Global illumination with many-light methods

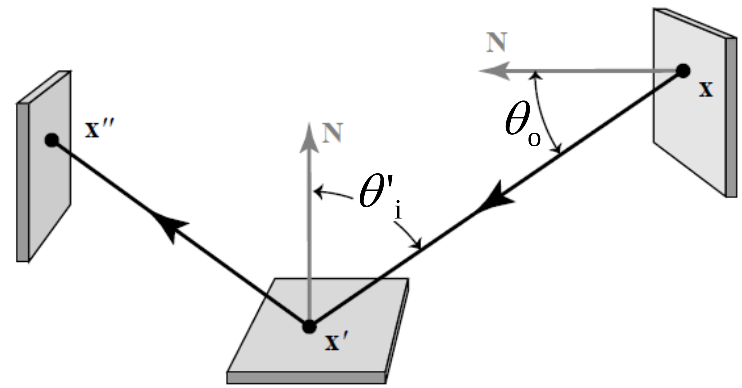
Jaroslav Křivánek

Charles University, Prague

Review:
Path integral formulation of light transport

Veach, 1998

Zobrazovací rovnice v 3b formulaci



$$L(\mathbf{x}' \to \mathbf{x}'') = L_e(\mathbf{x}' \to \mathbf{x}'') + \\ + \int_M L(\mathbf{x} \to \mathbf{x}') \cdot f_r(\mathbf{x} \to \mathbf{x}' \to \mathbf{x}'') \cdot G(\mathbf{x} \leftrightarrow \mathbf{x}') \, dA_{\mathbf{x}}$$

$$G(\mathbf{x} \leftrightarrow \mathbf{x}') = V(\mathbf{x} \leftrightarrow \mathbf{x}') \frac{\left|\cos\theta_o \, \cos\theta_i'\right|}{\left\|\mathbf{x} - \mathbf{x}'\right\|^2}$$

Měřicí rovnice v 3b formulaci

Rovnovážná radiance
(Řešení zobrazovací rovnice)

$$I_j = \int_{M \times M} W_e^{(j)}(\mathbf{x} \to \mathbf{x}') \cdot L(\mathbf{x} \to \mathbf{x}') \cdot G(\mathbf{x} \leftrightarrow \mathbf{x}')\, \mathrm{d}A_{\mathbf{x}}\, \mathrm{d}A_{\mathbf{x}'}$$

Důležitost emitovaná z **x**' do **x**
(Značení: šipka = směr šíření světla, nikoli důležitosti)

x' … na senzoru
x … na ploše scnény

Transport světla jako integrál přes prostor světelných cest

- **Cíl:** místo integrální rovnice chceme formulovat transport světla jako integrál přes cesty:

Příspěvek cesty x k hodnotě pixelu (contribution function)

Míra na množině světelných cest

$$I_j = \int_\Omega f_j(\bar{x})\, \mathrm{d}\mu(\bar{x})$$

Hodnota ("měření") j-tého pixelu

Prostor všech světelných cest
Spojujících zdroj světla s pixelem *j*

Obor integrování

Ω_k ... množina cest délky k

$$\bar{\mathbf{x}} = \mathbf{x}_0 \mathbf{x}_1 \ldots \mathbf{x}_k$$

$$\Omega = \bigcup_{k=1}^{\infty} \Omega_k$$ množina cest všech možných délek

Míra na prostoru cest

Diferenciální míra pro cesty délky k

$$d\mu(\bar{x}) = d\mu(\mathbf{x}_0 \ldots \mathbf{x}_k) \quad = \quad dA_{\mathbf{x}_0} \cdots dA_{\mathbf{x}_k}$$

Tj. násobný integrál přes plochu scény, pro každý vrchol cesty jedna „fajfka“

Aplikace integrálu přes cesty

$$I_j = \int_\Omega f_j(\bar{x})\, d\mu(\bar{x})$$

Odhad integrálu pomocí klasických Monte Carlo metod:

$$I_j \approx \frac{f_j(\bar{X})}{p(\bar{X})}$$

Jak definovat a spočítat hustotu na prostoru cest?

Hustota p-nosti na prostoru cest

- Hustota pravděpodobnosti cesty

$$\bar{\mathbf{x}} = \mathbf{x}_0 \mathbf{x}_1 \ldots \mathbf{x}_k$$

  - Sdružená hustota pozic vrcholů cesty:

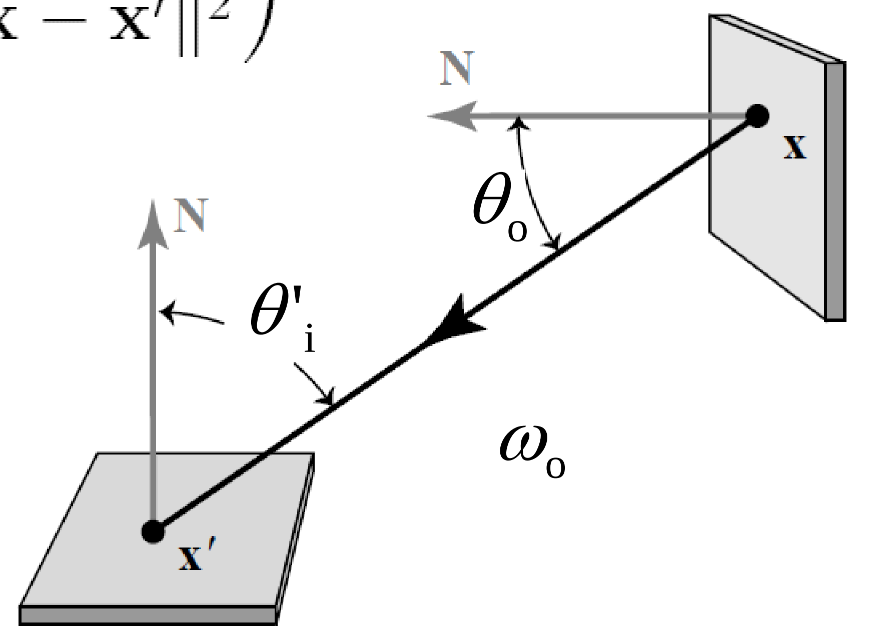
$$\begin{aligned} p(\bar{\mathbf{x}}) &= p(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k) \\ &= p(\mathbf{x}_0)\, p(\mathbf{x}_1 \,|\, \mathbf{x}_0)\, p(\mathbf{x}_2 \,|\, \mathbf{x}_0, \mathbf{x}_1) \ldots \end{aligned}$$

  - Součin podmíněných hustot pro jednotlivé vrcholy (vzhledem k plošné míře)

Hustota pro vzrokování směru

- Hustota p-nosti není invariantní vůči míře
- Nutno konvertovat z dω na dA

$$p(\mathbf{x}') \quad = \quad p(\omega_o) \left(\frac{|\cos(\theta'_i)|}{\|\mathbf{x} - \mathbf{x}'\|^2} \right)$$



Instant radiosity

Keller, 1997

Instant radiosity

Instant Radiosity

SIGGRAPH 1997

Alexander Keller*

Universität Kaiserslautern

Abstract

We present a fundamental procedure for instant rendering from the radiance equation. Operating directly on the textured scene description, the very efficient and simple algorithm produces photorealistic images without any finite element kernel or solution discretization

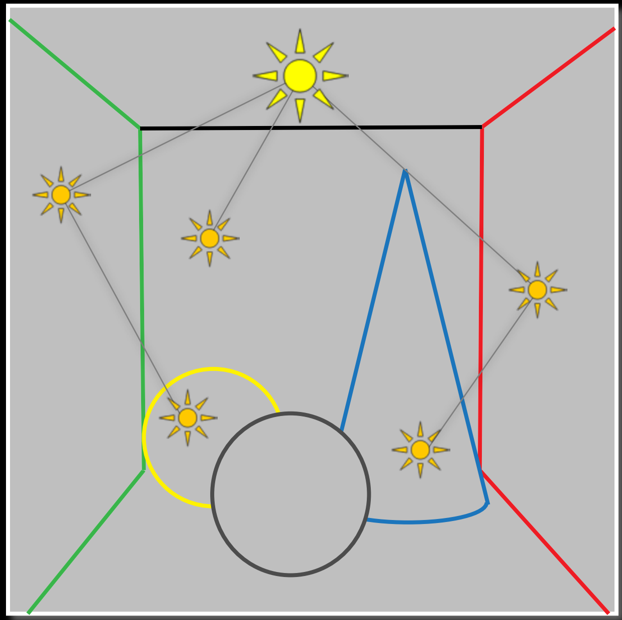
lation scheme based on low discrepancy sampling has been introduced in [Kel96b]. This deterministic scheme converges smoother at a slightly superior rate and exposes no variance as compared to stochastic algorithms. In bidirectional path tracing [LW93, VG94], even the discretization of the solution of the radiance equation has been avoided, but the rendering time is far from realtime.

- http://dl.acm.org/citation.cfm?id=258769
- The “original” many-light method
- Probably the first GPU-based GI algorithm

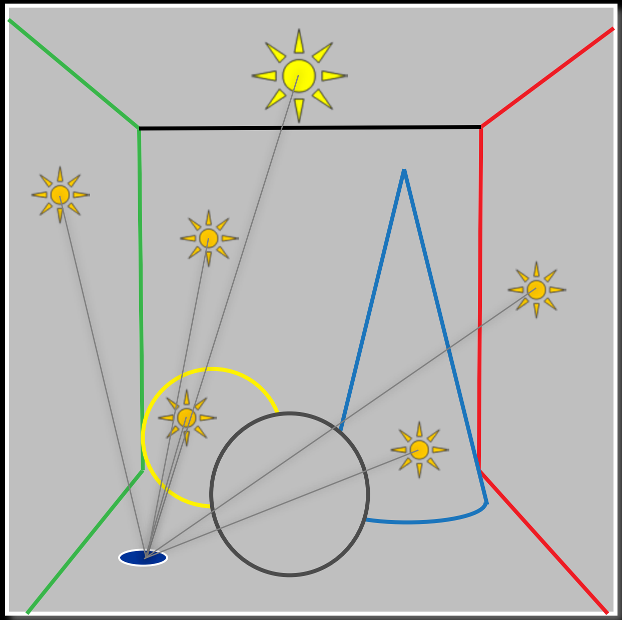
Instant radiosity

- Approximate indirect illumination by
  Virtual Point Lights (VPLs)

1. Generate VPLs



2. Render with VPLs



VPL Tracing

- Exactly the same as photon tracing
  - see e.g. CG III slides: http://cgg.mff.cuni.cz/~jaroslav/teaching/2011-pg3/slides/krivanek-10-npgr010-2011-pm.pptx

VPL Tracing

1. Pick a light source

2. Pick an initial point an direction

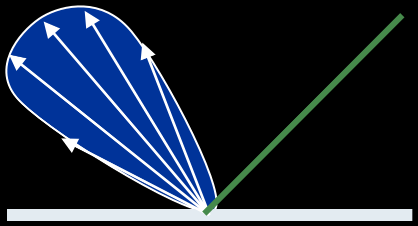
3. Trace particle

4. Create a VPL (photon) at every non-specular surface intersection

VPL

- **Diffuse VPL**
  - Position, surface normal
  - “Power”

- **Glossy VPL**
  - Position, surface normal
  - “Power”
  - BRDF parameters at VPL position
  - Incident direction



