Global illumination with many-light methods

Jaroslav Křivánek

Charles University, Prague

Instant radiosity

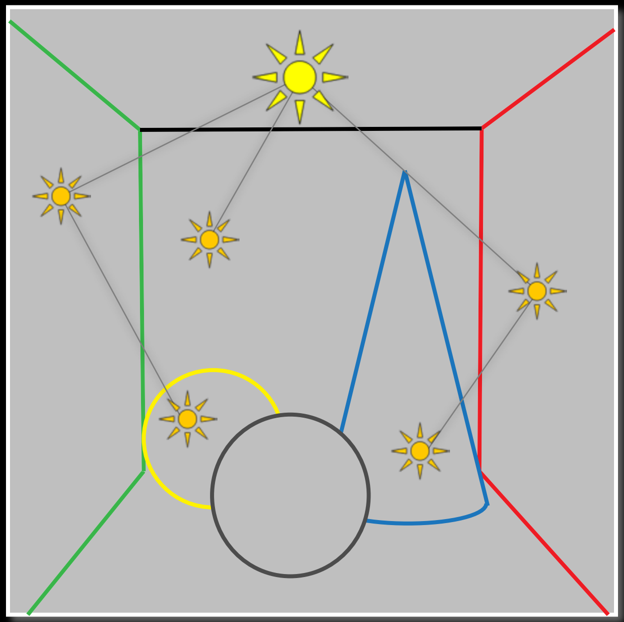
- Alexander Keller, 1997
- The “original” many-light method
- Probably the first GPU-based GI algorithm

Instant radiosity

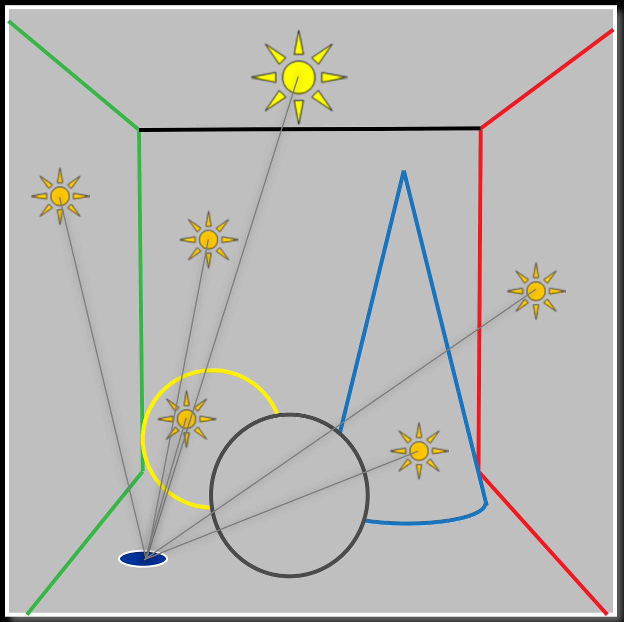
- Approximate indirect illumination by

Virtual Point Lights (VPLs)

1. Generate VPLs



2. Render with VPLs



Instant radiosity as BDPT

- VPLs = light sub-paths
- VPL contributions = sub-path connections

Instant radiosity

- Works well in diffuse scenes
- 100s of VPLs sufficient for ok-ish images
- Basis of many **real-time** GI algorithms

Real-time GI with Instant radiosity

- Reflective shadow maps *[Dachsbacher and Stamminger 05]*
  - Fast VPL generation

- Incremental Instant Radiosity *[Laine et al. 07]*
  - Only a few new VPLs per frame

- Imperfect Shadow Maps *[Ritschel et al. 08]*
  - Faster shadow mapping

Intuition behind VPLs

- There is nothing in global illumination images that a CG artist could not simulate otherwise
- VPLs „automate" the artist approach

Real-time | Minutes | Tens of hours

Slide credit: Miloš Hašan

Clamping & compensation

Kollig and Keller, 2004

- Singularity in light contribution

Biased result with clamping

Unbiased result with compensation

Scalability

Instant radiosity with glossy surfaces

- Large number of VPLs required
  - True even for diffuse scenes
  - Scalability issues

Scalable many-light methods

1. Generate many, many VPLs
2. Use only the most relevant VPLs for rendering

- Choosing the right VPLs
  - Per-image basis
    - Matrix Row Column Sampling [Hašan et al. 07]
  - Per-pixel basis
    - Lightcuts [Walter et al 05/06]

More lights may not do the trick…

artifacts **material change**

Dealing with gloss in many-light methods

Approach #1:

Virtual Spherical Lights

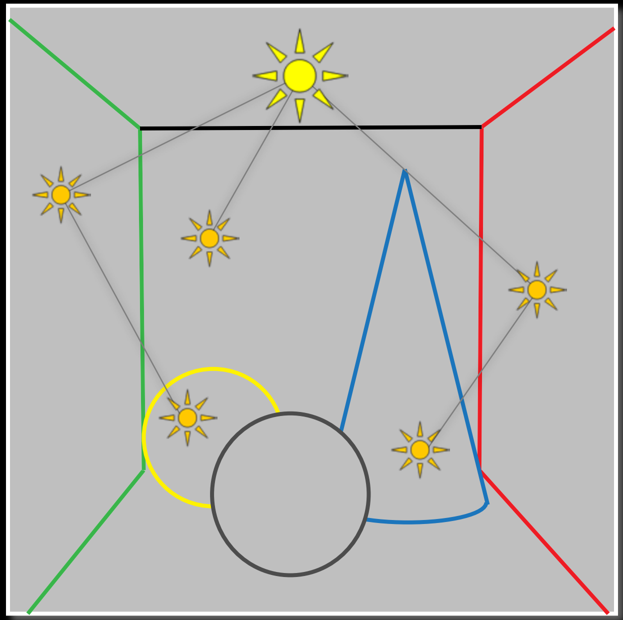
Hašan, Křivánek & Bala, SIGGRAPH Asia 2009

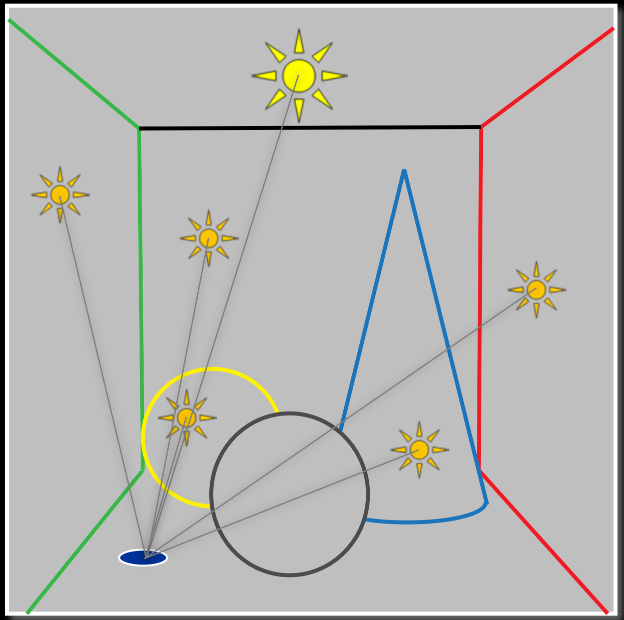
Instant radiosity

- Approximate indirect illumination by

  Virtual Point Lights (VPLs)

