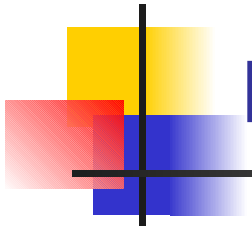
IMAGE BASED LIGHTING

di Francesco Banterle

f_banty@yahoo.it

http://www.geocities.com/frabante



Ringraziamenti

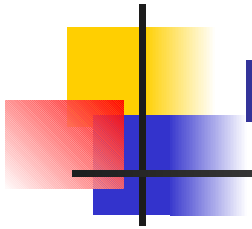
- Ringrazio il prof. Paul Debevec per avermi permesso di usare i lightprobe presenti sul sito:

  http://www.debevec.org/probes/

- Ringrazio il prof. Ravi Ramamoorthi per le delucidazioni sul metodo con le armoniche sferiche.

- Ringrazio il prof. Andrea Fusiello per le correzioni al tutorial "Introduzione all'Image Based Lighting".

- Ringrazio Davide Pasca, Davide Pirola, Emanuele Salvucci, Marco Corbetta, Nicola Ferruzzi, Sebastiano Mandalà e tutti gli altri di Playfields.net per le utili discussioni riguardanti l'IBL.

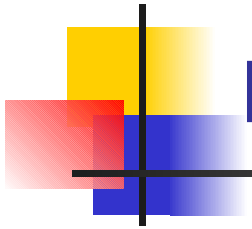


Introduzione

- Con il termine Image Based Lighting (IBL) si intende una serie di tecniche e algoritmi per l'illuminazione di oggetti sintetici utilizzando le informazioni delle fonti luminose presenti in fotografie o rendering.

- Le origini dell'IBL risalgono al 1976 quando Blinn e Newell introdussero l'envorinmental mapping per simulare riflessioni di oggetti sintetici. Negli ultimi si sono avuti grandi progressi grazie al lavoro di Paul Debevec.

- L'IBL negli ultimi anni si è sviluppato soprattutto grazie all'altissima resa realistica rispetto ai costi di computazione. Viene utilizzato in vari campi tra cui:
  - Computer Graphics
  - Realtà Potenziata
  - Film
  - Videogiochi

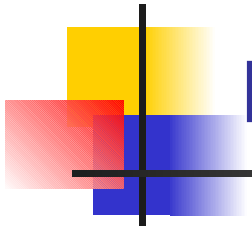


Low Dynamic Range Images

- Le Low Dynamic Range Images (LDRI) sono le consuete immagini digitali (Jpeg, Bmp, Tiff, etc..), nelle quali i valori di intensità dei pixel sono trattati come numeri naturali quantizzati ad 8-bit.

- Le LDRI sono state usate nelle prime fasi di sviluppo dell'IBL. Sono poi state abbandonate, in quanto non consentono di catturare tutta la gamma di luminosità del mondo reale. Infatti i valori dei singoli canali essendo compresi tra 0-255 non consentono di distinguere valori più elevati.

  Esempio: l'intensità di una finestra fortemente illuminata non si distingue dall'intensità del sole.



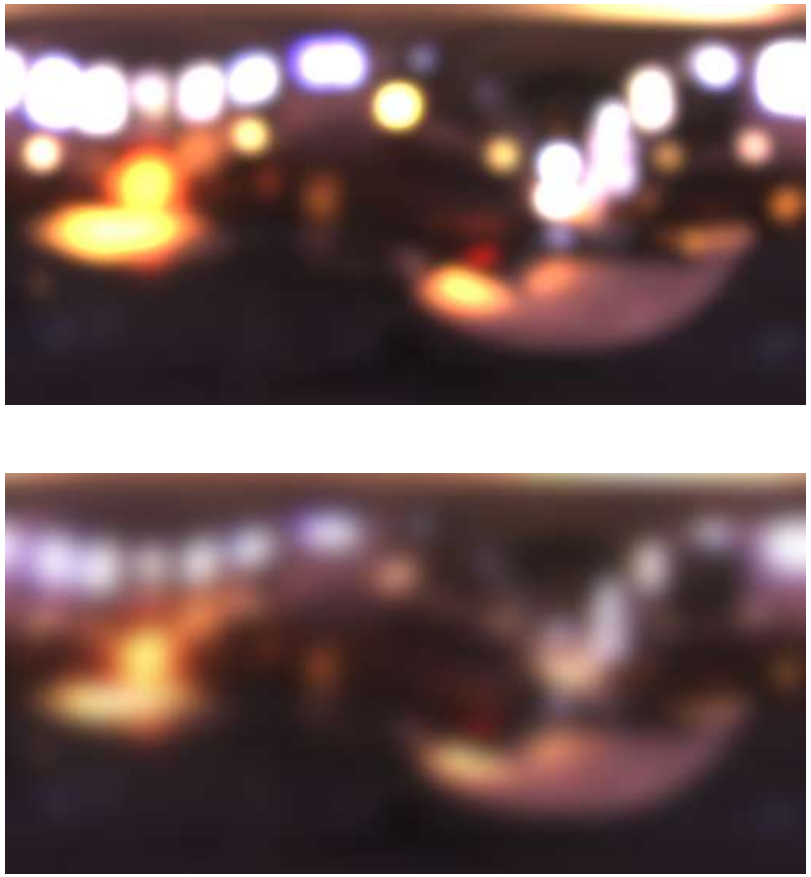
High Dynamic Range Images

- Le High Dynamic Range Images (HDRI), sono immagini che catturano tutta la gamma di luminosità del mondo reale, dal buio 0.0, alla luce del sole 100.000 .
- Per rendere possibile questo, per ogni canale l'informazione viene codificata tramite valori reali (in genere float, 32-bit) positivi al posto di valori naurali (di solito unsigned char, 8-bit).
- L'uso di numeri reali rende le HDRI molto pesanti per quanto riguardo lo spazio occupato, infatti occupano quattro volte la dimensione delle LDRI.
- In ambito real-time vengono ancora usate le LDRI dato che occupano poca memoria, inoltre il filtraggio e il blending di HDRI nella generazione nv3x e r3xx non è via hardware. Però giochi della prossima generazione come Unreal 3 e Half-Life 2 promettono l'uso estensivo di HDRI.

Confronto tra LDRI e HDRI

- Per rendere più chiare le differenze tra HDRI e LDRI seguiranno degli esempi in cui le stesse operazioni vengono effettuate sia su HDRI (in alto) che su LDRI (in basso) sulla famosa immagine della Cattedrale della Grazia (in coordinate longitudine/latitudine) catturata da Paul Debevec:

HDRI vs LDRI: Gaussian blur



HDRI vs LDRI: Variazione di esposizione

HDRI vs LDRI: Horizontal blur

Costruzione di HDRI (1)

- Le HDRI si costruiscono, partendo da due o più immagini a differenti esposizioni (utilizzando dei filtri sull'obiettivo, o variando l'apertura del diaframma o la velocità dell'otturatore), per esempio a 2s, 1s, 0.5s, 0.25s e così via fino a limiti consentiti dallo strumento fotografico.
- Una volta raccolte le fotografie, si devono comporre per ottenere la HDRI, che non è altro che una mappa di radianza;
- Ogni fotografia, presa ad una determinata esposizione per ogni pixel, non è altro che una funzione della radianza del pixel moltiplicata per l'esposizione:

