Irradiance & radiance caching

Jaroslav Křivánek
Charles University, Prague
Jaroslav.Krivanek@mff.cuni.cz

SEVENTH FRAMEWORK PROGRAMME

MARIE CURIE ACTIONS

Global illumination?

- Light bouncing around in a scene

www.photos-of-the-year.com

Diffuse inter-reflection

- May go unnoticed, but looks odd if missing

Slide credit: Michael Bunnell

Why is GI important?

- Architectural visualization
- Interior design
- Product design
- Animated movies, special effects
- Games

Outline

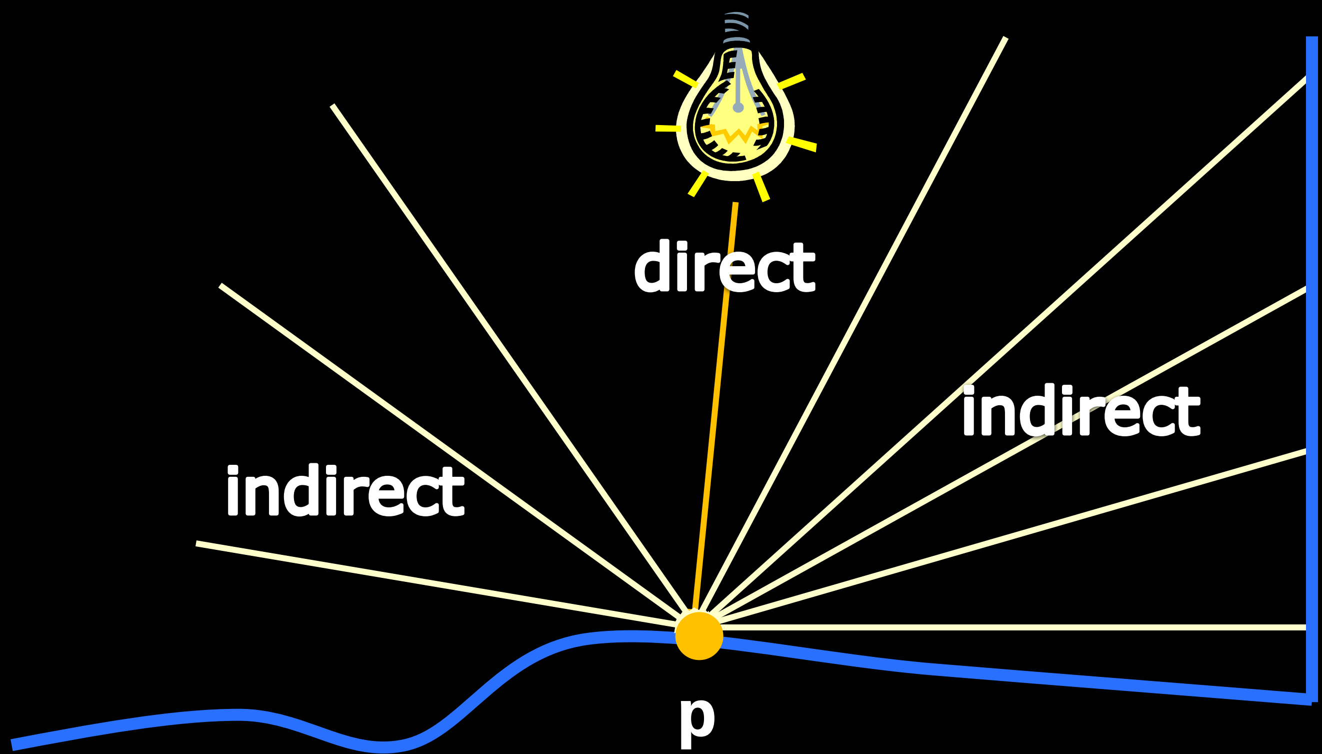
- Brief rehash of realistic rendering
- Irradiance caching

Realistic rendering

- For each visible point **p** in the scene
  - How much light is reflected towards the camera

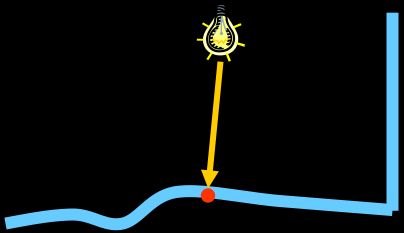
Where does the light come from?

- From light sources (*direct illumination*)
- From scene surfaces (*indirect illumination*)

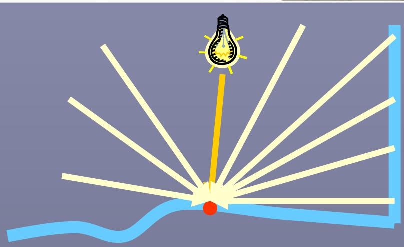


Direct and global illumination

Direct-only

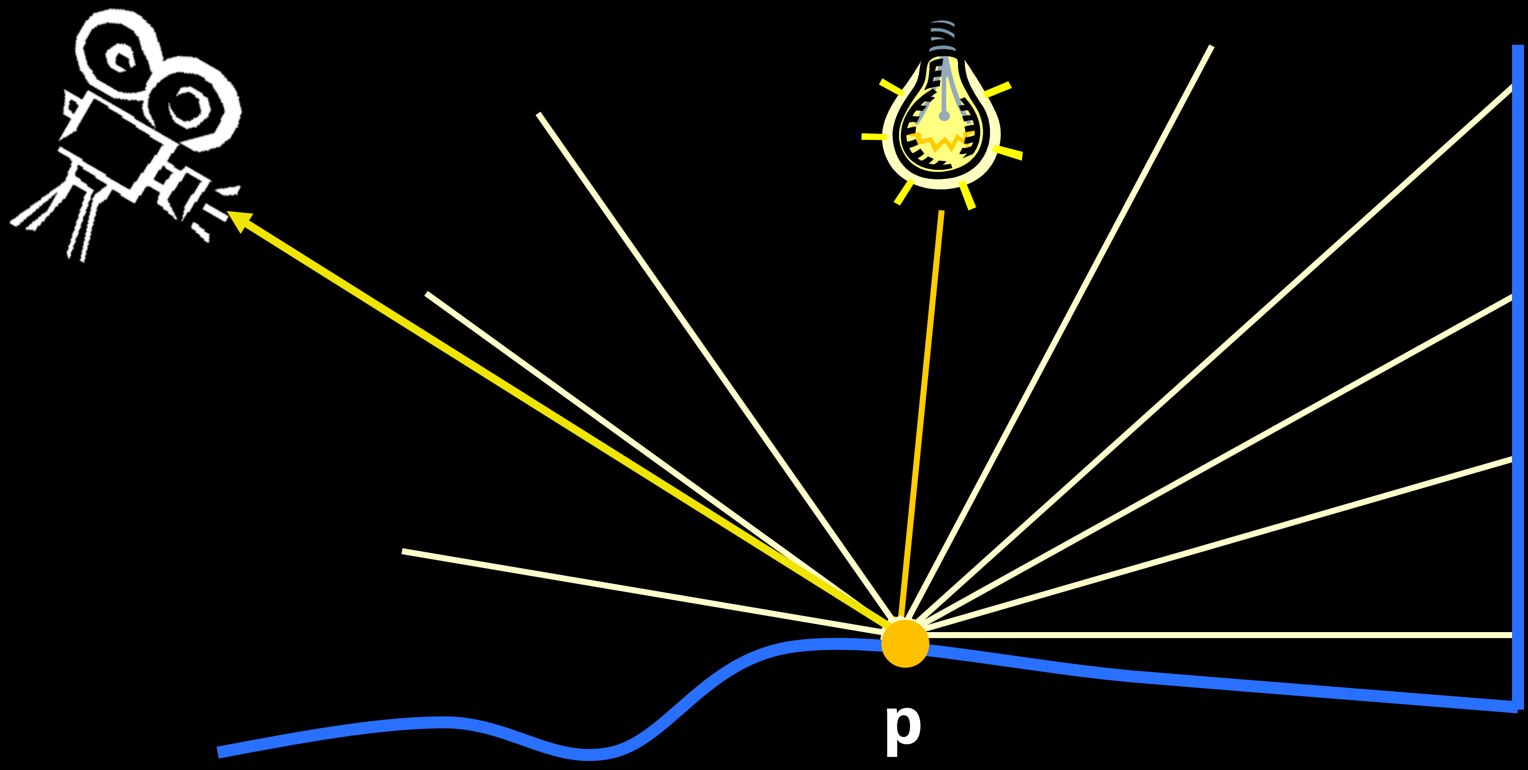


global =
direct +
indirect



Where does the light go then?

- Light reflection – material reflectance



Light reflection

- BRDF
- Shader

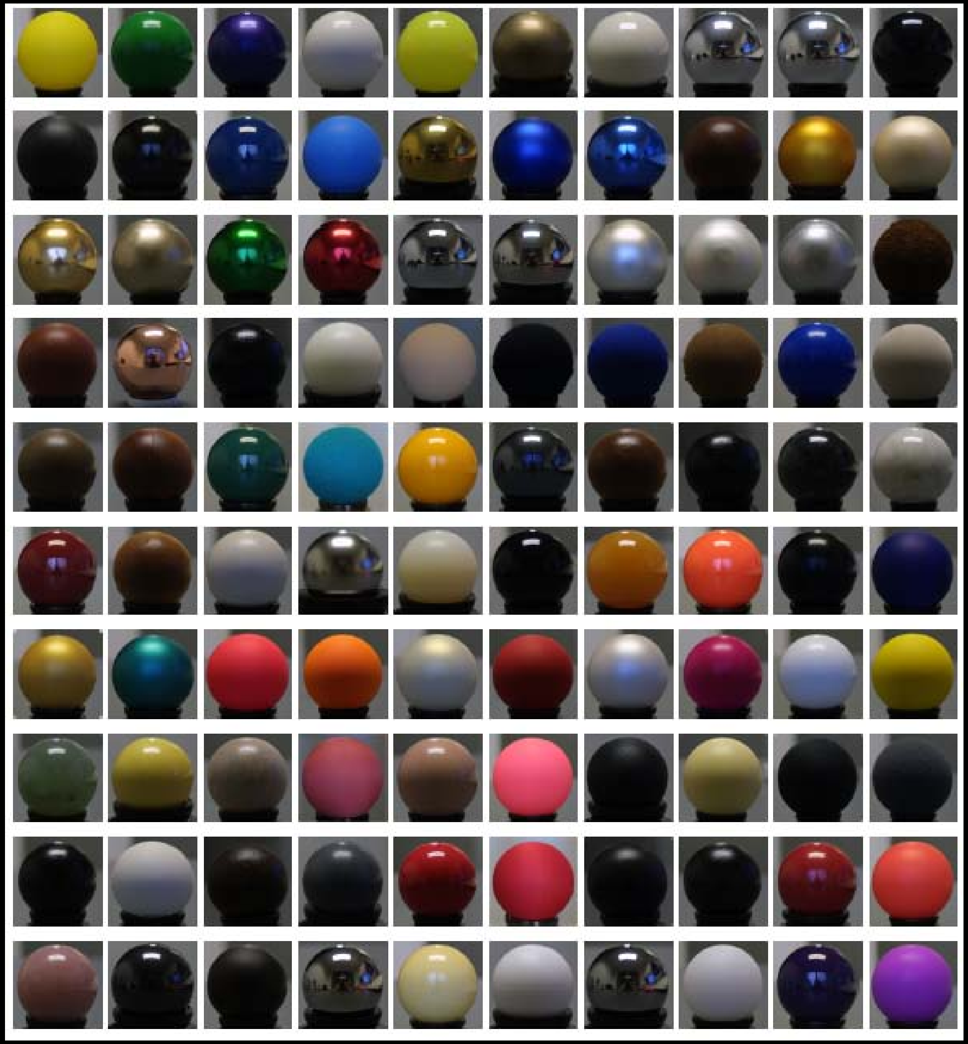
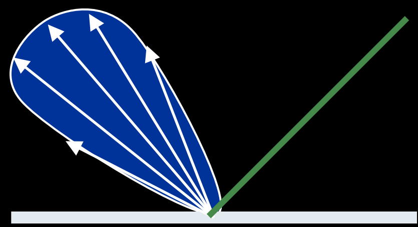