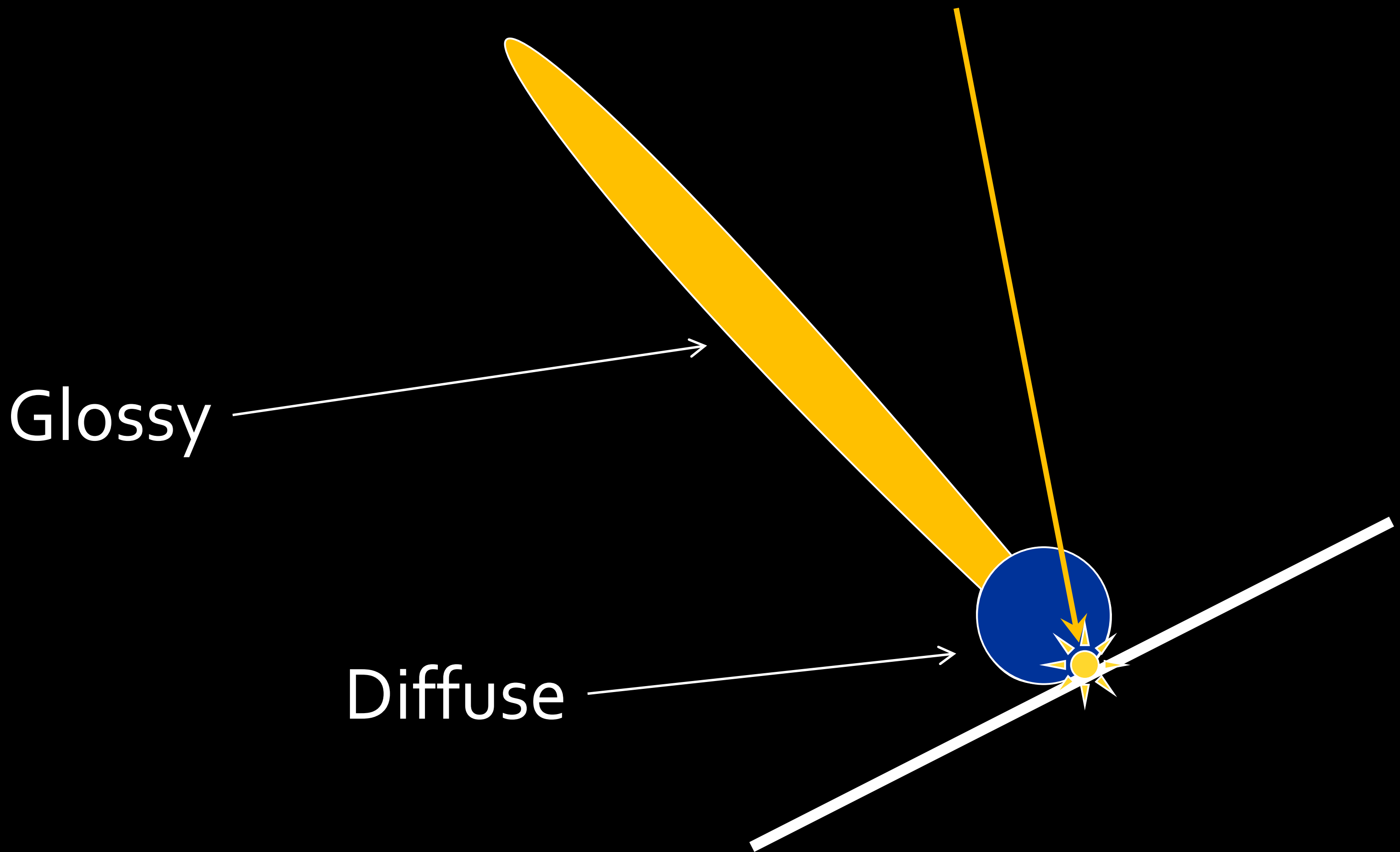
1. Generate VPLs
2. Render with VPLs



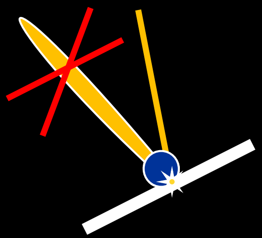


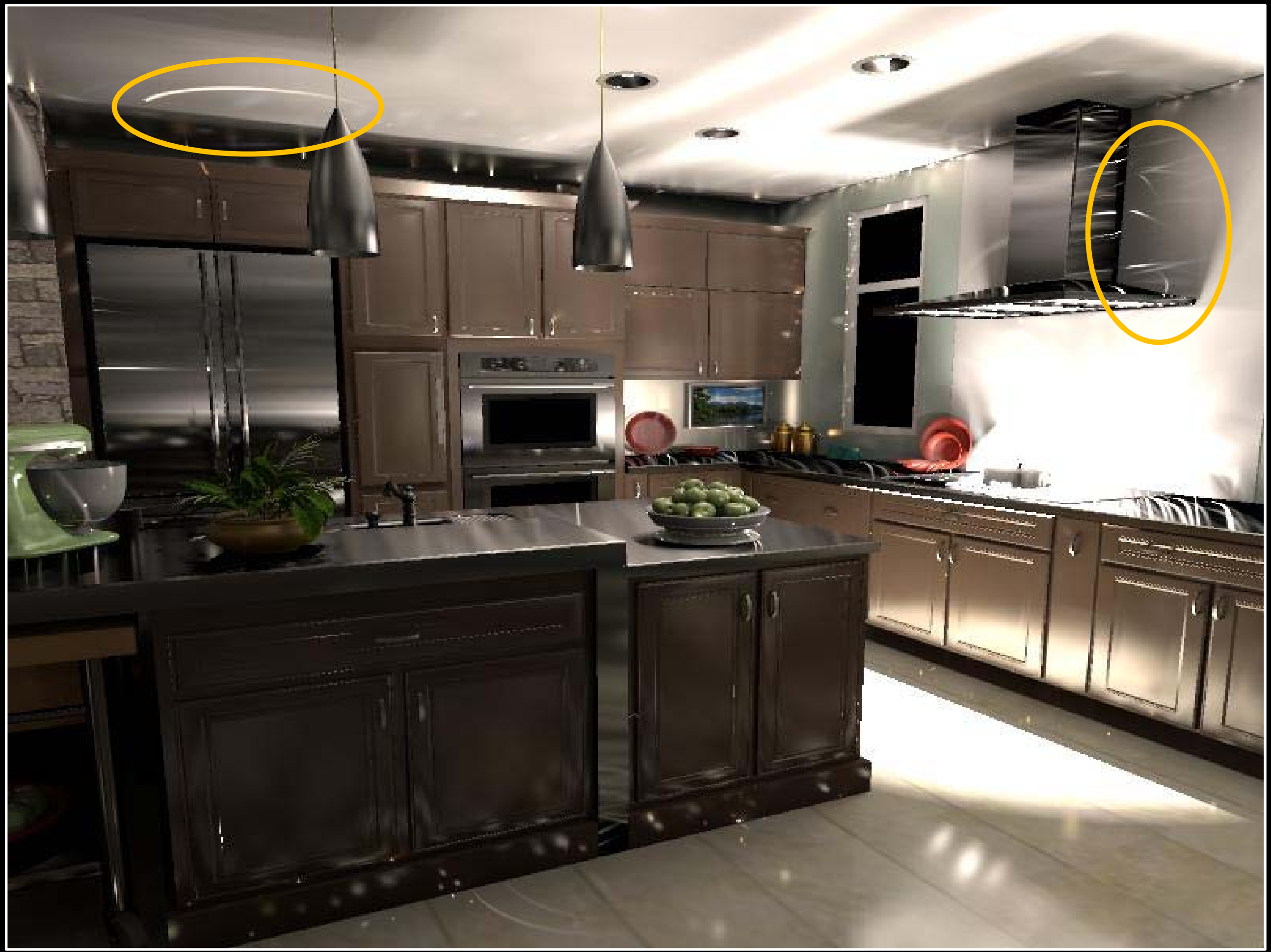
Emission distribution of a VPL

- Cosine-weighted BRDF lobe at the VPL location



Glossy VPL emission: illumination spikes

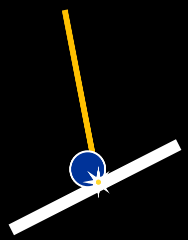


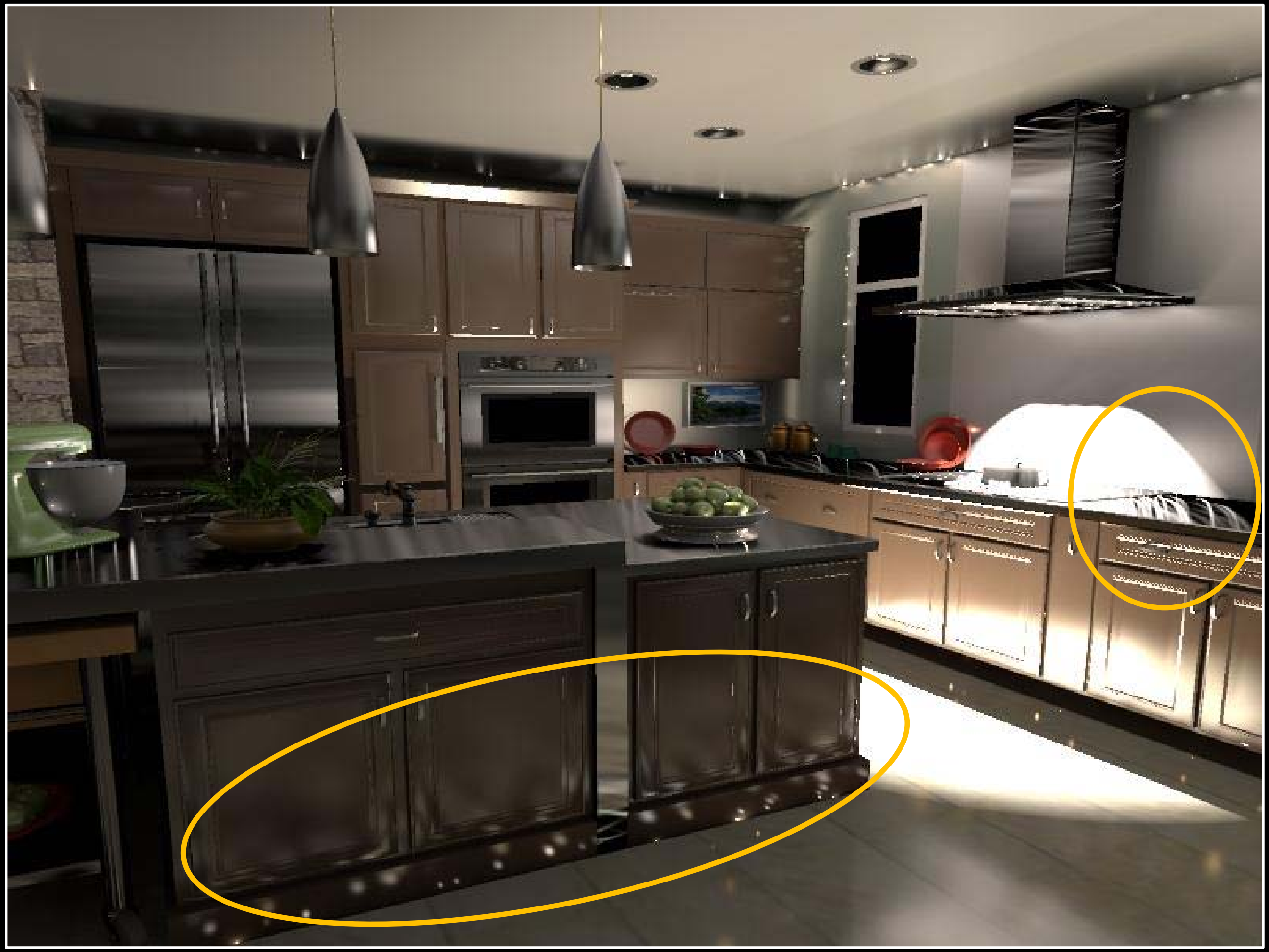


Common solution:

Only **diffuse** BRDF at light location

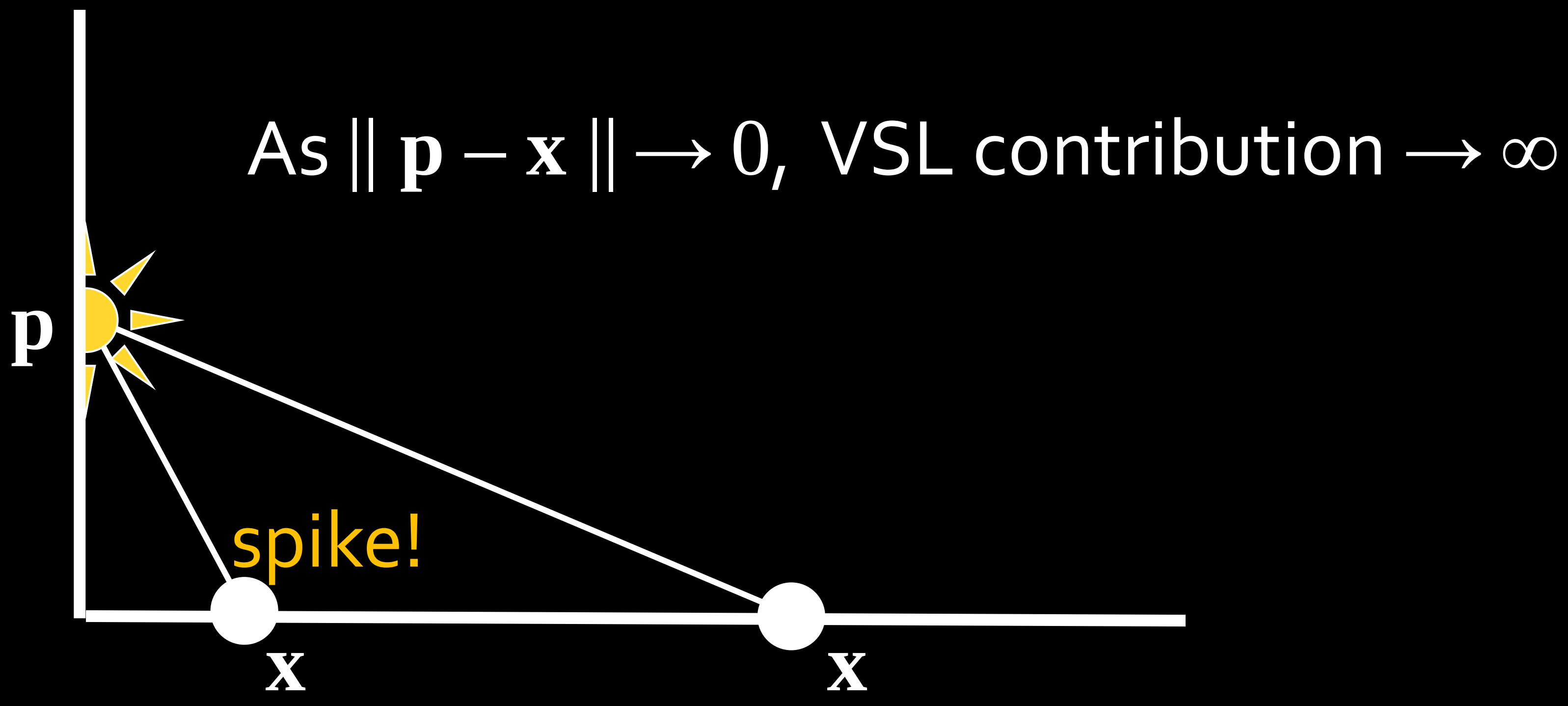
Remaining spikes





Remaining spikes

- VPL contribution =



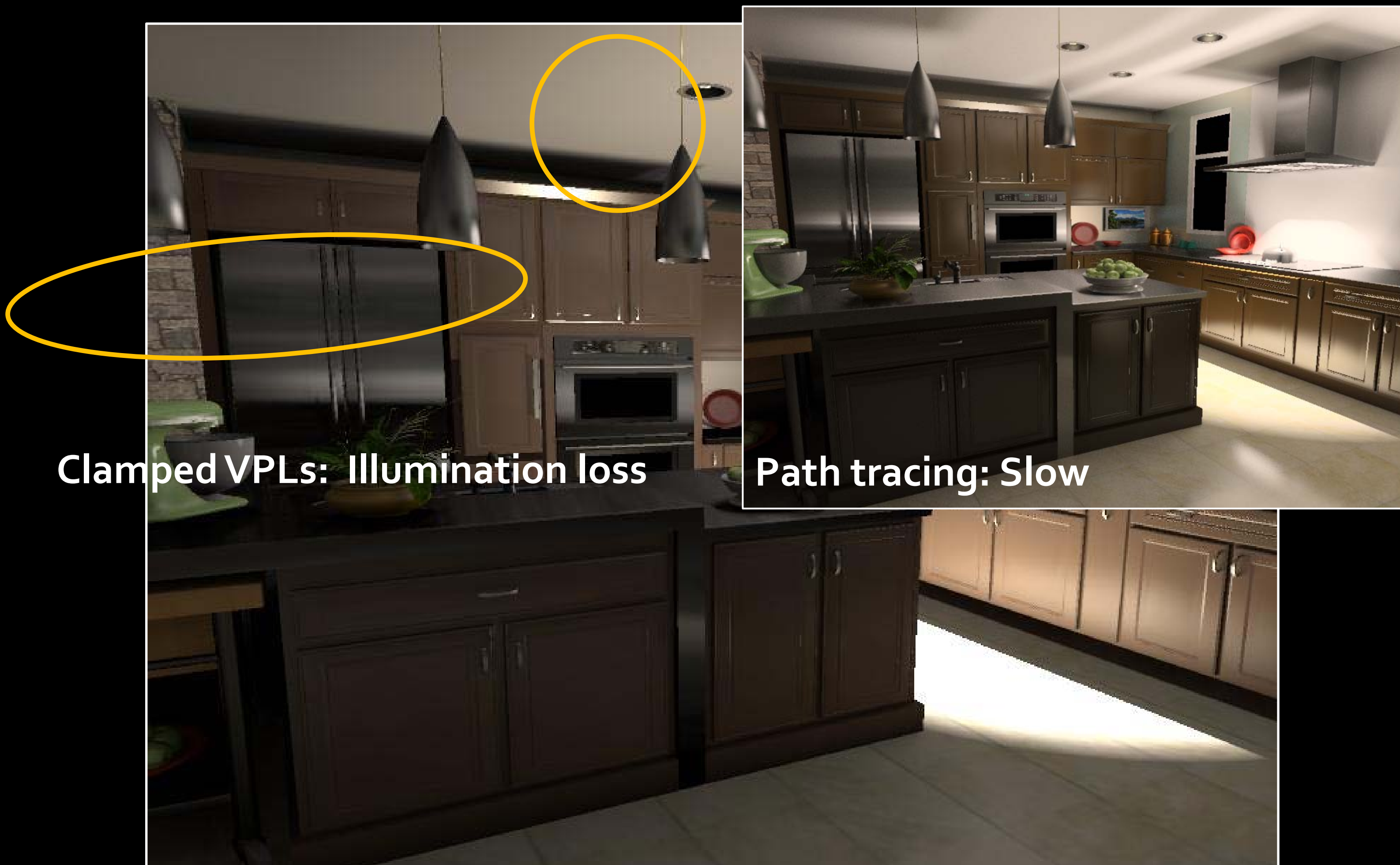
- Common solution: **Clamp** VPL contributions

Instant radiosity: The practical version

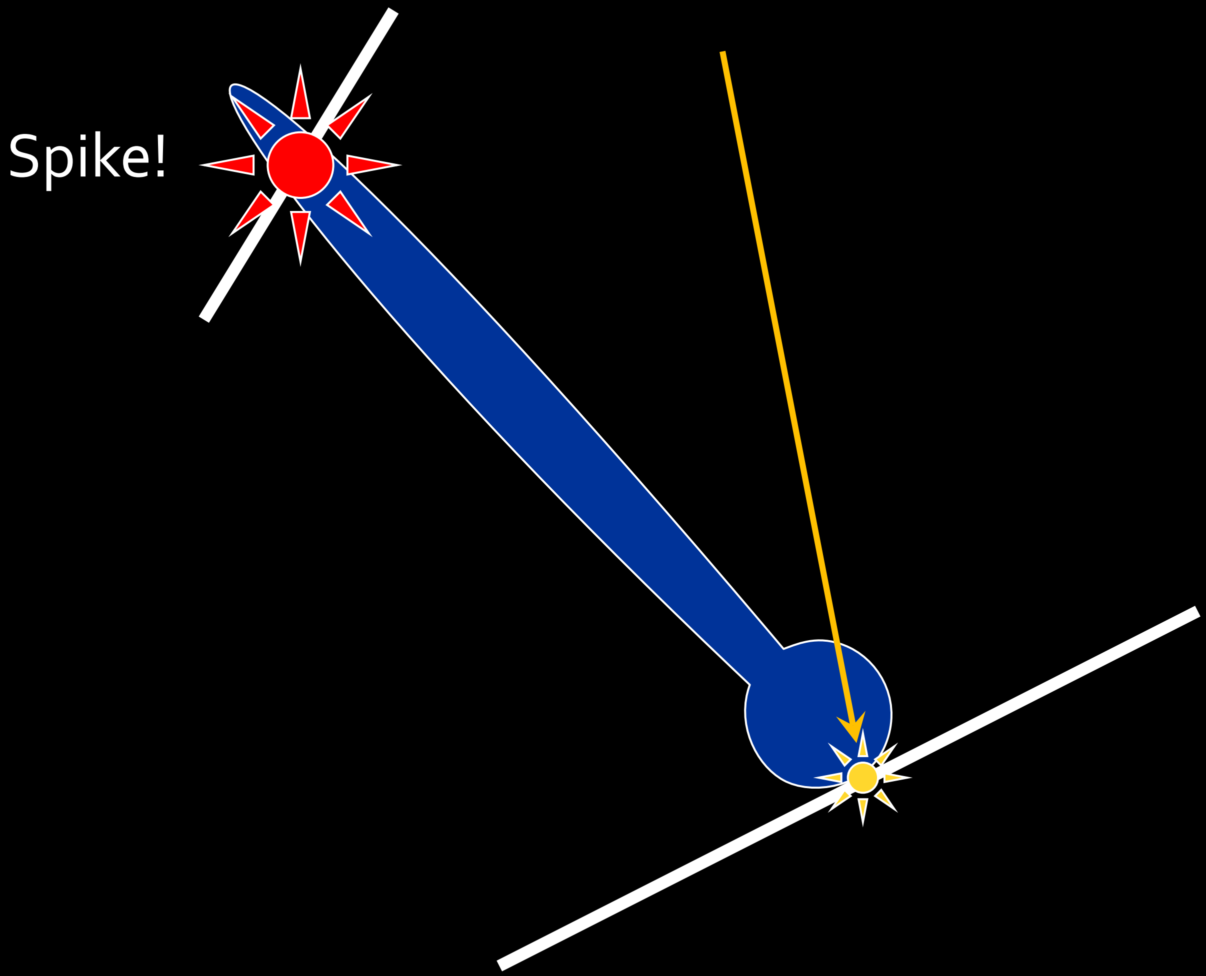
Clamping and diffuse-only VPLs:

Illumination is lost!

