Fondamenti di Grafica Tridimensionale:

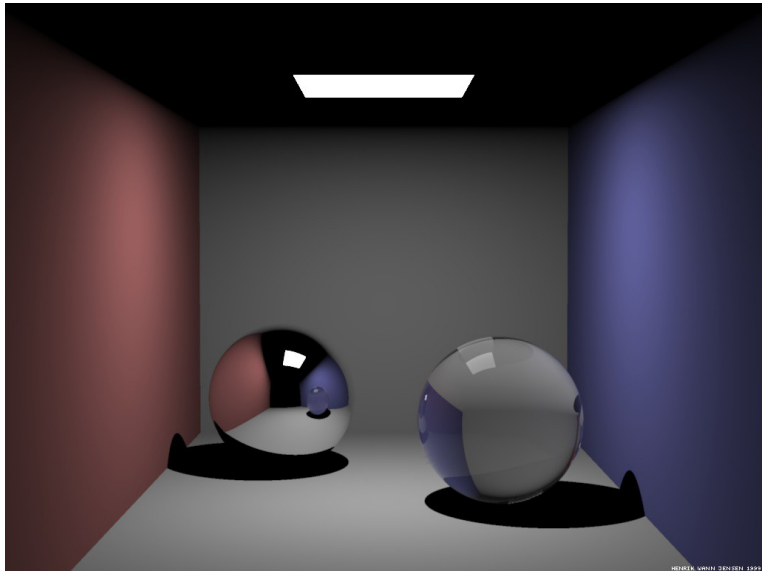
Global Illumination

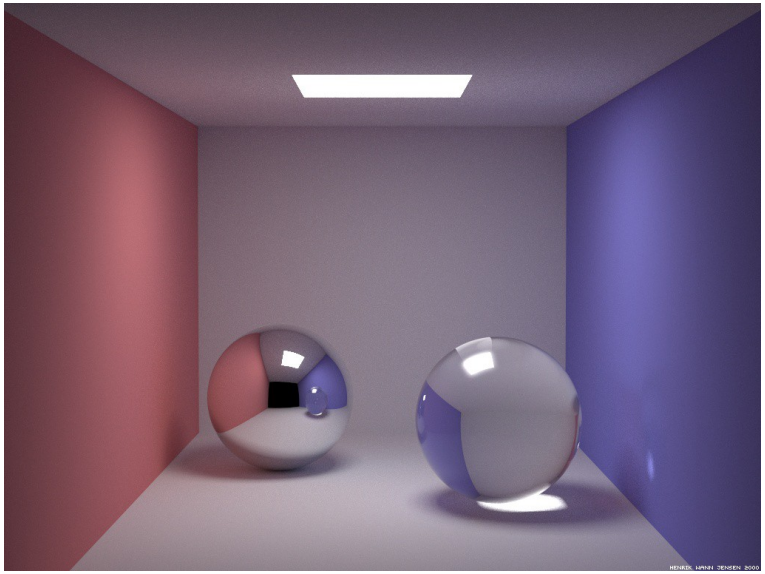
dr. Francesco Banterle
francesco.banterle@isti.cnr.it

slides basate su quelle di:

dr. Paolo Cignoni
p.cignoni@isti.cnr.it
http://vcg.isti.cnr.it/~cignoni

Global Illumination: Perché?

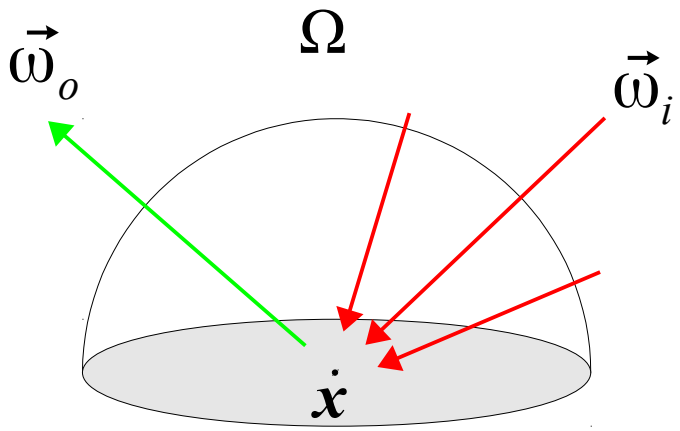




The Rendering Equation

"To provide an unified context for viewing rendering algorithms as more or less accurate approximations to the solution for a single equation" [Kajiya 1986]

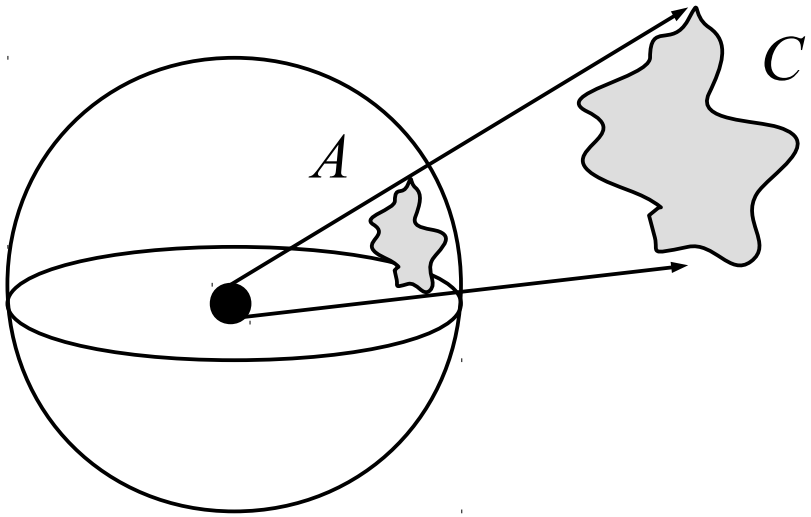
$$L_o(\boldsymbol{x}, \vec{\omega}_o) = L_e(\boldsymbol{x}, \vec{\omega}_o) + \int_{\Omega} f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_i(\boldsymbol{x}, \vec{\omega}_i)(\vec{\omega}_i \cdot \vec{n}) \, d\vec{\omega}_i$$



Radiometria

- ***Flusso radiante*** Φ: è la quantità di energia che arriva ad una regione di spazio per unità di tempo
- L'unità di misura è il watt (W) o Joule per secondo (J /s)

- ***Angolo solido***: $\Omega = A / r^2$ [sr]



Radianza

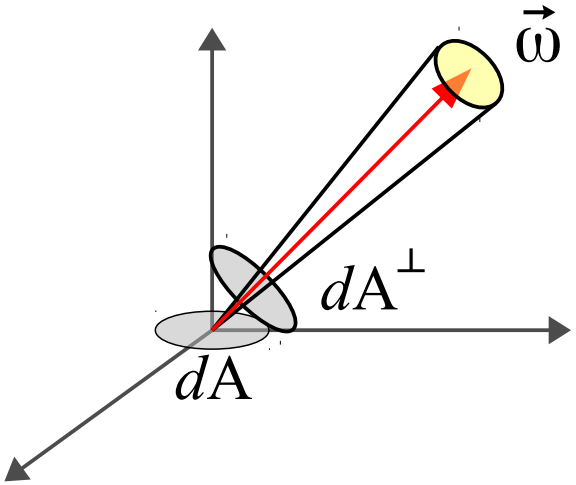
•La ***radianza*** L è una delle 8 unità fisiche radiometriche del S.I.:

$$L = \frac{d\,\Phi}{d\,\vec{\omega}\,dA^{\perp}}$$

•Misura l'intensità di un raggio luminoso definita come la potenza per angolo solido unitario per area proiettata.

•Si misura in: W / (sr m^2)

Nota: $dA^{\perp} = |\cos\theta| d\omega$



Irradianza

•***Irradianza**, E,* è l'integrale della radianza su tutte le direzioni; pesato per il coseno della direzione:

$$E(\boldsymbol{x}, \vec{n}) = \int_{\Omega(\vec{n})} L_i(\boldsymbol{x}, \vec{\omega}) \cos\theta \, d\vec{\omega}$$

- θ è l'angolo tra $\vec{n}$ e $\vec{\omega}$
- $L_i(\boldsymbol{x}, \vec{\omega})$ è la radianza in arrivo
- **NOTA**: $d\vec{\omega} = \sin\theta \, d\theta \, d\phi$

•L'irradianza si misura in W / m^2

Radiosity

- ***Radiosity***, *B*, è l'energia per unità di area che lascia una superficie per unità di tempo

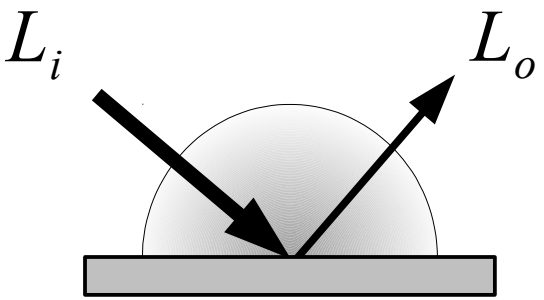
- Unità di misura: W/m^2

Radianza Riflessa

- La radianza riflessa è definita come:

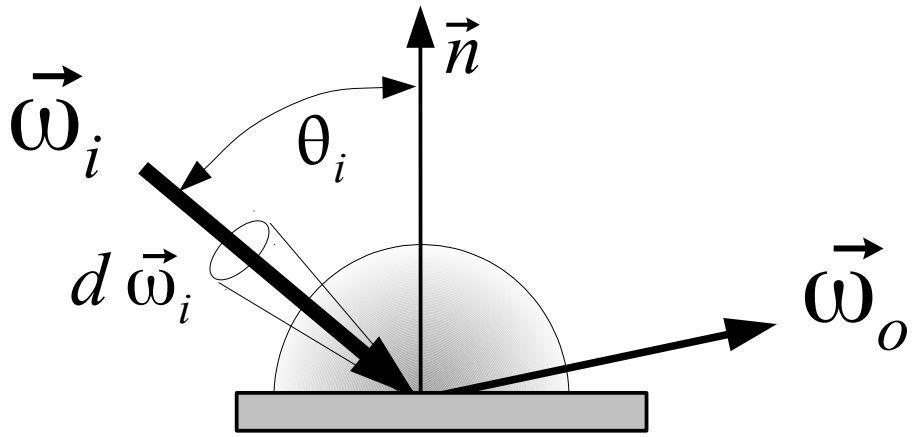
$$L_o(\boldsymbol{x}) = \int \rho \, L_i(\boldsymbol{x}, \vec{\omega}) \, d\vec{\omega}$$

e rappresenta la quantità di energia (luce) che rimbalza da una direzione verso un'altra.



