

Il risultato ottenuto approssima brutalmente i fenomeni di illuminazione reali, ma è comunque una relazione ottenuta a partire da leggi fisiche, quindi è abbastanza accurato. Adesso vediamo una soluzione storica al lighting adottata da OpenGL, che non si basa su leggi fisiche ma che cerca di imitare i fenomeni in modo ragionevole.

Radianza secondo OpenGL

In questo modello storico la luce percepita finale sarà data dalla somma di tre componenti, a cui ciascuno corrisponde a un valore RGB:

$$\text{luce finale} = \text{ambiente} + \text{riflessione} + \text{emissione}$$

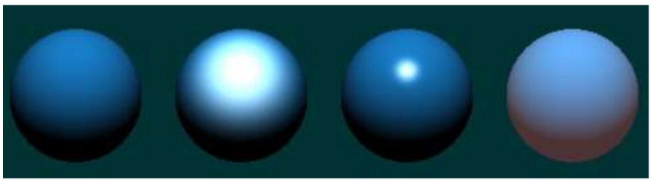
Emissione

Come per il modello precedente, si tratta di una costante RGB che viene sommata nel caso l'oggetto emetta luce propria.

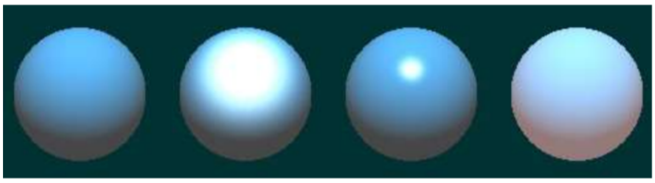
Ambiente

In questo caso si tratta di un valore che modella l'arrivo di un po' di luce da tutte le direzioni. Ad esempio è utile nel caso non si voglia far apparire gli oggetti neri se non vi è alcuna fonte luminosa. È anch'esso un valore RGB, che si ottiene moltiplicando componente per componente il *ambient color* del materiale(M) e il colore della luce(L):

$$\text{ambiente} = \begin{pmatrix} M^R \\ M^G \\ M^B \end{pmatrix} \otimes \begin{pmatrix} L^R \\ L^G \\ L^B \end{pmatrix}$$



senza



con

Un oggetto illuminato solo con la componente ambiente risulterà così:

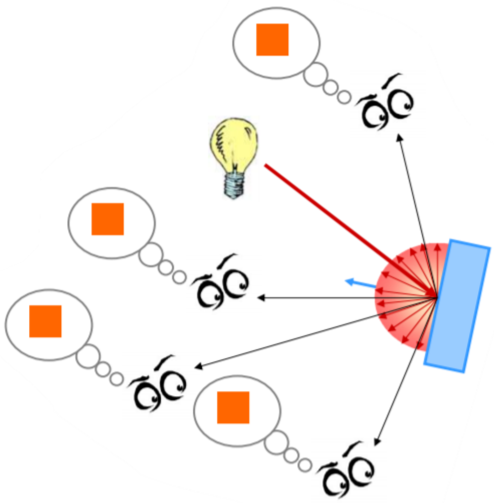
Riflessione

La componente riflessiva, somma 2 tipi di riflessione diversi:

- riflessione diffusa
- riflessione speculare

Riflessione diffusa

Si tratta della riflessione della luce in tutte le direzioni, similmente a come avevamo visto nel modello precedente quando avevamo semplificato il materiale a *materiale puramente diffusivo*. Dipende esclusivamente dalla legge del coseno, e quindi dipende solo dalla posizione della fonte luminosa rispetto alla superficie. La posizione dell'osservatore non viene considerata:



Per calcolare questa componente avremo:

$$I_{diff} = I_{luce} \otimes k_{base} \cdot \underbrace{(\vec{n} \cdot \vec{l})}_{\text{legge } cos}$$

dove:

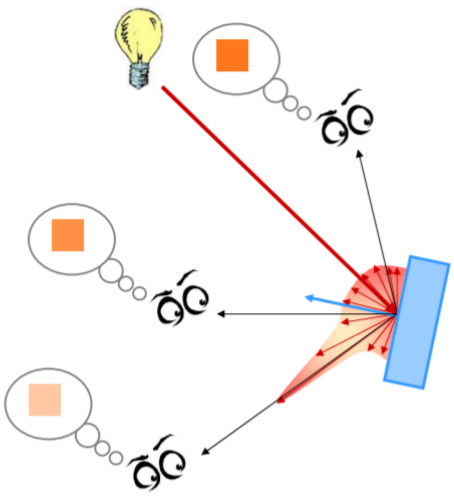
- I_{luce} è il colore assegnato alla luce da riflettere, di solito è (1,1,1)
- k_{base} è il base color del materiale
- $\vec{n}$ è la normale
- $\vec{l}$ è la direzione del raggio luminoso

Questa componente è l'unica che si basa su leggi fisiche di questo modello di lighting!