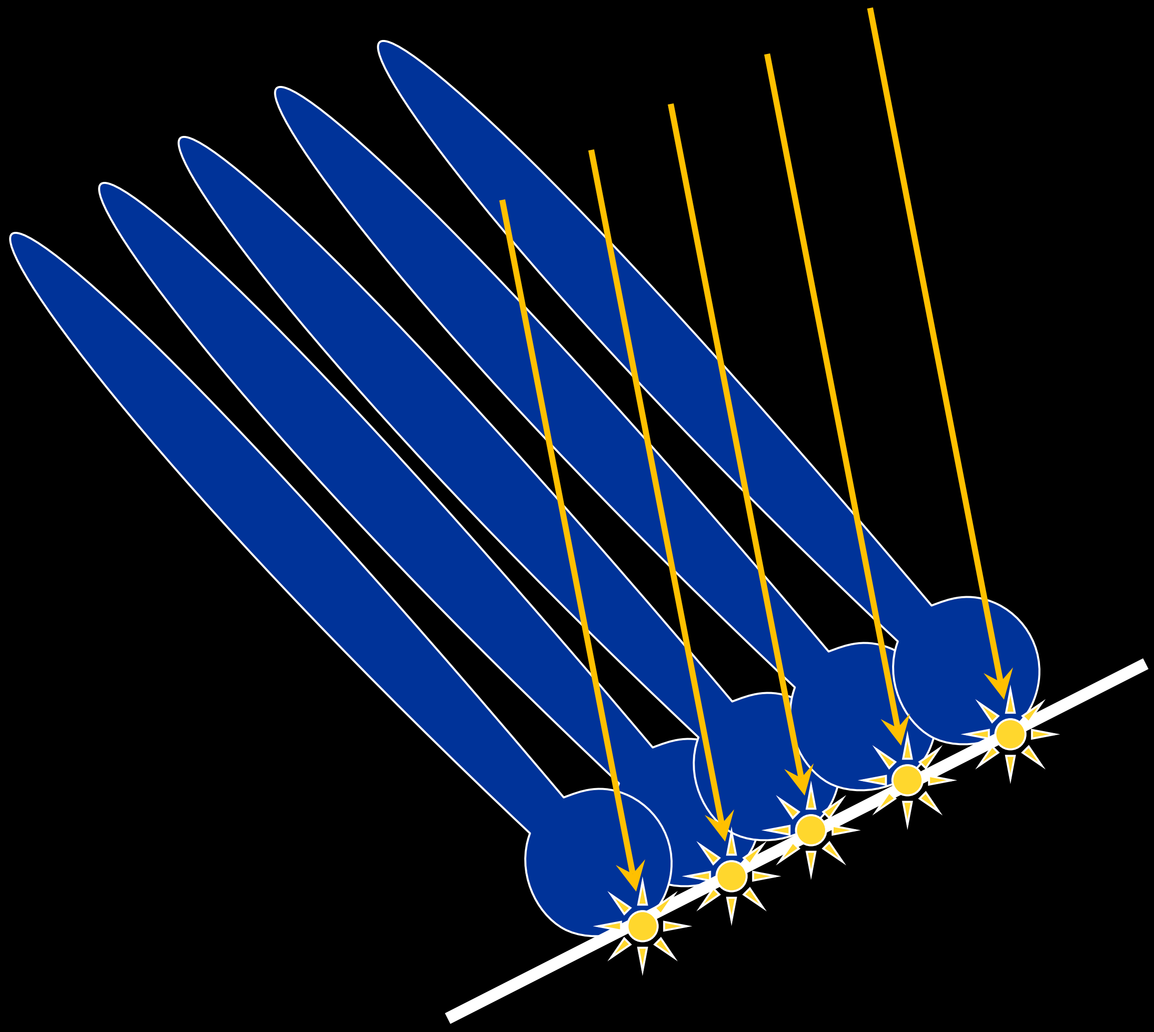
Comparison



Recall: Emission Distribution of a VPL

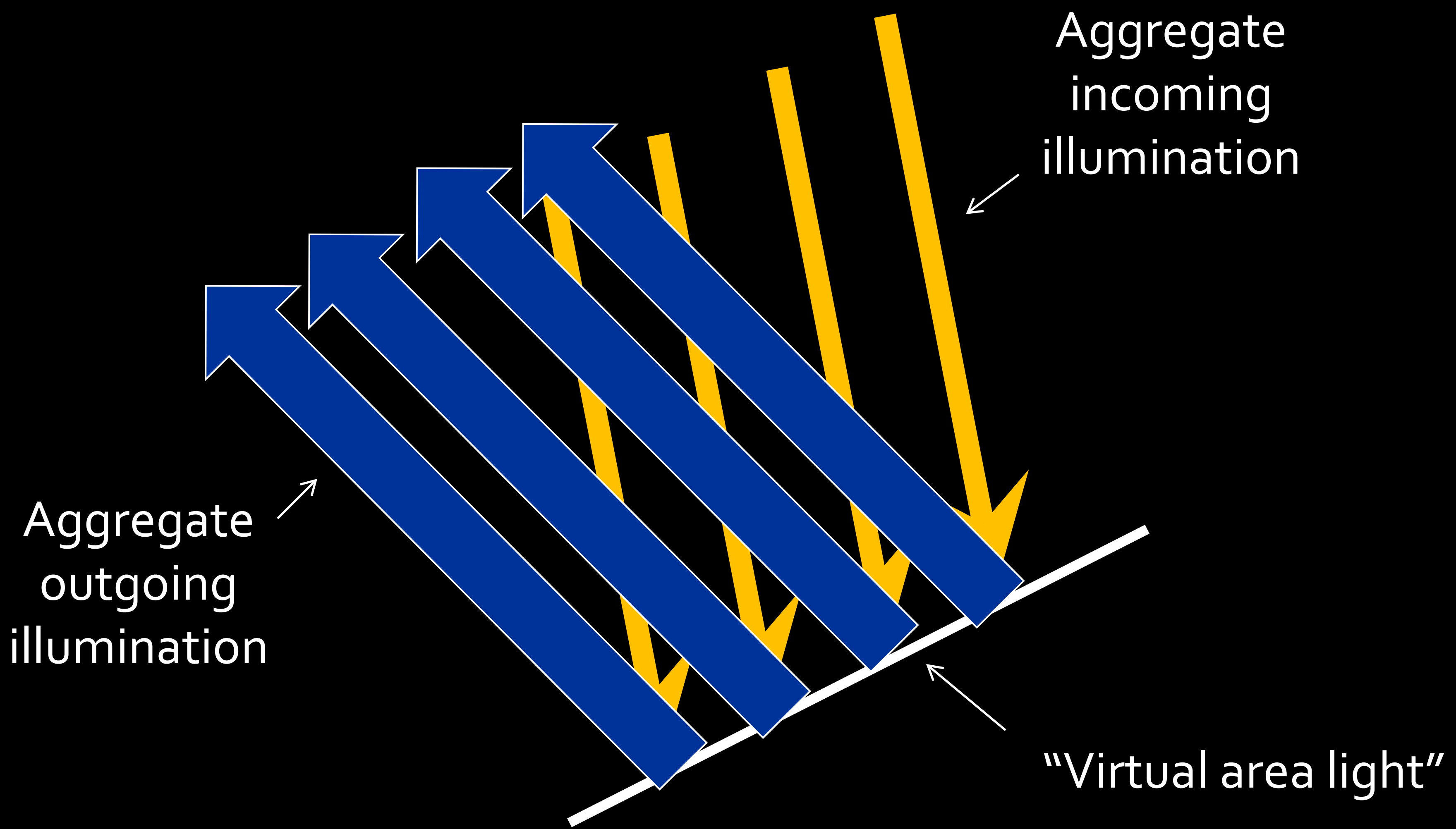


What happens as #lights $\rightarrow \infty$?



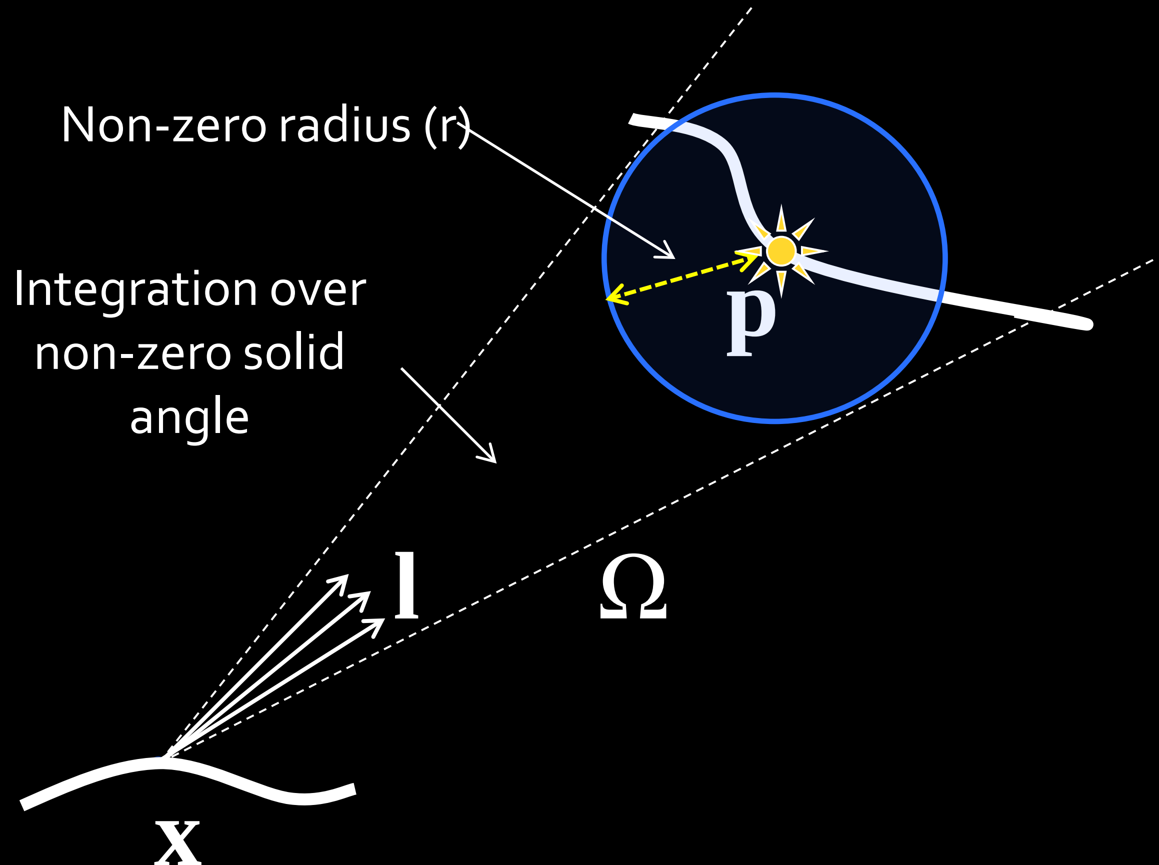
Spiky lights converge to a continuous function!