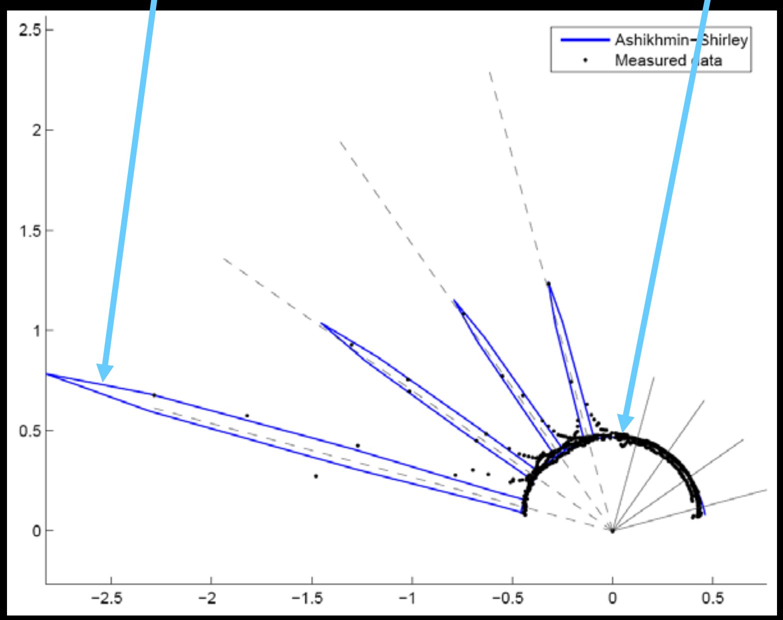
image courtesy Wojciech Matusik

BRDF components



Glossy / specular

Diffuse

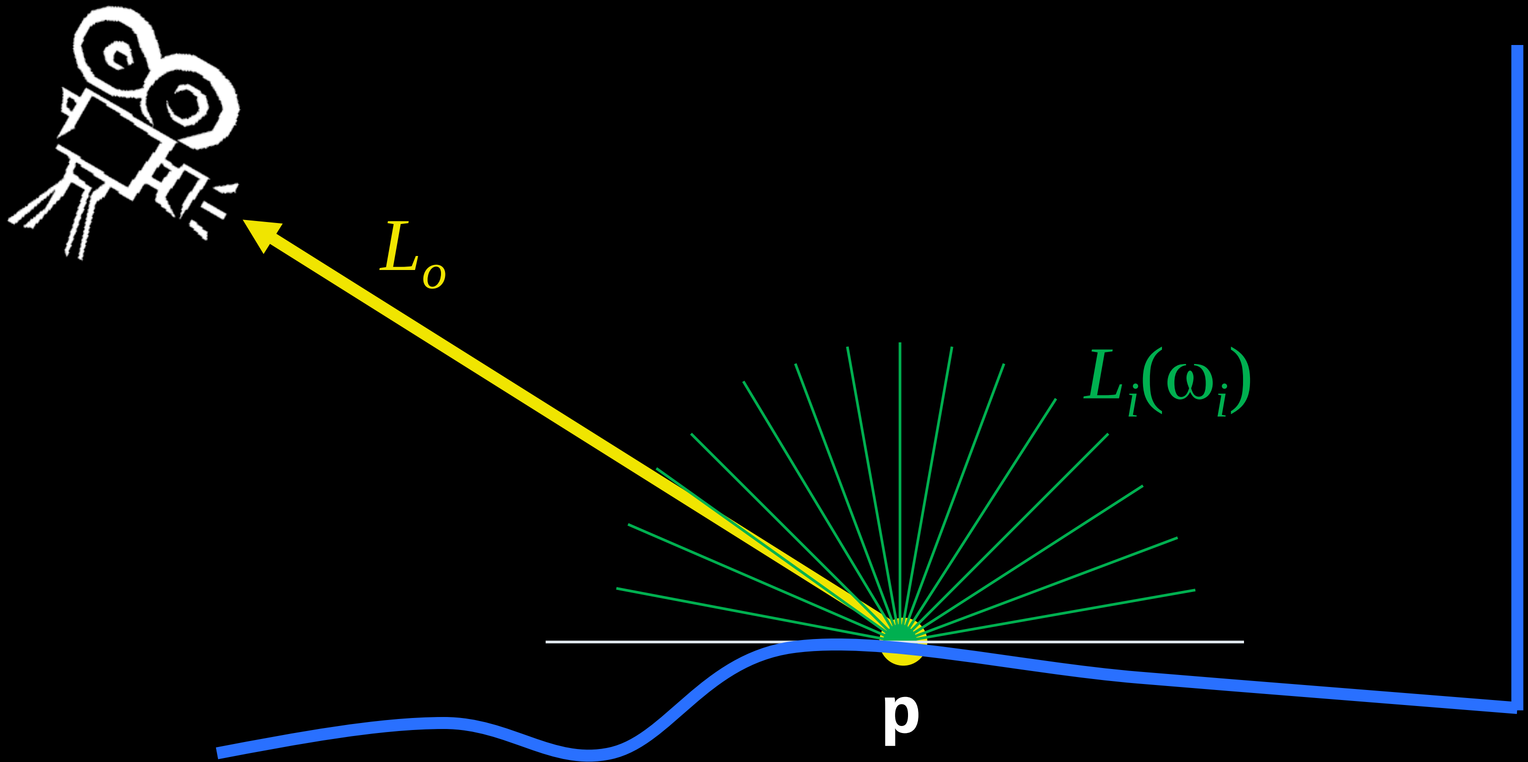


Images by Addy Ngan

Illumination integral

- Total amount of light reflected to ω_o:

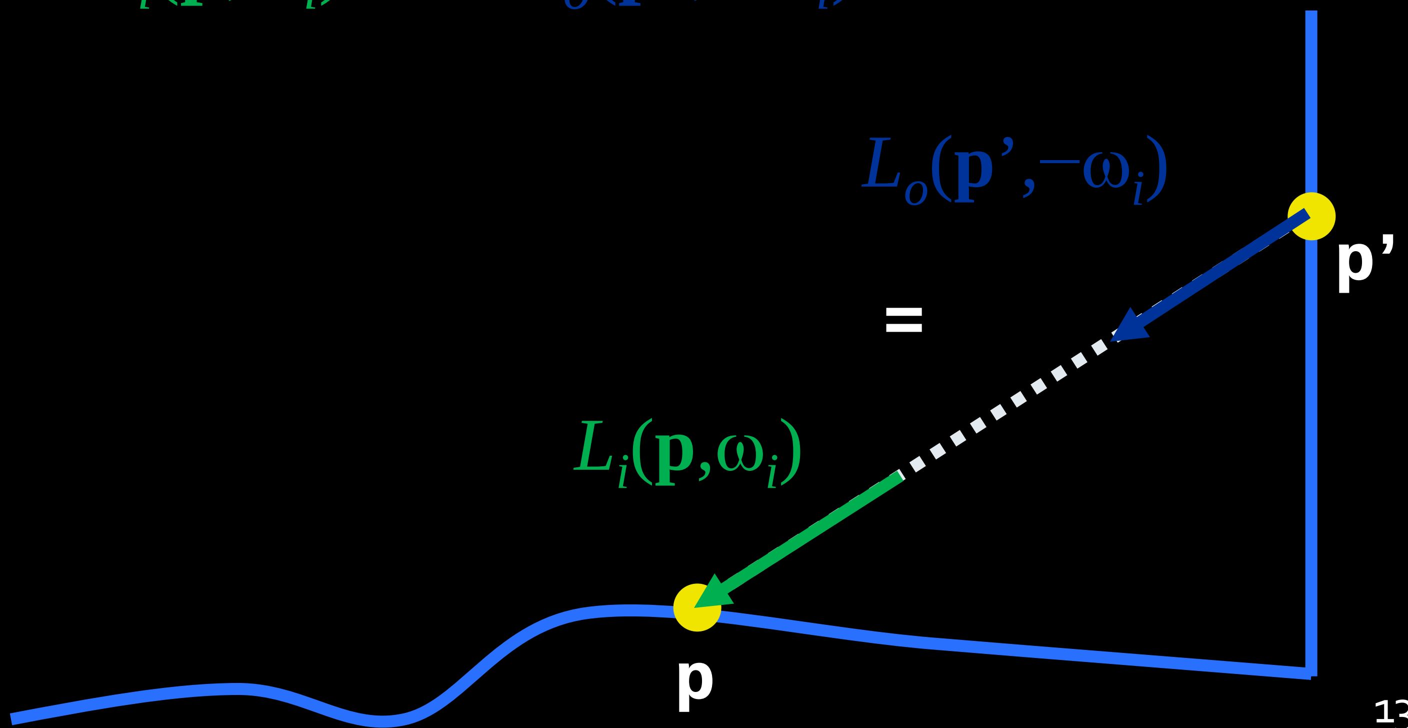
$$L_o = \int L_i(\omega_i)\ \mathrm{BRDF}\ (\omega_i)\ \cos\theta_i\ d\omega_i$$



Light transport

- **Q**: How much light is coming from ω_i ?