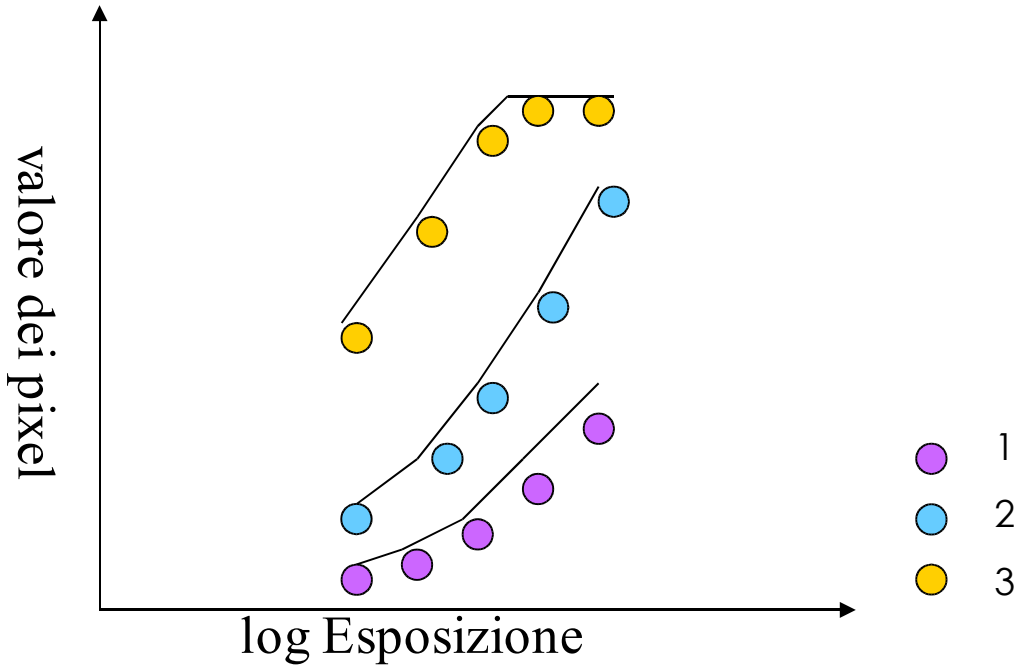
$$\text{pixel} = f\left(E \cdot \Delta t \right) \qquad (1)$$

Costruzione di HDRI (2)

- Il problema è calcolare la funzione *f* incognita.
- *"f si può assumere uguale per ogni pixel"*.

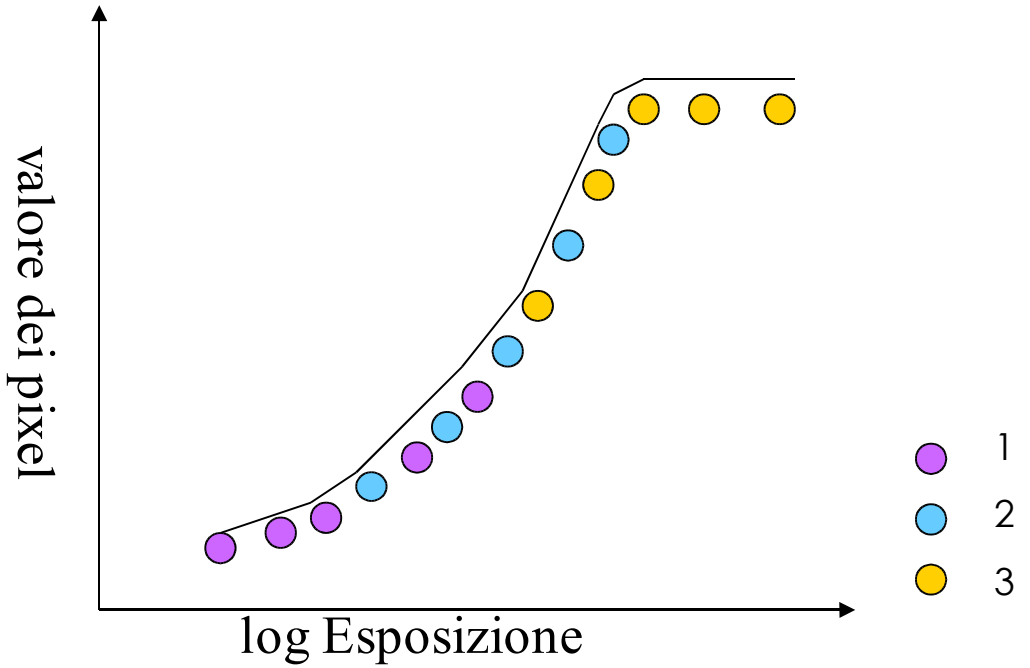
Costruzione di HDRI (3)

- Se calcoliamo g per alcuni campioni otteniamo, per ognuno una g diversa che si discosta.



Costruzione di HDRI (4)

- Dobbiamo fare degli aggiustamenti alle curve, in modo da ottenerne una sola:



Costruzione di HDRI (5)

- Dobbiamo calcolarci l'inversa di f:

$$f^{-1}(\text{pixel}) = E \cdot \Delta t \quad (2)$$

- Introducendo i logaritmi otteniamo:

$$\log f^{-1}(\text{pixel}) = \log E + \log \Delta t \quad (3)$$

$$g(\text{pixel}) = \log E + \log \Delta t \quad (4)$$

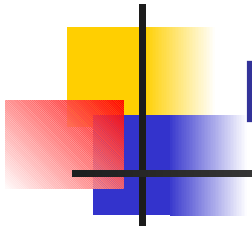
$$\log E = g(\text{pixel}) - \log \Delta t \quad (5)$$

- L'aggiustamento si effettua minimizzando la seguente funzione obiettivo:

$$\mathcal{O} = \sum_{i=1}^{N} \sum_{j=1}^{P} [\, g(\text{pixel}) - \log \Delta t - \log E \,]^2 \quad (6)$$

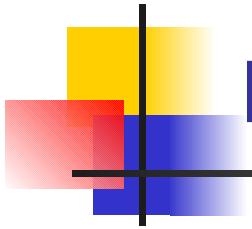
dove N è l'indice del pixel e P il numero di fotografie.

- La minimizzazione di $\mathcal{O}$ si può risolvere facilmente riconducendosi al problema dei minimi quadrati



HDRI: IBL Ready?

- Non tutte le HDRI sono adatte per l'IBL, nello specifico servono delle immagini che siano in grado di rappresentare tutto lo spazio circostante al punto di acquisizione.
- Ci sono vari modi per ottenere una HDRI con tale caratteristica:
  - Si scattano sei foto con grandangolo della scena (è un processo lento perché necessita di ritocchi con Photoshop per assemblare bene le immagini);
  - Si scattano due foto a una pallina totalmente riflettente (tipo le palline degli alberi di Natale), una foto per emisfero;
  - Si scattano due foto (in due direzioni opposte) utilizzando un obiettivo ad occhio di pesce (fisheye);
  - Si scattano foto utilizzando macchine fotografiche particolari (tipo quelle di Panoscan o Spheron), il processo è totalmente automatico ma il *costo delle macchine è molto elevato*.
- Una volta ottenute le immagini che rappresentano tutto l'ambiente posso essere convertite con HDRShop in formato Longitudine/Latitudine o in Mappe Angolari.



Formati HDRI

- I principali formati delle HDRI sono:
  - PFM: Portable Float Map, un formato non compresso, è composto da un header semplice seguito dai dati organizzati come un file .raw.
  - OpenEXR: formato sviluppato da ILM per le sue produzioni cinematografiche. Il formato utilizza il tipo di dato half, ossia un numero a virgola mobile a 16-bit ottenuto dimezzando la specifica IEEE. Il formato permette compressione con wavelet con un risparmio del 40%.
  - RGBE: formato sviluppato da Greg Ward per Radiance. In sintesi i canali RGB sono salvati come half, però l'esponente è comune. Questo metodo porta perdita di rappresentazione, ma consente una dimensione del file più compatta.
  - PIXAR LOG TIFF: E' il formato di Pixar Studios. Ogni canale occupa 11-bit, dove il valore è in forma logaritmica.

Formati HDRI: RGBE code/decode

- Codifica RGBE:
  - Calcolo del massimo tra R, G, B. Se il massimo è minore di una soglia ε (1^{-32}) la quadrupla RGBE è impostata a zero;
  - Calcolo l'esponente (in base 2)del massimo e la mantissa (frexp). E porto la mantissa nell'intervallo $[\,0, 255\,]$;
  - Trasformo i valori R, G, B nella codifica RGBE moltiplicandoli per il valore della mantissa;
  - Aggiungo all'esponente E, 128 (per evitare valori negativi).
- Decodifica RGBE:
  - Se tutte le componenti sono zero si ritorna un vettore nullo;
  - Altrimenti per ogni componente si effettua il calcolo (ldexp):

    $x \cdot 2^{E-128}$. Dove E è l'esponente comune e x è una delle componenti R, G, B in codifica RGBE.

