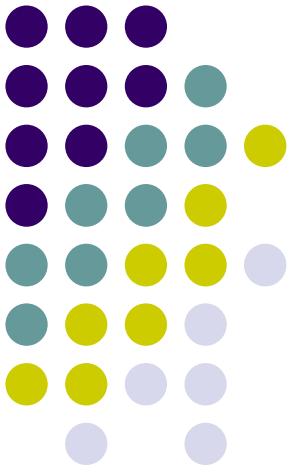
Advanced Computer Graphics
CS 563: *Making Imperfect Shadow Maps View-Adaptive*

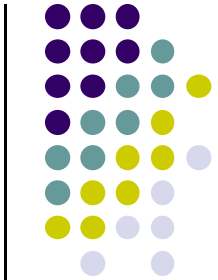
Frederik Clinckemaillie

Computer Science Dept.

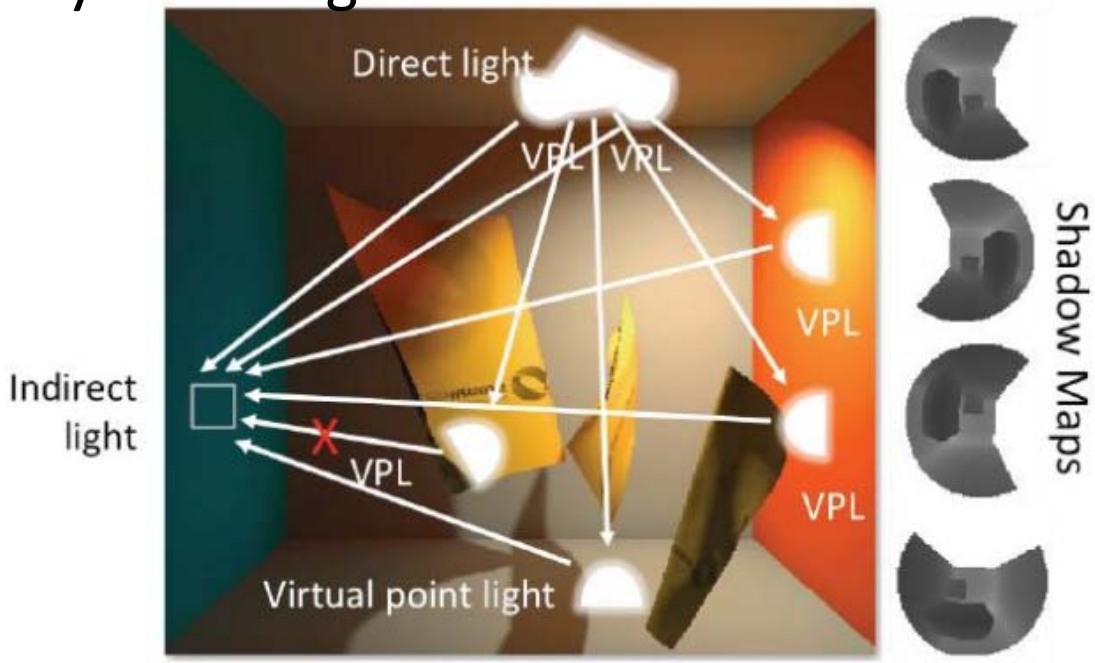
Worcester Polytechnic Institute (WPI)



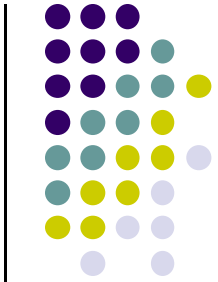
Background: Virtual Point Lights



- Simulates indirect illumination by creating VPL
- Evaluates shading combining illumination from all VPLs
  - Costly with large scenes



Background: Reflective Shadow Maps



- Renders the scene from the light's point of view
- Used to generate VPLs efficiently
- Uses different methods to determine which points should become VPLs
  - Randomly
  - Based on Outgoing Irradiance