$$L_i(\mathbf{p},\omega_i) \quad = \quad L_o(\mathbf{p}',-\omega_i)$$

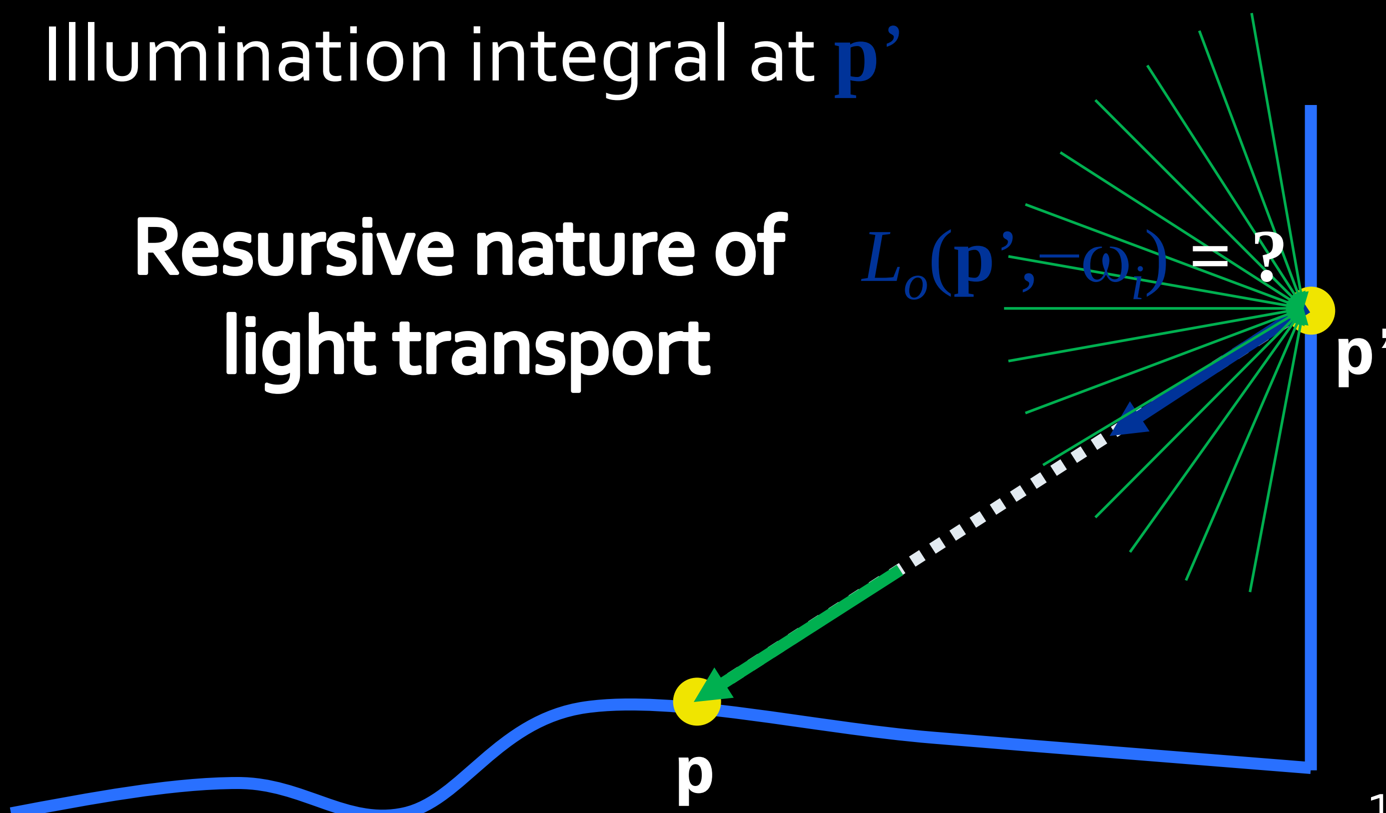


Recursion

- **Q**: How much light is reflected from **p'** ?

Illumination integral at **p'**

Resursive nature of light transport



GI computation

- Many techniques exist
  - You have seen path tracing
- All of them transport light among surfaces
- Different practical consequences
- Today: „irradiance caching”

Unbiased vs. biased estimators

- Path tracing
  - unbiased but noisy images
- Practice
  - Prefer less noise at the cost of bias
  - Systematic bias more acceptable than noise if "looks good" is our measure of image quality

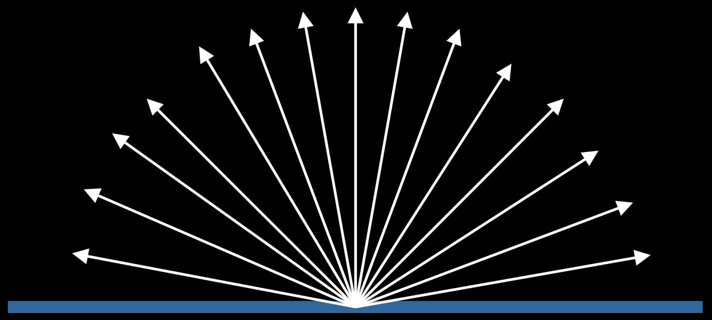
Unbiased vs. consistent estimator

- Unbiased estimator
  - No systematic error, only variance
- Consistent estimator
  - Has systematic error
  - Converges to the correct result
  - E.g. irradiance caching

Irradiance Caching

Motivation

- Distribution path tracing (DPT)
  - Estimate illumination integral at a point by tracing many rays (500-5000)
  - Costly computation



- Irradiance caching accelerates DPT for diffuse indirect illumination

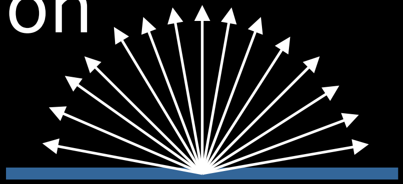
Motivation

- Spatial coherence
  - Diffuse indirect illumination changes slowly over surfaces

Indirect irradiance – changes slowly

Irradiance caching

- Sparse locations for full DRT computation
- Resulting irradiance stored in a cache
- Most pixels interpolated from cached records



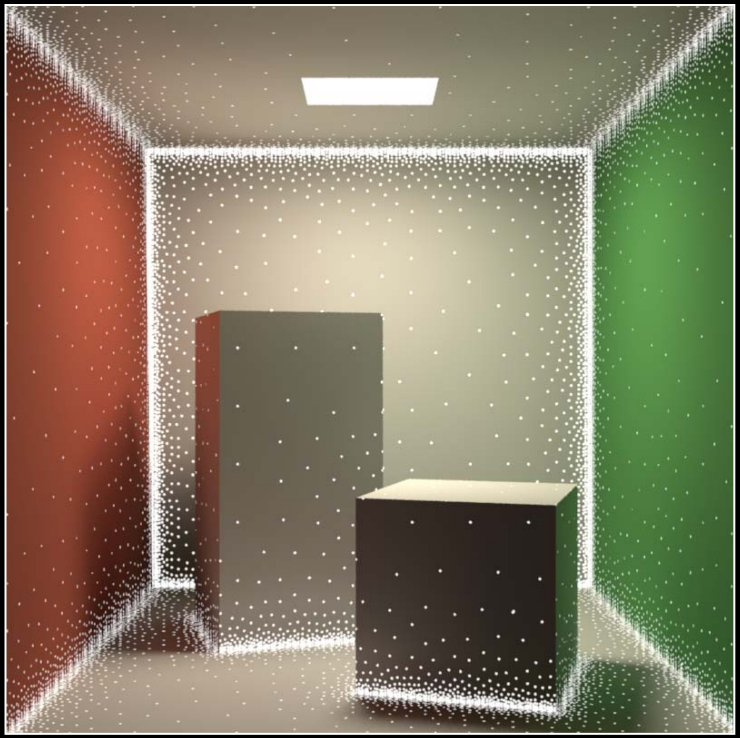


Image credit: Okan Arikan

Irradiance caching

- Faster computation of the *diffuse component* of indirect illumination

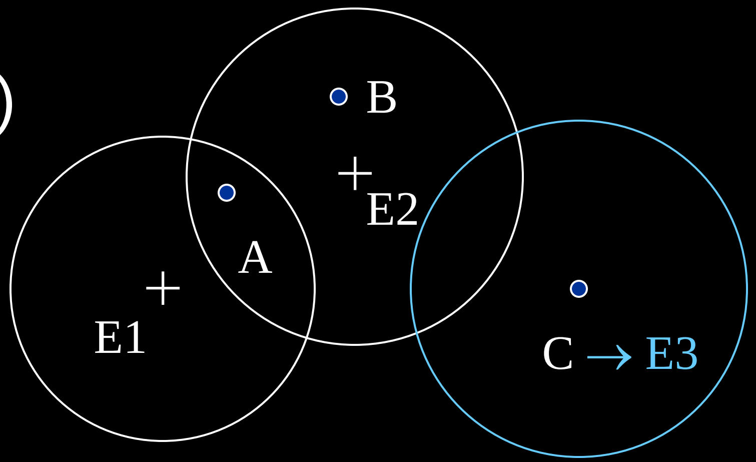
- Diffuse reflection

$$L_o(\mathbf{p}) = E(\mathbf{p}) * \rho_d(\mathbf{p}) / \pi$$

- View-independence
  - Outgoing radiance independent of view direction
  - Total irradiance is all we need => cache irradiance

Irradiance caching

- Lazy evaluation of new irradiance values
  - Only if cannot be interpolated from existing ones
- Example: Values E1 and E2 already stored
  - Interpolate at A (fast)
  - Extrapolate at B (fast)
  - Add new record at C (slow)

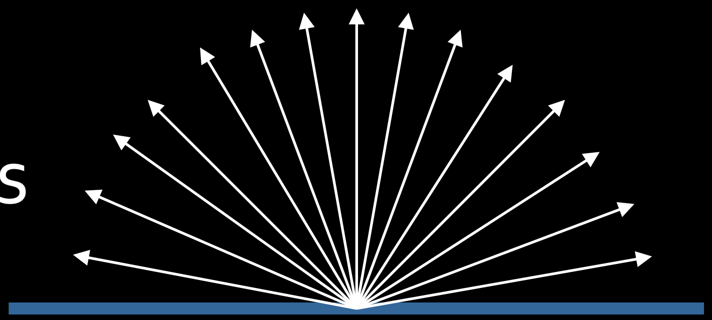


Irradiance caching pseudocode

```
GetIrradiance(p):
    Color E = InterpolateFromCache(p);
    if( E == invalid )
        E = SampleHemisphere(p);
        InsertIntoCache(E, p);
    return E;
```

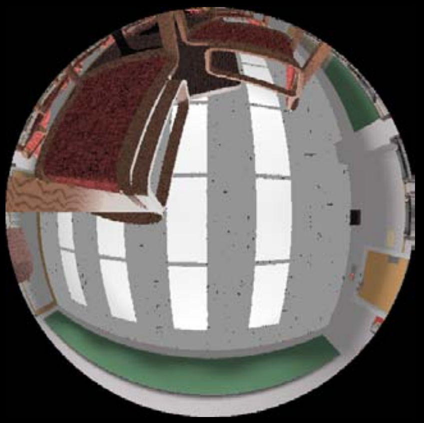
Indirect irradiance calculation

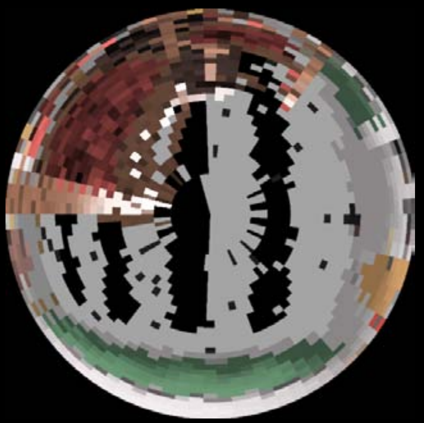
```
E = SampleHemisphere(p);
```



- Cast 500-5000 secondary rays (user-specified)
- Compute illumination at intersection
  - Direct illumination only
  - Path tracing
  - Photon map radiance estimate
  - Query in (another) irradiance cache

  - No emission taken into account!