VPL contribution

$$\text{VPL contrib} =$$

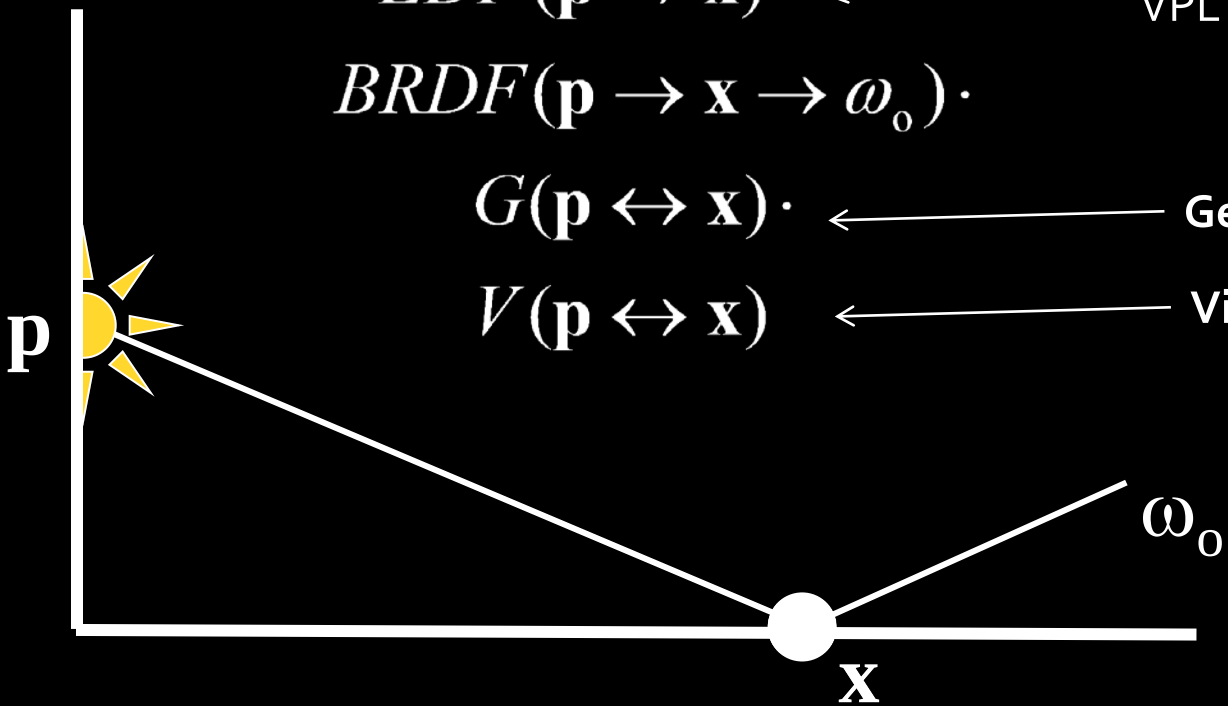
$$\Phi \cdot$$ ← **VPL power**

$$EDF(\mathbf{p} \to \mathbf{x}) \cdot$$ ← **VPL emission distribution** (BRDF lobe at p – for a diffuse VPL can be folded into Φ)

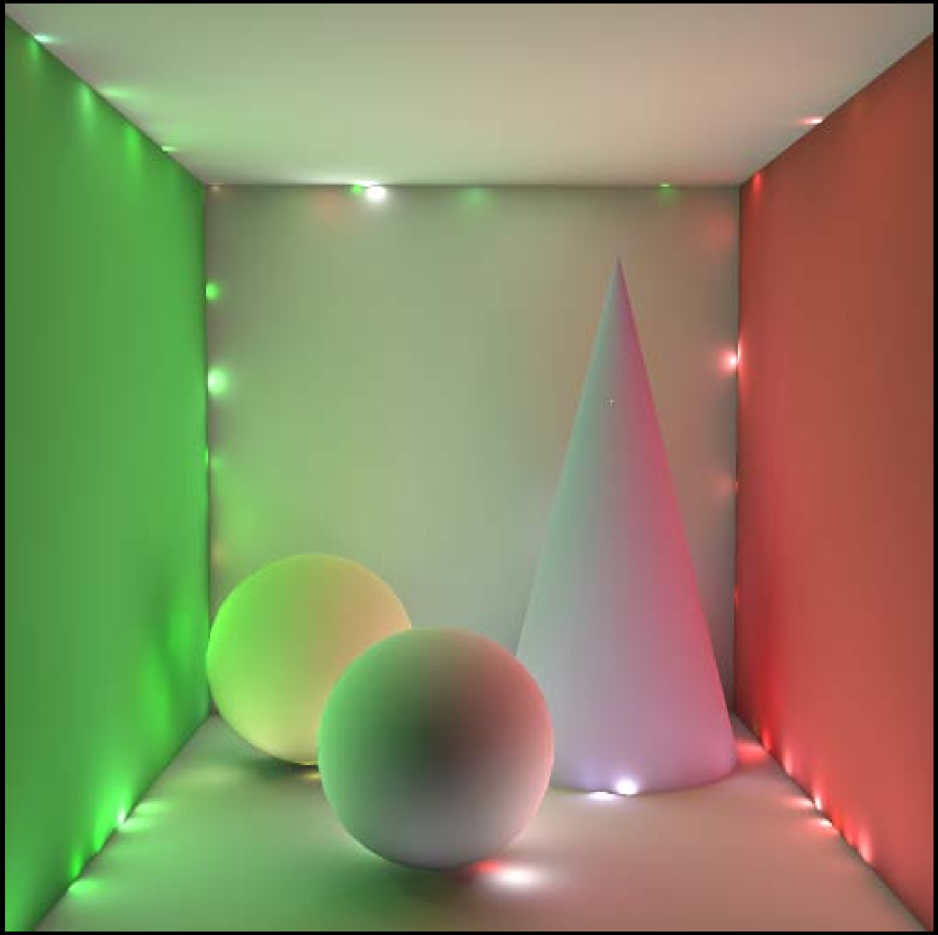
$$BRDF(\mathbf{p} \to \mathbf{x} \to \omega_o) \cdot$$

$$G(\mathbf{p} \leftrightarrow \mathbf{x}) \cdot$$ ← **Geometry term**

$$V(\mathbf{p} \leftrightarrow \mathbf{x})$$ ← **Visibility**



Effect of variance



"correlated noise"

Getting rid of variance – Clamping

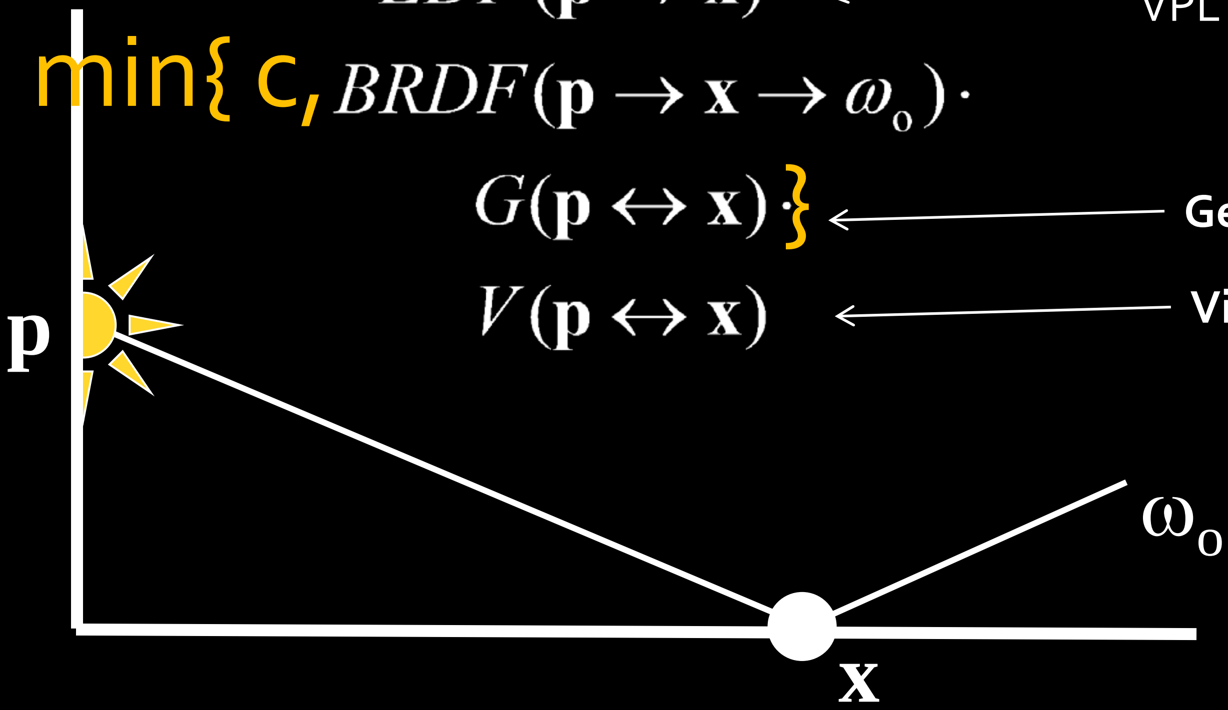
$$\text{VPL contrib} = \Phi \cdot EDF(\mathbf{p} \rightarrow \mathbf{x}) \cdot \min\{\, c,\ BRDF(\mathbf{p} \rightarrow \mathbf{x} \rightarrow \omega_o) \cdot G(\mathbf{p} \leftrightarrow \mathbf{x}) \} \cdot V(\mathbf{p} \leftrightarrow \mathbf{x})$$

VPL power → Φ

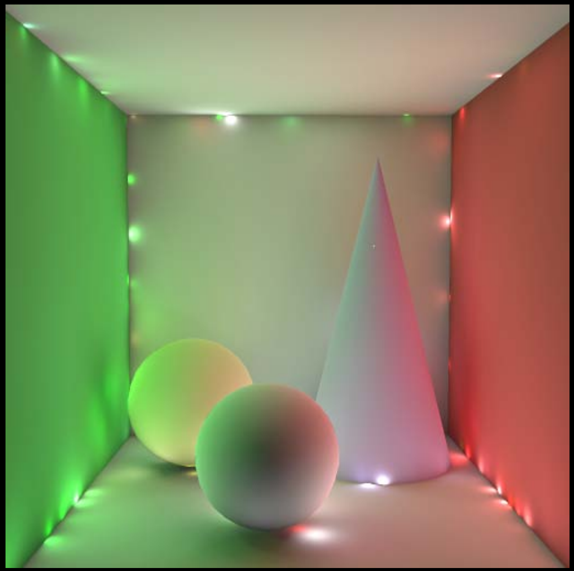
VPL emission distribution (BRDF lobe at p – for a diffuse VPL can be folded into Φ) → $EDF(\mathbf{p} \rightarrow \mathbf{x})$

Geometry term → $G(\mathbf{p} \leftrightarrow \mathbf{x})$

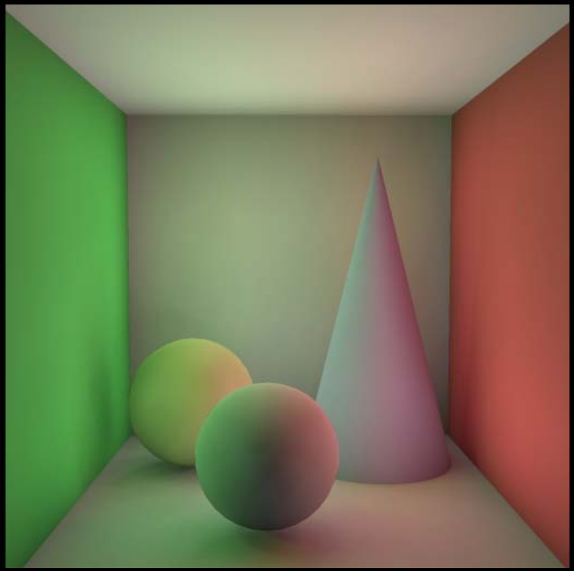
Visibility → $V(\mathbf{p} \leftrightarrow \mathbf{x})$



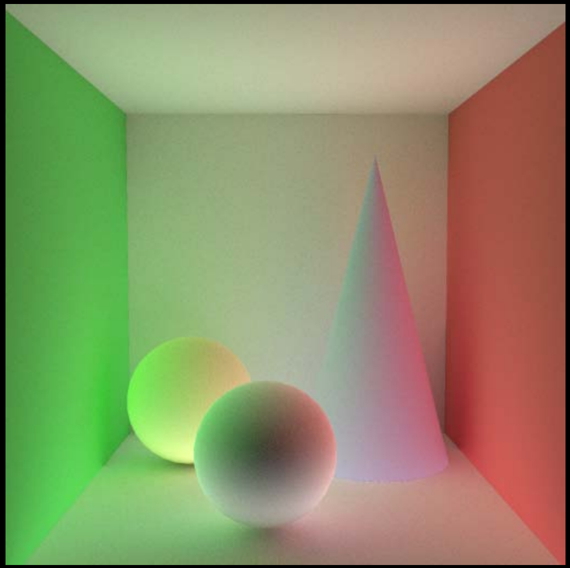
Effect of clamping



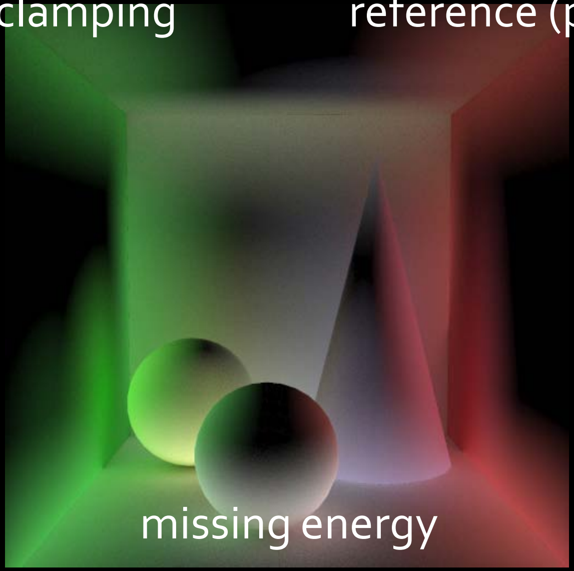
1000 VPLs - no clamping



1000 VPLs - clamping



reference (path tracing)