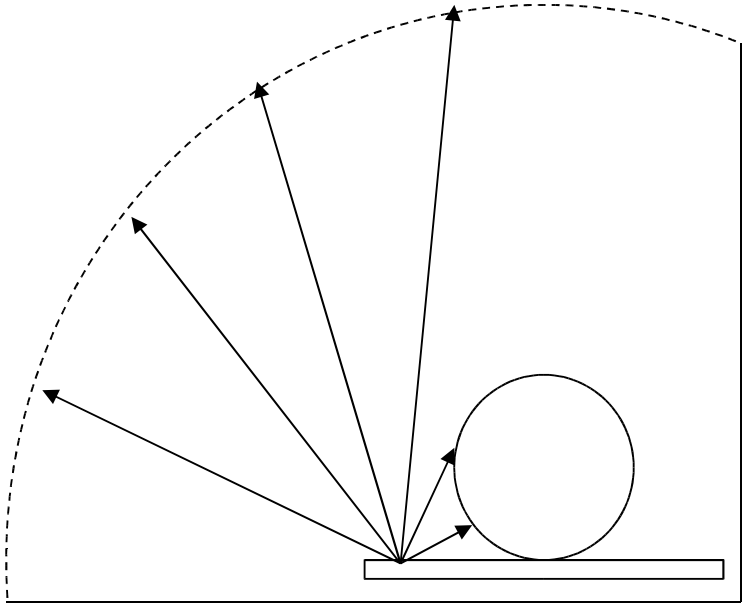
Il calcolo della luce: introduzione

- Il calcolo dell'intensità luminosa, cioè l'irradianza E, nel punto x con normale $\boldsymbol{n}$, si effettua svolgente il seguente integrale:

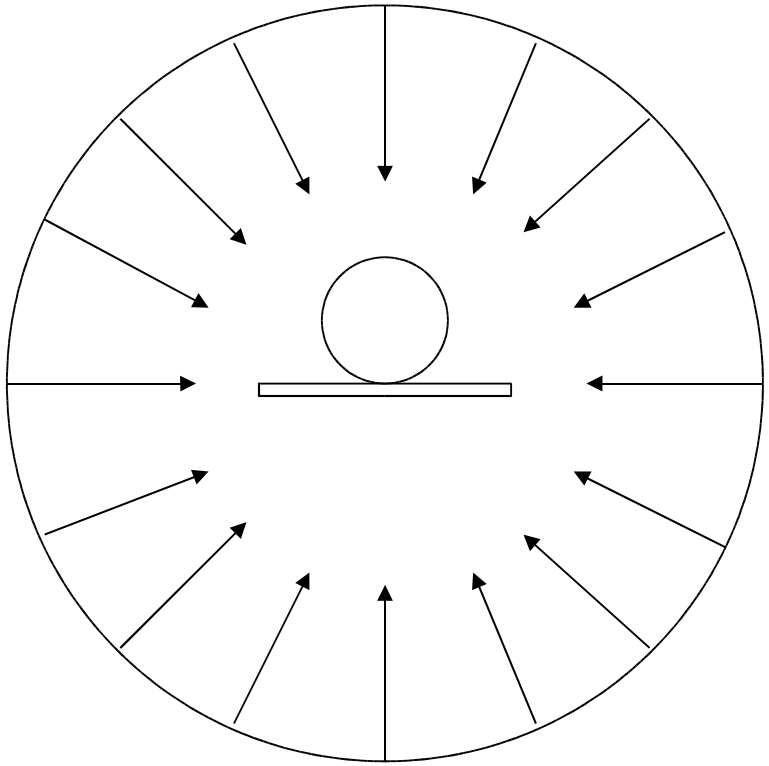
$$E(x) = \int_{\Omega 2\pi} L_i(\boldsymbol{\omega}) S(x, \boldsymbol{\omega})(\boldsymbol{n} \cdot \boldsymbol{\omega}) d\boldsymbol{\omega} \quad (8)$$

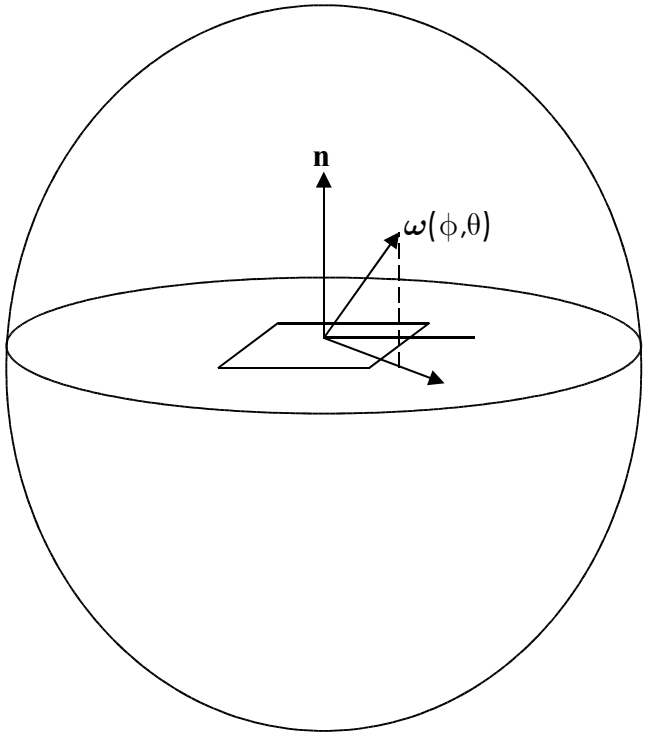
- Dove L_i rappresenta la radianza incidente, che nel caso dell'IBL è una HDRI. La funzione S o di visibilità, è una funzione booleana che ritorna 1 se un raggio partendo dal punto x in direzione $\boldsymbol{\omega}$ è privo di occlusori, 0 altrimenti.
- L'integrale si risolve facilmente tramite l'integrazione di Monte Carlo (MC).
- Il problema può essere inquadrato in un'altra ottica: tante luci direzionali quante i pixel dell'HDRI e di intensità pari all'intensità dei rispettivi pixel dell'HDRI.

Il calcolo della luce: visualizzazione (1)



Il calcolo della luce: visualizzazione (2)





Il calcolo della luce: discretizzazione

- L'integrale per il calcolo di *E(x)*, in genere viene discretizzato nella seguente forma :

$$E(x) = \frac{\pi}{MN} \sum_{j=1}^{M} \sum_{i=1}^{N} L_i(\theta_j, \phi_i) \quad (9)$$