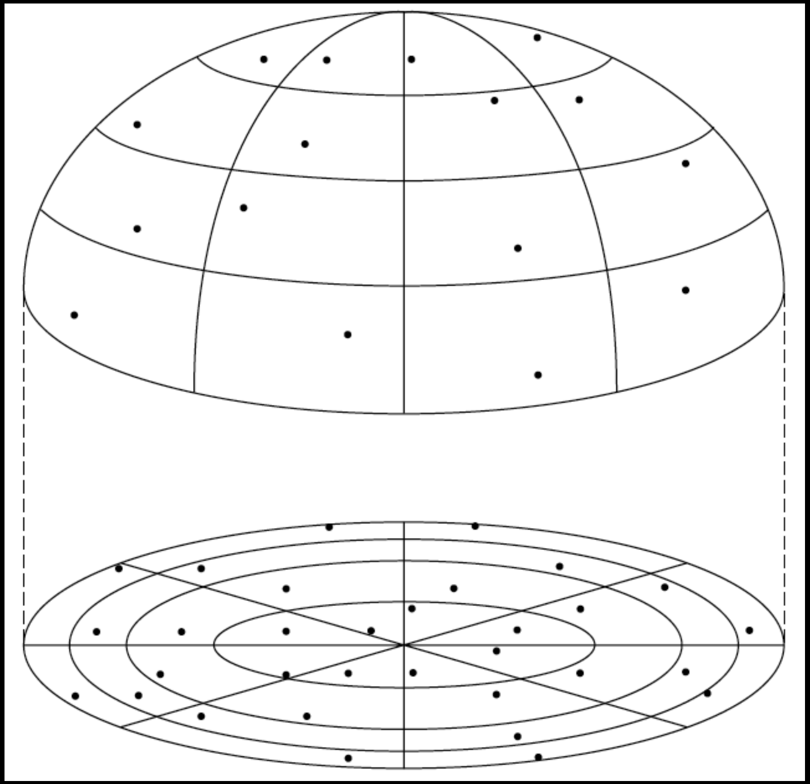




Indirect irradiance calculation

```
E = SampleHemisphere(p);
```

- *Stratified Monte Carlo* hemisphere sampling
  - Subdivide hemisphere into cells
  - Choose a random direction in each cell and trace ray



Indirect irradiance calculation

```
E = SampleHemisphere(p);
```

- Estimating irradiance a **p**:

$$E(\mathbf{p}) = \int L_i(\mathbf{p}, \omega_i) \cos\theta_i \, \mathrm{d}\omega_i$$

- General form of the estimator

$$E(\mathbf{p}) \approx \frac{1}{MN} \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} \frac{f(\theta_{j,k}, \phi_{j,k})}{p(\theta_{j,k}, \phi_{j,k})}$$

Indirect irradiance calculation

```
E = SampleHemisphere(p);
```

- For irradiance calculation, the integrand is:

$$L(\theta, \phi)\ \cos\theta$$

- PDF:

$$p(\theta, \phi) = \frac{\cos\theta}{\pi}$$

Indirect irradiance calculation

- Irradiance estimator for IC:

$$E(\mathbf{p}) \approx \frac{\pi}{MN} \sum_{j=0}^{M-1} \sum_{k=0}^{N-1} L_{j,k}$$

- $L_{j,k}$ ... radiance sample from direction:

$$(\theta_{j,k}, \phi_{j,k}) = \left(\arccos \sqrt{1 - \frac{j + \zeta_{j,k}^1}{M}}, 2\pi \frac{k + \zeta_{j,k}^2}{N} \right)$$

- M, N ... number of divisions along θ and ϕ
- $\zeta_{j,k}^1, \zeta_{j,k}^2$ ... random numbers from R(0,1)

Irradiance caching pseudocode

```
GetIrradiance(p):
   Color E = InterpolateFromCache(p);
   if( E == invalid )
      E = SampleHemisphere(p);
      InsertIntoCache(E, p);
   return E;
```

Record spacing

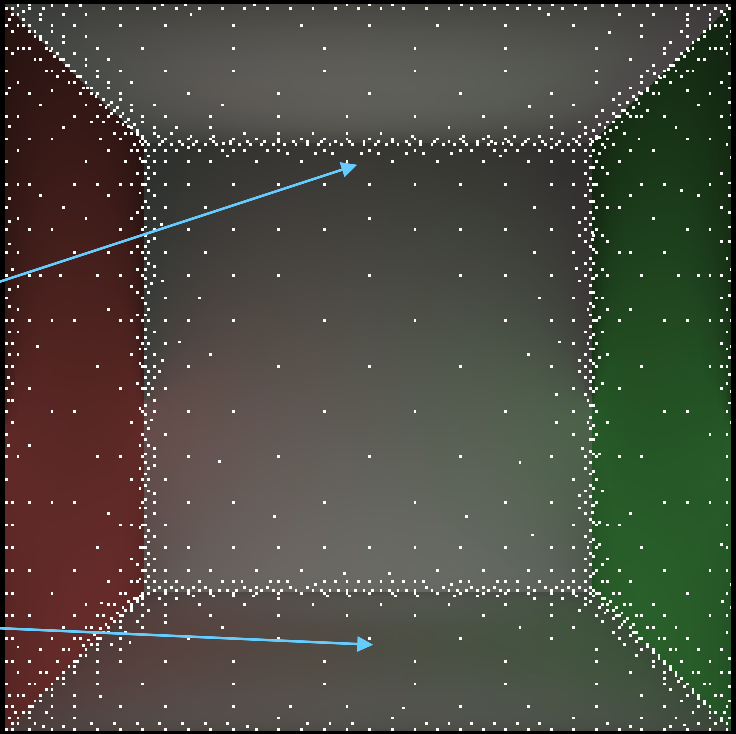
- If $E(\mathbf{p})$ changes slowly => interpolate more
- If $E(\mathbf{p})$ changes quickly => interpolate less

- What is the upper bound on rate of change (i.e. gradient) of irradiance?
- Answer from the "worst case" analysis (omitted)

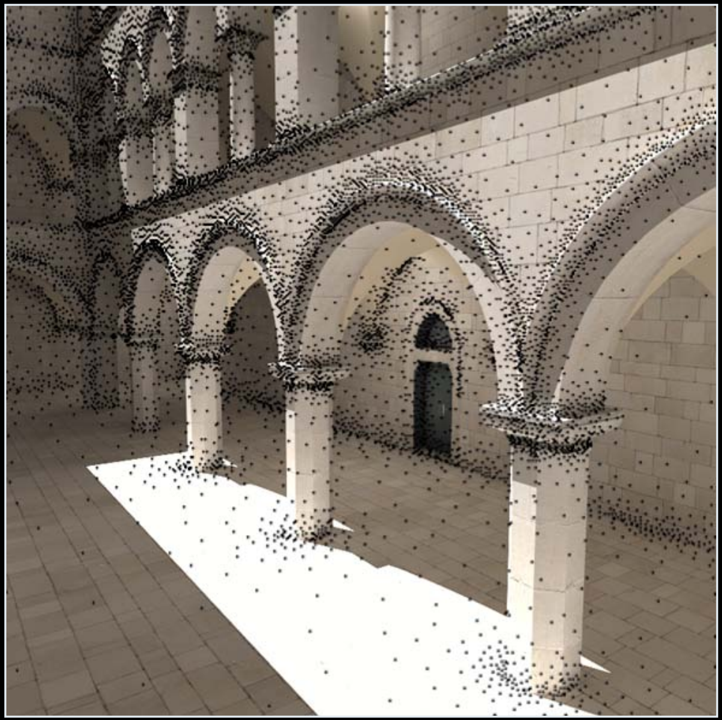
Record spacing

- Near geometry
  → dense spacing
  - Geometry = source of indirect illumination

- Open spaces
  → sparse sampling



Record spacing



Irradiance interpolation

```
E = InterpolateFromCache(p)
```

- Weighted average: $$E(\mathbf{p}) = \frac{\sum_{i \in S(\mathbf{p})} E_i(\mathbf{p}) w_i(\mathbf{p})}{\sum_{i \in S(\mathbf{p})} w_i(\mathbf{p})},$$

- Records used for interpolation:

$$S(\mathbf{p}) = \{i;\ w_i(\mathbf{p}) > 0\}$$

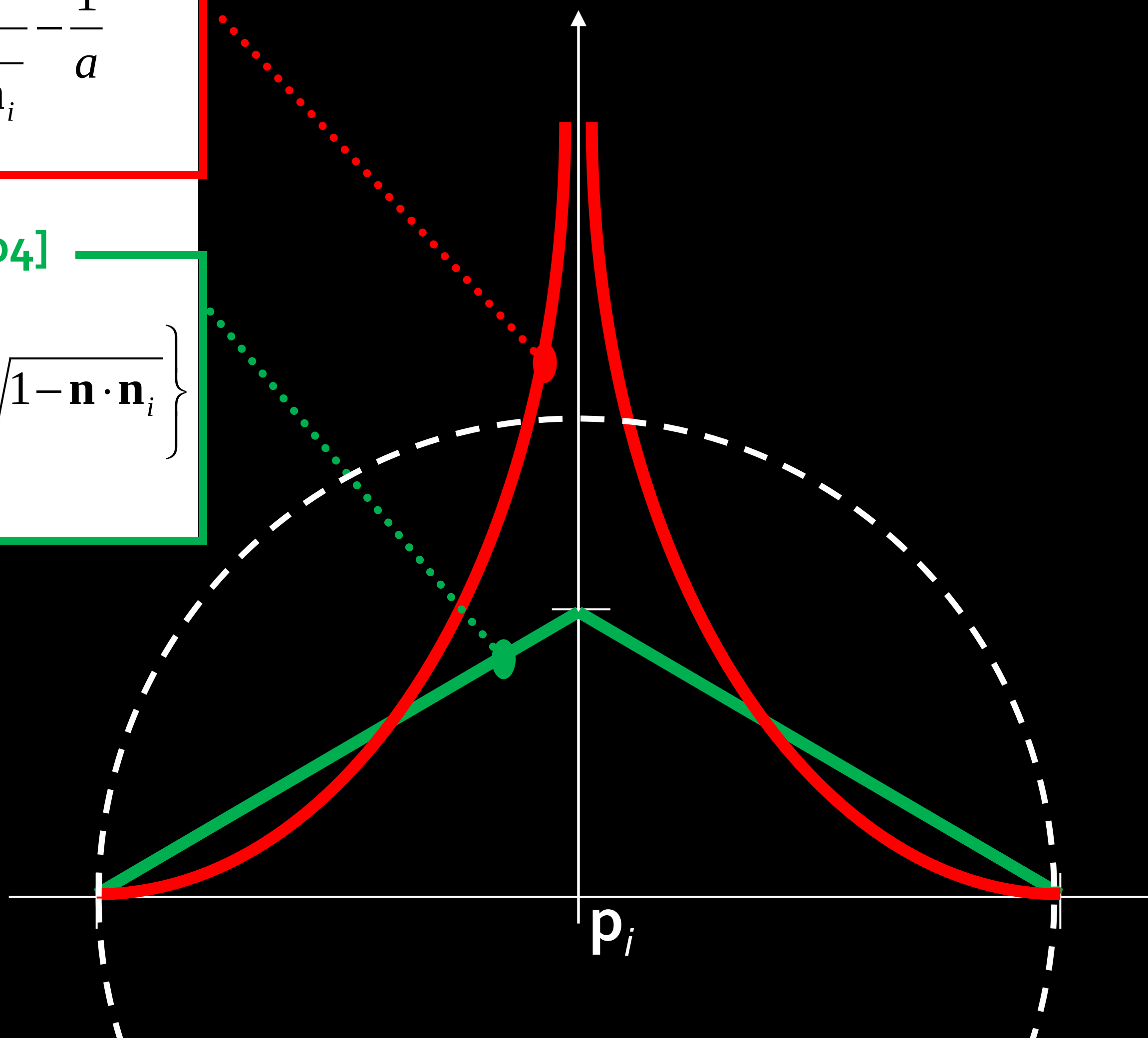
Weighting function

[Ward 88 (modified)]