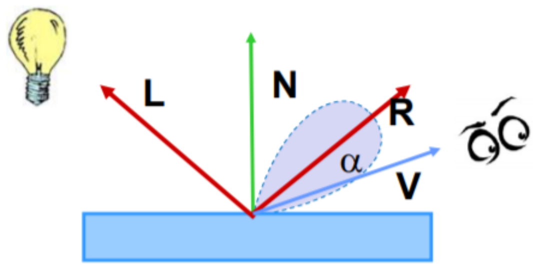
Un oggetto che ha sia componente ambientale sia compenente di riflessione diffusa sarà visualizzato così:

Riflessione speculare

Per poter dare al materiale dei riflessi specifici che dipendono dalla posizione dell'osservatore, è necessario decidere una direzione "preferenziale" della luce riflessa:



In questo modo la posizione dell'osservatore diventa rilevante! Per simulare questo effetto dobbiamo quindi trovare una relazione tra la luce riflessa e la posizione dell'osservatore:



Nella situazione data, avremo:

- L è la direzione raggio luminoso
- N è la normale
- R è il raggio riflesso
- V è la direzione che va verso l'osservatore

R è calcolato allo stesso modo in cui vengono calcolati i raggi riflessi nel ray_tracing.
L'idea è quella di trovare un modo per far sì che più il vettore V è allineato a R, più la luce percepita sarà intensa.

Come nella legge del coseno, il coseno dell'angolo tra R e V sarà tanto maggiore tanto più questi sono allineati (angolo piccolo). Possiamo quindi esprimere il calcolo della luce riflessa in questo modo:

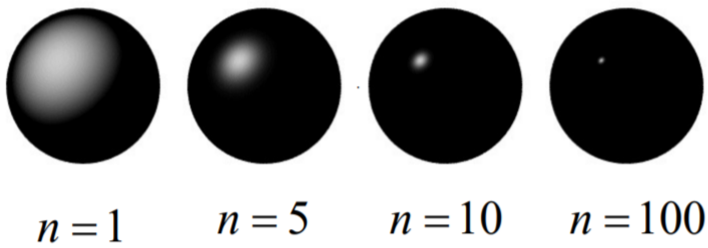
$$I_{spec} = I_{luce} \otimes k_{spec} \cdot \cos\alpha$$

dove:

- I_{luce} è come prima il colore della luce
- k_{spec} è il colore (e quindi anche l'intensità) del riflesso (*shinyness*)

In realtà la formula trovata è perfezionabile aggiungendo un modo per calcolare la *glossyness*, che corrisponde alla concentrazione dei riflessi. Per farlo basta ridurre il valore del coseno, quindi (essendo sempre un valore tra 0 e 1) basta moltiplicarlo per lui stesso, cioè elevarlo a potenza:

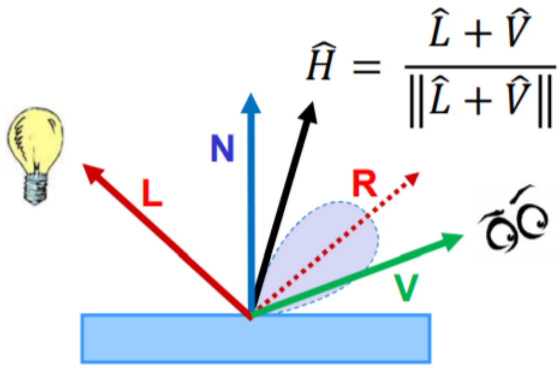
$$I_{spec} = I_{luce} \otimes \underbrace{k_{spec}}_{\text{shiny}} \cdot (\cos\alpha)^{\overbrace{n}^{\text{glossy}}}$$



Dopodiché possiamo passare dal coseno al prodotto dot, che come sappiamo sono equivalenti:

$$I_{spec} = I_{luce} \otimes \underbrace{k_{spec}}_{\text{shiny}} \cdot (R \cdot V)^{\overbrace{n}^{\text{glossy}}}$$