IR as a path-sampling technique

- VPLs = light sub-paths
- VPL contributions = sub-path connections
- Path splitting at VPL position

Instant radiosity

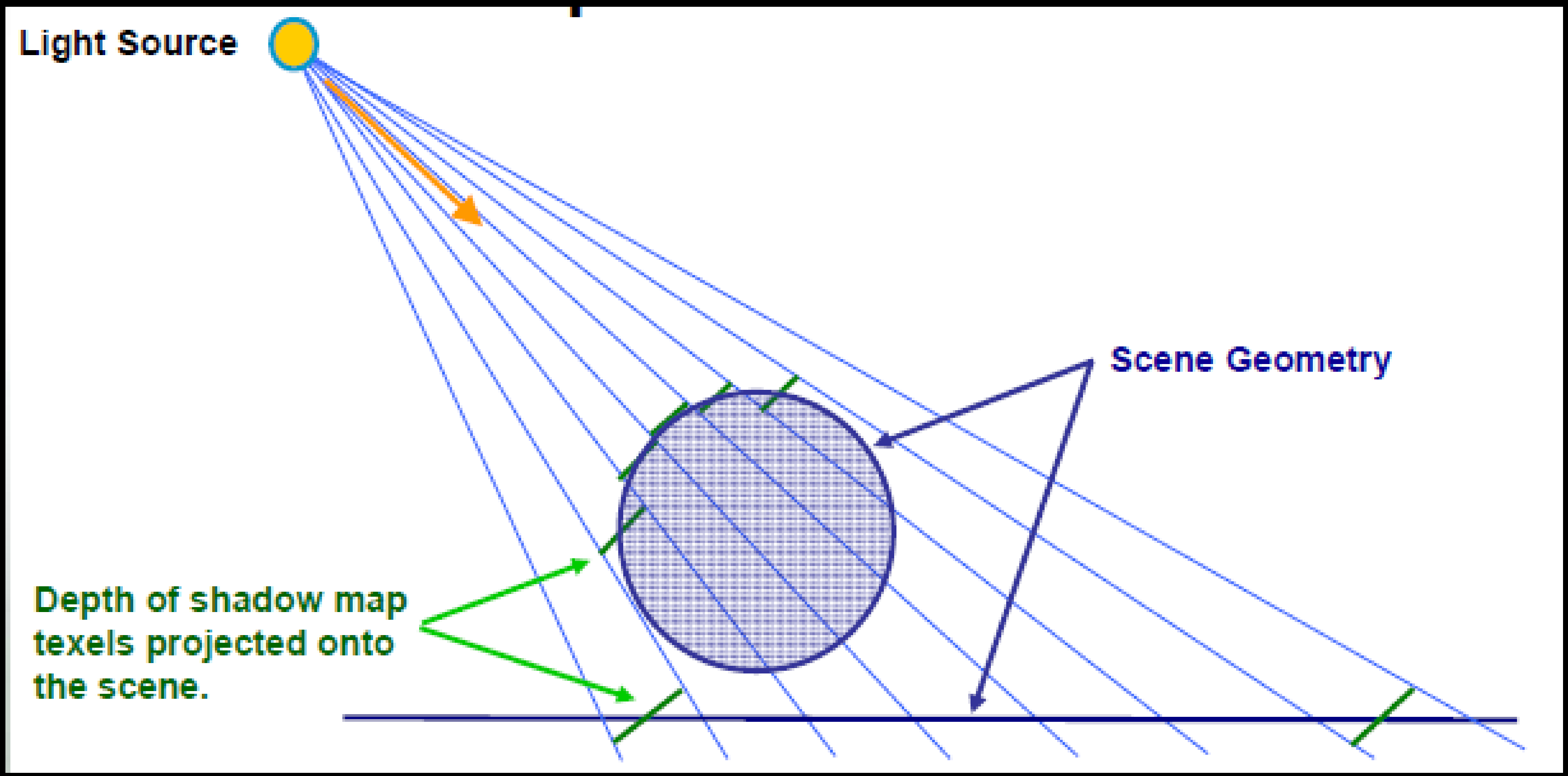
- Works great in diffuse scenes
- 100s of VPLs sufficient for ok-ish images
- Basis of many **real-time** GI algorithms
- Efficiency: accumulate VPL contribs using GPU (shadow mapping for visibility)

IR: Results from the original paper

- 128 VPLs

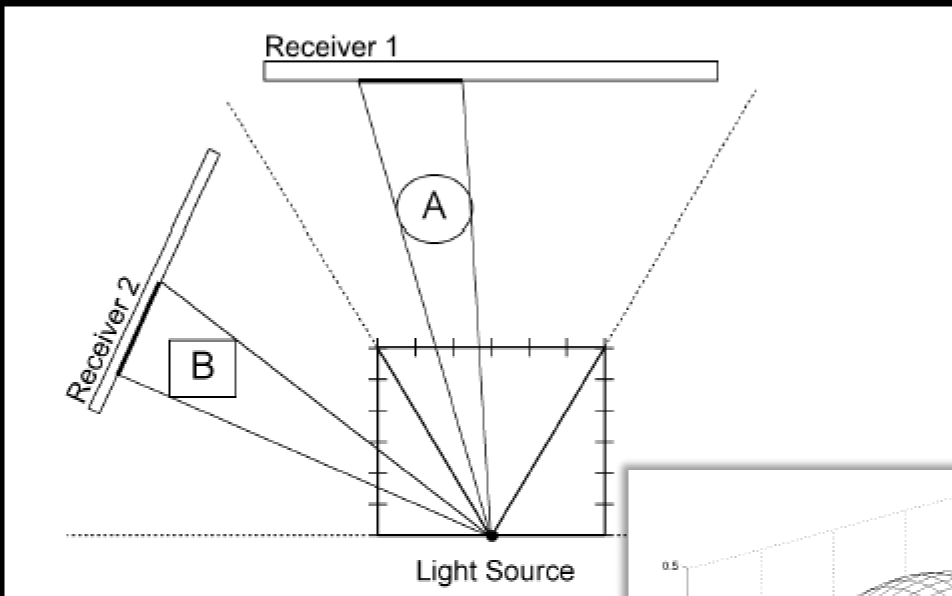
 

Digression: Shadow Mapping



Digression: Shadow Mapping

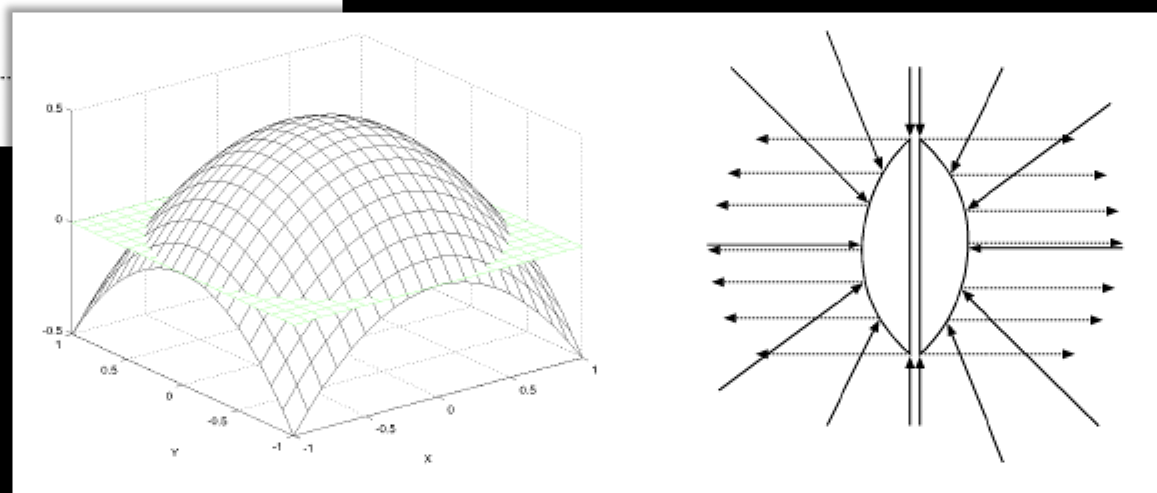
- Shadow maps for 180 degree lights (VPLs)



Images: Brabec et al. 2002

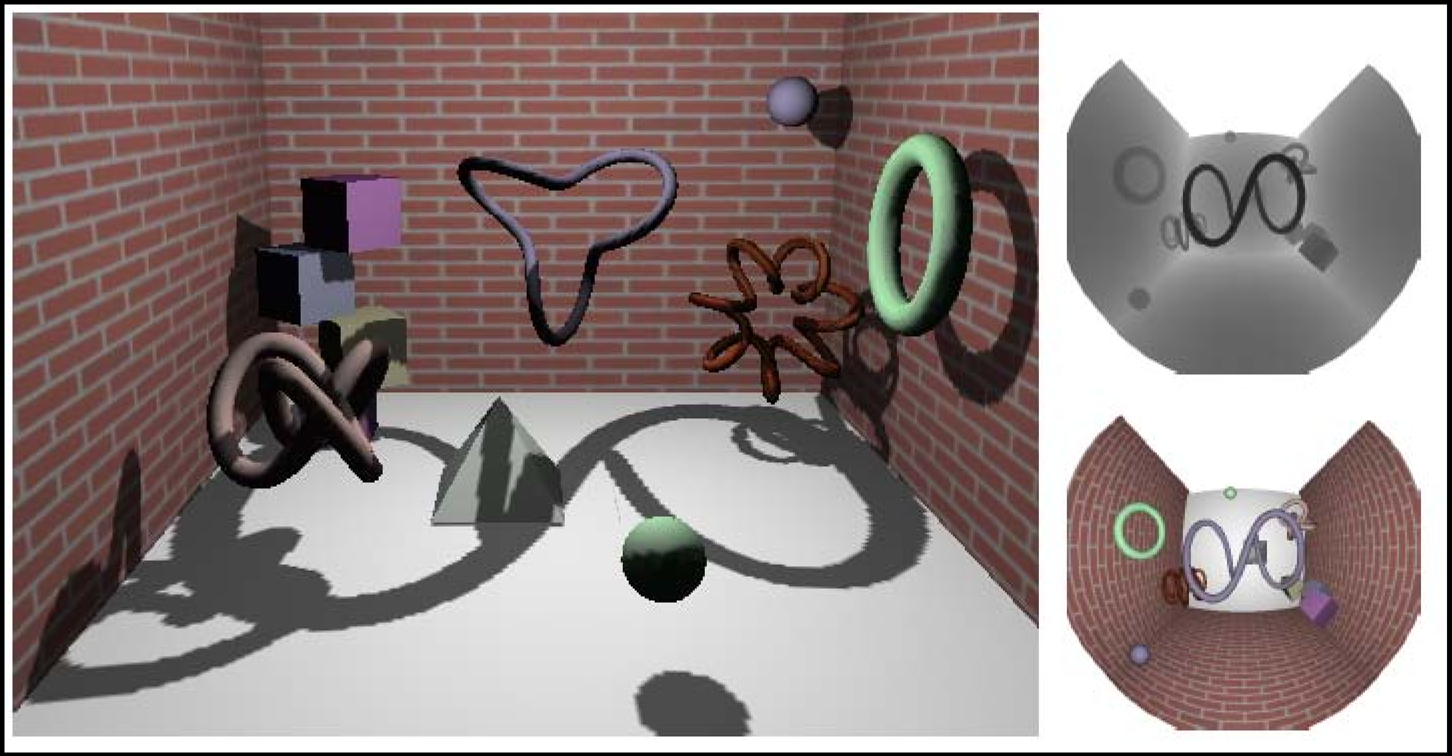
Option 1:
Hemicube shadow maps.
slow (render scene 5 times)