Con variazione direzionale

$$L_o(\boldsymbol{x}, \vec{\omega}_o) = L_e(\boldsymbol{x}, \vec{\omega}_o) + \int_{\Omega} f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o)\, L_i(\boldsymbol{x}, \vec{\omega}_i) \cos\theta \, d\,\vec{\omega}_i$$

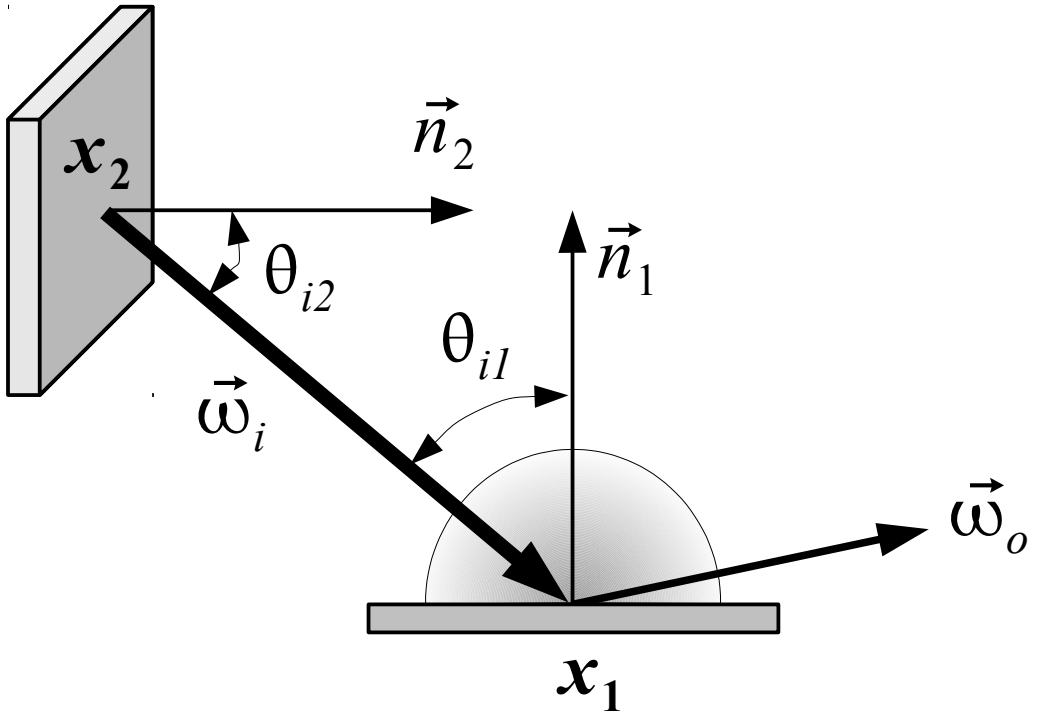


Integrazioni di Superficie

$$L_o(\boldsymbol{x}, \vec{\omega}_o) = L_e(x, \vec{\omega}_o) +$$

$$+ \int_S f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_i(\boldsymbol{x}, \vec{\omega}_i) g(\boldsymbol{x}, \boldsymbol{x}') \frac{\cos\theta_{i1} \cos\theta_{12}}{\|\boldsymbol{x} - \boldsymbol{x}'\|^2} dA$$

$$d\vec{\omega} = \frac{dA\cos(\theta)}{r^2}$$



BRDF: Bidirectional Reflectance Distribution Function

- Formalizzazione per descrivere come la luce interagisce con un materiale

$$f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o) = \frac{dL_o(\boldsymbol{x}, \omega_o)}{dE(p, \omega_i)} = \frac{dL_o(\boldsymbol{x}, \omega_o)}{L_i(\boldsymbol{x}, \omega_i)\cos(\theta_i)\, d\,\vec{\omega}_i}$$

- Proprietà:
  - $f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o) = f_r(\boldsymbol{x}, \vec{\omega}_o \rightarrow \vec{\omega}_i)$
  - $\int_{\Omega(\vec{n})} f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o)\cos(\theta_i)\, d\,\omega_i \leq 1$

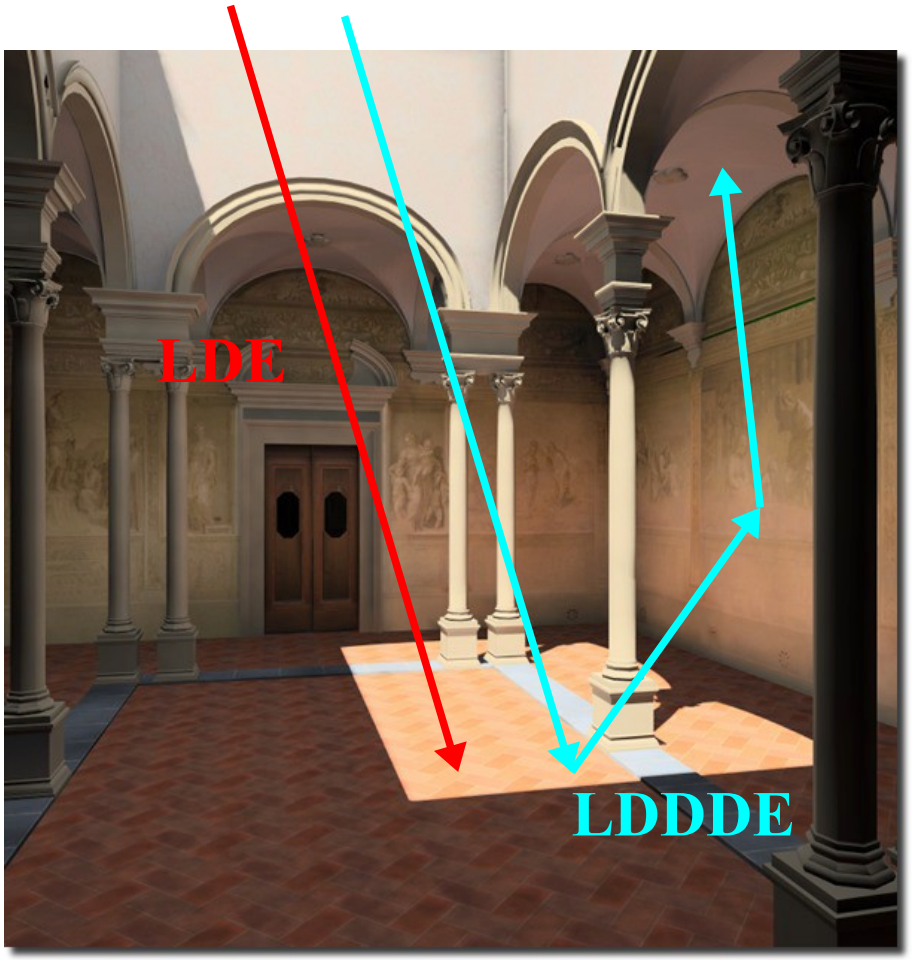
Notazione Light Paths

- Una grammatica regolare introdotta da Heckbert, utile per la classificazione
- Si descrive il percorso della luce dalla sorgente luminosa all'occhio:

$$L(S|D)^*E$$

- L: sorgente luminosa (light)
- D: riflessione diffusa (diffuse)
- S: riflessione speculare (specular)
- E: occhio (eye)

Esempi di Light Paths



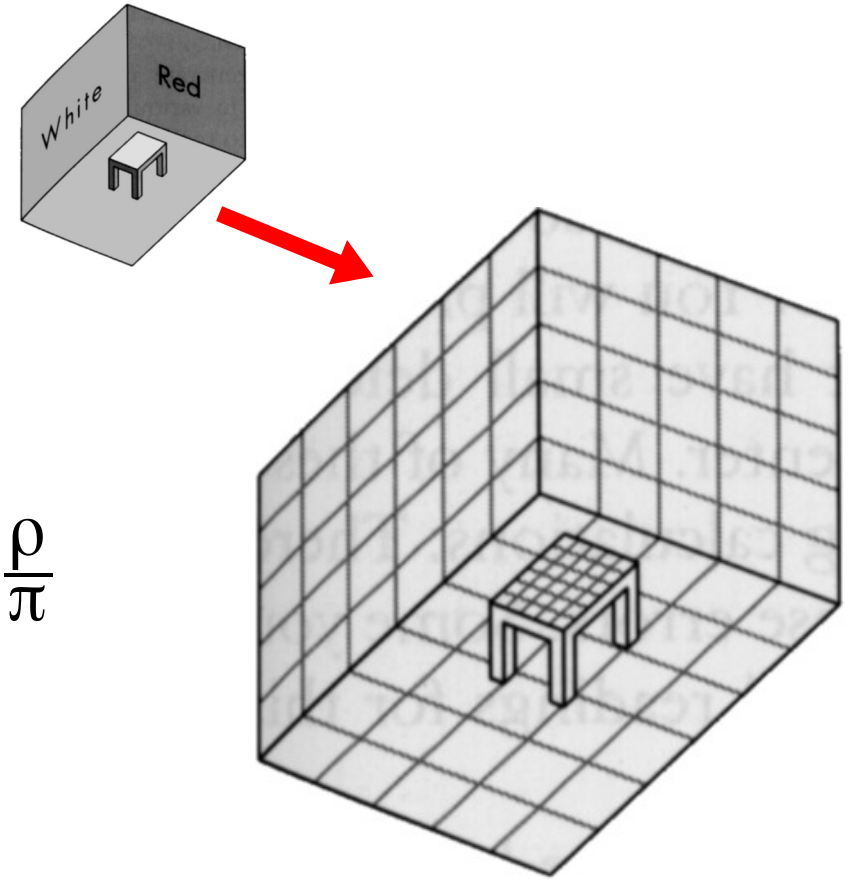
Esempi di Light Paths: Caustics

Algoritmi di Global Illumination

- The Radiosity Method
- Whitted Ray-Tracing
- Distried Ray-Tracing
- Path-Tracing + Irradiance Caching
- Photon Mapping
- Instant Radiositybuted Ray