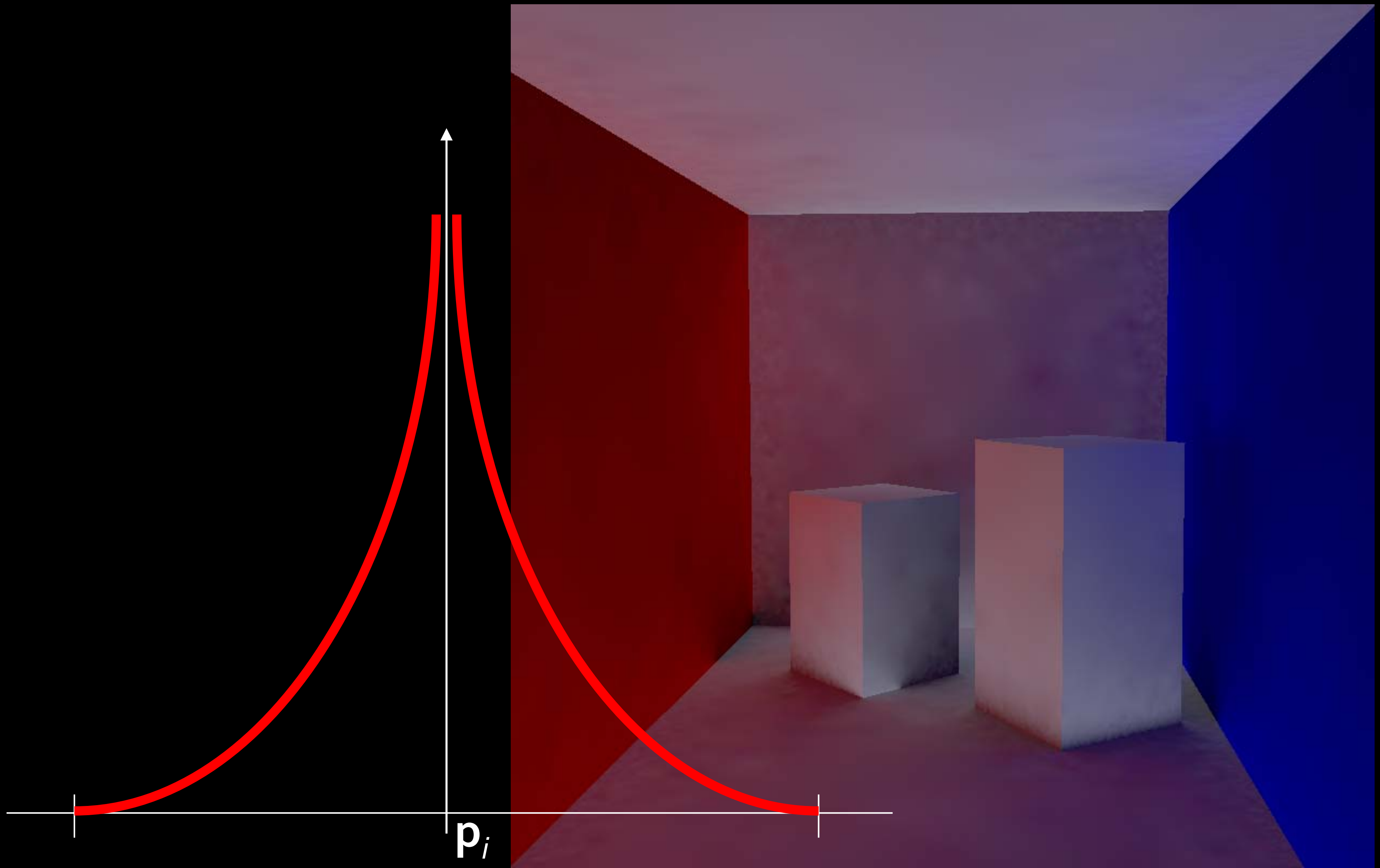
$$w_i^1(\mathbf{p}) = \frac{1}{\frac{\|\mathbf{p}-\mathbf{p}_i\|}{R_i} + \sqrt{1-\mathbf{n}\cdot\mathbf{n}_i}} - \frac{1}{a}$$

[Tablellion and Lamorlette 04]

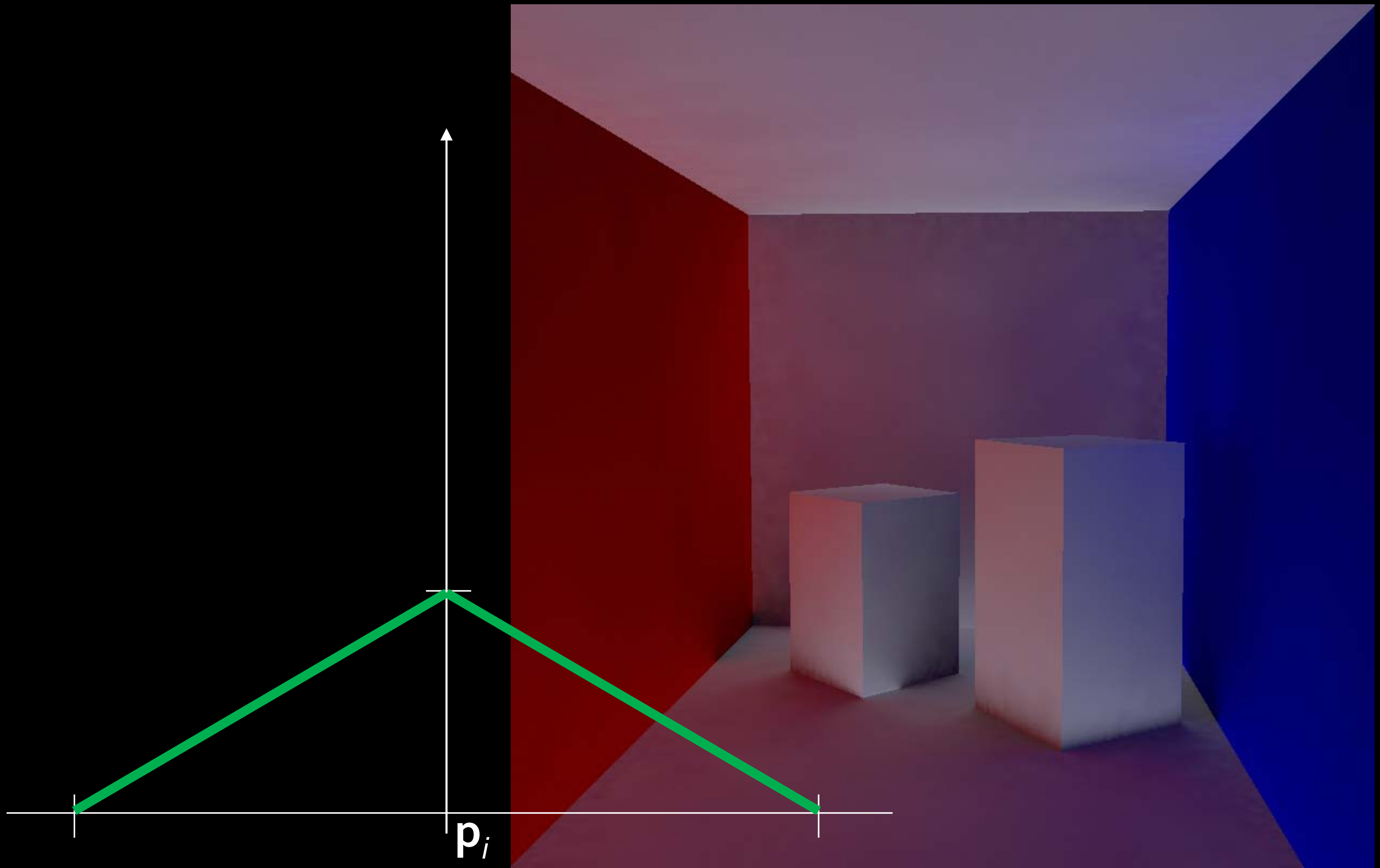
$$w_i^2(\mathbf{p}) = 1 - \frac{1}{a}\max\left\{\frac{\|\mathbf{p}-\mathbf{p}_i\|}{R_i},\quad \sqrt{1-\mathbf{n}\cdot\mathbf{n}_i}\right\}$$



Weighting function

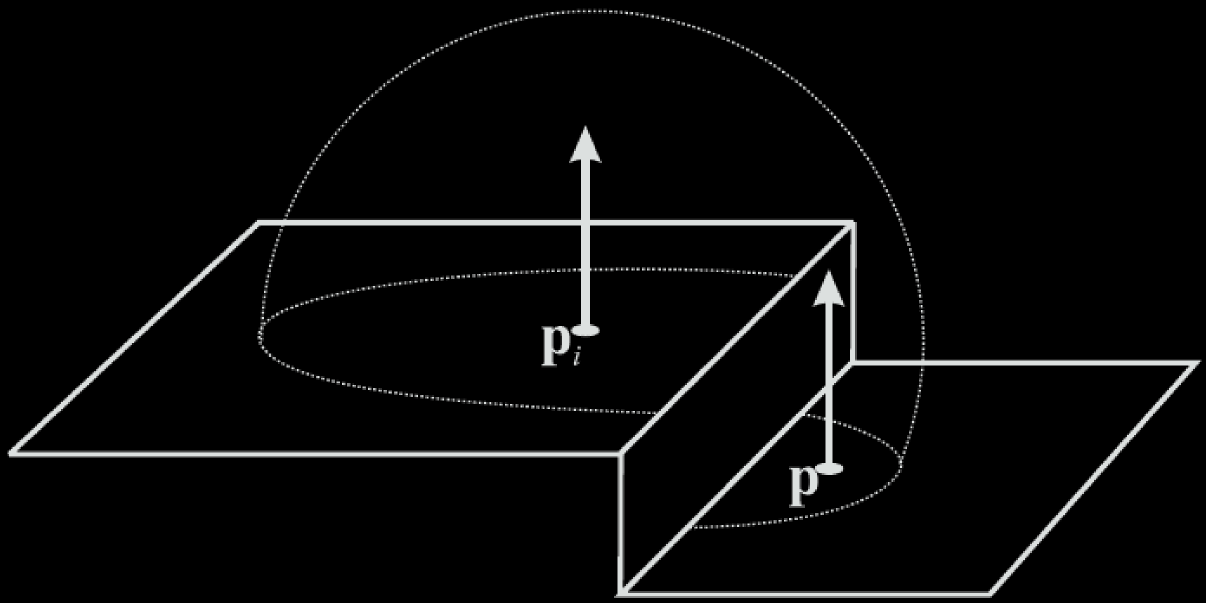


Weighting function



Heuristic "behind" test

- Record at $\mathbf{p}_i$ rejected from interpolation at $\mathbf{p}$ if $\mathbf{p}$ is "behind" $\mathbf{p}_i$



Irradiance caching pseudocode

```
GetIrradiance(p):
   Color E = InterpolateFromCache(p);
   if( E == invalid )
      E = SampleHemisphere(p);
      InsertIntoCache(E, p);
   return E;
```

Irradiance cache record

`InsertIntoCache(E, p);`

• Vector3 position;	Position in space
• Vector3 normal;	Normal at P
• float R;	Validity radius
• Color E;	Stored irradiance
• Color dEdP[3];	Gradient w.r.t. translation
• Color dEdN[3];	Gradient wrt rotation