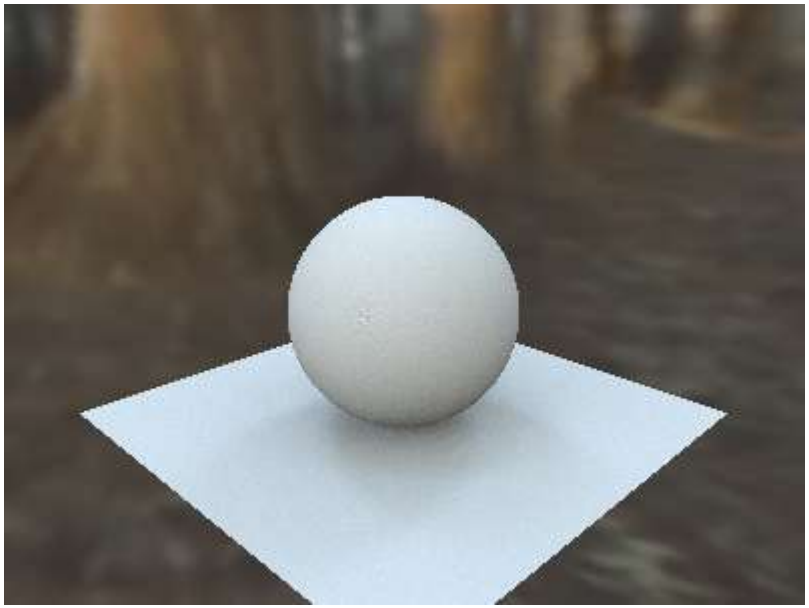
dove

$$\theta_j = sin^{-1}(((j - \xi_1)/M)^{0.5})$$

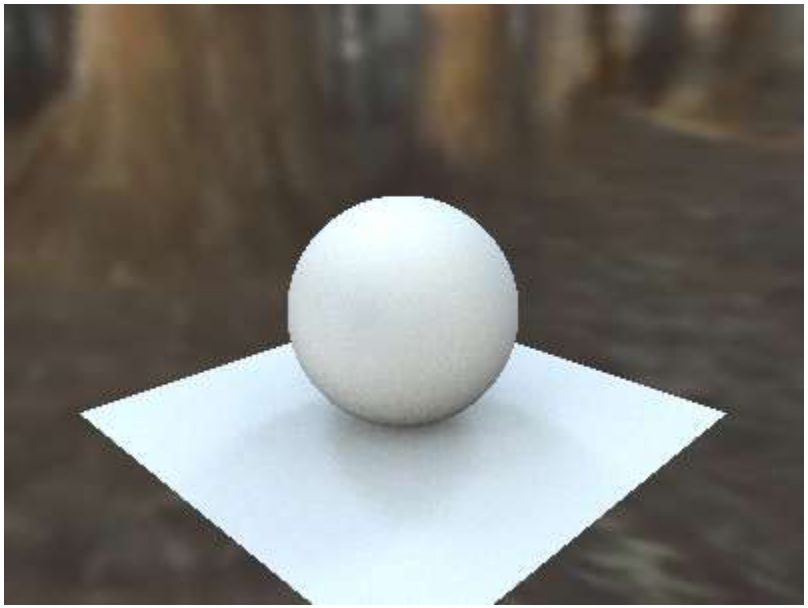
$$\phi_i = 2\pi((i - \xi_2)/N)$$

$$\xi_1 \in [0, 1] \text{ e } \xi_2 \in [0, 1]$$

Il calcolo della luce:discretizzazione(3)



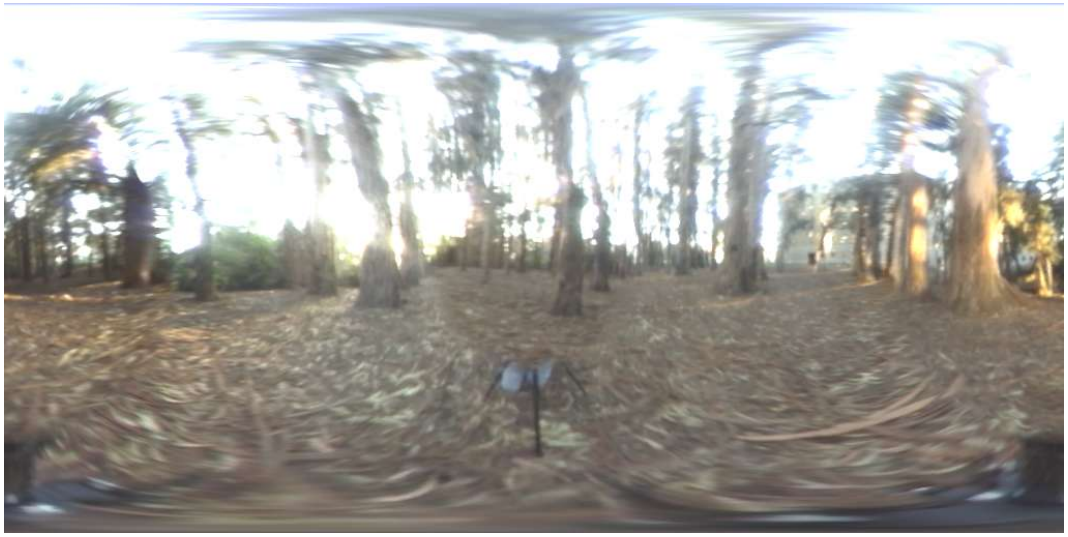
Integrazione con la *(9)*



Integrazione considerando ogni pixel una luce direzionale

Integrazione: tempi/campioni

- L'integrazione con Monte Carlo (utilizzando la stratificazione dei campioni per migliorare la qualità) è molto lenta nel raggiungere la convergenza ed è rumorosa; questo è dovuto principalmente alla presenza di regioni a forte intensità nella HDRI. Per dimostrare la scarsa bontà del metodo verranno presentati l'integrazione a vari campioni/pixel con i relativi tempi di computazione. Le immagini sono renderizzata a 400x300 utilizzando la HDRI di RNL:

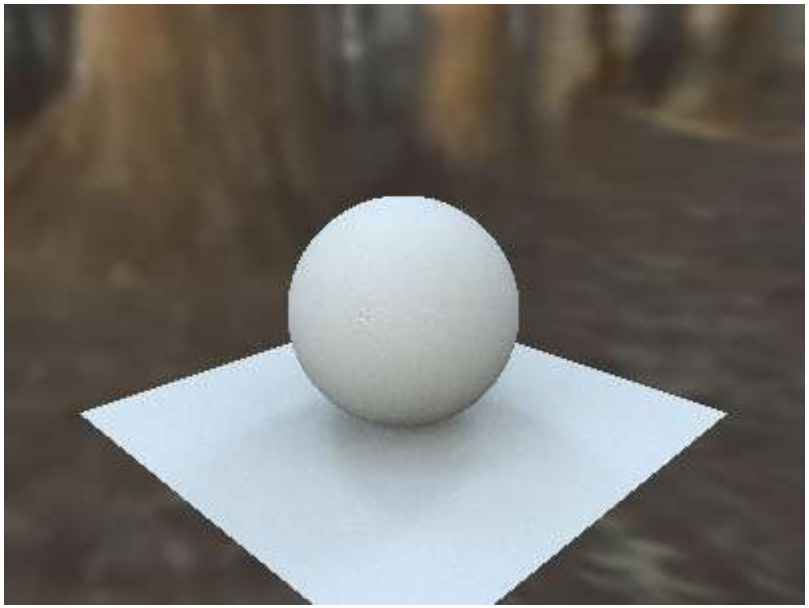


Esempi: MonteCarlo

Integrazione con 1 campioni, 0.167 sec

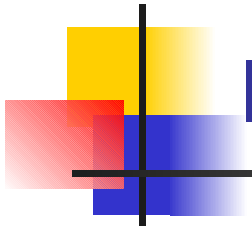
Integrazione con 100 campioni, 4.579 sec