The Radiosity Method

- Nel Radiosity, la scena viene suddivisa in *patch*, ovvero in poligoni piatti e di dimensioni limitate, ciascuno dei quali è considerato perfettamente diffusivo
- Questo implica che:

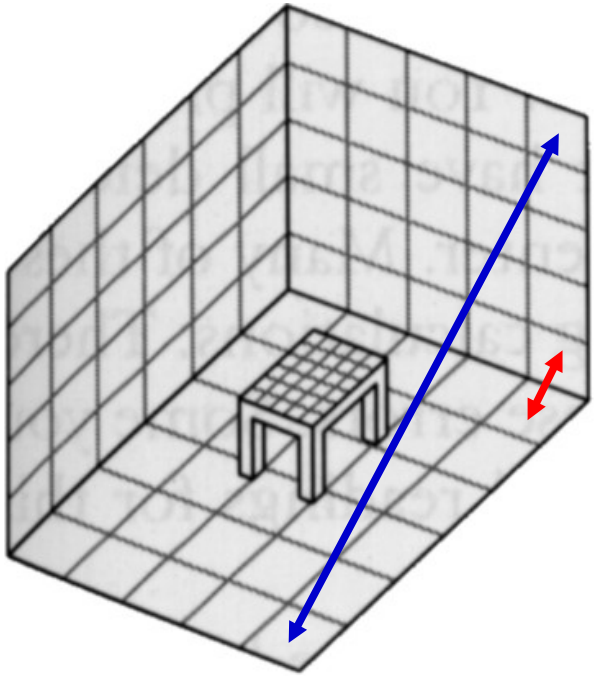


$$f_r(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o) = \frac{\rho}{\pi}$$

$$L_o(\boldsymbol{x}, \vec{\omega}_o) = L_e(\boldsymbol{x}, \vec{\omega}_o) + \frac{\rho}{\pi} \int_{\Omega} L_i(\boldsymbol{x}, \vec{\omega}_i)(\vec{\omega}_i \cdot \vec{n})\, d\vec{\omega}_i$$

The Radiosity Method: Primo Passo

- Il primo passo consiste nel determinare, per ogni coppia di *patch*, i *form factor* (FF).
- I *form factor* definiscono quanta energia lascia un patch verso un'altra tenendo conto di:
  - Occlusioni
  - Orientamento
  - Distanza tra le patch
- Il calcolo dei *form factor* è $O(n^2)$; la maggior parte dei *form factor* sono nulli:
  - *Patch* lontane non si influenzano



The Radiosity Method: Secondo Passo

- Nel secondo passo, la distribuzione della luce all'interno di una scena corrisponde ad un sistema di equazioni lineari

- La somma della radianza che arriva su una *patch* deve essere uguale alla radianza che esce più la luce assorbita dalla *patch* stessa

The Radiosity Method: Equazione e Form Factor

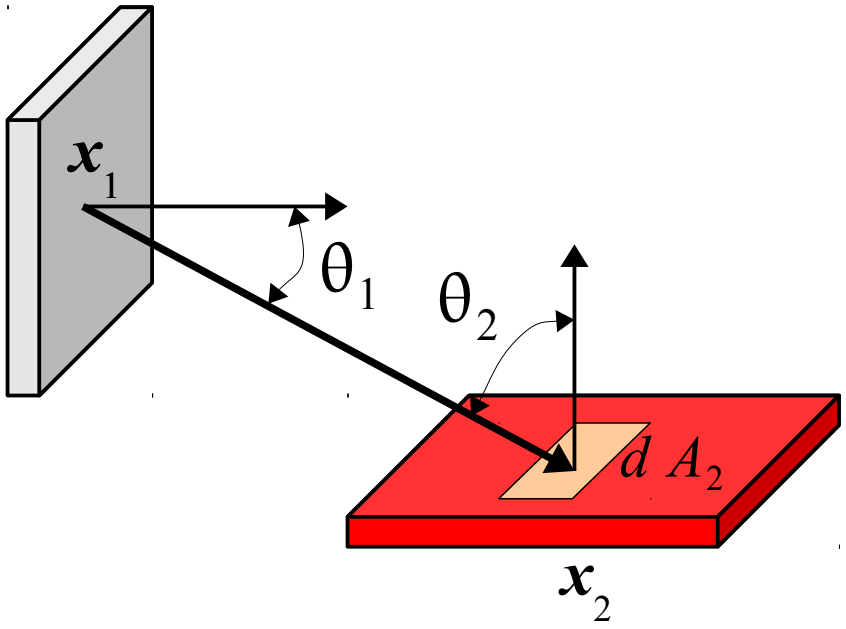
- *Form Factor*:

$$F(\boldsymbol{x}_1, \boldsymbol{x}_2) = V(\boldsymbol{x}_1, \boldsymbol{x}_2) \frac{\cos\theta_1 \cos\theta_2}{\pi \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2} dA_2$$

- Vale che:

$$F(\boldsymbol{x}_1, \boldsymbol{x}_2) A_1 = F(\boldsymbol{x}_2, \boldsymbol{x}_1) A_2$$

$$F(\boldsymbol{x}_1, \boldsymbol{x}_2) = F(\boldsymbol{x}_1, \boldsymbol{x}_2) \frac{A_2}{A_1}$$



The Radiosity Method: Equazione

- In termini di L.E.

$$B(\boldsymbol{x}')d\,\boldsymbol{x}' = \varepsilon(\boldsymbol{x}')d\,\boldsymbol{x}' + \rho(\boldsymbol{x}')\int_S B(\boldsymbol{x})F(\boldsymbol{x},\boldsymbol{x}')d\,\boldsymbol{x}$$

L'integrale viene tipicamente discretizzato ottenendo una sommatoria:

- *Patch* discrete invece di infinitesimali della superficie

$$B(p_i)A(p_i) = \varepsilon(p_i)A(p_i) + \rho(p_i)\sum_{j=1}^{n} B(p_j)A(p_j)F_{j,i}$$

The Radiosity Method: Discretizzazione

- Sfruttando il principio di reciprocita dei *form factor*, si divide per

$$A(p_i)$$

ottenendo il seguente risultato:

$$B(p_i) = \varepsilon(p_i) + \rho(p_i) \sum_{j=1}^{n} B(p_j) F_{j,i}$$

The Radiosity Method: Sistema Lineare

$$B(p_i) = \varepsilon(p_i) + \rho(p_i) \sum_{j=1}^{n} B(p_j) F_{j,i}$$

$$B_i = E_i + R_i \sum_{j=1}^{n} B_j F_{j,i}$$

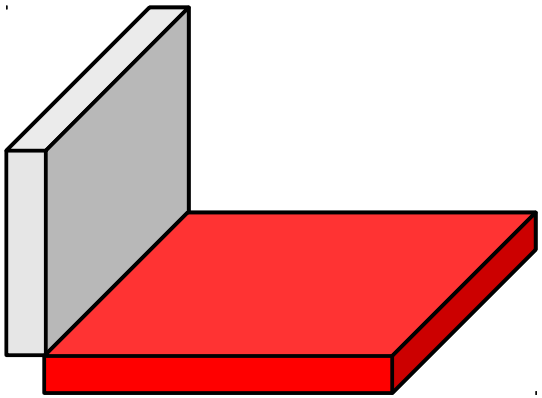
$$\begin{bmatrix} 1-R_1F_{11} & -R_1F_{12} & \cdots & -R_1F_{1n} \\ -R_2F_{21} & 1-R_2F_{22} & \cdots & -R_2F_{2n} \\ \vdots & & \ddots & \vdots \\ -R_nF_{n1} & \cdots & & 1-R_nF_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}$$

The Radiosity Method: Calcolo del Form Factor

- I *form factor* discreti tra due *patch* sono definiti come

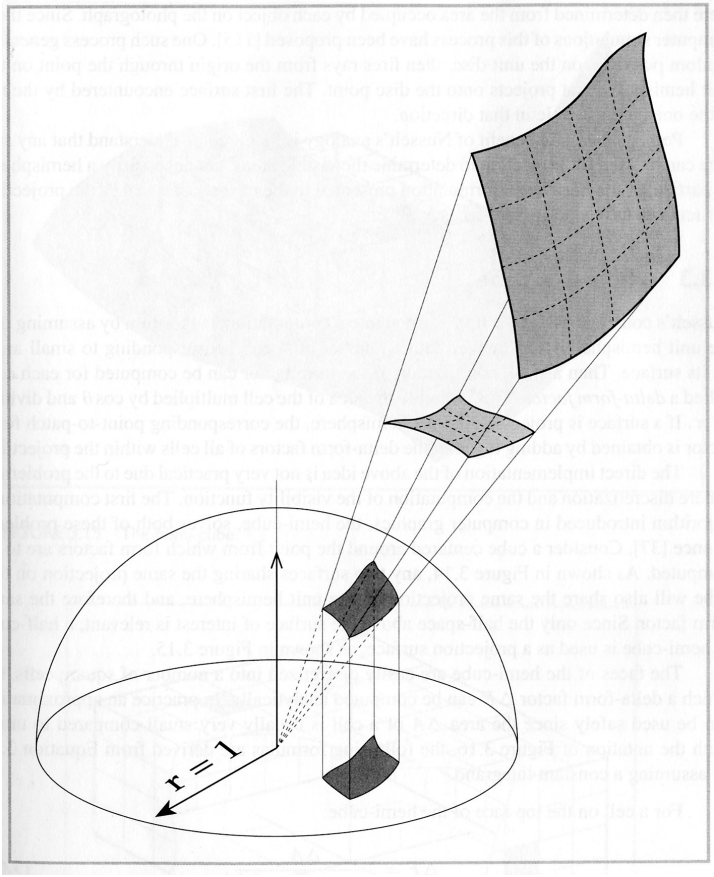
$$F_{i,j} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos(\varphi_i)\cos(\varphi_j)}{\pi r_{i,j}^2} dA_j \, dA_i$$

- In casi molto semplici esiste una forma chiusa
  - Caso rettangolo-rettangolo, totalmente non occlusi
  - In generale sono risultati solo teorici...