Option 2:
Paraboloid mapping



Digression: Shadow Mapping

- Paraboloid shadow mapping



Images: Brabec et al. 2002

Real-time GI with instant radiosity

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 map rendering

Reflective shadow maps

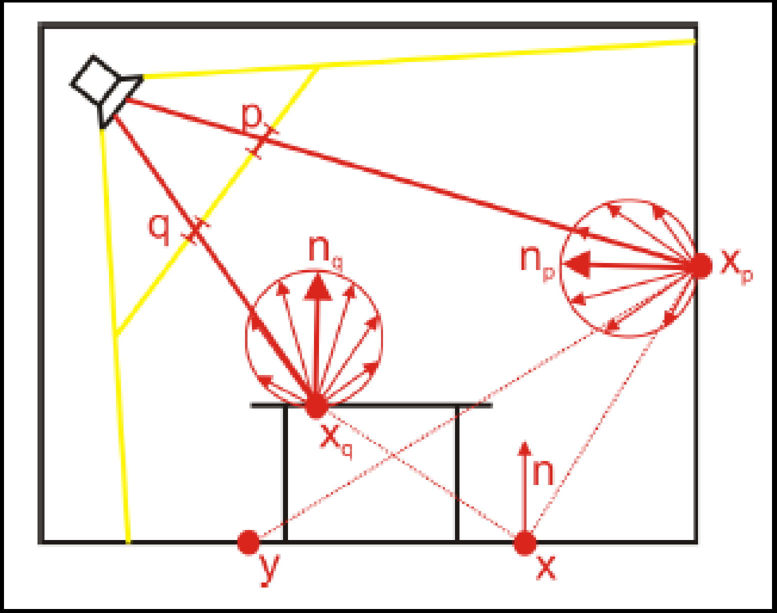
Reflective Shadow Maps

I3D 2005

Carsten Dachsbacher*
University of Erlangen-Nuremberg

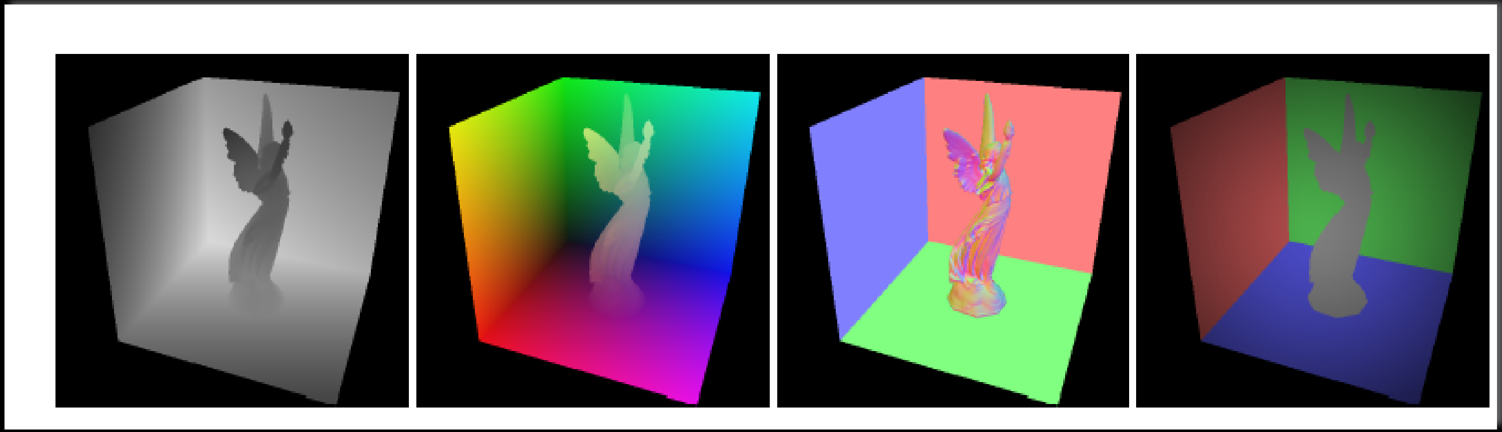
Marc Stamminger†
University of Erlangen-Nuremberg

- http://cg.ibds.kit.edu/publikationen.php
- **Key idea**: Interpret shadow map pixels as VPLs



Reflective shadow maps

- **Key idea**: Interpret shadow map pixels as VPLs



Reflective shadow maps

- **Key idea**: Interpret shadow map pixels as VPLs
- Problem
  - Too many SM pixels -> too many VPLs
- Solution
  - Subsample the RSM
  - Different samples for each pixel

Reflective shadow maps

- Consider **x** at which we compute indirect illum.
  - Project **x** onto the RSM
  - Select RSM pixels close to the projection

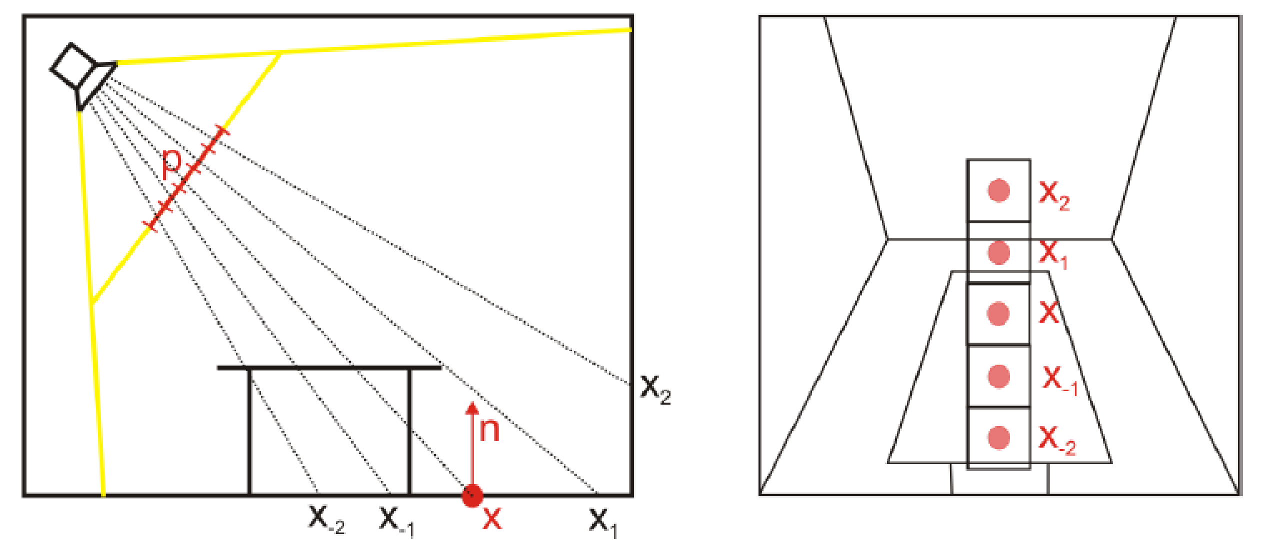
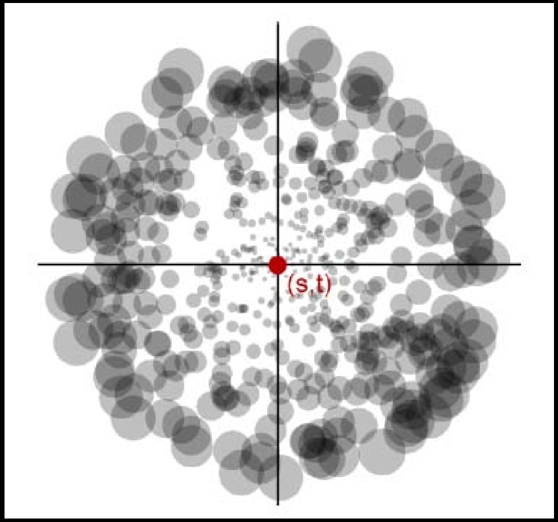


Figure 3: RSM sampling



Sampling pattern w/ sample weights

Reflective shadow maps

- Only one-bounce indirect illumination

- Further optimizations
  - no visibility testing in indirect calculation
  - screen-space subsampling

- Results
  - 5fps for 400 VPLs on an GeForce Quadro FX4000

Incremental Instant Radiosity

Incremental Instant Radiosity for Real-Time Indirect Illumination

EGSR 2007

Samuli Laine[1,3] Hannu Saransaari[3] Janne Kontkanen[2,3] Jaakko Lehtinen[3,4] Timo Aila[1,3]

[1]NVIDIA Research [2]PDI/DreamWorks
[3]Helsinki University of Technology [4]Remedy Entertainment

- *http://www.tml.tkk.fi/~samuli/*

- **Key idea**: reuse VPLs from previous frames

VPL Reuse

- Reuse VPLs from previous frame
  - Generate as few new VPLs as possible
  - Stay within budget, e.g. 4-8 new VPLs/frame
- \+ Benefit: Can reuse shadow maps!
- ! **Disclaimer: Scene needs to be static (only light positions can change)**

Slide courtesy Samuli Laine

How To Reuse VPLs

- Every frame, do the following:
  - Delete invalid VPLs
  - Reproject existing VPLs to a 2D domain according to the new light source position
  - Delete more VPLs if the budget says so
  - Create new VPLs
  - Compute VPL intensities

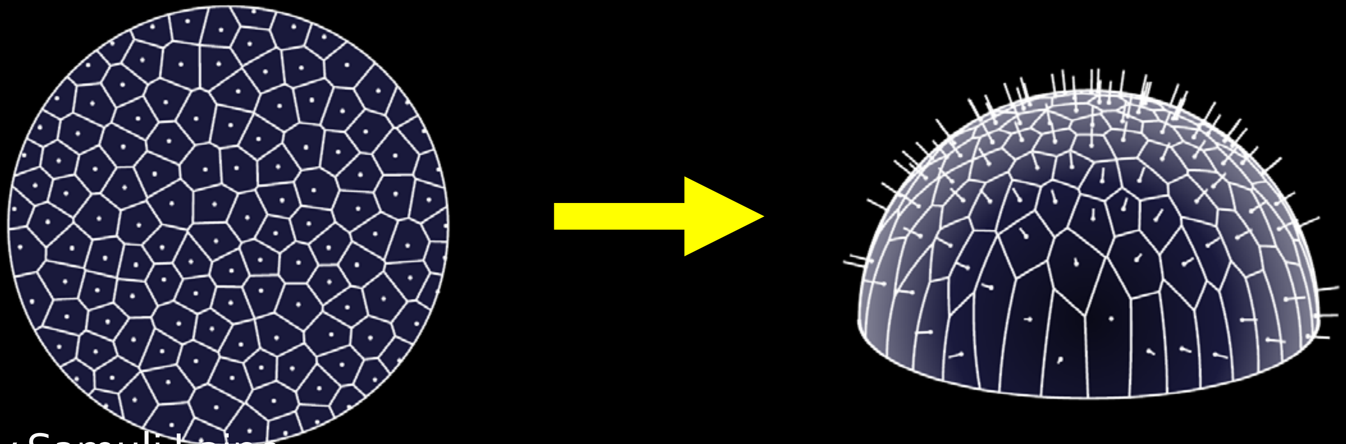
Slide courtesy Samuli Laine

2D Domain for VPLs

- Let's concentrate on 180° cosine-falloff spot lights for now
- Nusselt analog

  Uniform distribution in unit disc

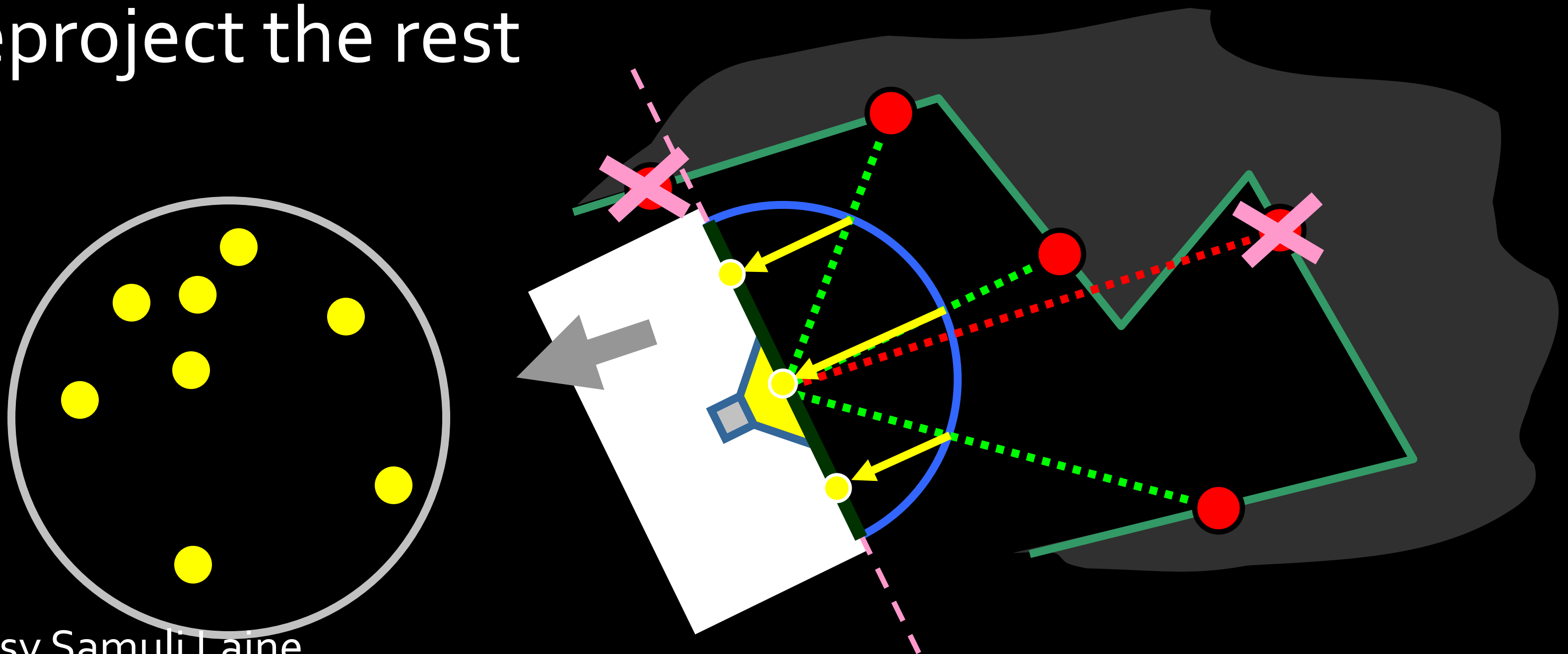
  = Cosine-weighted directional distribution