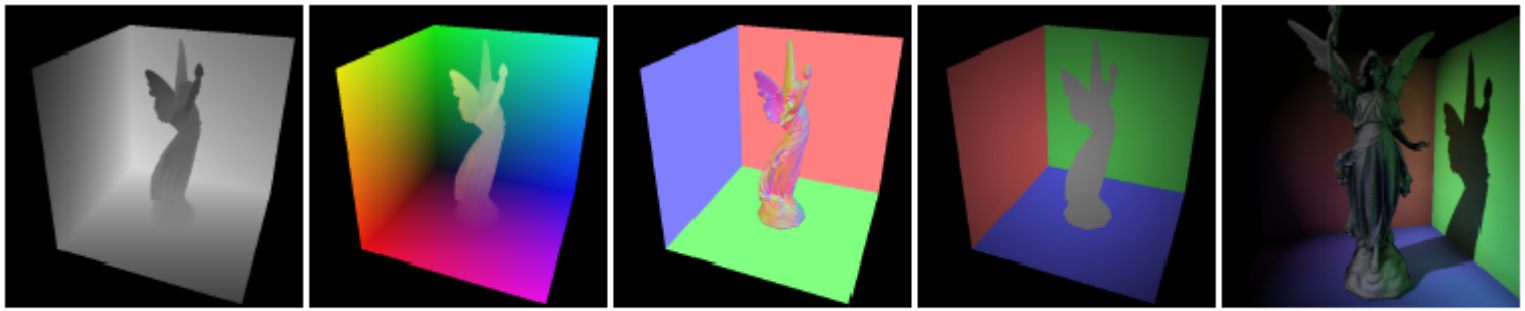
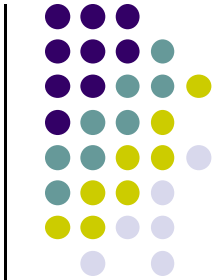
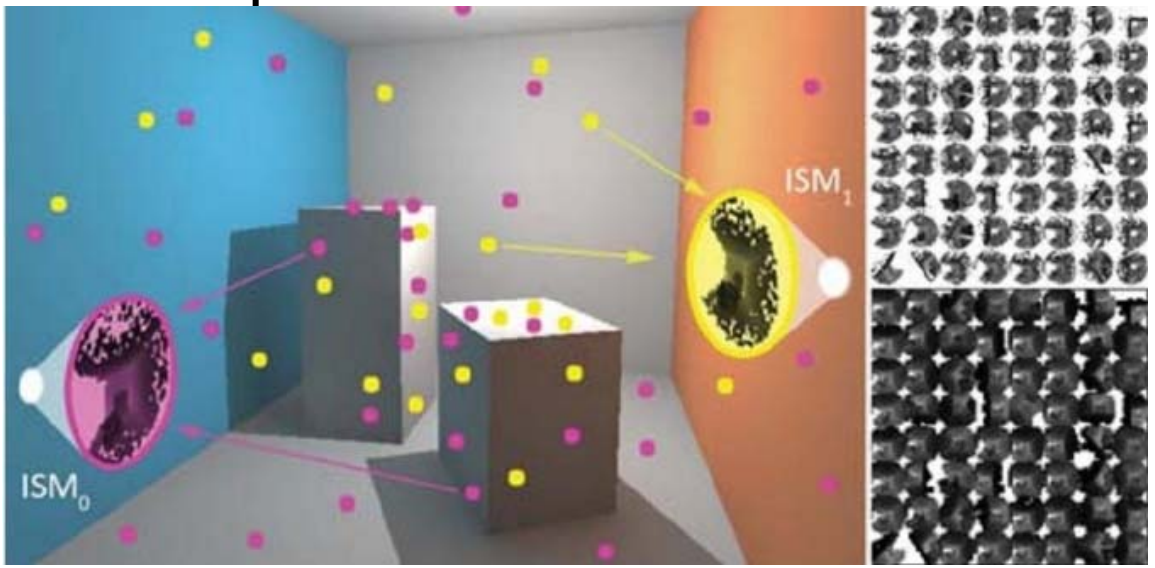


Figure 1: This figure shows the components of the reflective shadow map (depth, world space coordinates, normal, flux) and the resulting image rendered with indirect illumination from the RSM. Note that the angular decrease of flux is shown exaggerated for visualization.

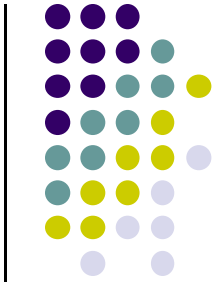
Background: Imperfect Shadow Maps



- Computed to resolve visibility
- Preprocess image to reduce it to points instead of triangles
- Points are then added to ISMs if visible
- Pull-Push Interpolation is used to fill in holes.