Idea: We want a “virtual area light”

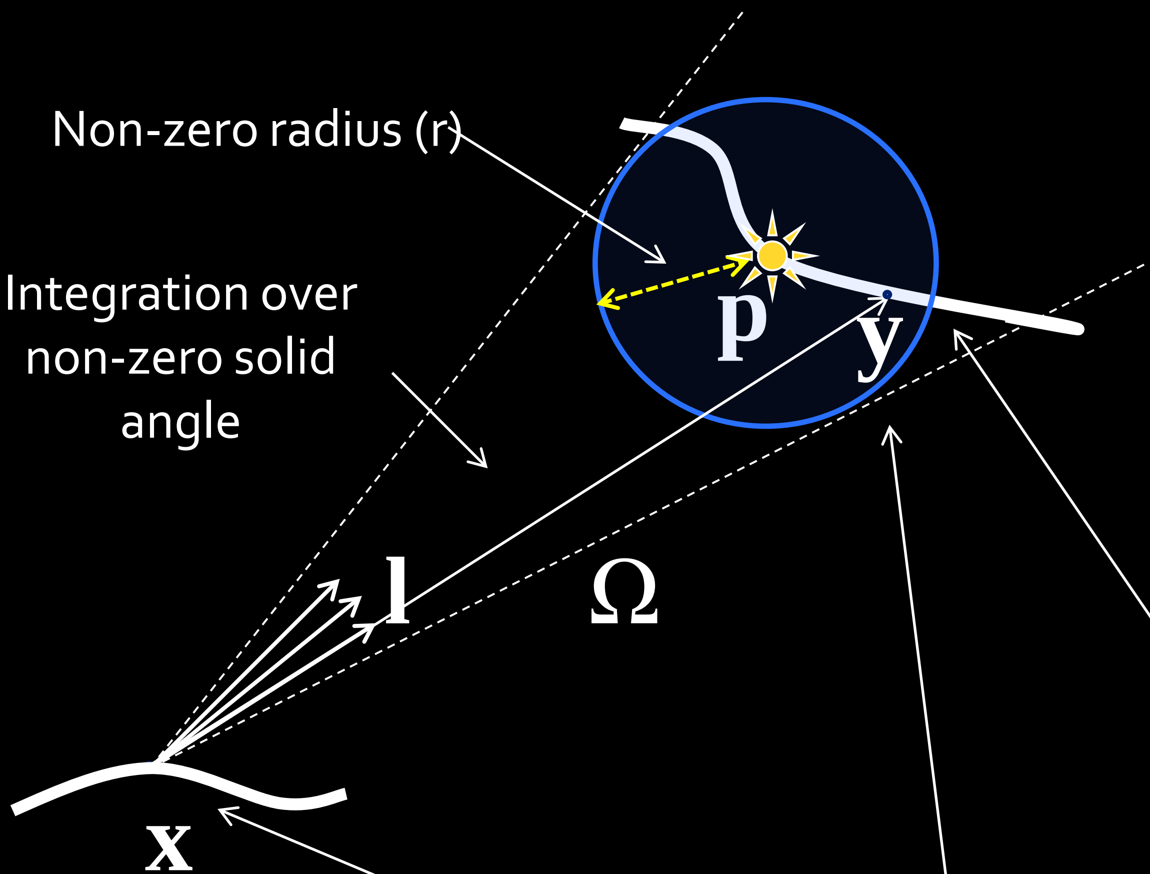


Problem: What if surface is not flat?