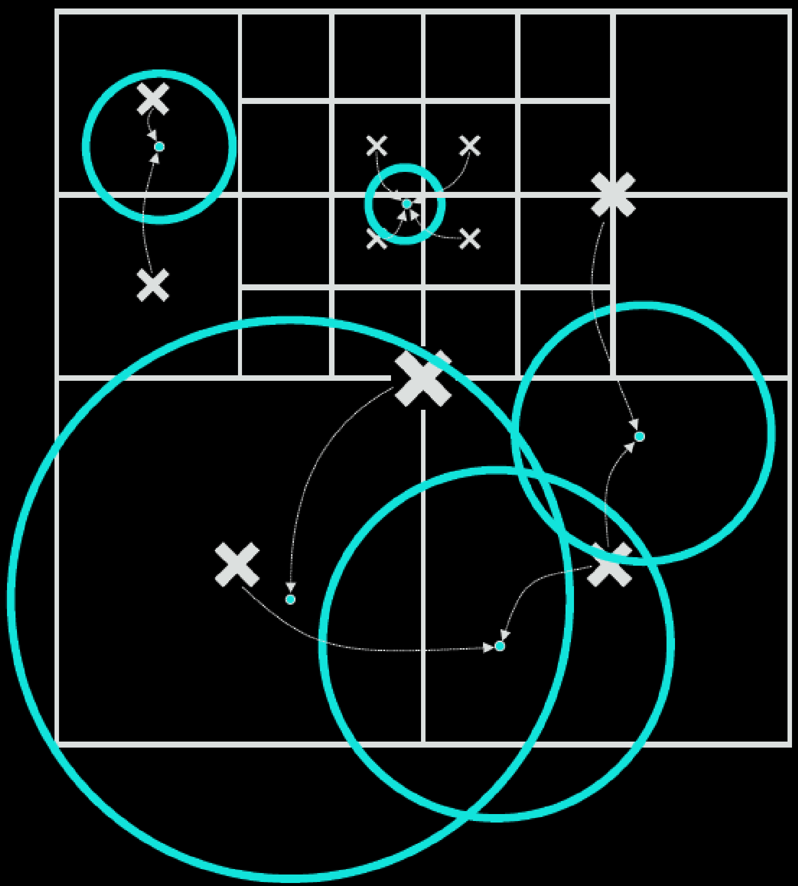
Irradiance cache data structure

```
InsertIntoCache(E, p);
```

- Requirements
  - Fast incremental updates (records stored on the fly)
  - Fast query for all records (spheres) overlapping a given point **p**

Data structure: Octree

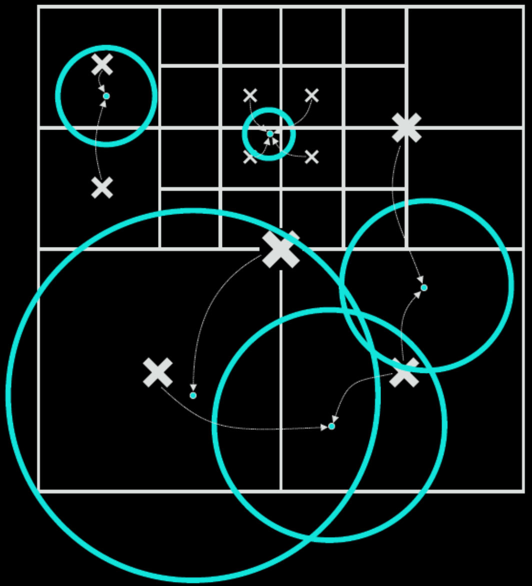
```
InsertIntoCache(E, p);
```



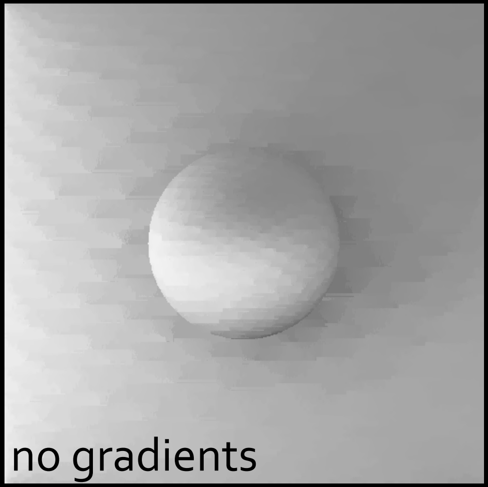
Data structure: Octree

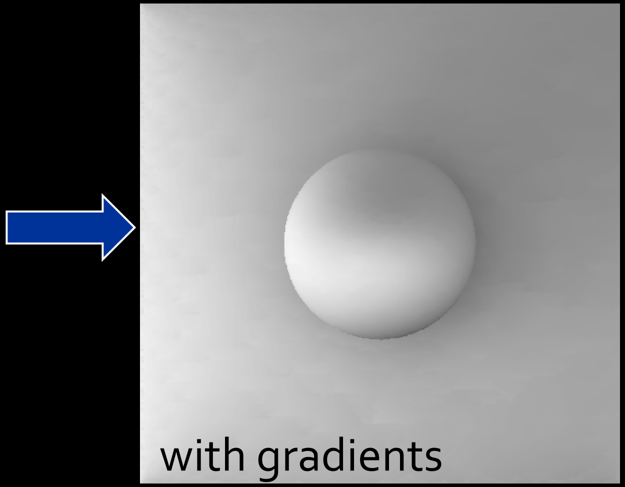
back to ... `E = InterpolateFromCache(p)`

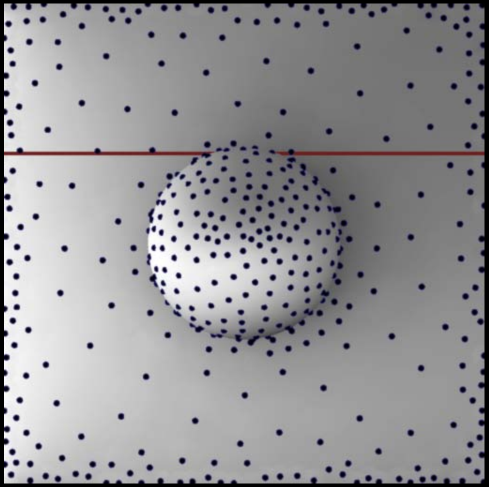
procedure LookUpRecordsMR(**p**, **n**)
    node $\leftarrow$ root
    while node $\neq$ NULL **do**
        for all records i stored in node **do**
            if $(w_i(\mathbf{p}) > 0)$ **and** ($\mathbf{p}_i$ not in front of **p**) **then**
                Include record in $S(\mathbf{p})$.
            end if
        end for
        node $\leftarrow$ child containing **p**
    end while
end procedure

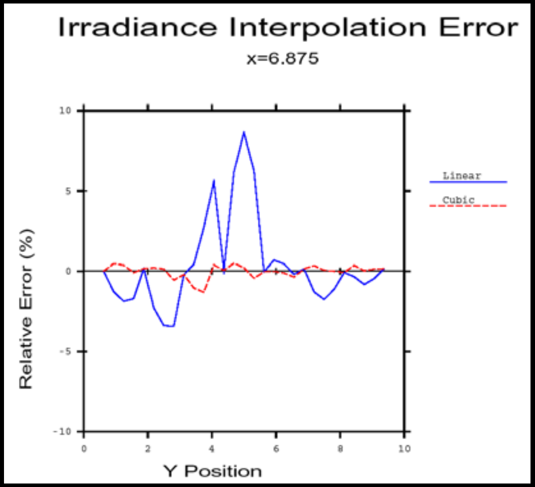


Irradiance gradients









Irradiance gradients

- Essential for smooth interpolation
- Calculated during hemisphere sampling
  - i.e. no extra rays, little overhead
- Stored as a part of the record in the cache
- Used in interpolation

Rotation gradient

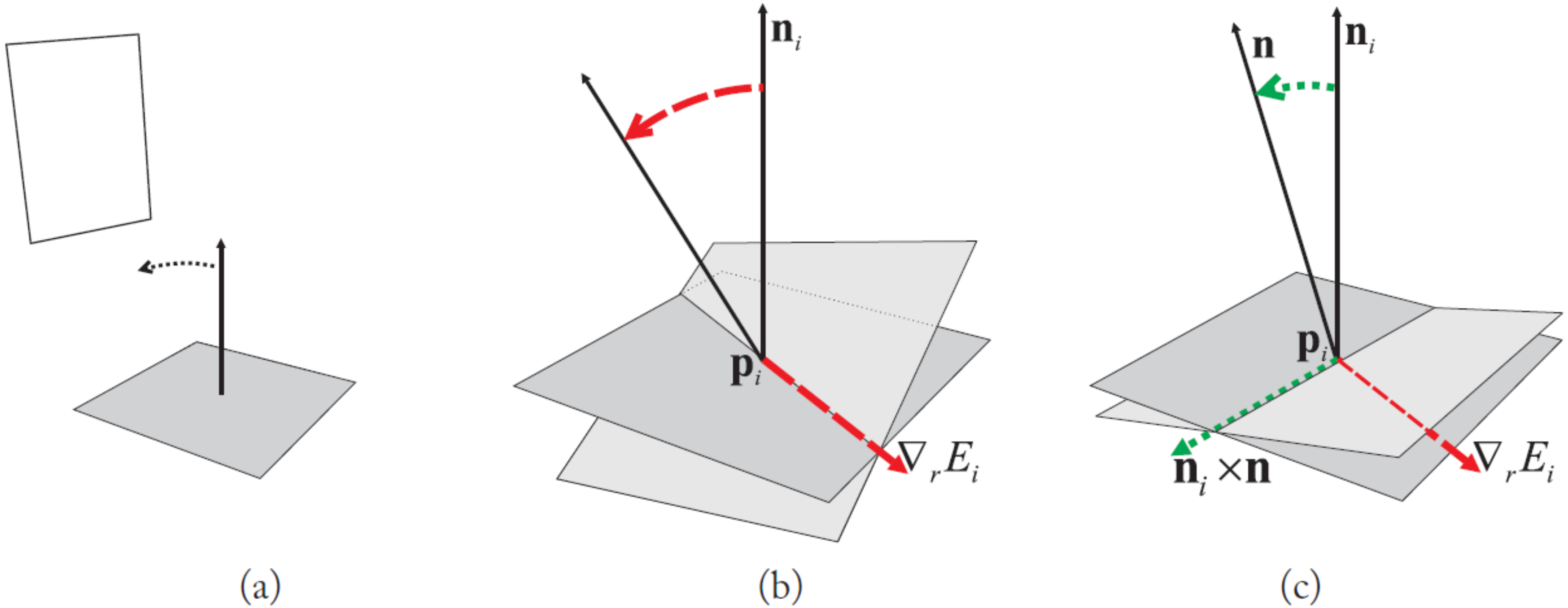
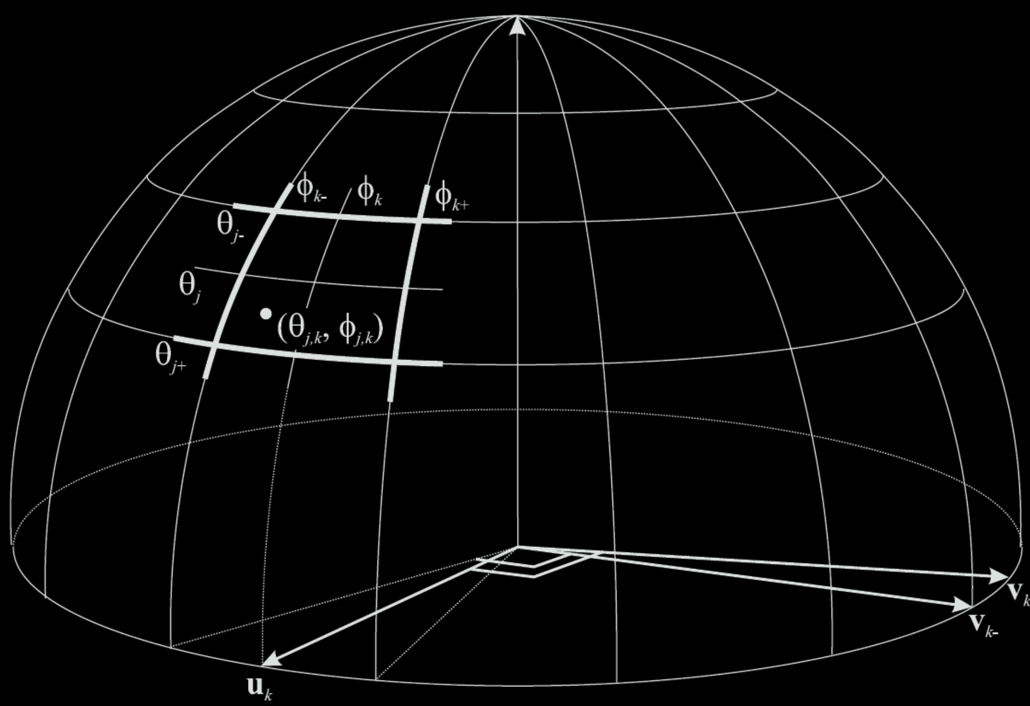