The Radiosity Method: Calcolo del Form Factor

- Ogni patch con la stessa proiezione sulla semisfera ha lo stesso *form factor*
- Metodo di Montecarlo:
  - Campionare (uniformemente in area proiettata sul piano) la semisfera
  - "Lanciare" un raggio
  - *Form factor* per ogni *patch* è la somma pesata del numero di raggi che colpisce quella *patch*



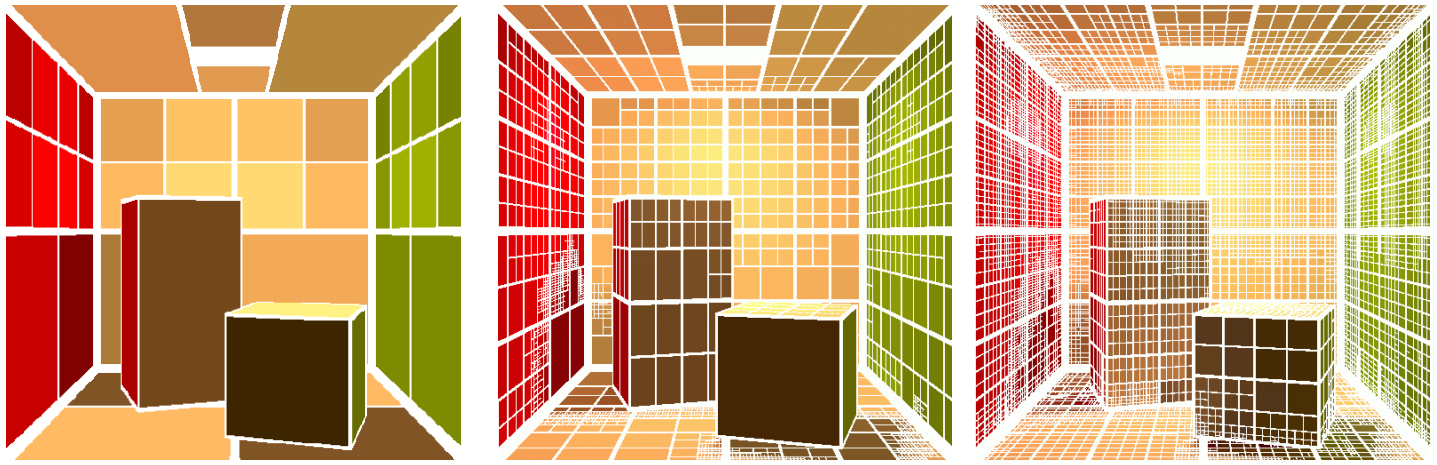
The Radiosity Method: Approccio Iterativo

- Approccio iterativo:
  - Evitare di calcolarsi e memorizzarsi tutti i *form factor* subito
  - Avere una soluzione approssimata subito che migliora ad ogni iterazione
- Iterazione:
  - Selezionare la *patch i*-esima; massima differenza di *unshot*
  - Calcolare F_{ij}, per ogni *patch j*-esima:
    - Aggiornamento del radiosity della *patch j*-esima
    - Aggiornamento dell'emissione della *patch j*-esima
  - Assegnare l'emissione della *patch i*-esima a zero

The Radiosity Method: Approccio Iterativo

0 rimbalzi 1 rimbalzi 2 rimbalzi

The Radiosity Method: Raffinamento della Mesh



The Radiosity Method: Conclusioni

- Path Notation: LD*E
- Considera l'interazione tra le superfici della scena
- La soluzione è tipicamente *view-independent*

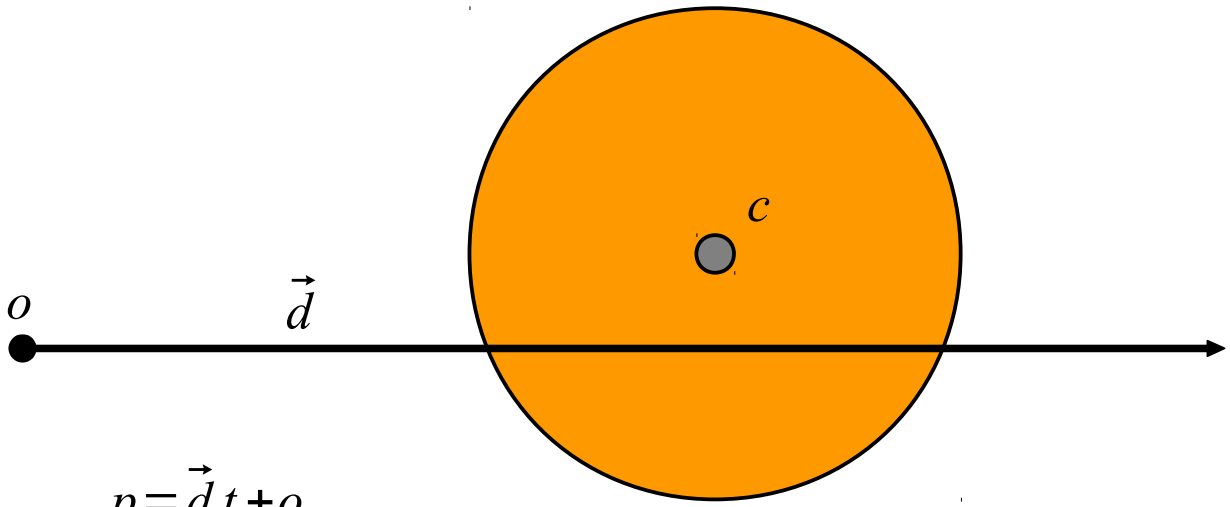
- **Problemi**:
  - Discretizzazione della scena in *patch*
  - Non modella riflessioni speculari
  - Calcolare *form factor* per tutte le coppie di *patch*
  - Risolvere il sistema

Whitted Ray-Tracing

- Un algoritmo ricorsivo; partenza dalla camera virtuale
- Per ogni pixel del piano immagine:
  - Viene “lanciato” nella scena un raggio, *R*
  - Se *R* colpisce una superficie diffusa, ***x***, viene calcolata l'illuminazione diretta, lanciando un raggio *S* (*shadow ray*) verso la luce per controllare se il punto ***x*** è in ombra
- Se la superficie è riflettente o rifrattiva viene lanciato un raggio riflesso o rifratto

Whitted Ray-Tracing: Intersezione con la scena

$$(p_x - c_x)^2 + (p_y - c_y)^2 + (p_z - c_z)^2 = r^2$$

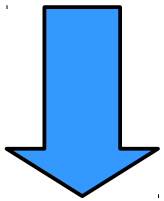


$$p = \vec{d}\,t + o$$

Whitted Ray-Tracing: Intersezione con la scena

$$(p_x - c_x)^2 + (p_y - c_y)^2 + (p_z - c_z)^2 = r^2$$

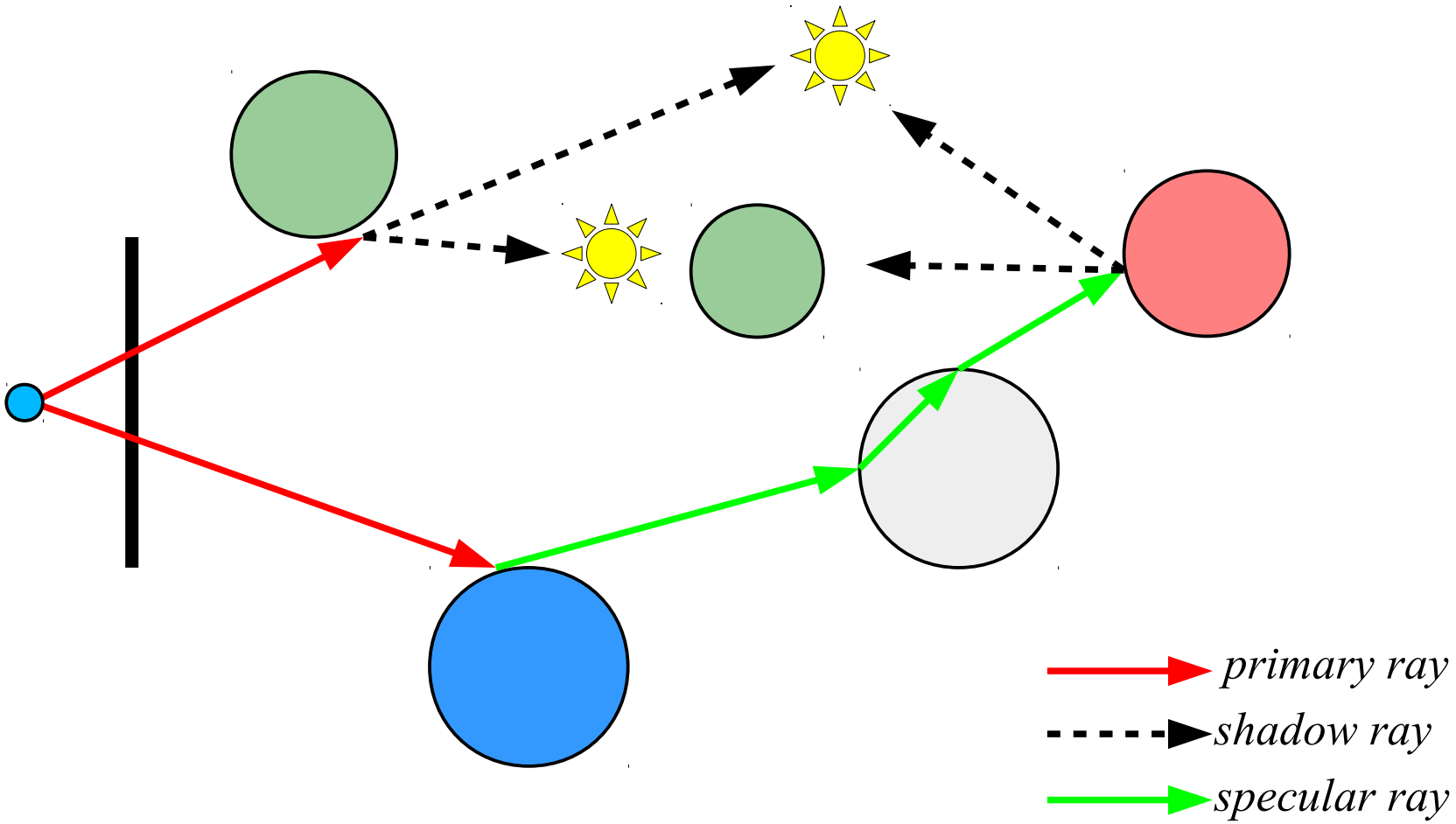
$$p = \vec{d}\,t + o$$



$$(d_x t + o_x - c_x)^2 + (d_y t + o_y - c_y)^2 + (d_z t + o_z - c_z)^2 = r^2$$

Whitted Ray-Tracing: path

- Path Notation: LD[S]*E



Whitted Ray-Tracing: Pseudo-code

Per ogni pixel nell'immagine:

    Crea un raggio, *R*, dalla telecamera per il pixel

    Colore del pixel = Trace(*R*)

Trace(*R*):

    Trova la più vicina intersezione (***x***, *n*) tra *R* e la scena

    Ritorna Shade(***x***, *n*)

Shade(***x***, *n*):

    Colore = 0

    Per ogni luce nella scena:

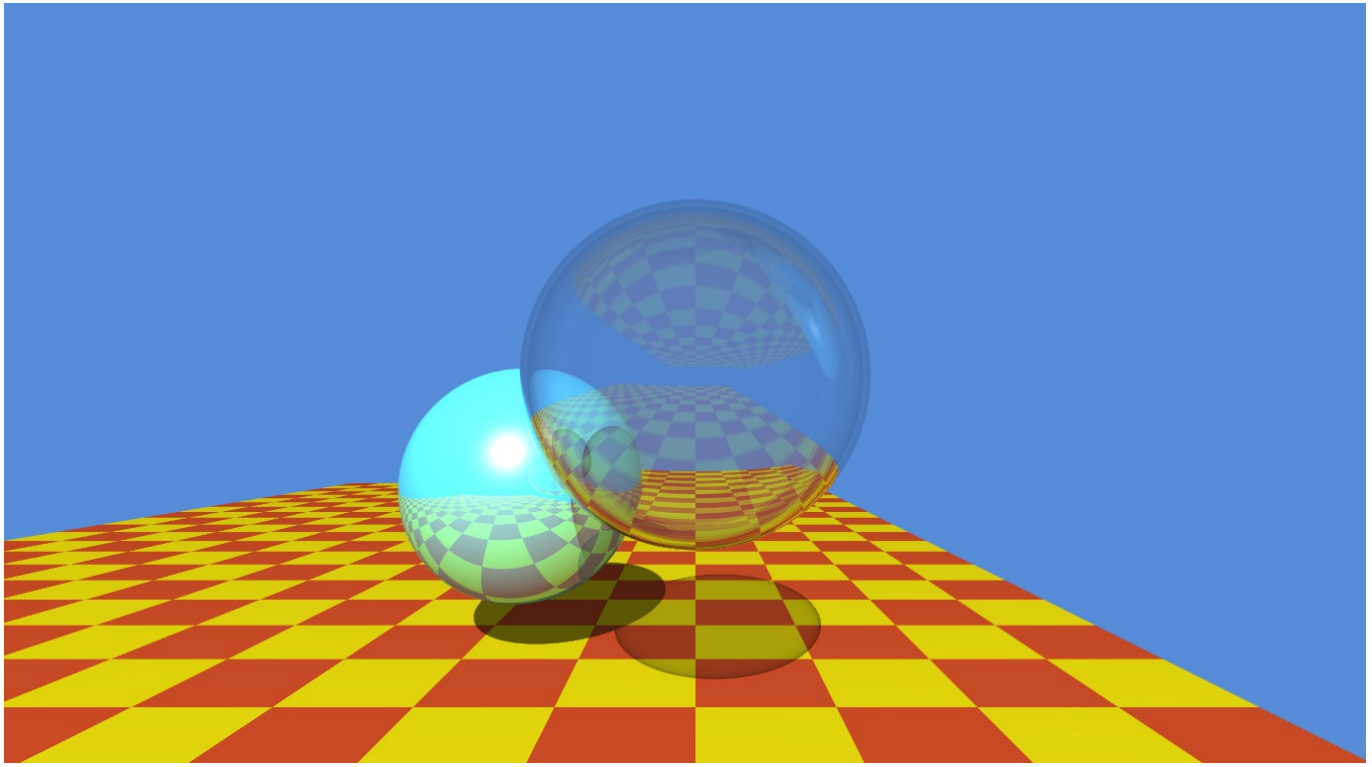
        Crea uno *shadow ray*, *S*, verso la luce

        Se S interseca la luce → Colore = Colore + luce diretta

        Se speculare → Colore = Colore + Trace(riflesso / rifratto)