Slide courtesy Samuli Laine

Reprojecting VPLs

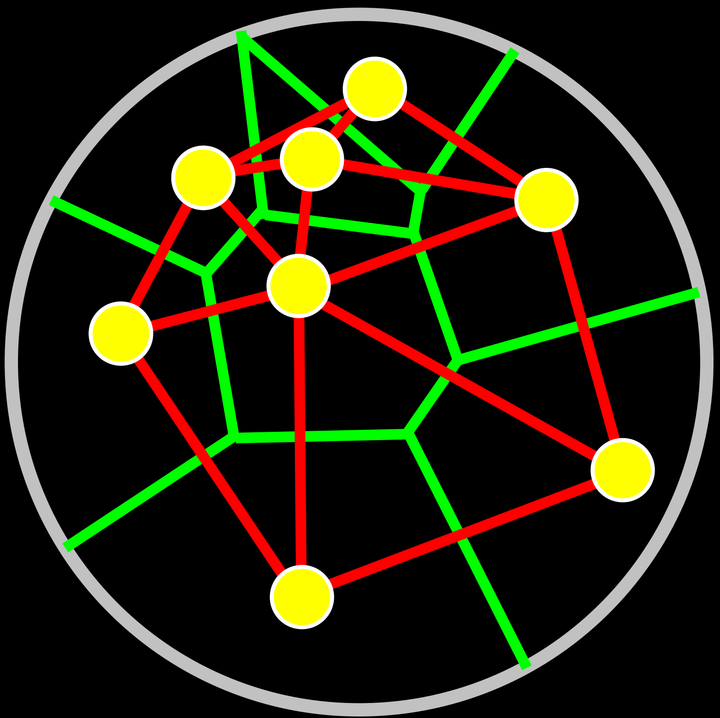
- So we have VPLs from previous frame
- Discard ones behind the spot light
- Discard ones behind obstacles
- Reproject the rest



Slide courtesy Samuli Laine

Spatial Data Structures

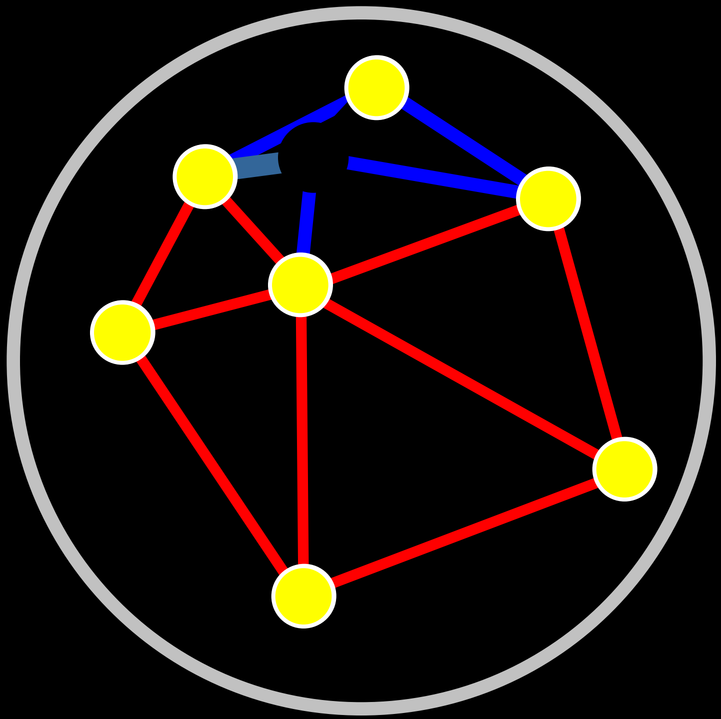
- Compute Voronoi diagram and Delaunay triangulation for the VPL point set



Slide courtesy Samuli Laine

Deleting VPLs

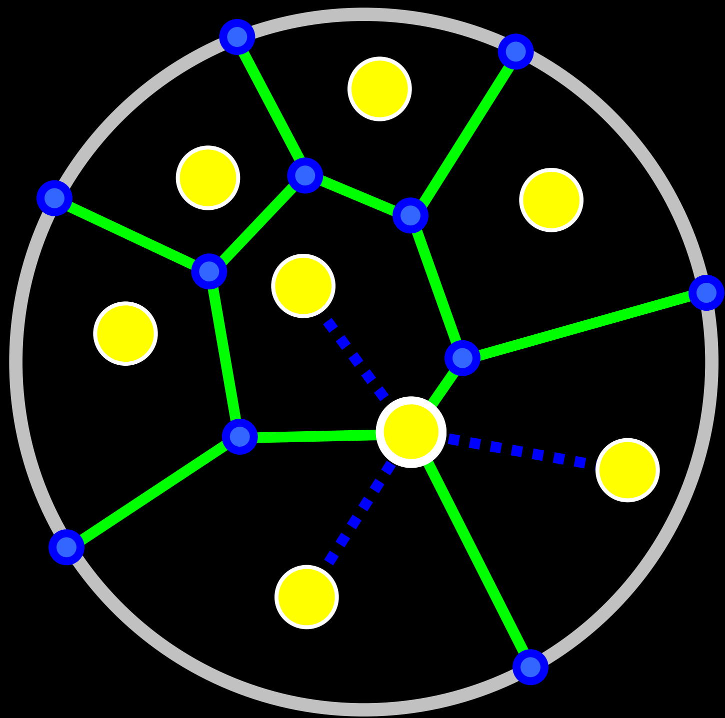
- Greedily choose the "worst" VPL
  = The one with shortest Delaunay edges



Slide courtesy Samuli Laine

Generating New VPLs

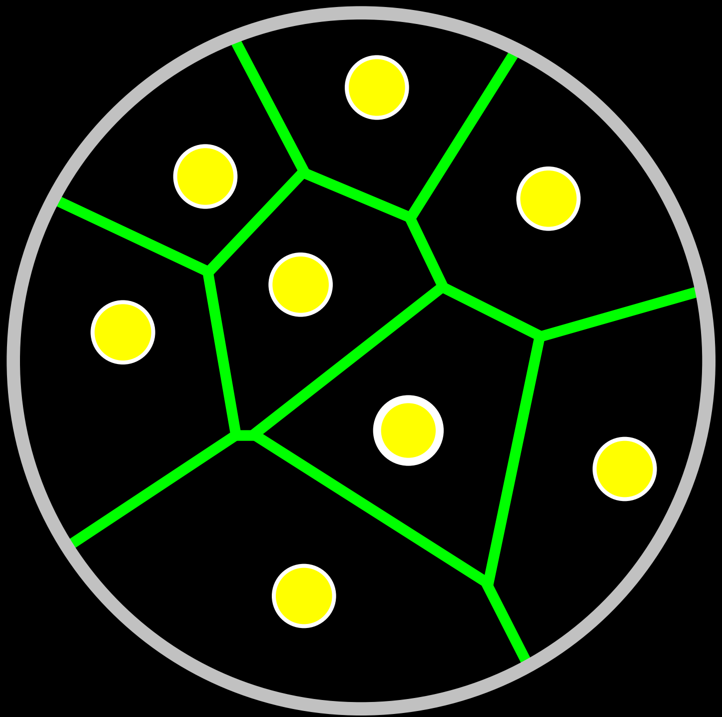
- Greedily choose the "best" spot
  = The one with longest distance to existing VPLs



Slide courtesy Samuli Laine

Computing VPL Intensities

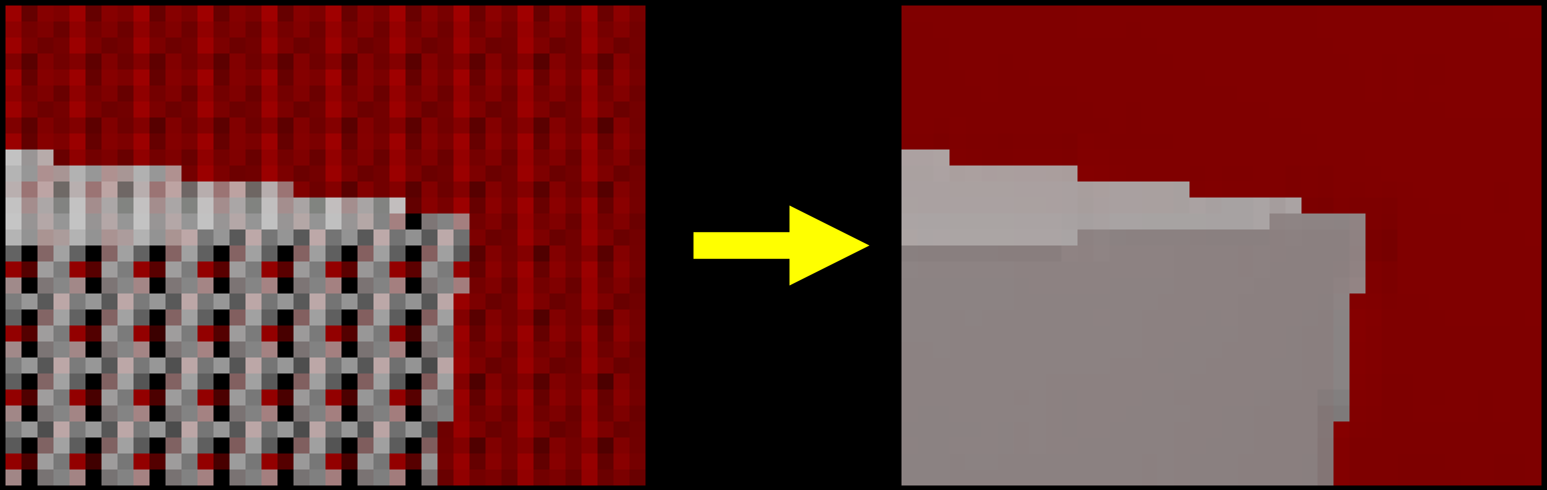
- Since our distribution may be nonuniform, weight each VPL according to Voronoi area



Slide courtesy Samuli Laine

Interleaved Sampling

- Reduces the number of shadow map lookups per pixel
- For each pixel, use a subset of all VPLs
- Apply geometry-aware filtering