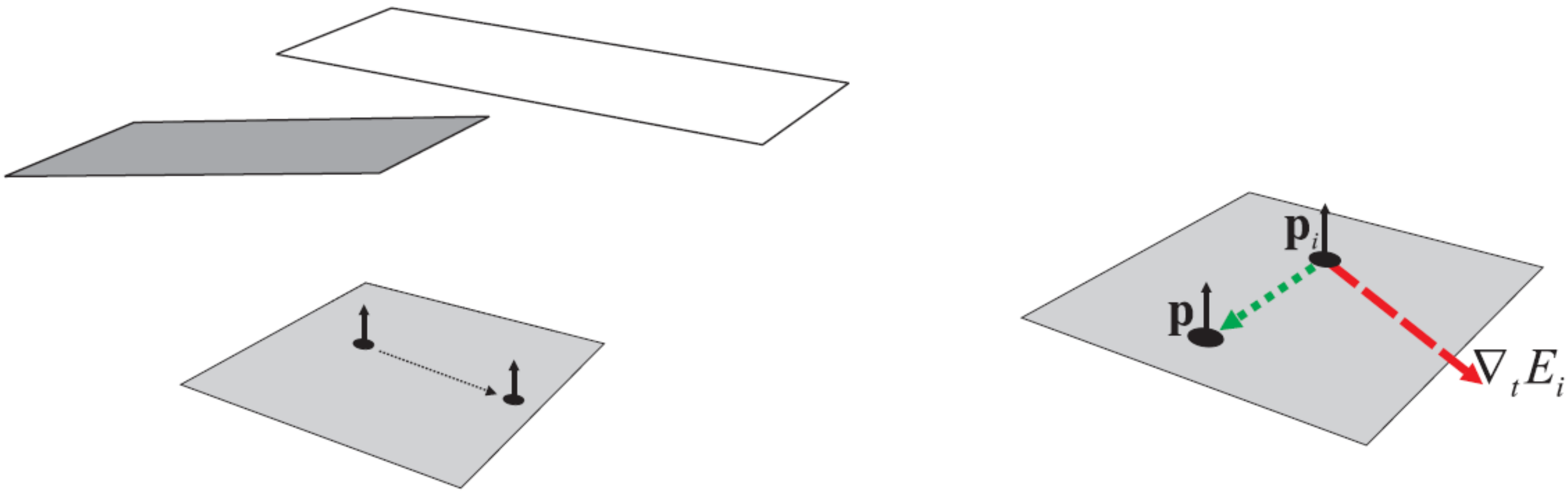
Figure 2.4: (a) As the surface element is rotated towards the bright surface, irradiance increases. (b) The rotation gradient $\nabla_r E_i$ of cache record i gives the axis of rotation that produces maximum increase in irradiance. The gradient magnitude is the irradiance derivative with rotation around that axis. (c) When the surface element is rotated around any arbitrary axis (in our example determined by the change in surface normal as $\mathbf{n}_i \times \mathbf{n}$) the irradiance derivative is given by the dot product of the axis of rotation and the rotation gradient: $(\mathbf{n}_i \times \mathbf{n}) \cdot \nabla_r E_i$.

Rotation gradient formula

$$\nabla_r E \approx \frac{\pi}{MN} \sum_{k=0}^{N-1} \left(\mathbf{v}_k \sum_{j=0}^{M-1} -\tan\theta_j \cdot L_{j,k} \right)$$



Translation gradient



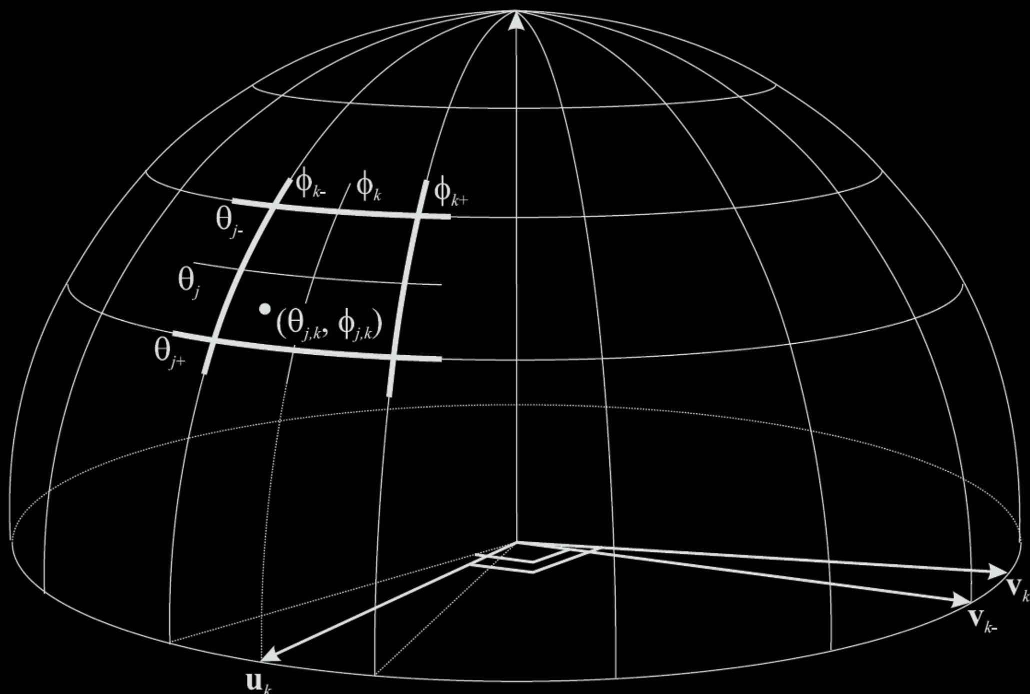
(a) (b)

Figure 2.6: (a) As the surface element is translated, it becomes more exposed to the bright surface, and irradiance increases. (b) The translation gradient $\nabla_t E_i$ of record i gives the direction of translation that produces the maximum increase in irradiance. The gradient magnitude is the irradiance derivative with respect to translation along that direction. When a surface element is translated along any arbitrary direction, a first-order approximation of the change in irradiance is given by the dot product of the translation vector and the translation gradient: $(\mathbf{p} - \mathbf{p}_i) \cdot \nabla_t E_i$.

Translation gradient formula

$$\nabla_t E \approx \sum_{k=0}^{N-1} \Bigg[\quad \mathbf{u}_k \frac{2\pi}{N} \sum_{j=1}^{M-1} \frac{\cos^2 \theta_{j\_} \sin \theta_{j\_}}{\min\{r_{j,k}, r_{j-1,k}\}} (L_{j,k} - L_{j-1,k}) +$$

$$\mathbf{v}_{k\_} \sum_{j=0}^{M-1} \frac{\cos \theta_j (\cos \theta_{j\_} - \cos \theta_{j_+})}{\sin \theta_{j,k} \min\{r_{j,k}, r_{j,k-1}\}} (L_{j,k} - L_{j,k-1}) \Bigg]$$



Irradiance interpolation w/ grads

```
E = InterpolateFromCache(p)
```

- Weighted average: $$E(\mathbf{p}) = \frac{\sum_{i \in S(\mathbf{p})} E_i(\mathbf{p}) w_i(\mathbf{p})}{\sum_{i \in S(\mathbf{p})} w_i(\mathbf{p})},$$

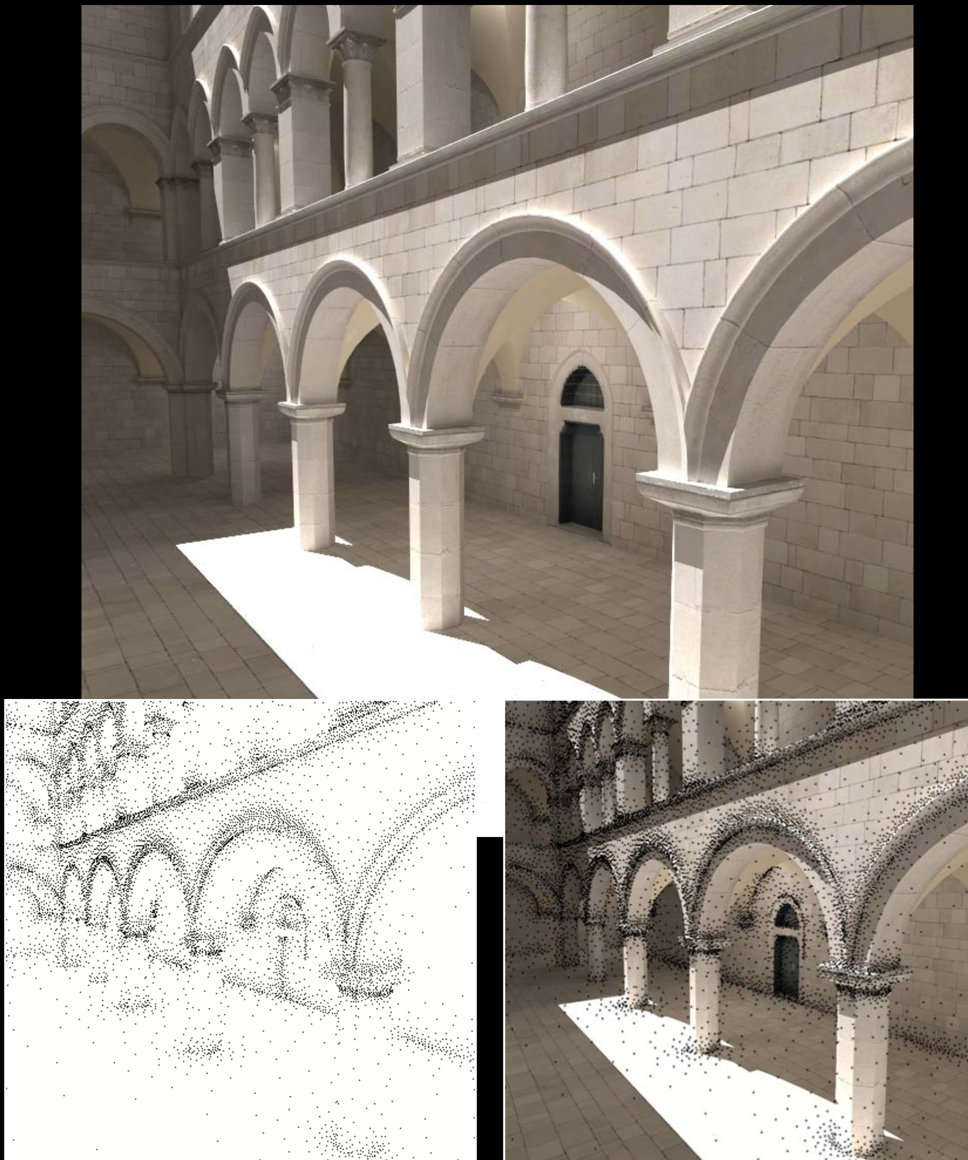
$$E_i(\mathbf{p}) = E_i + (\mathbf{n}_i \times \mathbf{n}) \cdot \nabla_r E_i + (\mathbf{p} - \mathbf{p}_i) \cdot \nabla_t E_i$$