    Ritorna Colore

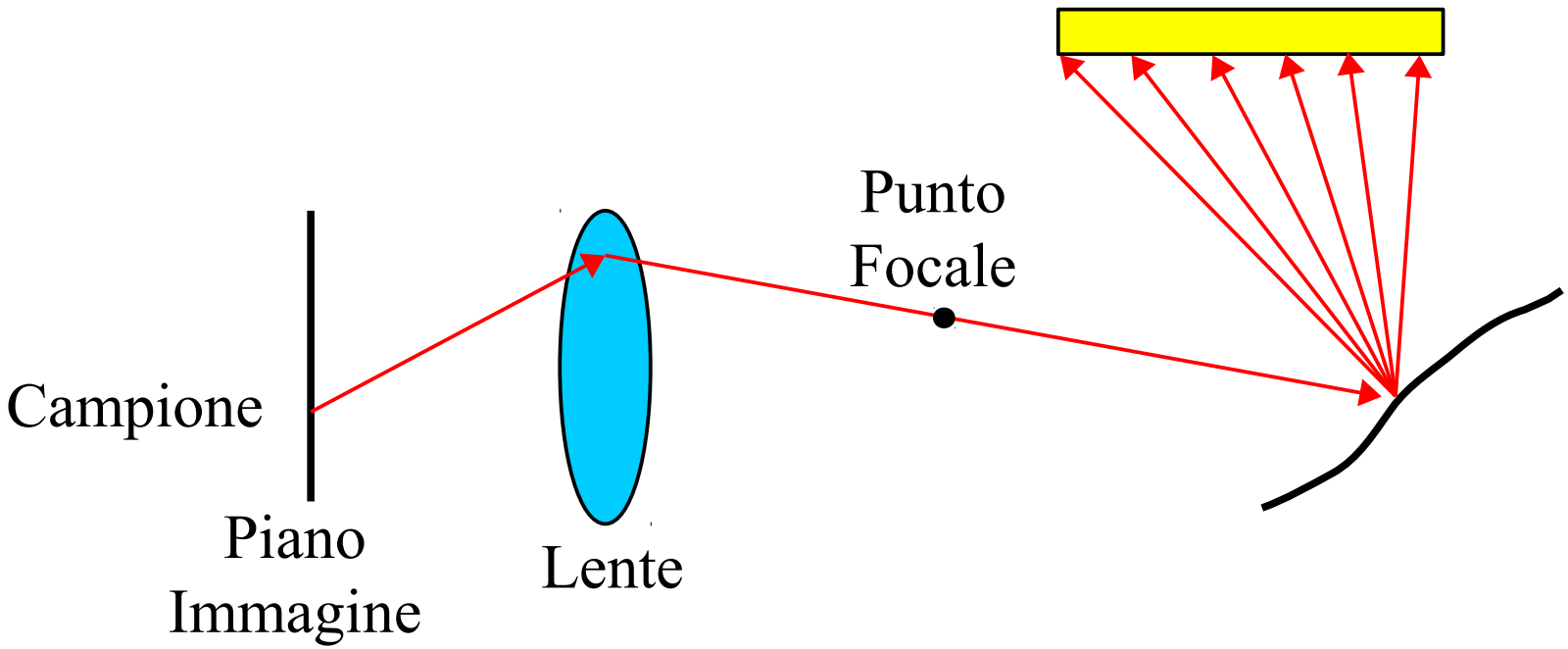
Whitted Ray-Tracing: Esempio



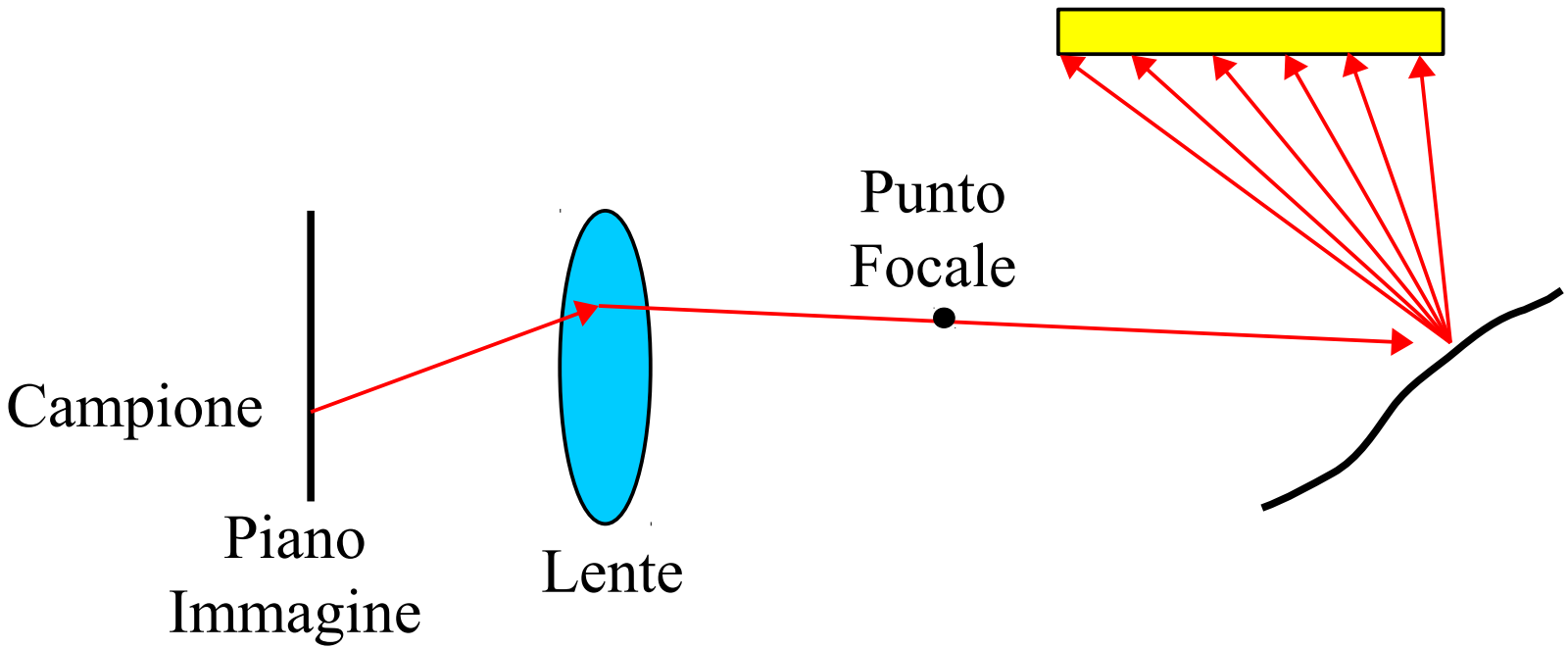
Distributed Ray-Tracing

- Un'estensione del Whitted Ray-Tracing, per simulare i seguenti effetti:
  - Anti Aliasing
  - Area Lights
  - Soft Reflections
  - Motion Blur
  - Depth of Field
- L'idea di base è di lanciare più raggi utilizzando metodi di Monte-Carlo:
  - Buon generatore di numeri casuali; Mersenne Twister
  - Distribuzione di punti: Jittering, Hammersly, Halton, Poisson

Distributed Ray-Tracing: Diagramma



Distributed Ray-Tracing



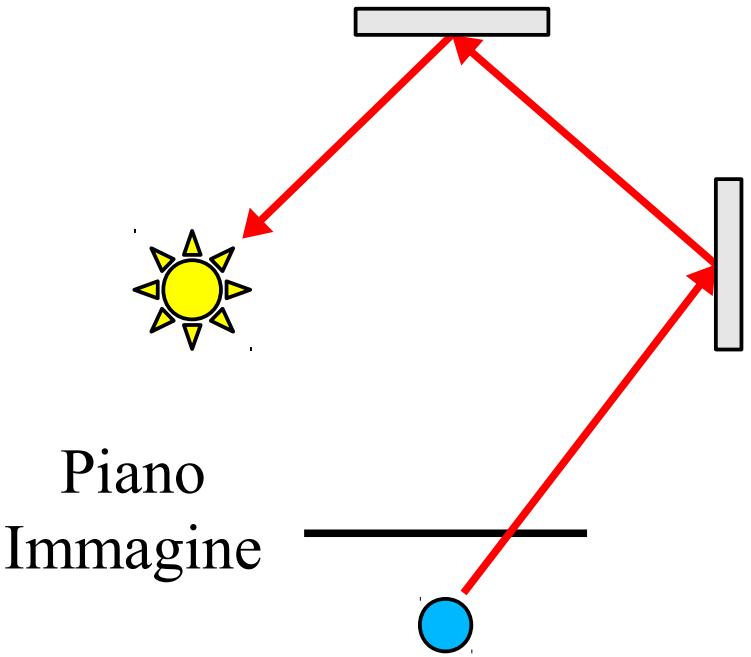
Distributed Ray-Tracing: Esempio

Path-Tracing

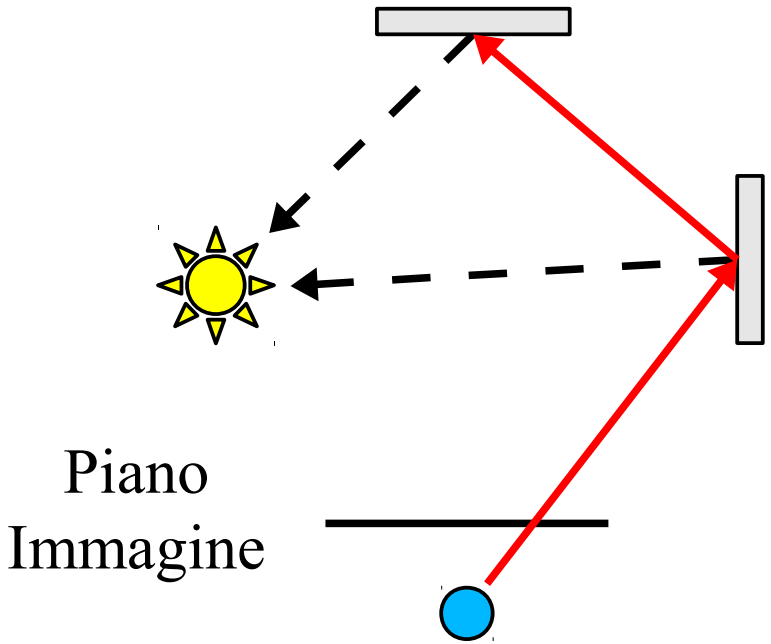
- Naturale estensione del Ray-Tracing e Distributed Ray-Tracing
- Per ogni pixel vengono generati *N* path come nel Distributed Ray-Tracing
- **Differenza**: quando viene colpita una superficie viene sempre creato un raggio di riflessione in base alle proprietà del BRDF

- Path Notation: L(S|D)*E

Path-Tracing: Implicito ed Esplicito

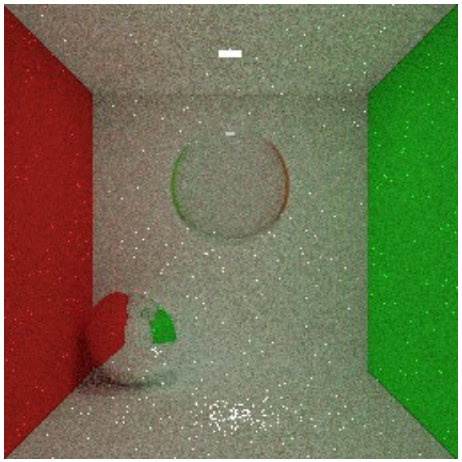


Path-Tracing Implicito

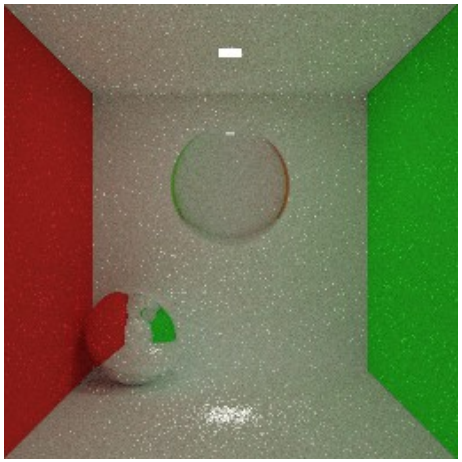


Path-Tracing Esplicito

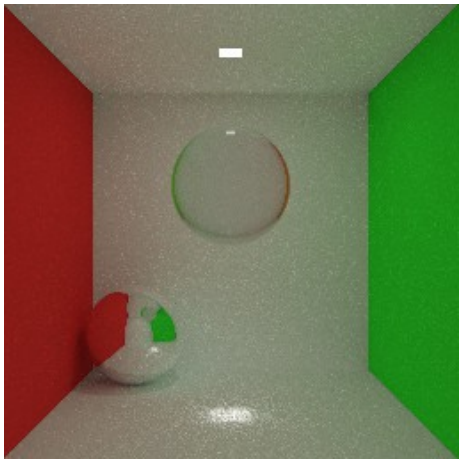
Path-Tracing: Esempio



4 path per pixel



16 path per pixel



64 path per pixel

Path-Tracing Conclusioni

- Simulazione di tutti i *path* di illuminazione in modo unbiased

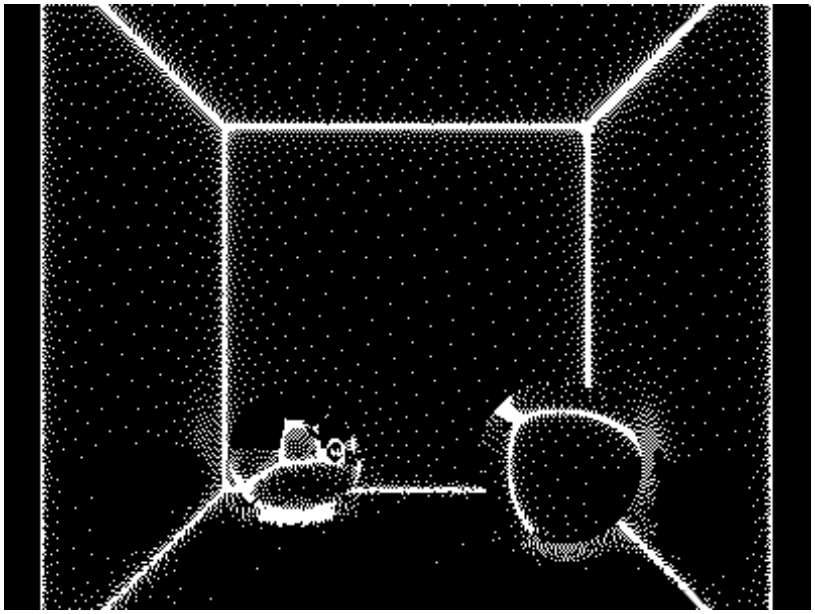
- Le immagini richiedono un'elevato numero di path per ridurre il rumore, circa 1000 *path* per pixel. La varianza del rumore diminuisce proporzionalmente $\frac{1}{\sqrt{N}}$

- Non appena vi è un oggetto riflettente o rifrattivo il rumore aumenta in modo elevato, più di 10000 *path* per pixel sono necessari per ridurre il rumore!