Slide courtesy Samuli Laine

Sibenik

Triangles:

tessellated 109k

Resolution	Time (ms)	FPS
1024×768	17.0	48.6
1600×1200	30.1	25.9

Imperfect Shadow Maps

Imperfect Shadow Maps for Efficient Computation of Indirect Illumination

T. Ritschel* T. Grosch* M. H. Kim† H.-P. Seidel* C. Dachsbacher‡ J. Kautz†

MPI Informatik* University College London† Universität Stuttgart‡

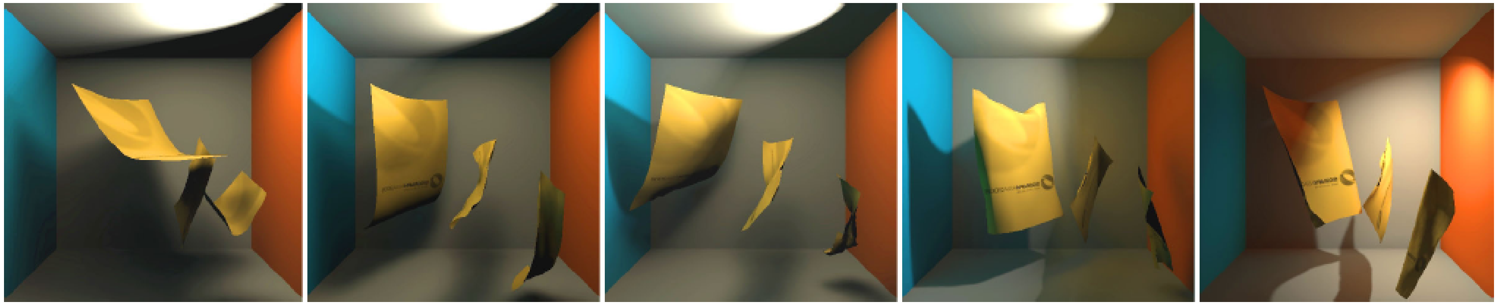
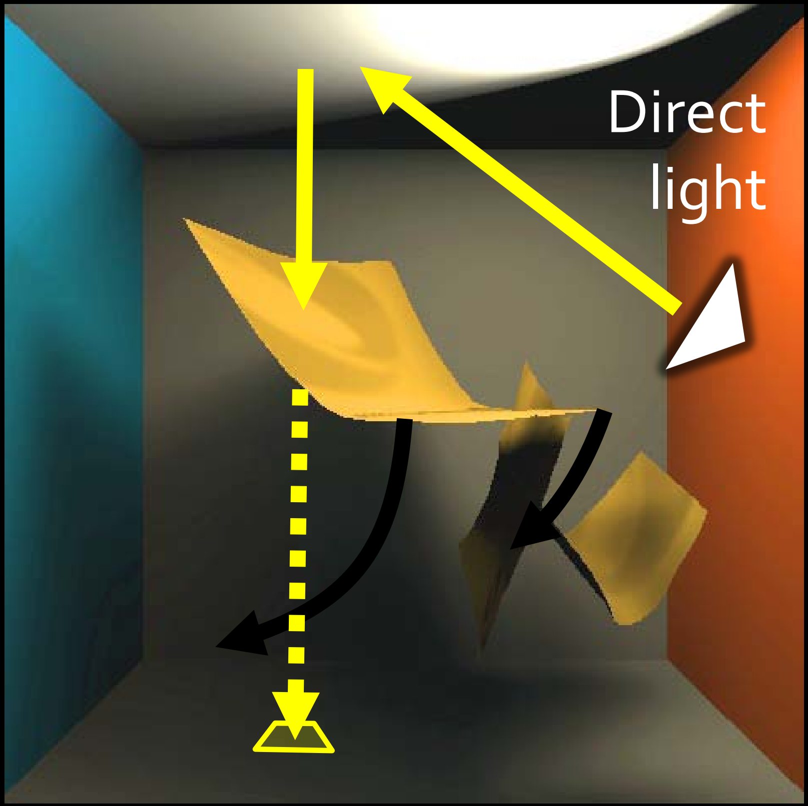


Figure 1: *Global illumination for a completely dynamic scene (light, view, geometry, material) rendered at 19 fps on an NVIDIA GeForce 8800 GTX. The scene is illuminated with a small spot light (upper right); all other illumination and shadowing is indirect (one bounce).*

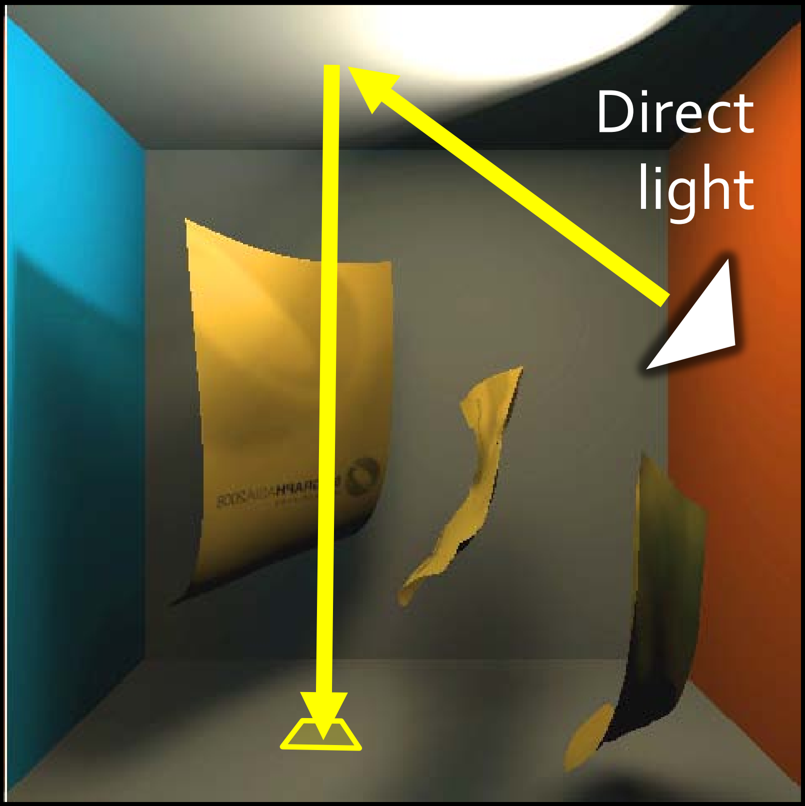
- http://www.mpi-inf.mpg.de/resources/ImperfectShadowMaps/
- **Key idea:** Faster shadow map rendering using a point-based geometry representation