Esempi: MonteCarlo



Integrazione con 2500 campioni, 95.322 sec

Zoom, il rumore non è scomparso

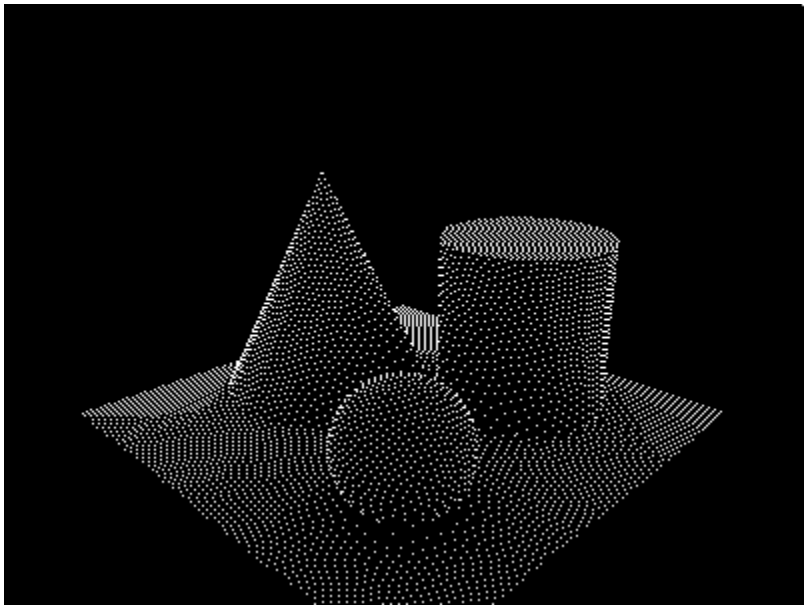


Il calcolo della luce: accelerazione

- L'accelerazione della convergenza può essere effettuata in vari modi, ognuno con i suoi pregi e difetti:
  - Integrazione Iterativa:
    - Si generano campioni finché non si raggiunge una precisione desiderata. Serve a poco e a niente
  - Irradiance Cache (Ward 88, Ward 92) :
    - Si calcola l'illuminazione in alcuni punti utilizzando tanti campioni, per i restanti punti si effettua interpolazione. Per migliorare l'interpolazione se si utilizzano i gradienti dell'irradianza;
    - Il metodo è molto veloce (soprattutto se i campioni sono in una gerarchia, octree), ma funziona solo con BRDF diffusi;
  - Armoniche Sferiche;
  - Importance Sampling;
  - Structured Importance Sampling;

Esempi: Irradiance Cache

Integrazione 2500 campioni/pixel, 15.654 sec



Punti di integrazione

Armoniche Sferiche

- Nel 1984 Greene propose di precalcolare ogni direzione ω il valore di $E(x)$. Visto che nel precalcolo non si ha a disposizione la posizione del punto x la *(8)* viene semplificata eliminando il termine di visibilità $S(\omega)$:

$$E(x) = \int_{\Omega 2\pi} L_i(\omega)(n \cdot \omega) d\omega \quad (11)$$

- Il metodo proposto da Greene è molto oneroso, infatti ha una complessità di $O(n^4)$.
- Nel 2001 Ravi Ramamoorthi e Pat Hanrahan proposero di utilizzare le armoniche sferiche per ridurre la complessità portandola a $O(n^2)$, con un errore massimo del 9%.
- Il metodo si presta molto bene ad un'implementazione hardware utilizzando la Shader Technology. In assenza di shader le armoniche possono essere tabulate su una cubemap o una envmap.

Armoniche Sferiche: Filtraggio (1)

- Dato che $\boldsymbol{n}$ e $\boldsymbol{\omega}$ sono dei versori e $E(n)$ e $L_i(\boldsymbol{\omega})$ sono parametrizzabili in coordinate polari (θ, ϕ) sulla sfera unitaria, possiamo rappresentare E e L tramite espansioni di armoniche sferiche:

$$E(\theta, \phi) = \Sigma_{l,m} E_{l,m} Y_{l,m}(\theta, \phi) \qquad (12)$$

$$L_i(\theta, \phi) = \Sigma_{l,m} L_{l,m} Y_{l,m}(\theta, \phi) \qquad (13)$$

- A questo punto definiamo A come $\boldsymbol{n}\cdot\boldsymbol{\omega}$; la sua espansione in armoniche sferiche ha con i solo coefficienti A_l, dato che non ha nessuna dipendenza dall'azimuth (e quindi m=0).

$$A(\theta) = \Sigma_l \mathrm{A}_l Y_{l,0}(\theta, \phi) \qquad (14)$$

- Dalla *(14)* ragionando sulla *(12)* e sulla definizione di armoniche sferiche si ricava che:

$$E_{l,m} = (4\pi/(2l+1))^{0.5} A_l L_{l,m} \qquad (15)$$

Armoniche Sferiche: Filtraggio (2)

- A questo punto della $E(\theta, \phi)$ ci rimangono incogniti solo i coefficienti di $L_{l,m}$. Per calcolarli basta integrare L_i con le funzioni base dell'armonica sferica, cioè con $Y_{l,m}$.

$$L_{lm} = \int_{\Omega 4\pi} L_i(\omega) Y_{l,m}(\omega) d\omega \quad (16)$$

- Le funzioni di base per Y_{lm} per i primi nove coefficienti sono:

$$(x, y, z) = (\sin(\theta)\cos(\phi); \cos(\theta); \sin(\theta)\sin(\phi))$$

$$Y_{00}(\theta, \phi) = 0.282095$$

$$(Y_{11}; Y_{10}; Y_{1-1})(\theta, \phi) = 0.488603\ (x; z; y)$$

$$(Y_{21}; Y_{2-1}; Y_{2-2})(\theta, \phi) = 1.092548\ (xz; yz; xy)$$

$$Y_{20}(\theta, \phi) = 0.315392\ (3z^2 - 1)$$

$$Y_{22}(\theta, \phi) = 0.546274\ (x^2 - y^2)$$

Armoniche Sferiche: valutazione

- Una volta calcolati i coefficienti $L_{l,m}$ si può valutare l'irradianza organizzando meglio la *(15)* sostituita nella *(12)*:

$$E(\mathbf{n}) = \mathbf{n}^t M \mathbf{n} \qquad (17)$$

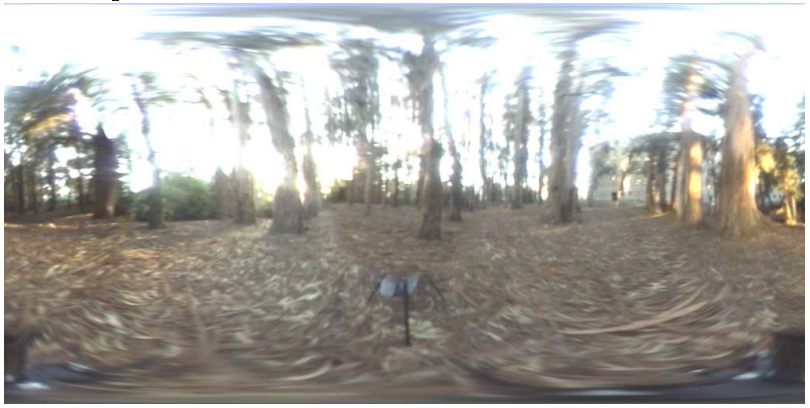
dove

$$M = \begin{bmatrix} c_1 L_{22} & c_1 L_{2\text{-}2} & c_1 L_{21} & c_2 L_{11} \\ c_1 L_{2\text{-}2} & -c_1 L_{22} & c_1 L_{2\text{-}1} & c_2 L_{1\text{-}1} \\ c_1 L_{21} & c_1 L_{2\text{-}1} & c_3 L_{20} & c_2 L_{10} \\ c_2 L_{21} & c_2 L_{1\text{-}1} & c_2 L_{10} & c_4 L_{00} - c_5 L_{20} \end{bmatrix}$$

$c_1 = 0.429043$ $c_2 = 0.511664$

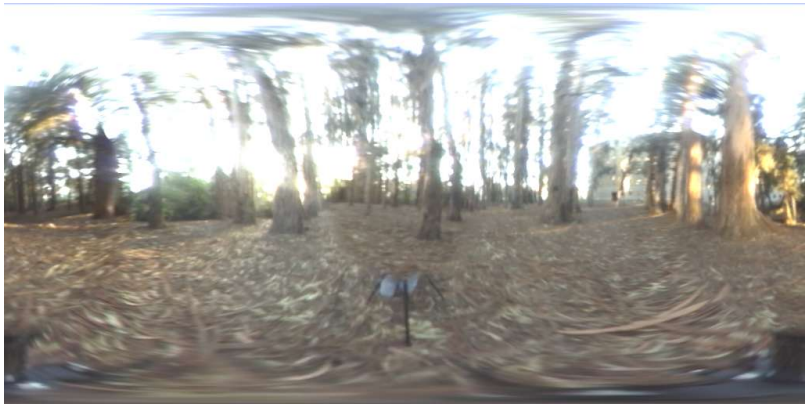
$c_3 = 0.743125$ $c_4 = 0.886227$ $c_5 = 0.247708$