Questo modello di simulazione della luce riflessa è detto modello **Phong**, il suo inventore (1975). Tale modello per questione di semplicità è stato rivisto da **Blinn**, usando invece che il coseno tra R e V, il coseno tra H e N, dove H è il vettore in mezzo (*half-way vector*) tra L e V:



La formula quindi diventa:

$$I_{spec} = I_{luce} \otimes \underbrace{k_{spec}}_{\text{shiny}} \cdot (H \cdot N)^{\overbrace{n}^{\text{glossy}}}$$

Ora simulando la componente ambientale, la componente di riflessione diffusa e anche quella speculare, un oggetto apparirà così:

Radianza secondo OpenGL (versione 1.0)

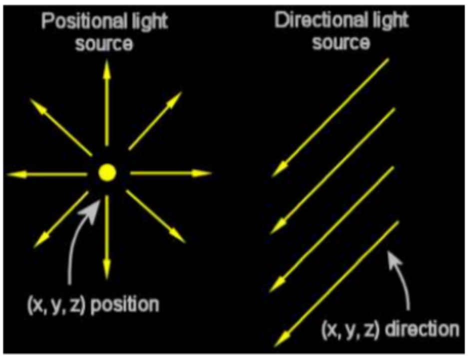
Quindi sommando tutte e tre le componenti, il modello può essere scritto come:

$$I_{tot} = \sum_{\text{luci}} \left(\begin{array}{c} \boxed{I_{luce\ ambient}} \otimes \boxed{k_{materiale\ ambient}} + \\ \boxed{I_{luce\ diffuse}} \otimes \boxed{k_{materiale\ diffuse}} \cdot (\vec{L} \cdot \vec{N}) + \\ \boxed{I_{luce\ specular}} \otimes \boxed{k_{materiale\ specular}} \cdot (\vec{H} \cdot \vec{N})^{\boxed{n}} \end{array} \right) + \boxed{k_{materiale\ emission}}$$

Dove in giallo vi sono i fattori che dipendono dalla luce, e in marroncino quelli che dipendono dal materiale.

Considerazione sulla direzione della luce *L*

Nello sviluppo delle applicazioni, è possibile scegliere se avere un rendering basato su una illuminazione generalizzata e costante della scena, come se ci fosse la luce solare, oppure un'illuminazione che varia in base alla posizione degli oggetti, come una lampadina. Nel primo caso parliamo di **luce direzionale**, nel secondo di **luce posizionale**.



Nel caso di luce direzionale, il vettore $\vec{L}$ sarà uguale per tutti i punti della scena, quindi non mi serve calcolarlo punto per punto, invece nel secondo caso $\vec{L}$ andrà calcolato, data la posizione della fonte luminosa L_{POS} per ogni punto della scena P:

$$\vec{L} = \frac{P - L_{POS}}{||P - L_{POS}||}$$

Inoltre, avendo a disposizione L_{POS} può essere utile introdurre un fattore di attenuazione della luce, in questo modo oltra alla direzione dell'osservatore rispetto all'oggetto, conterà anche la sua distanza da esso!
In fisica per attenuare l'intensità della luce in funzione della distanza faremmo:

$$f_{attenuazione\ luce} = \frac{1}{c \cdot d_L^2}$$

dove c è la velocità della luce e d_L^2 è il quadrato della distanza dall'osservatore.
Nella pratica però questa legge riduce l'intensità in modo troppo repentino.
Quindi si usa un polinomio di secondo grado al denominatore:

$$f_{attenuazione\ luce} = \min\left(\frac{1}{\boxed{c_1} + \boxed{c_2} d_L + \boxed{c_3} d_L^2}, 1\right)$$

con c_1, c_2 e c_3 costanti arbitrarie dipendenti dalla luce corrente.

Radianza secondo OpenGL (versione finale)

$$I_{tot} = \sum_{\text{luci}} \left(\left(\begin{matrix} \boxed{I_{luce\ ambient}} \otimes \boxed{k_{materiale\ ambient}} + \\ \boxed{I_{luce\ diffuse}} \otimes \boxed{k_{materiale\ diffuse}} \cdot (\vec{L} \cdot \vec{N}) + \\ \boxed{I_{luce\ specular}} \otimes \boxed{k_{materiale\ specular}} \cdot (\vec{H} \cdot \vec{N})^{\boxed{n}} \end{matrix} \right) \cdot f_{al} \right) + \boxed{k_{mat.\ emis.}}$$