Motivation

- Challenging: Dynamic indirect visibility



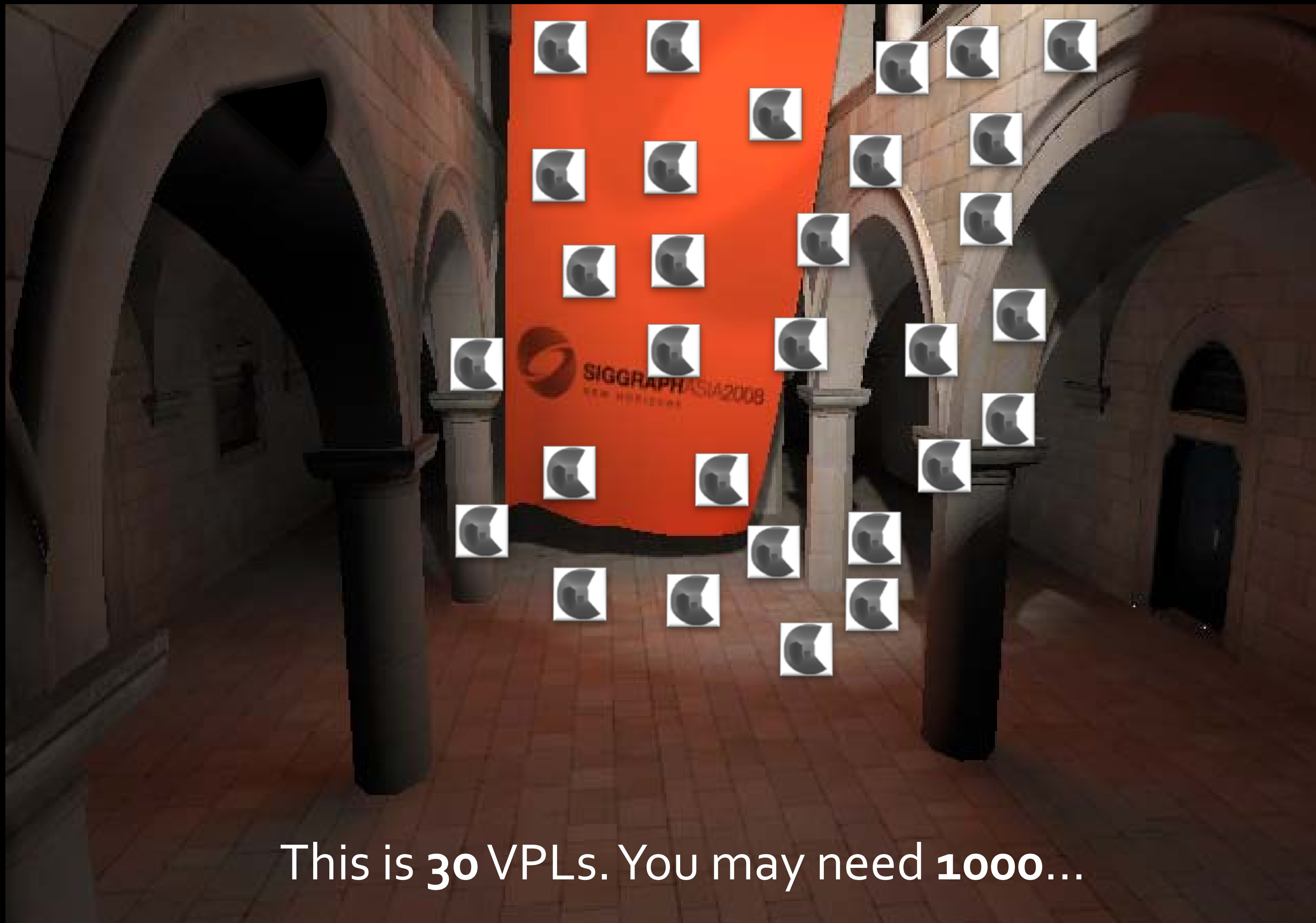
Indirect shadow



No indirect shadow

Slide courtesy Tobias Ritschel

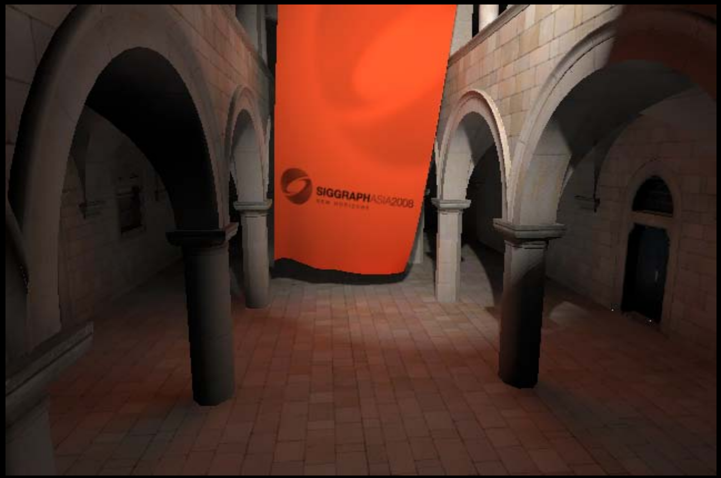
Instant Radiosity

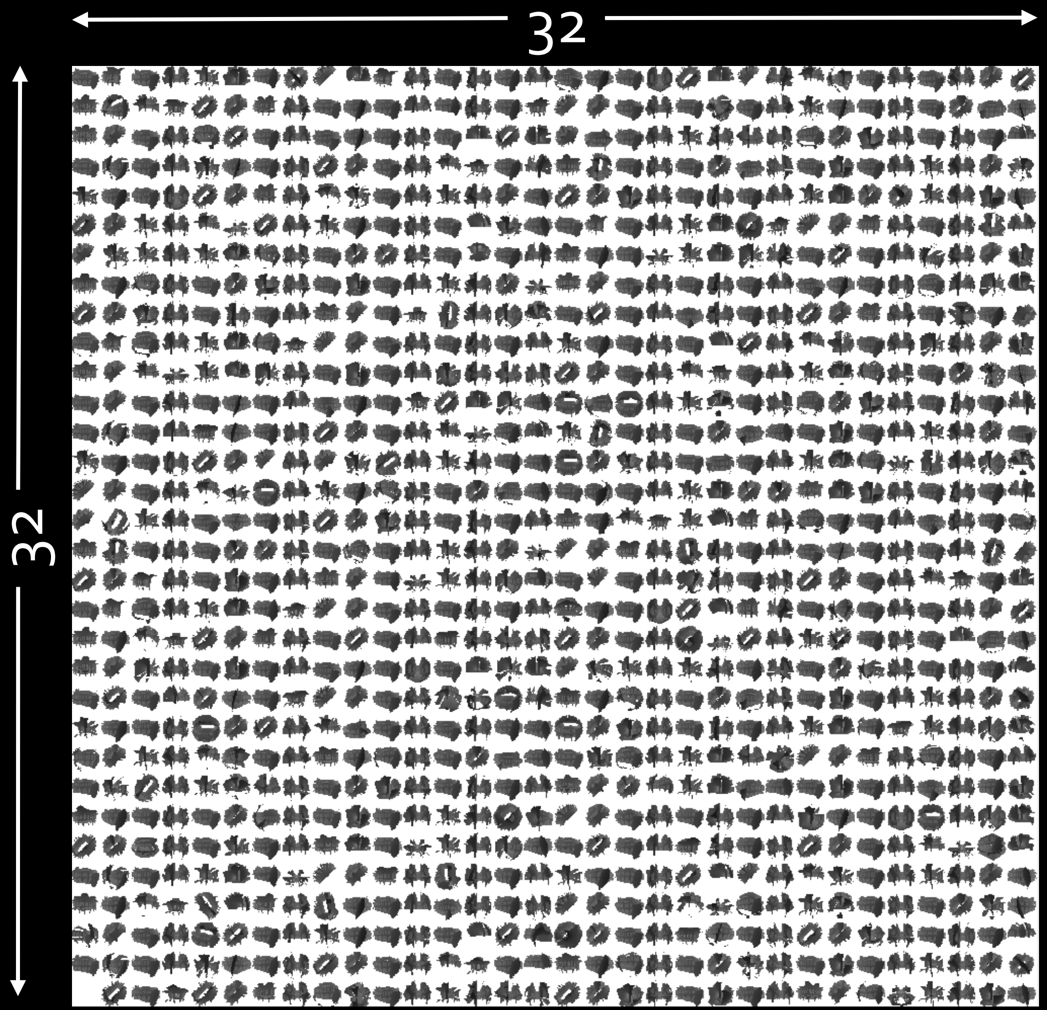


This is **30** VPLs. You may need **1000**…

Slide courtesy Tobias Ritschel

Instant Radiosity bottleneck





- 1024 VPLs
- 100k 3D model
- 32x32 depth map
- ~300M transforms
- 100x overdraw

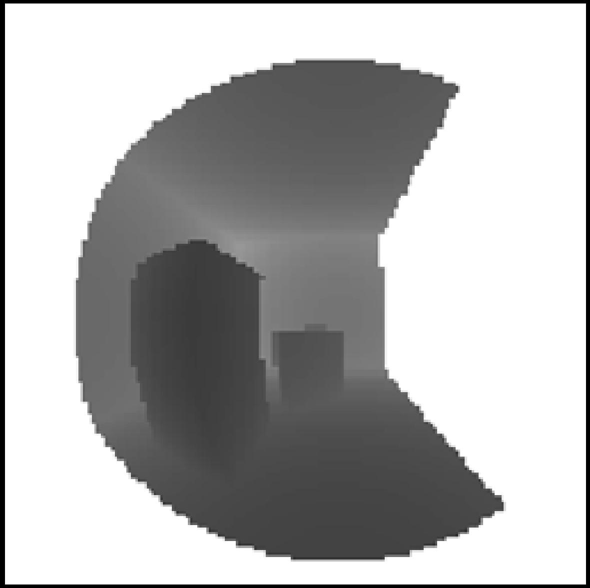
Slide courtesy Tobias Ritschel

Imperfect shadow maps

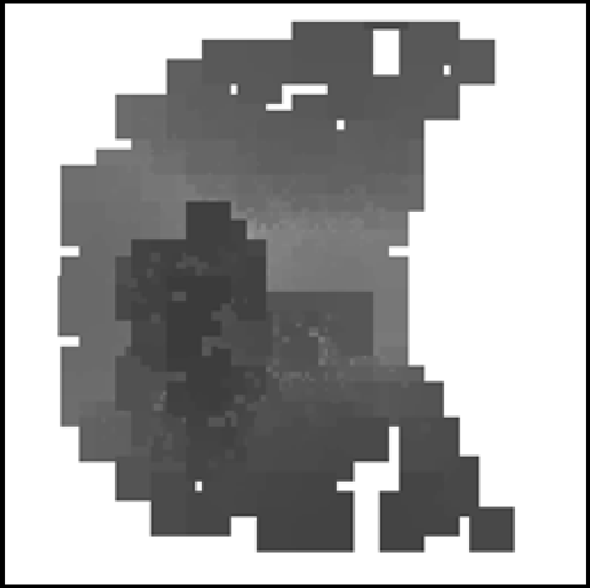
- **Observations**:
  Low quality (imperfect) depth maps sufficient for many faint VPLs that form smooth lighting
- **Contribution**:
  Efficient generation of low quality depth maps
- Main steps (detailed next)
  1. VPL generation
  2. Point-based depth maps
  3. Pull-push to fill holes
  4. Shading

Slide courtesy Tobias Ritschel

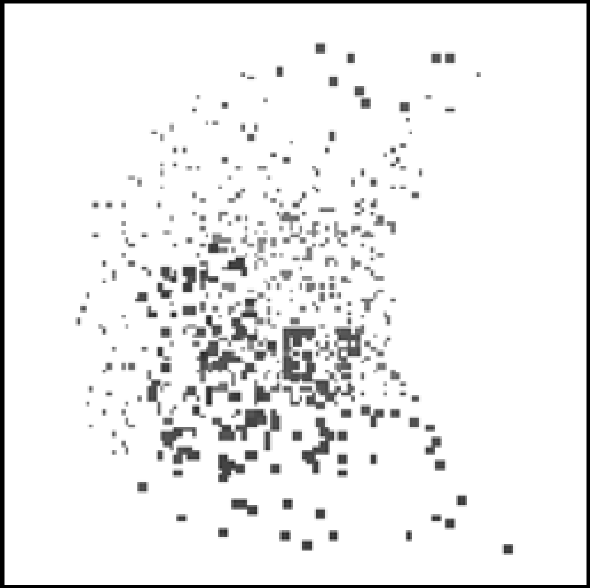
Step 2: Point-based depth maps



Classic



Imperfect

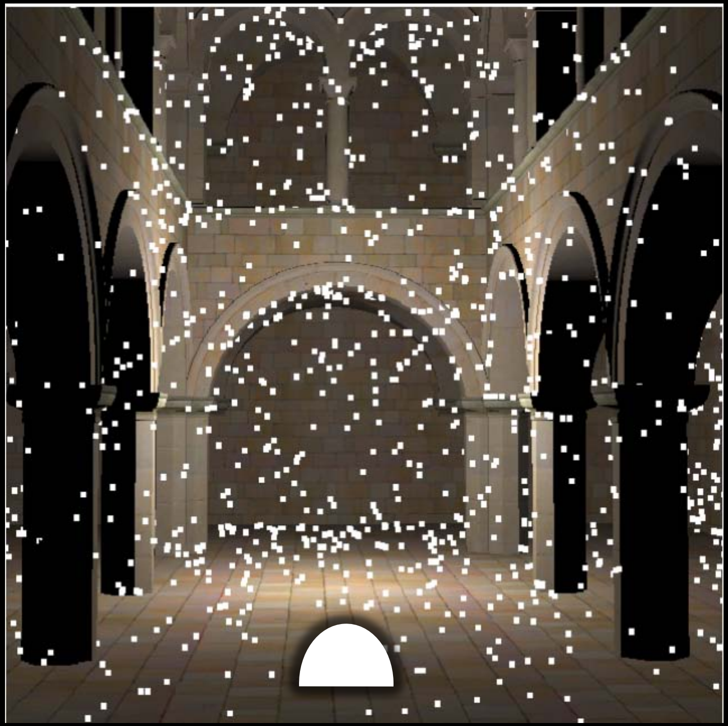


Imperfect
Smaller points
Less points

Slide courtesy Tobias Ritschel

Step 2: Point-based depth maps

- **Pre-process**:
  Distribute points on surface
  - ~8k points for every VPL
  - Different set for every VPLs