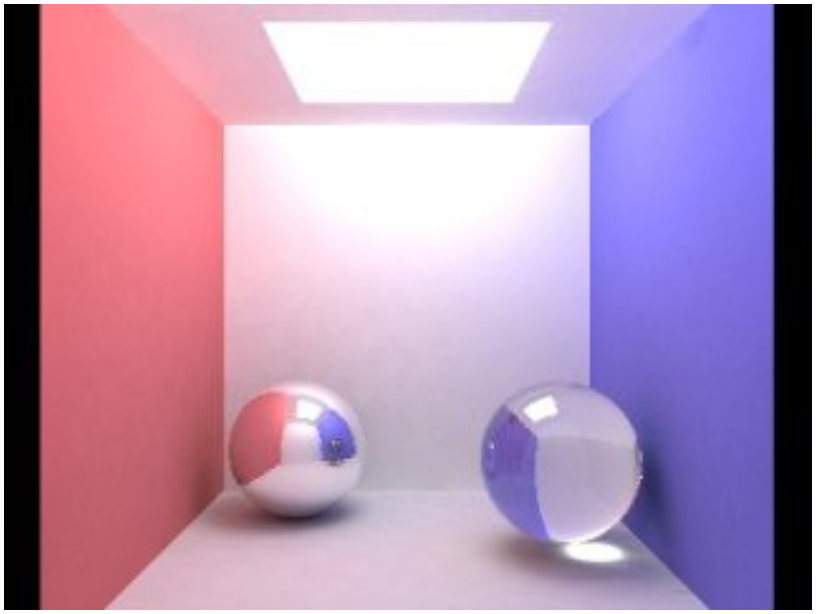
Irradiance Cache

- Si calcola l'irradianza dei *path* solo in alcuni punti della scena in spazio oggetto assumendo che:
  - BRDF diffusivo o a bassa frequenza
  - Geometria *smooth*; bassa frequenza
- Per il resto della scena l'irradianza viene calcolata usando interpolazione
- Punti di cache:
  - Dove la geometria cambia; spigoli e lati
  - Dove l'illuminazione cambia; piano a punti distanti

Irradiance Cache: Esempio



Punti della Cache



Rendering Finale

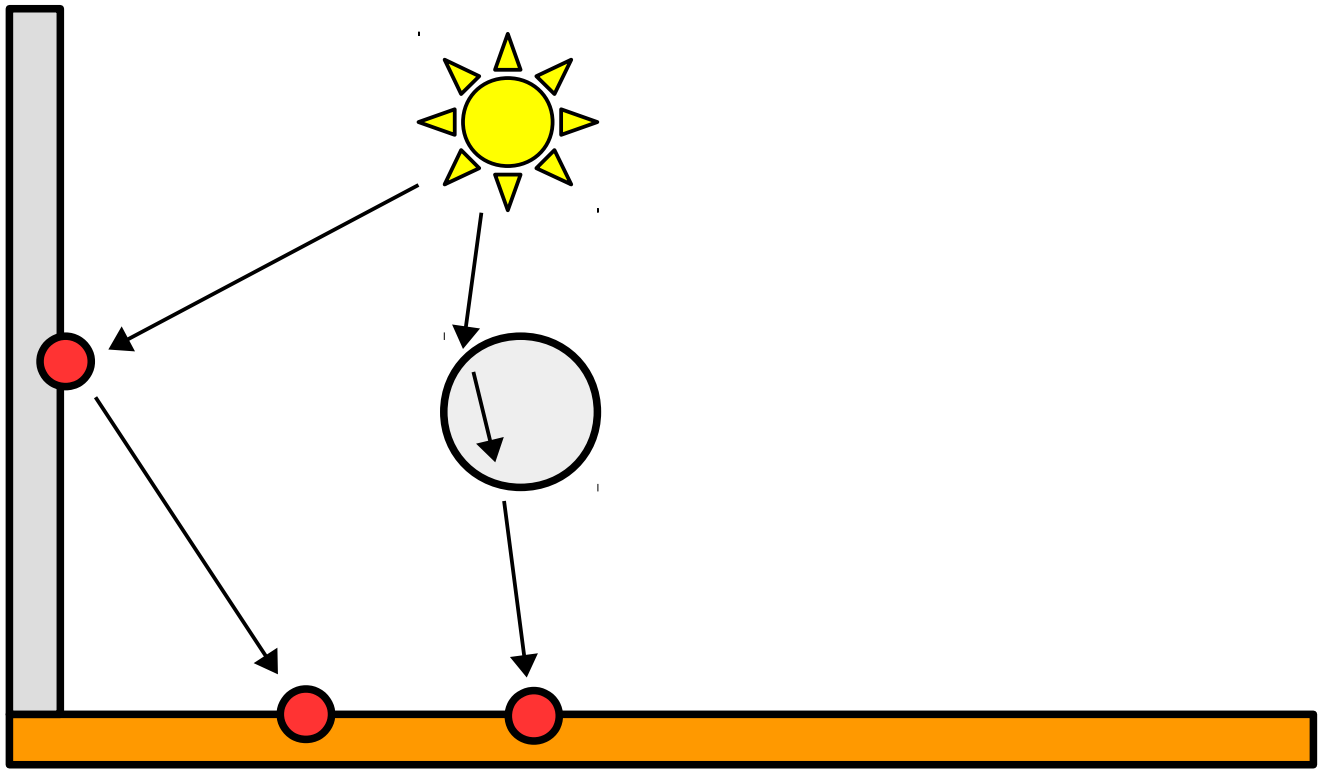
Photon Mapping

- Un algoritmo a due passate:
  - **Prima passata o Photon Tracing**: si lanciano delle particelle, dette fotoni, che partono dalla luce e rimbalzano nella scena finché non vengono assorbite da un materiale
  - **Seconda Passata o Density Evaluation**: viene eseguito il Ray-Tracing classico; quando si colpisce una superficie diffusa si calcola la stima di densità dei fotoni anziché l'illuminazione diretta:

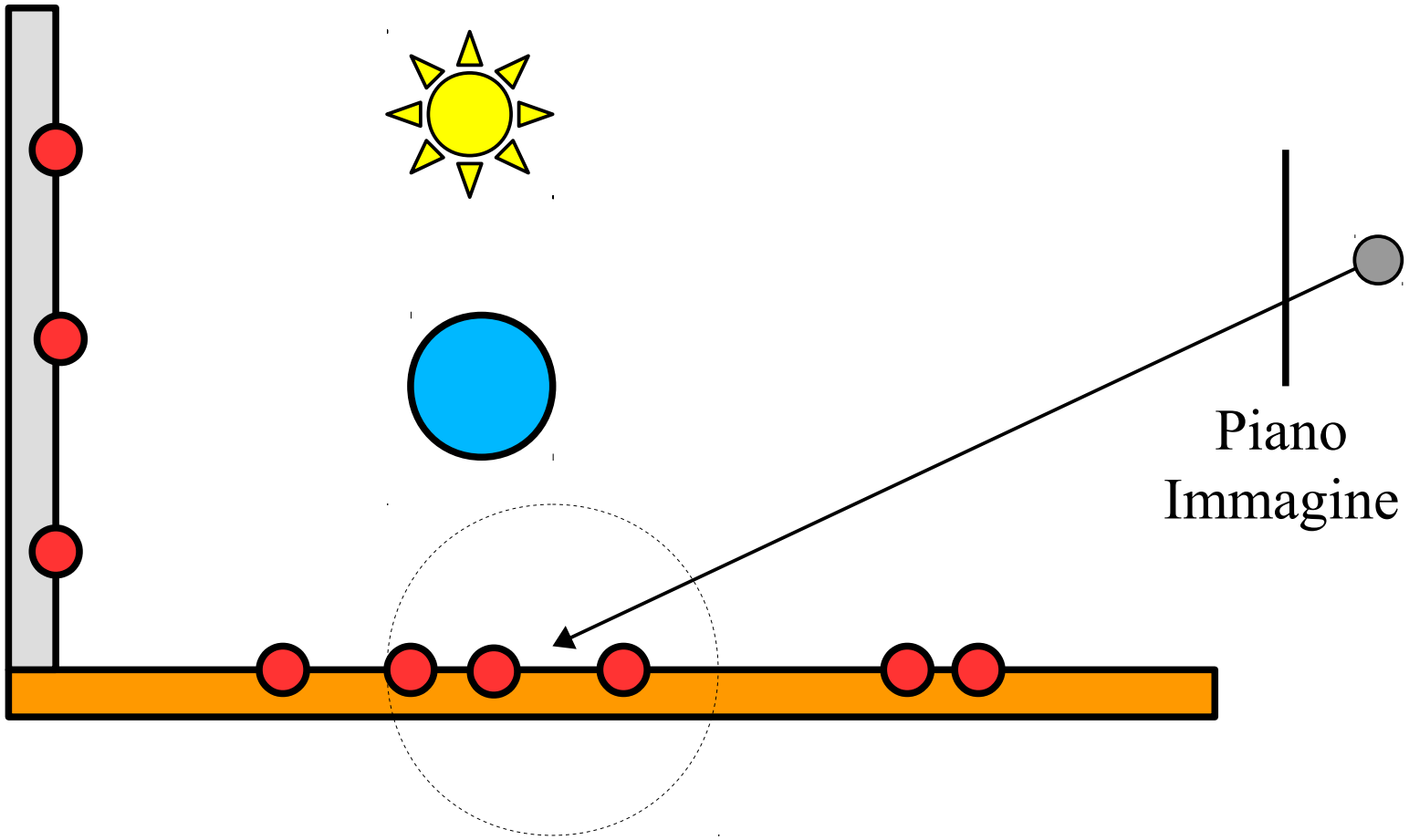
$$\frac{1}{\pi r^2}\sum_{i=1}^{N} f_r\left(\boldsymbol{x}, \vec{\omega}_i \rightarrow \vec{\omega}_o\right)\Phi_i$$