VPL to VSL

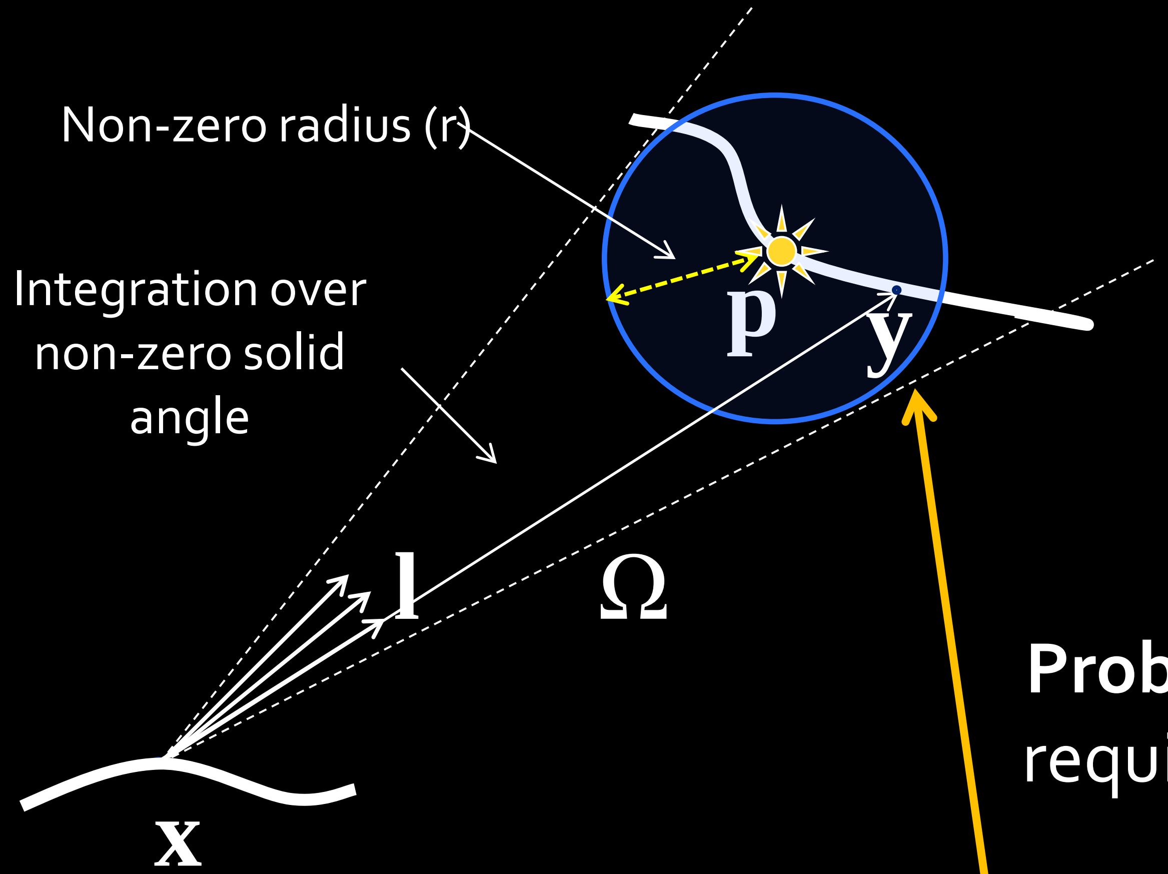

Non-zero radius (r)
Integration over non-zero solid angle
p
l
Ω
x

Light Contribution



$$\int_{\Omega} \quad \quad \quad \mathrm{d}\mathbf{l}$$

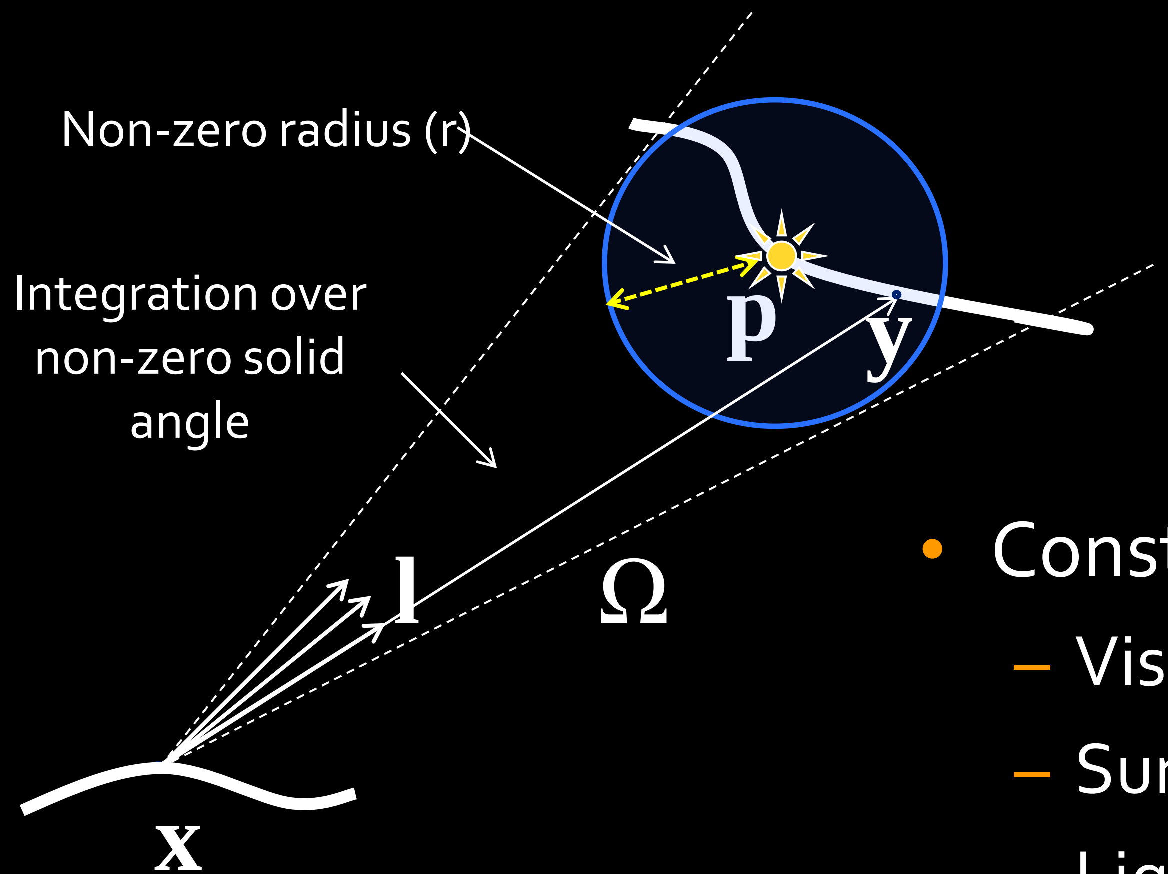
Light Contribution



Problem: Finding **y** requires ray-tracing

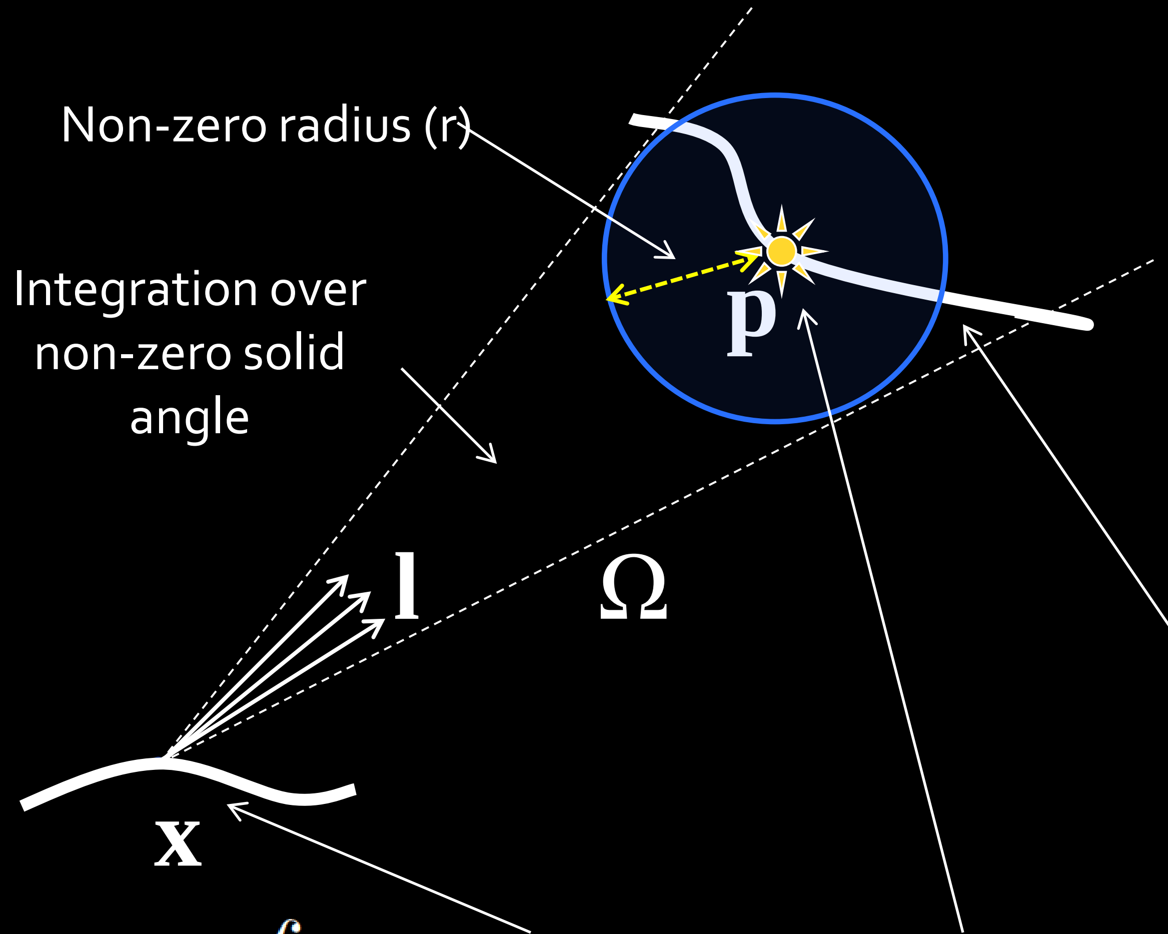
$$\frac{\Phi}{\pi r^2}\int_{\Omega} f_r(\mathbf{x})\cos\theta_{\mathbf{x}}\ f_r(\mathbf{y})\ (\|\mathbf{p}-\mathbf{y}\|<r)\mathrm{d}\mathbf{l}$$

Simplifying Assumptions



- Constant in Ω:
  - Visibility
  - Surface normal
  - Light BRDF
- Taken from **p**, the light location

Light Contribution Updated



$$V \int_{\Omega} \boxed{} \; \boxed{f_r(\mathbf{p})} \; \boxed{\cos\theta_{\mathbf{p}}} \; \mathrm{d}\mathbf{l}$$