Esempi SH: HDRI vs LDRI



HDRI prima del filtraggio

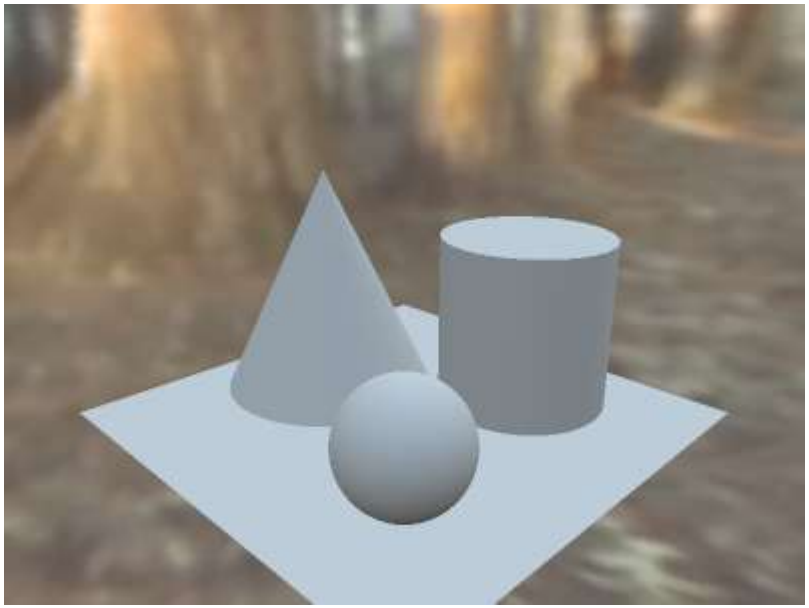
HDRI dopo il filtraggio



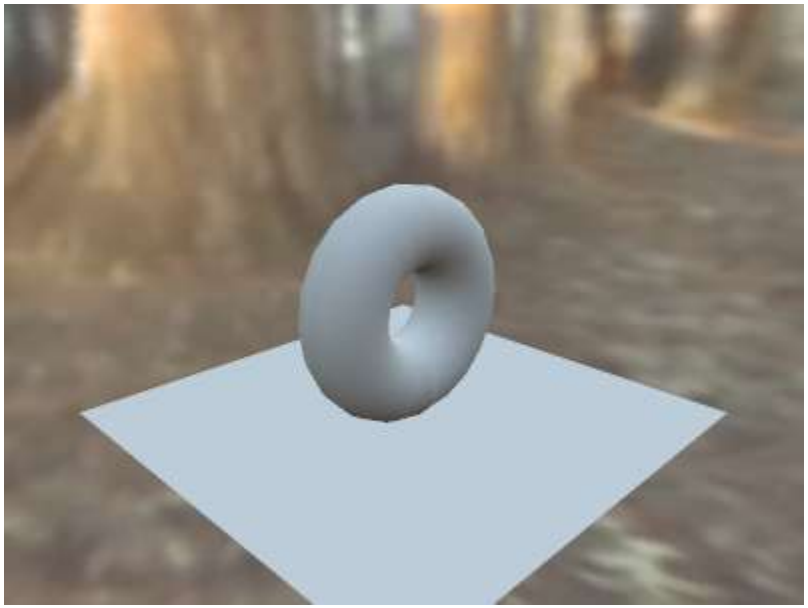
LDRI prima del filtraggio

LDRI dopo il filtraggio

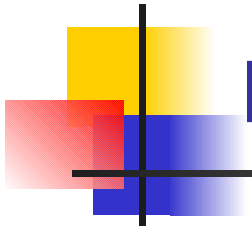
Esempi: SH Rendering



Armoniche sferiche 0.572 sec



Armoniche sferiche 0.835 sec



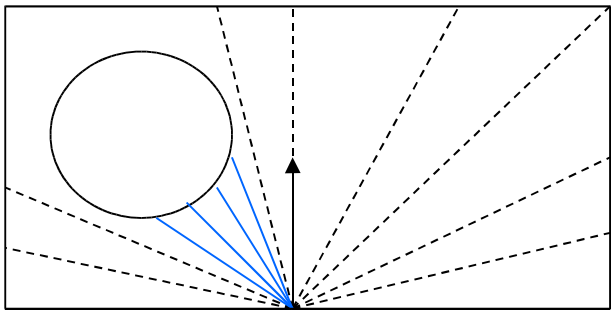
L'aggiunta dell'Occlusion Ambient

- Il filtraggio di HDRI per il calcolo dei coefficienti delle armoniche sferiche non tiene conto dell'occlusione da parte degli oggetti presenti nella scena.
- Un metodo per rimediare a ciò è la combinazione del valore calcolato con le armoniche sferiche con il termine di Occlusion Ambient, ossia moltiplicandoli tra loro.
- Il metodo è un'euristica, teoricamente sbagliata, tuttavia il risultato è abbastanza convincente.
- Questa tecnica è consigliata per visualizzazioni veloci o in ambito real-time.

Occlusion Ambient

- L'occlusion ambient si calcola facendo il casting di un determinato numero di raggi campionati nell'emisfero definito dalla normale del punto. Il fattore di occlusion si determina dividendo il numero di raggi liberi (cioè non occlusi) per il numero dei raggi totali lanciati.
- Si può calcolare in real-time accumulando lo zbuffer di render effettuati in punti di un emisfero.

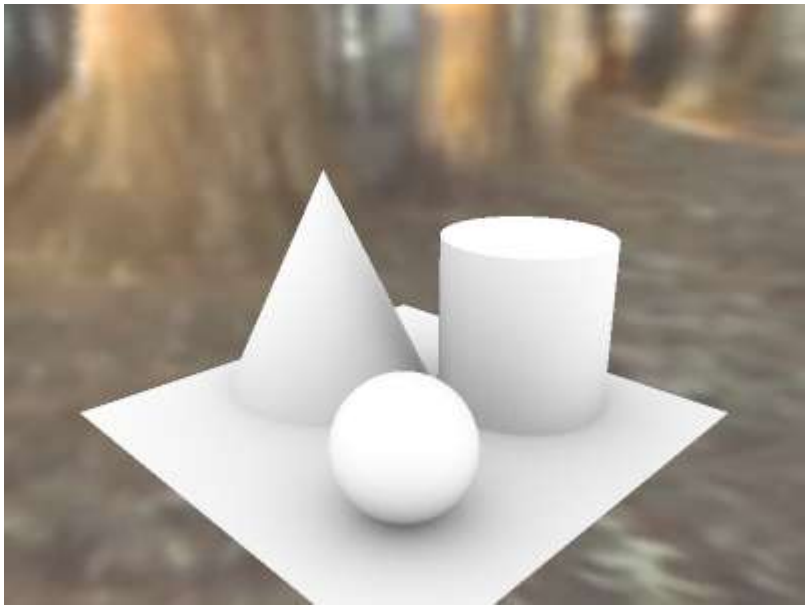
$$Occ = \frac{|\ Liberi\ |}{|\ Raggi\ |} \qquad (18)$$