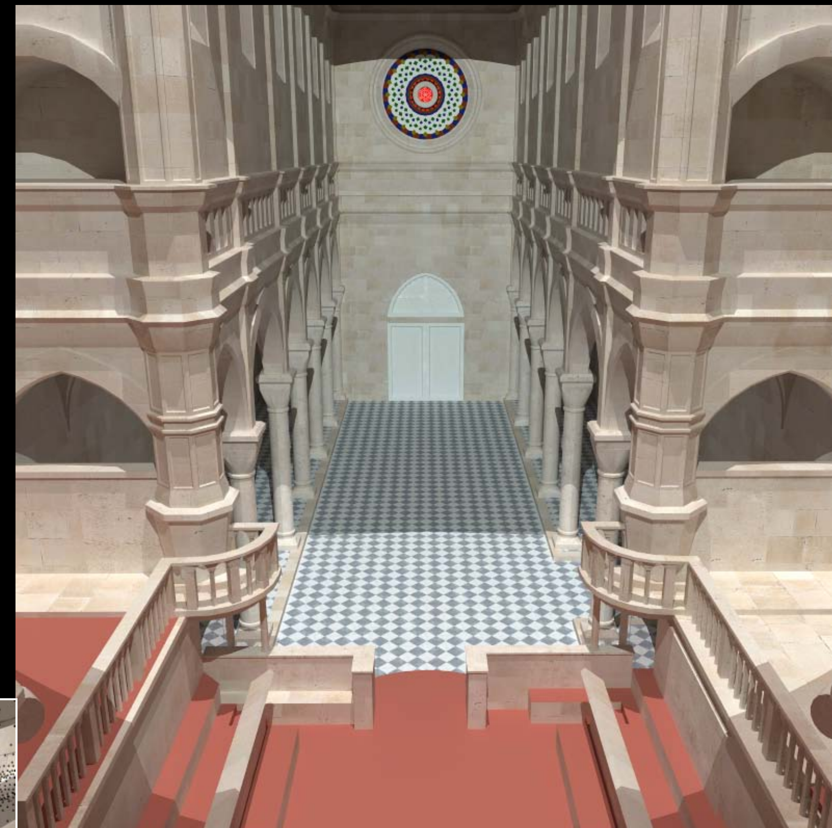
Irradiance caching history

- 1988: Ward et al.
  - Original idea
- 1992: Ward and Heckbert
  - Irradiance Gradients
- 1996: Jensen
  - IC part of Photon maps
- 2004: Tabellion and Lamorlette
  - First use of GI in film (Shrek2)
- 2005: Krivanek et al.
  - Extension to glossy surfaces

Image credit: Eric Tabellion, PDI DreamWorks

Irradiance caching examples





Irradiance caching examples

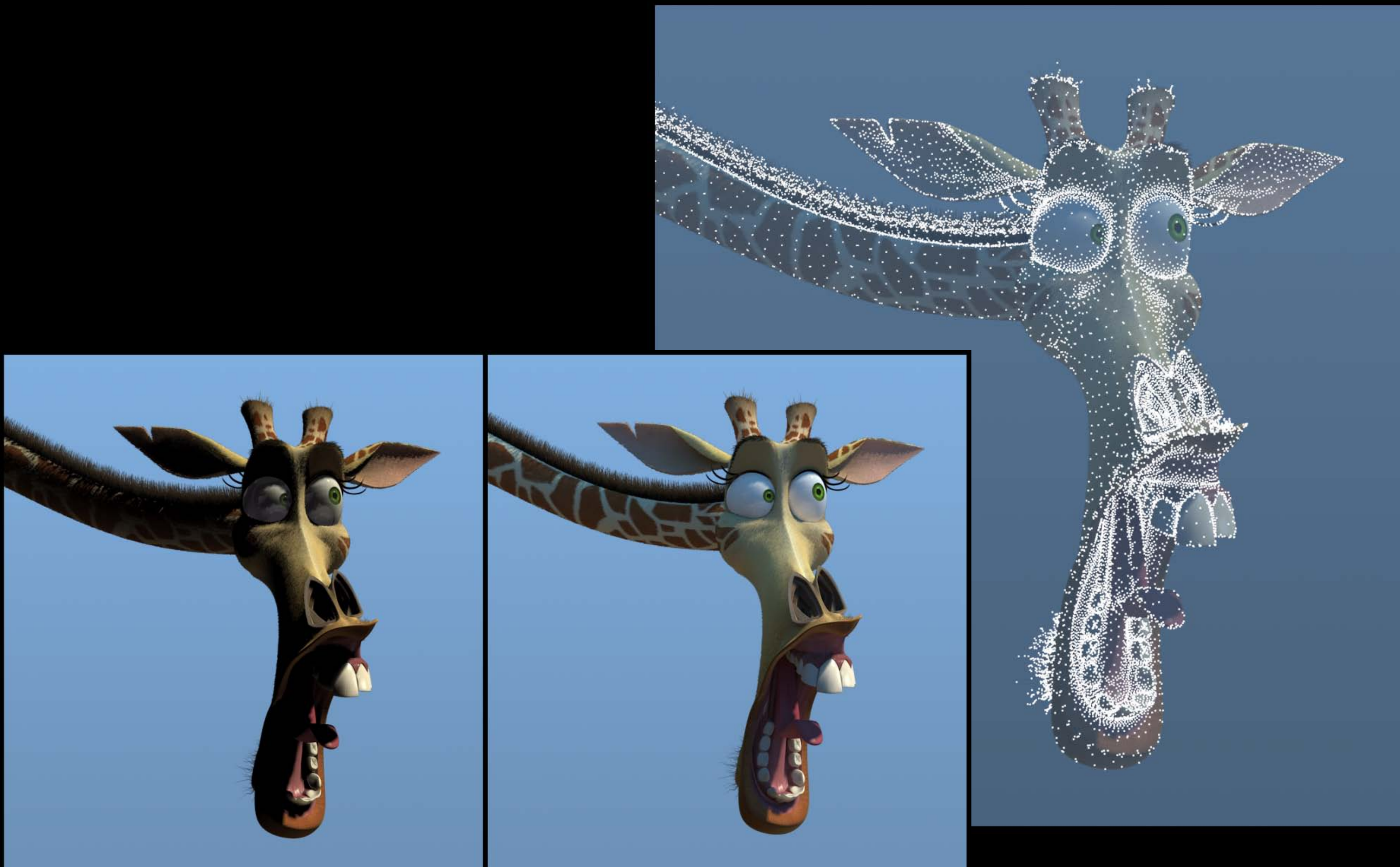


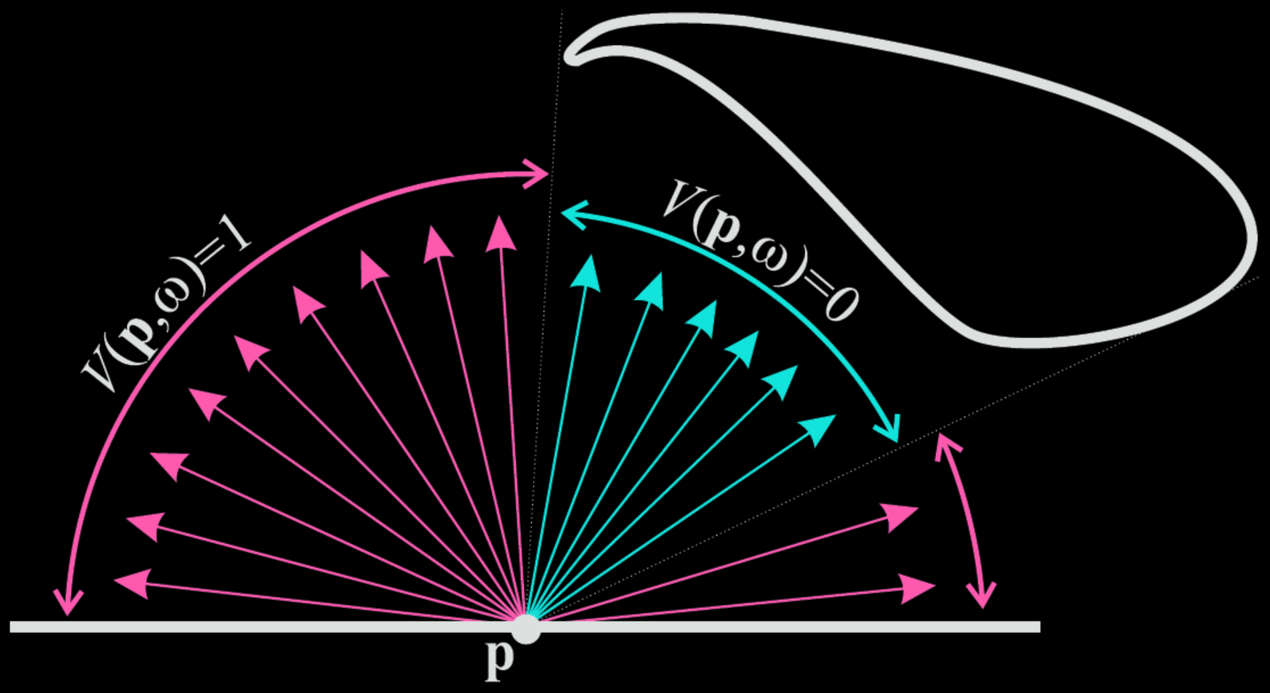
Image credit: Eric Tabellion, PDI DreamWorks

Irradiance caching examples

Image credit: Eric Tabellion, PDI DreamWorks

Ambient occlusion

$$A(\mathbf{p}) = \frac{1}{\pi}\int_{H+} V(\mathbf{p},\omega)\,\cos\theta\,\mathrm{d}\omega$$



Ambient occlusion

X

=

Ambient occlusion caching

Conclusion

- Fast indirect illumination of diffuse surfaces
  - Sprase sampling & fast interpolation
- Consistent but not unbiased
- Tons of implementation details that I did not discuss here

Further reading

- Practical Global Illumination with Irradiance Caching
  - SIGGRAPH Course: 2008, Křivánek et al.
  - Book, 2009, Křivánek & Gautron
  - Both give references to further resources