Virtual Spherical Light

- All inputs taken from $\mathbf{x}$ and $\mathbf{p}$
  - Local computation
- Same interface as any other light
  - Can be implemented in a GPU shader
- Visibility factored from the integration
  - Can use shadow maps

$$V \frac{\Phi}{\pi r^2} \int_{\Omega} f_r(\mathbf{x}) \cos\theta_{\mathbf{x}} \; f_r(\mathbf{p}) \; \cos\theta_{\mathbf{p}} \; \mathrm{d}\mathbf{l}$$

Implementation

- Matrix row-column sampling
  - Shadow mapping for visibility
  - VSL integral evaluated in a GPU shader

- Need more lights than in diffuse scenes

Results: Kitchen

- Most of the scene lit indirectly
- Many materials glossy and anisotropic

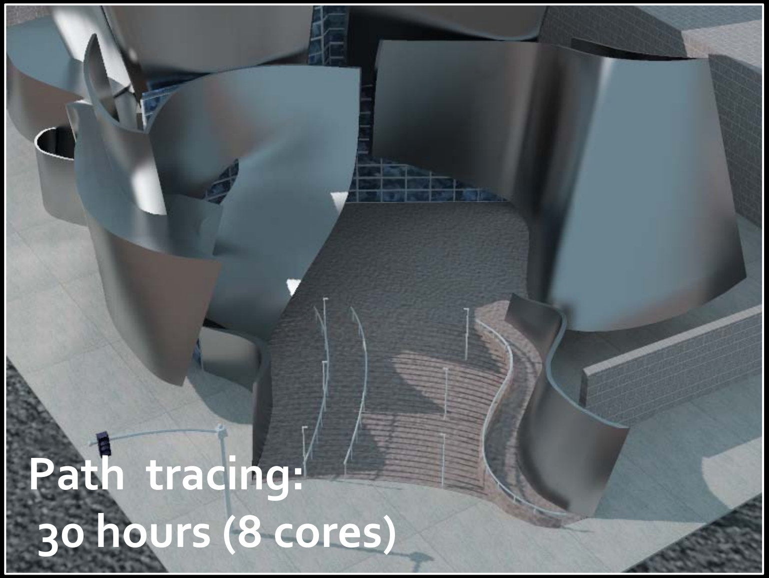
Path tracing:
316 hours (8 cores)