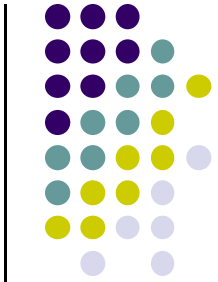


Problem



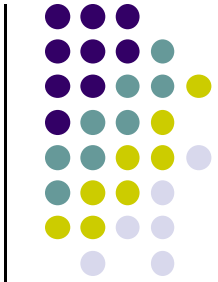
- Method does not scale well with large scenes
  - Will require many VPLs for acceptable LOD
  - VPL distribution does not consider contributions on the final image
  - Large scene may cause point-based representation (for ISM) to be too coarse.

Previous Work



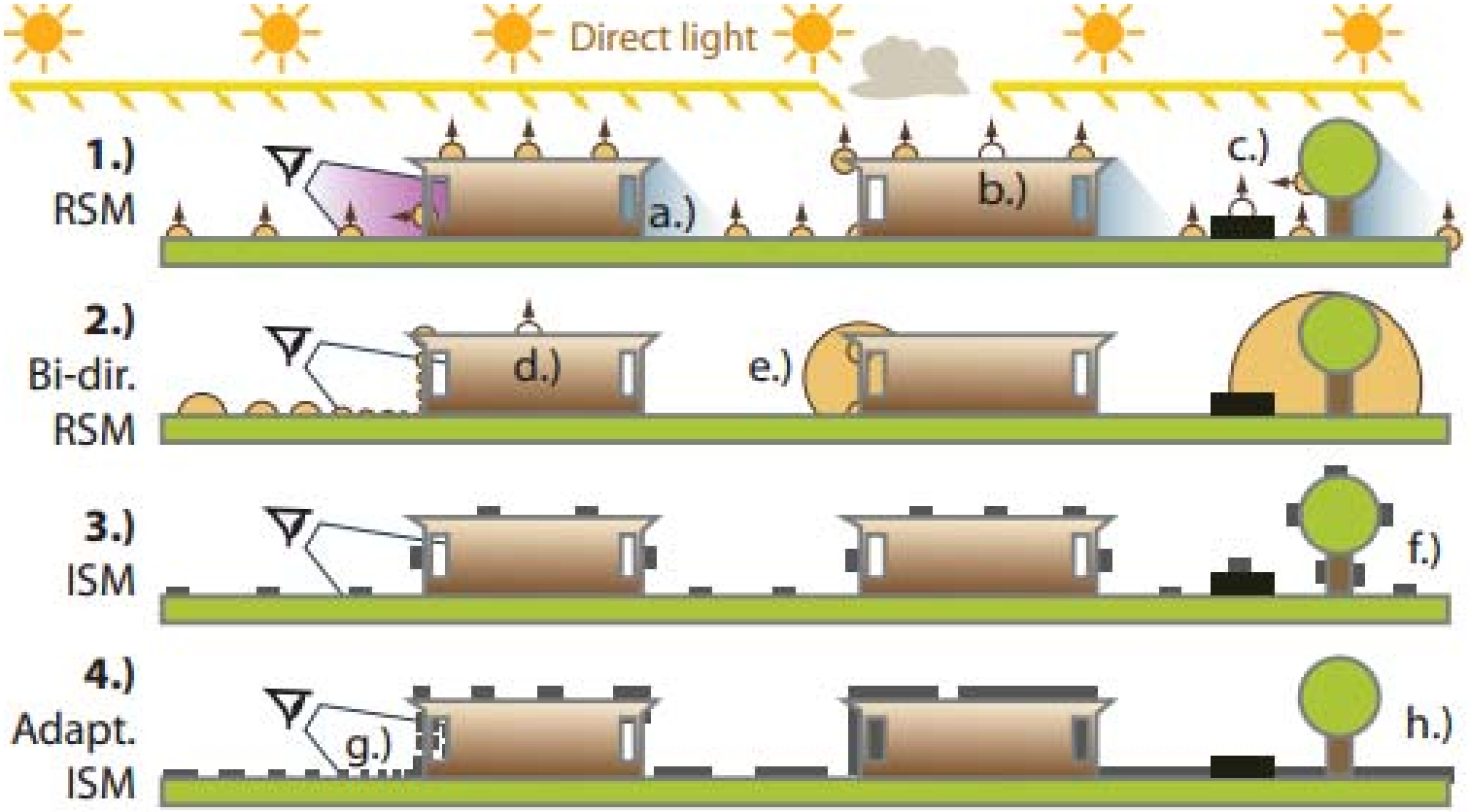
- Reusable VPLs[Laine & Al.]
  - Cannot handle dynamic scenes
- Ray-tracing Bidirectional Importance Sampling [Segovia & Al.]
  - Too slow to manage large and dynamic scenes
- Importance of nearby visibility events [Arikan & Al.]
  - Used coarse approximation to improve performance

Proposed Solution

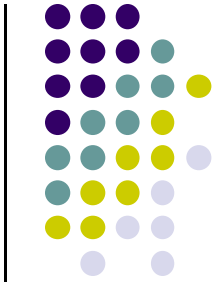


- Use different VPL distributions to provide more detail in areas of high importance
- Adapt the scene representation for indirect visibility to ensure geometric detail

Proposed Solution (cont.)