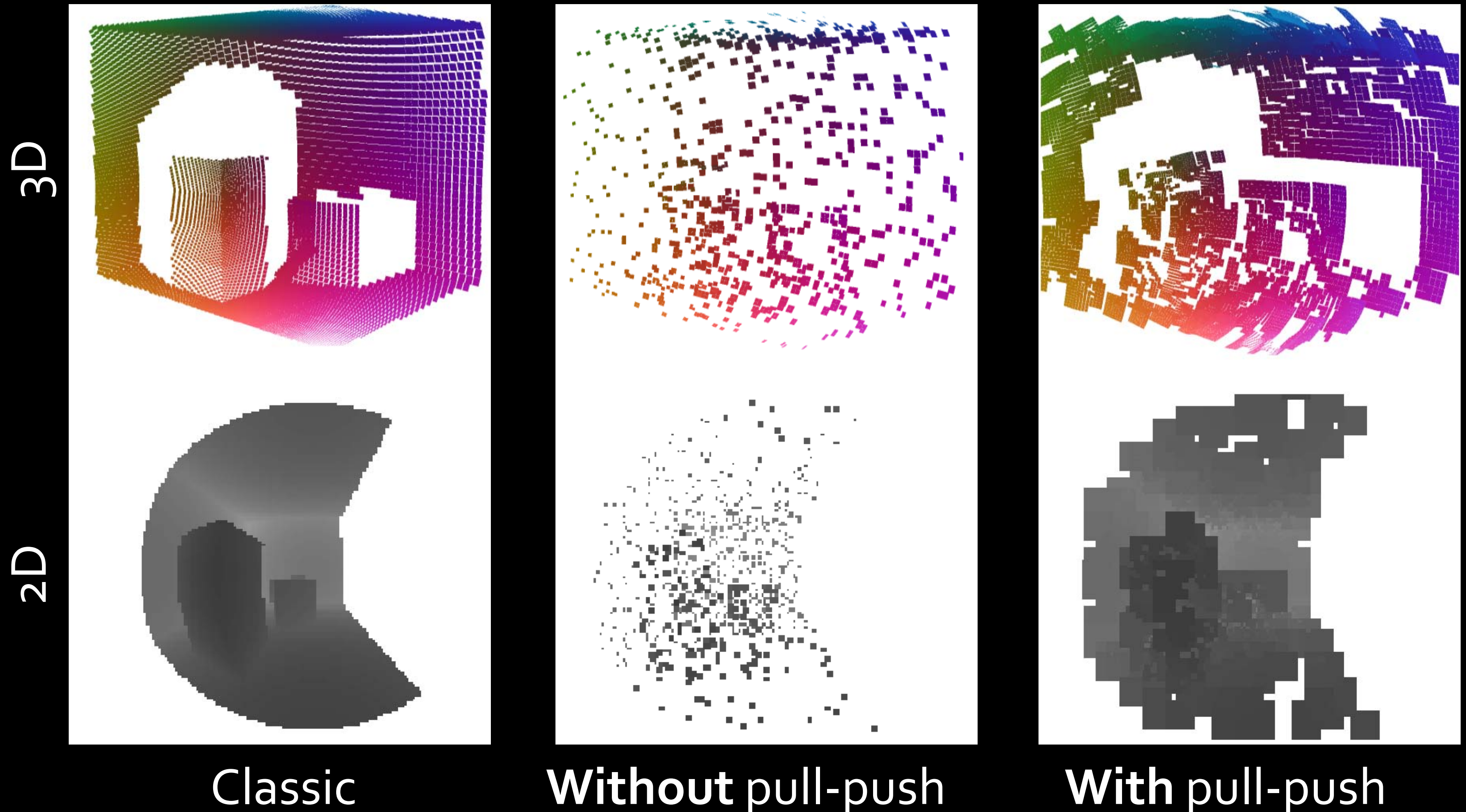


VPL / Depth map

Slide courtesy Tobias Ritschel

Step 3: Pull-Push

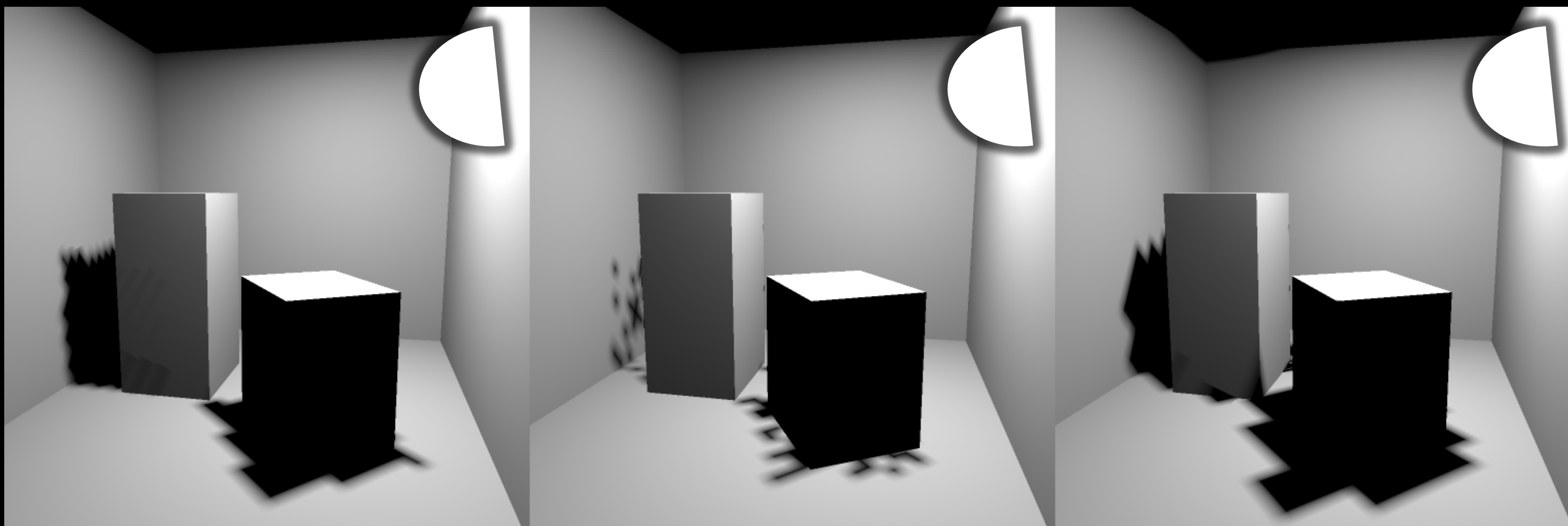
- Depth maps from points have holes



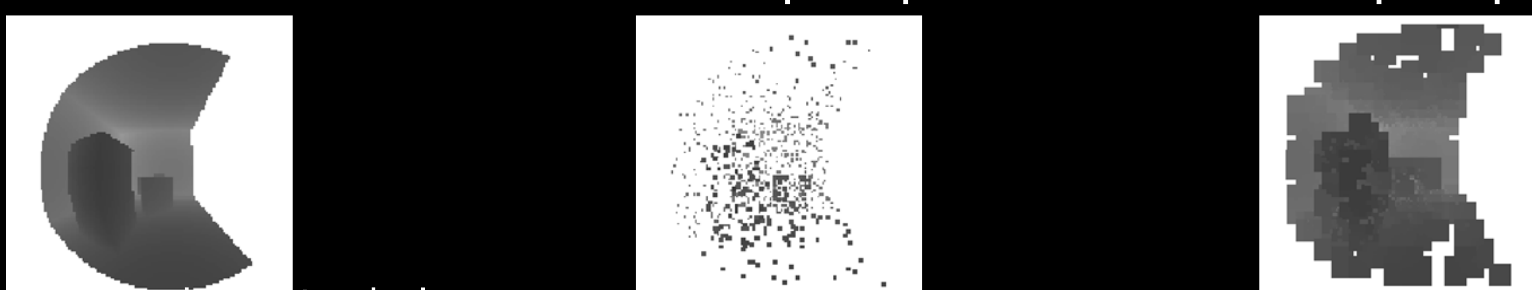
Slide courtesy Tobias Ritschel

Step 3: Pull-Push

- We fill those holes using pull-push ..



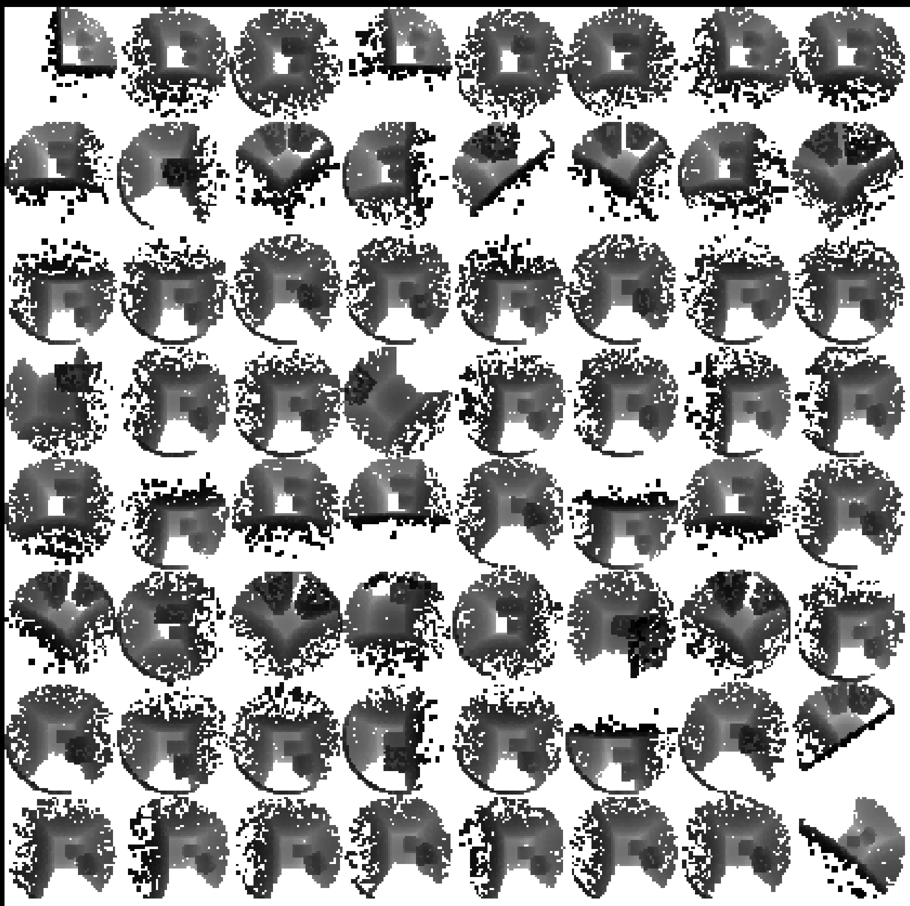
Classic | **Without** pull-push | **With** pull-push



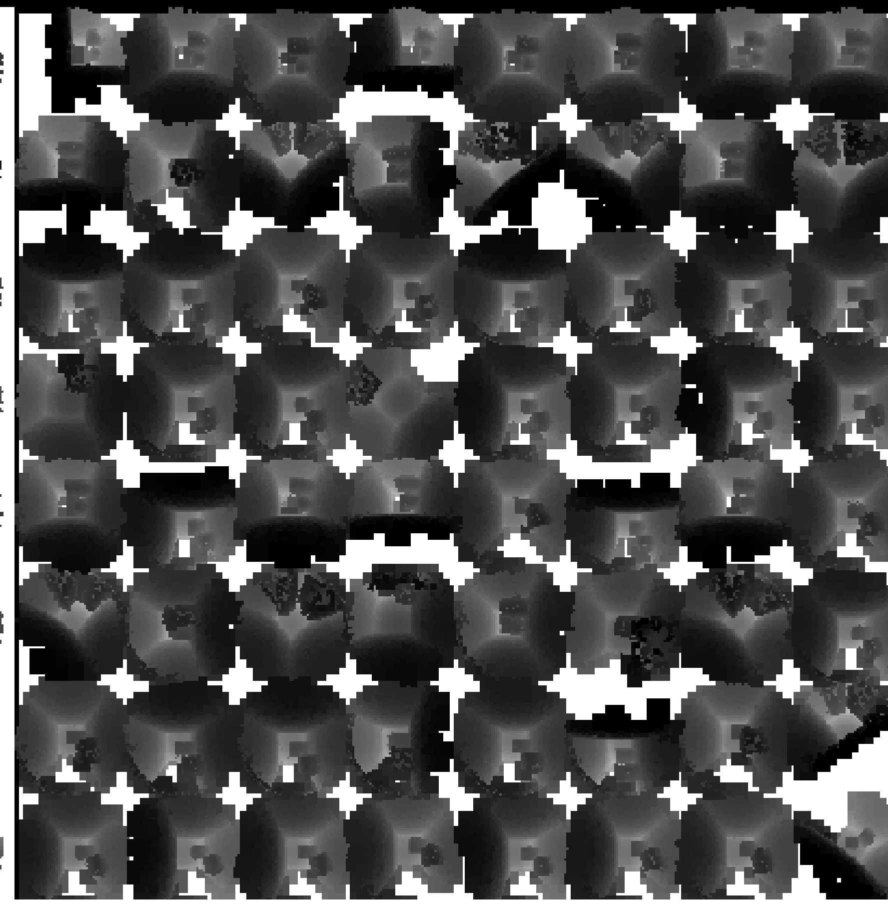
Slide courtesy Tobias Ritschel

Step 3: Pull-Push

- .. on all depth maps in parallel.



Without pull-push



With pull-push

Slide courtesy Tobias Ritschel

Step 4: Shading

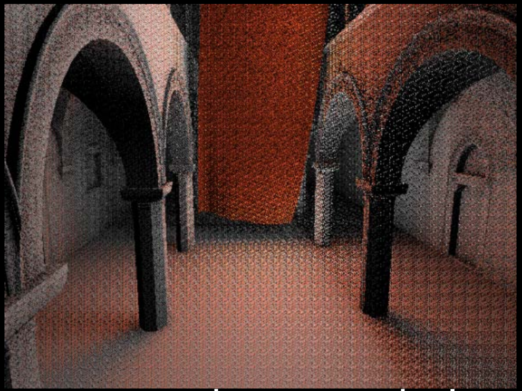
- Separate direct and indirect, both deferred
- Indirect: Interleaved sampling, geometry aware blur

Direct + Indirect

Direct only

Indirect only

G-Buffer

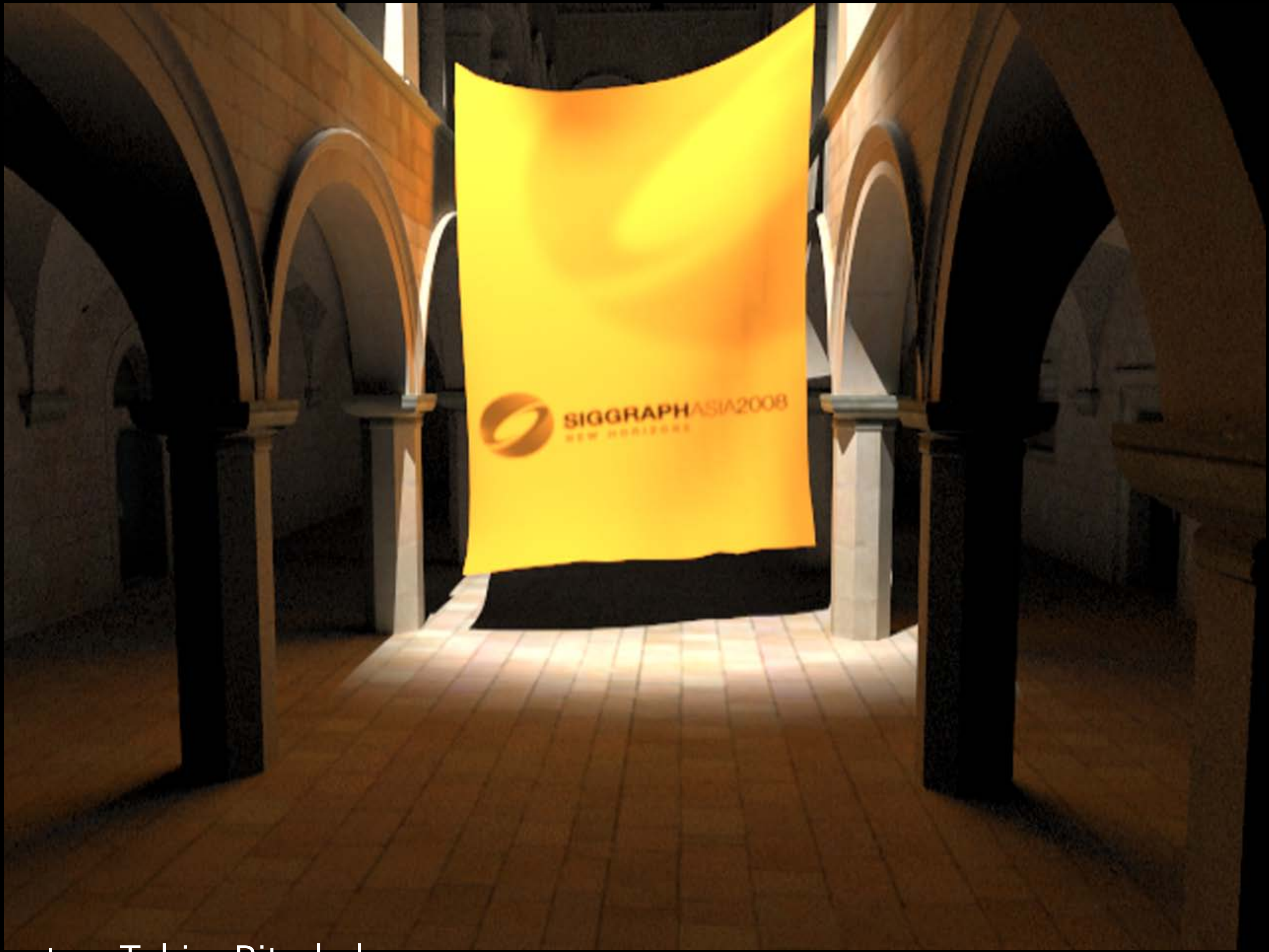


Simple blur

Edge-aware

Slide courtesy Tobias Ritschel

Results: Quality (*PBRT, hours*)



Slide courtesy Tobias Ritschel

Results: Quality (*Ours*, 11 fps)

Slide courtesy Tobias Ritschel

Imperfect shadow maps: Conclusion

- Doesn't really work that great…
  - No contact indirect shadows
  - Large scenes don't work well