- Path Notation: L(S|D)*E

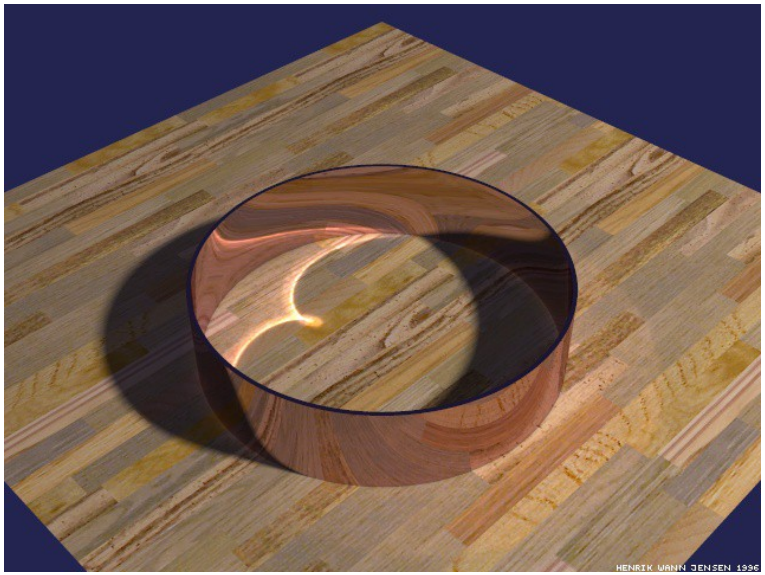
Photon Mapping: Photon Tracing



Photon Mapping: Density Estimation



Photon Mapping: Esempi



Photon Mapping: Conclusioni

- Il metodo riesce a simulare tutti i path di illuminazione, molto efficace per le caustiche (LSDE)

- Può essere velocizzato con le cache

- Non vi è rumore, **ma** è sostituito dal *bias*

- I parametri dipendono dalla scena!

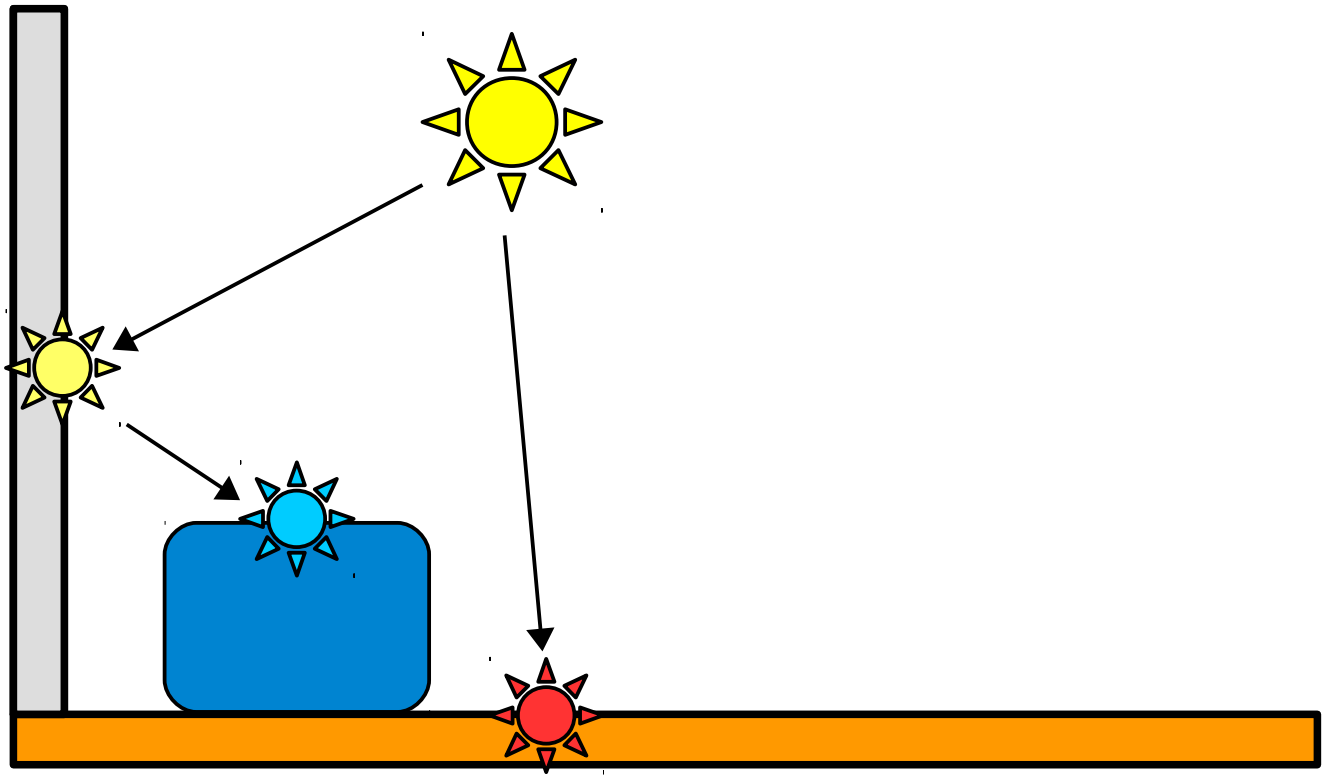
- Elevato consumo di memoria; l'ordine dei Gbyte per avere almeno 50M di fotoni

Instant Radiosity

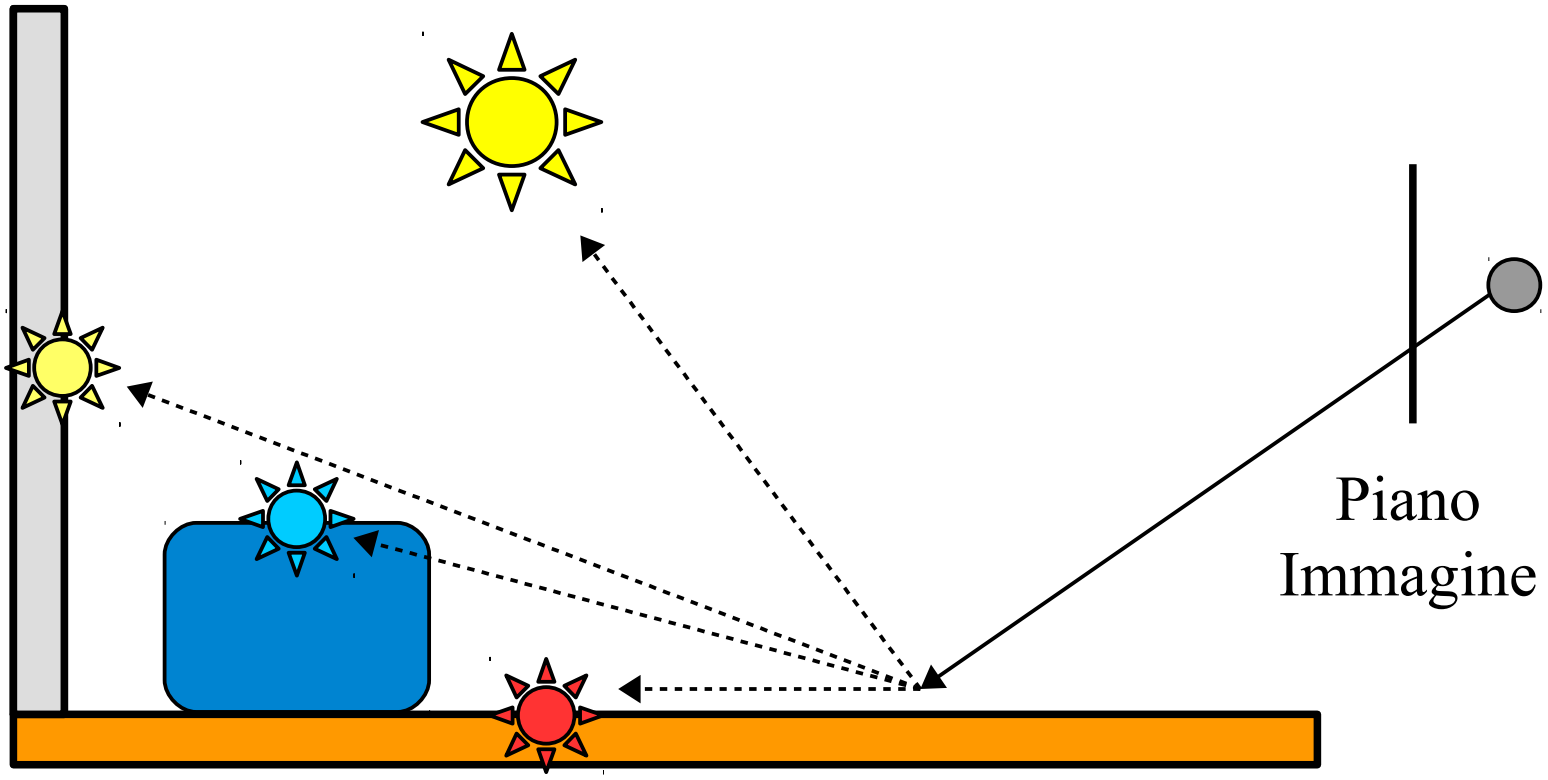
- Un algoritmo a due passate:
  - **Prima Passata o Virtual Light tracing**: simile al photon mapping, si lanciano delle luci virtuali dalla sorgente luminosa
  - **Seconda Passata**: è uguale al Whitted Ray-Tracing o al Distributed, nel quale si valutano anche le luci virtuali oltre a quelle reali

- Path Notation: L(S|D)*E

Instant Radiosity: Tracing



Instant Radiosity:
Direct and Indirect Lighting



Instant Radiosity: Esempio

Instant Radiosity: Conclusioni

- L'algoritmo è veloce e può essere:
  - Implementato su GPU
  - Usato con cache
  - Produce risultati convincenti
- L'algoritmo tuttavia non riproduce bene:
  - BRDF ad alta frequenza
  - Caustiche
- Inoltre l'algoritmo può produrre aliasing se si generano pochi VPL:
  - Almeno 256 VPL per una stanza chiusa!
- In questo caso un numero elevato di VPL è necessario