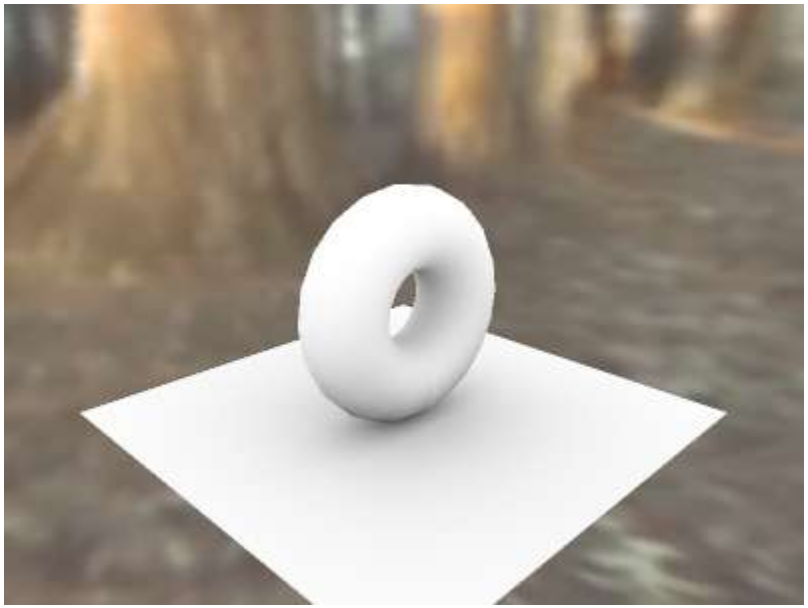
Liberi

Occlusi

Occlusion Ambient: esempi

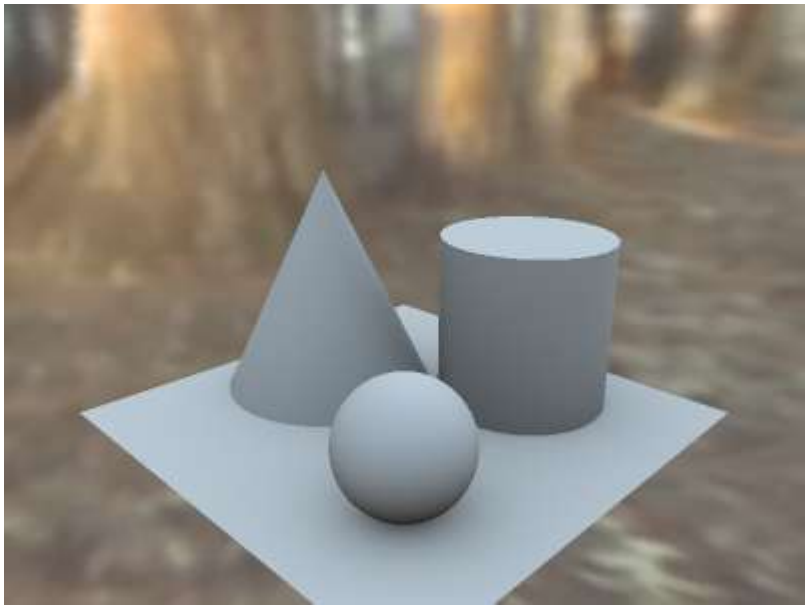


AA, 256 campioni per pixel, 21.198 sec

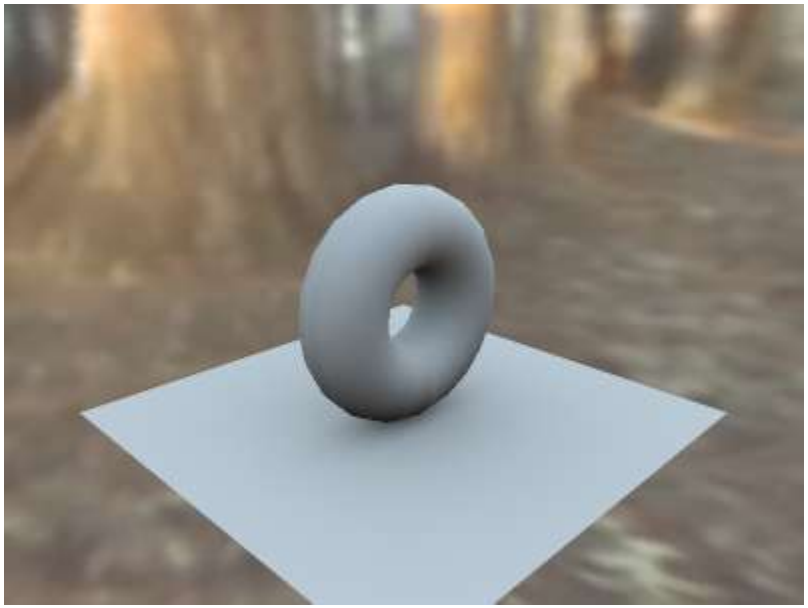


AA, 256 campioni per pixel 98.95 sec

Esempi: Armoniche sferiche + OA

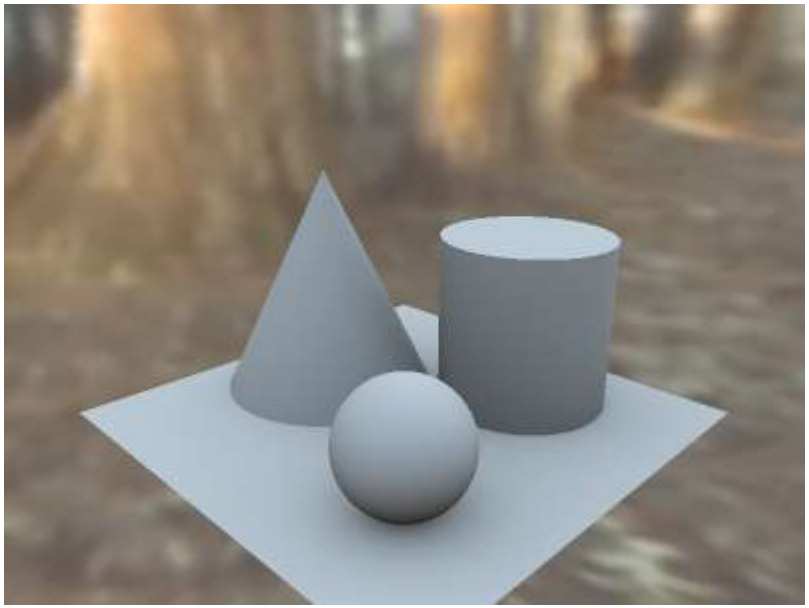


Armoniche Sferiche + OA

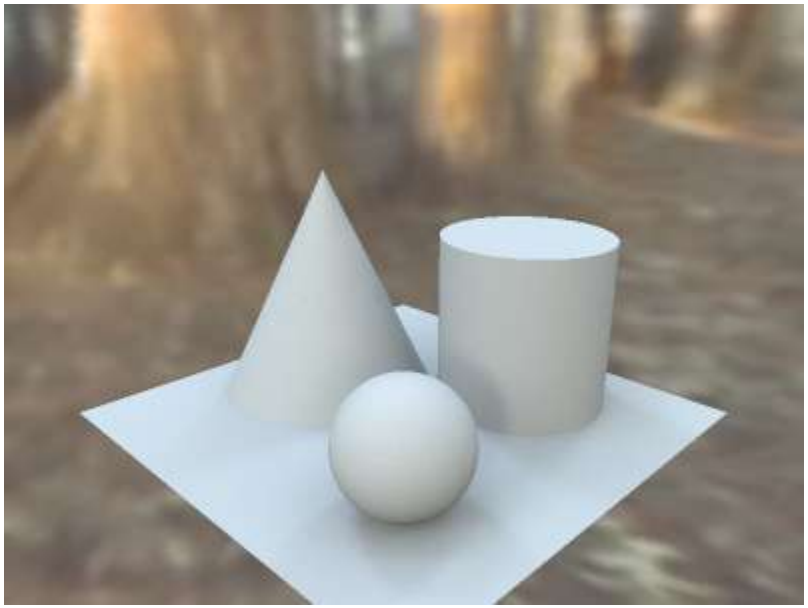


Armoniche Sferiche + OA

Differenze: SH+OA vs MonteCarlo

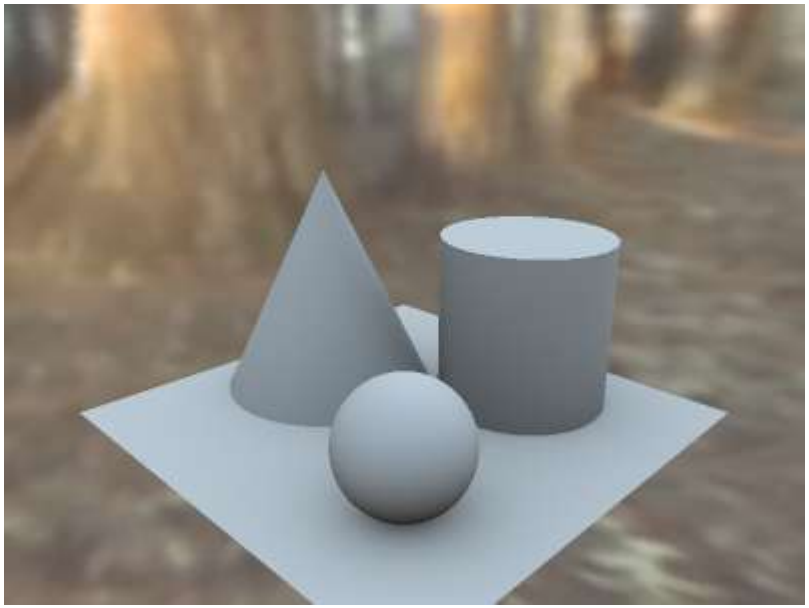


Armoniche Sferiche + OA

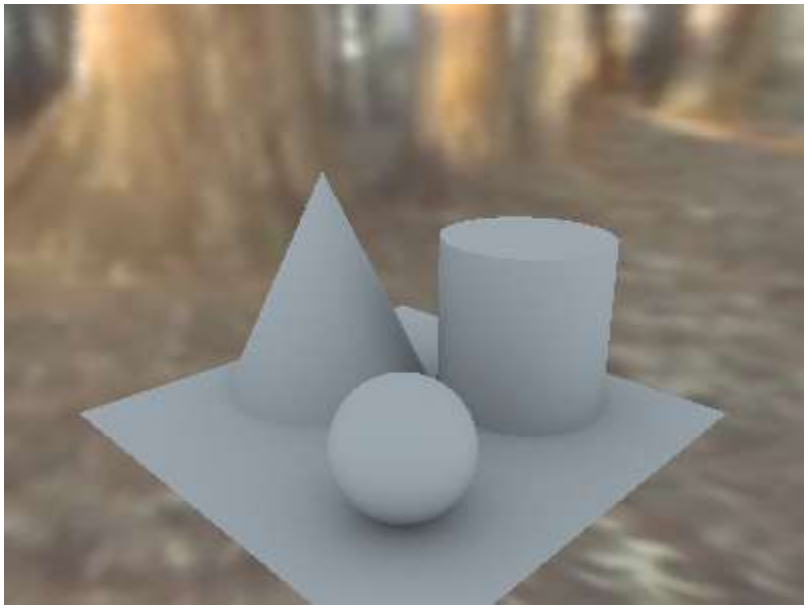


Monte Carlo

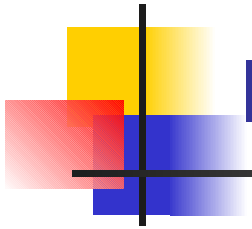
Differenze: SH+OA vs Media+OA



Armoniche Sferiche + OA

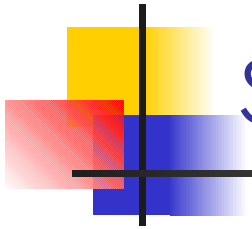


Media + OA



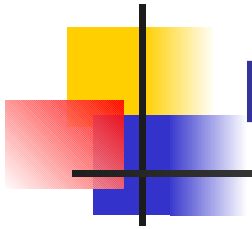
Importance Sampling

- L'Importance sampling, sfrutta il fatto che i pixel ad alto contenuto energetico contribuisco alla maggior parte dell'illuminazione della scena;
- Quindi è naturale scegliere con maggior probabilità questi pixel ad alta energia.
- L'Importance Sampling è il primo metodo a sfruttare le peculiari caratteristiche del problema;
- L'assegnamento della probabilità ai pixel in genere si effettua tramite l'istogramma. Ma le HDRI non hanno l'istogramma. In genere si procede esplorando l'emisfero, campionando le zone a più alta energia. Altrimenti si può cercare di definire una nuova funzione che funzioni da istogramma.
- Il miglioramento della convergenza è buono, seppur il rumore rimanga.



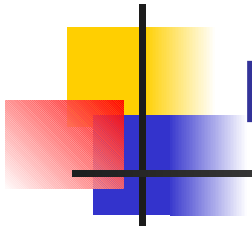
Structured Importance Sampling

- L'algoritmo è stato introdotto da Sameer Agarwal, Ravi Ramamoorthi, Serge Belongie, e Henrik Wann Jensen nel 2003.
- E' un'evoluzione dell'Importance Sampling; ora si strutturano le informazioni presenti nell'HDRI. E' la soluzione più veloce esistente per il raytracing, infatti raggiunge la convergenza con 4.7 raggi/pixel di integrazione in media per BRDF diffusi e glossy.
- E' suddiviso in due parti:
  - Nella prima parte si effettua una sogliatura gerarchica in base alla varianza dell'HDRI. Si calcolano le componenti connesse di ogni gerarchia, e per ogni componente connessa si determina il numero di campioni. Quindi utilizzando l'algoritmo di Hochbaum-Shmoys si partiziona ogni componente connessa in k (dove k è il numero dei campioni associati) nuove componenti di area uguale (strati).