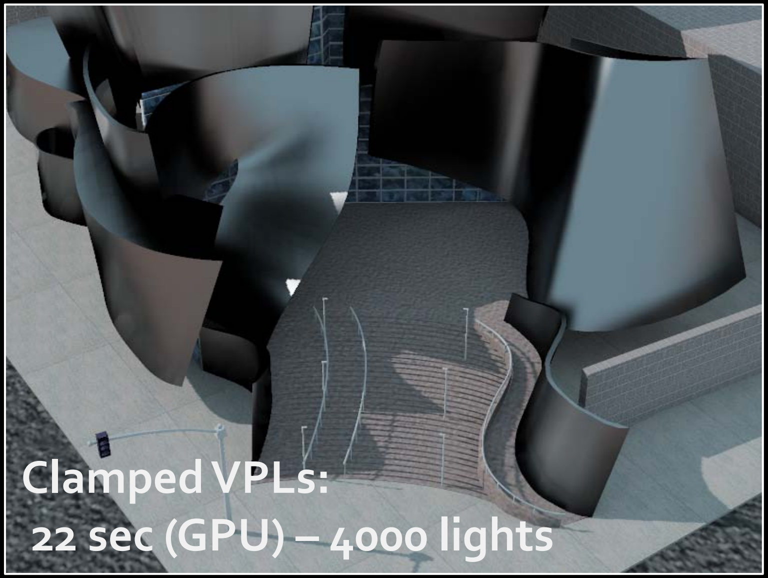
Clamped VPLs
34 sec (GPU) – 2000 lights

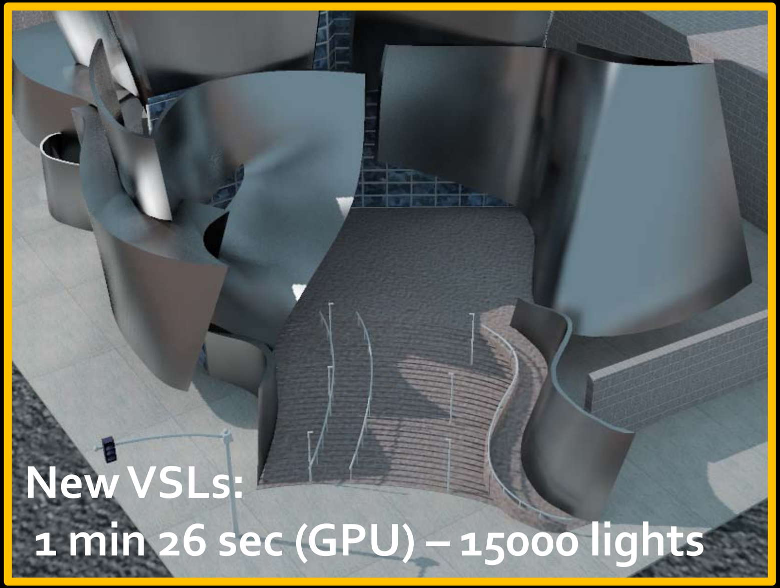
New VSLs:
4 min 4 sec (GPU) – 10000 lights

Results: Disney concert hall

- Curved walls with no diffuse component
- Standard VPLs cannot capture any reflection from walls





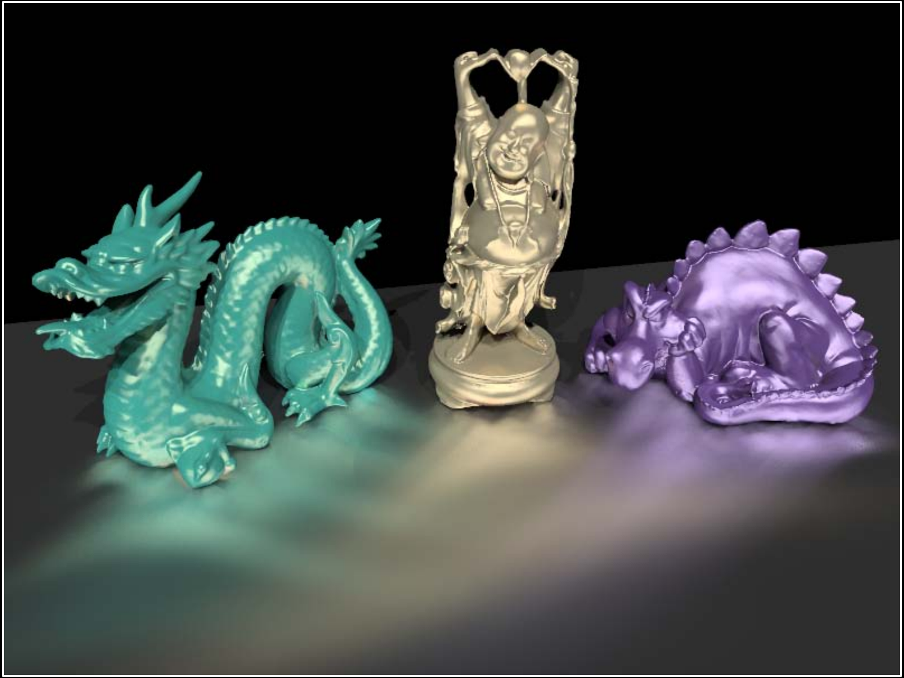


Results: Anisotropic tableau

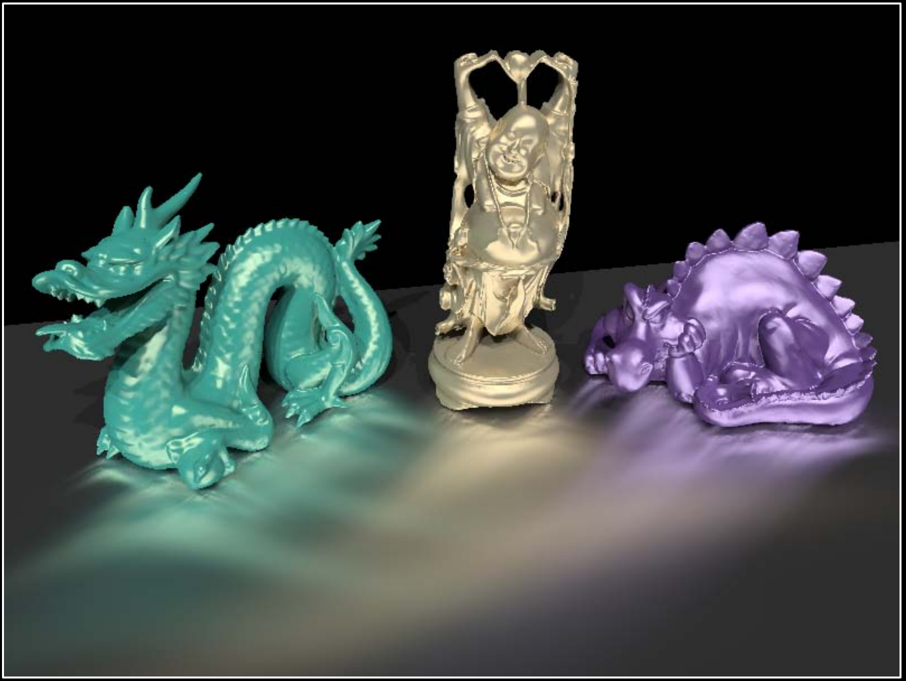
- Difficult case
- Standard VPLs capture almost no indirect illumination

Limitations: Blurring

- VSLs can blur illumination
- Converges as number of lights increases



5,000 lights - blurred



1,000,000 lights - converged

Conclusions

- Many-light methods do not deal well with glossy scenes
  - Artifacts or energy loss
  - Energy loss -> change of material perception
- Handling glossy effects with many-lights
  - Virtual Spherical Lights