  - Nella seconda parte, si valuta $E(x)$ prendendo una direzione casuale per ogni strato valutando l'illuminazione e il BRDF.



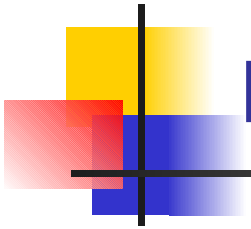
Links Utili

http://www.debevec.org/probes/
Una serie di HDRI di ottima qualità
http://www.debevec.org/hdrshop/
Il sito ufficiale di HDRShop
http://www.openexr.org/
Il sito ufficiale delle OpenEXR
http://www.daionet.gr.jp/~masa/rthdribl/index.html
La demo di Masaki Kawase di IBL in real-time.



Bibliografia (1)

“A ray tracing solution for diffuse interreflection”
Gregory Ward e Paul Heckbert. Siggraph ’88.
“Adaptive Sampling and Bias Estimation in Path Tracing”,
Rasmus Tamstorf and Henrik Wann Jensen in Rendering Techniques '97
"An Efficient Representation for Irradiance Environment Maps" .
di Ravi Ramamoorthi e Pat Hanrahan. Siggraph 2001.
“Efficient Illumination by High Dynamic Range Images”
di Thomas Kollig e Alexander Keller. Eurographics 2003.
“Environment mapping and other applications of world projections”.
di N. Greene .*IEEE Computer Graphics & Applications* 1986.
“Frequency Space Environment Map Rendering”
di Ravi Ramamoorthi e Pat Hanrahan. Siggraph 2003.
"High Dynamic Range and other Fun Shader Tricks".
di Simon Green. GDC 2002.
"High Dynamic Range Imaging".
Gregory Ward. 2001.
“HDRI and Image based Lighting course note”.
Paul Debevec, Gregory Ward, Rob Bogart, Dan Lemmon, e Frank Vitz. Siggraph 2003.



Bibliografia (2)

"Image based Modeling, Rendering, and Lighting in Fiat Lux"

di Paul Debevec. GDC 2002.

"Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments".

di Peter-Pike Sloan, Jan Kautz, and John Snyder. SIGGRAPH 2002.

"Real-Time High Dynamic Range Texture Mapping".

di Cohen Jonathan, Tchou Chris, Hawkins Tim, e Debevec Paul. Eurographics 2001.

"Recovering High Dynamic Range Radiance Maps from Photographs".

di Paul Debevec e Jitendra Malik. Siggraph 1997.

"Rendering Synthetic Objects into Real Scenes".

di Paul Debevec. Siggraph 1998.

"Spherical Harmonic Lighting: The Gritty Details".

di Robin Green. GDC 2002.

"Structured Importance Sampling of Environment Maps"

di Sameer Agarwal Ravi Ramamoorthi Serge Belongie Henrik Wann Jensen. Siggraph 2003.