Bidirectional Reflective Shadow Maps



- Performs Bidirectional Importance Sampling without the use of ray tracing
- Scene is rasterized from :
  - Current view into a frame buffer
  - Point of view of the light into a reflective shadow map
- RSM texels represent potential VPLs (pVPL) from which VPLs have to be selected

Bidirectional Reflective Shadow Map

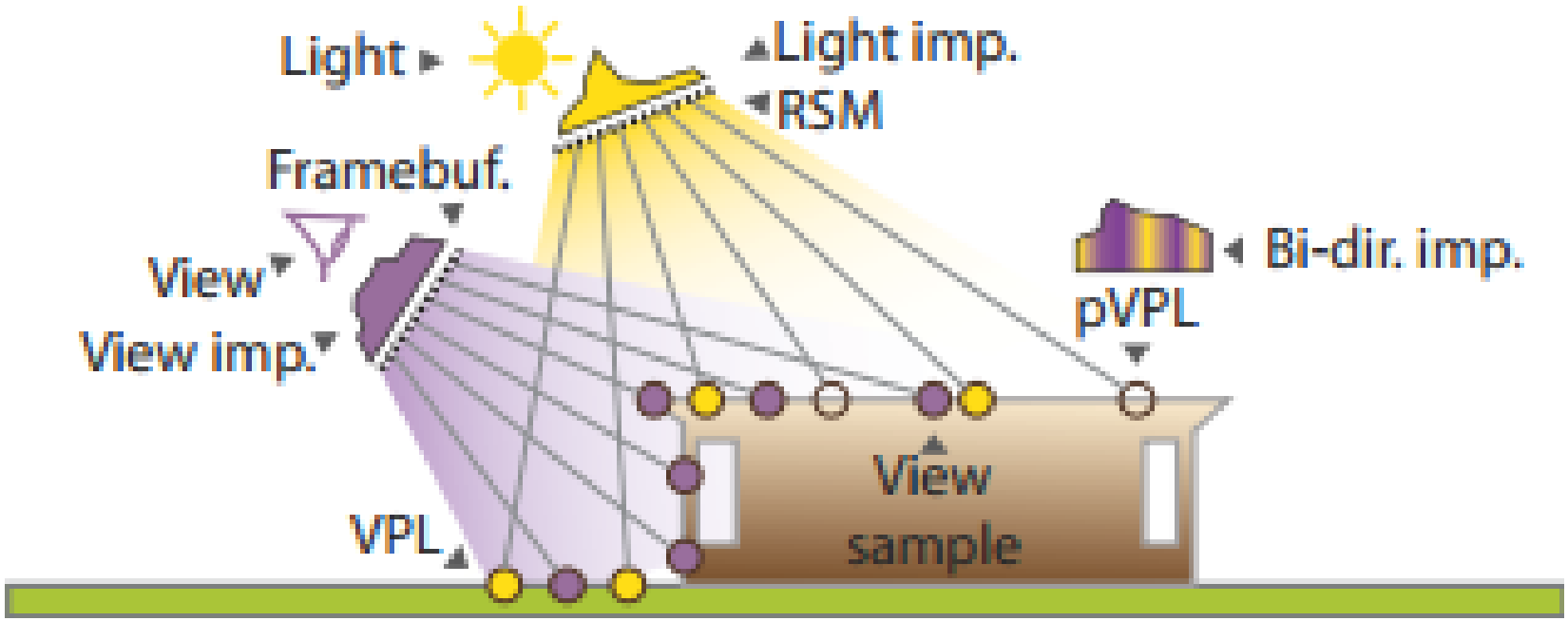
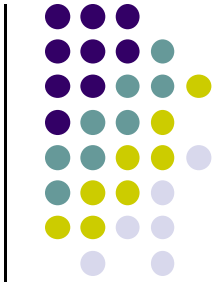


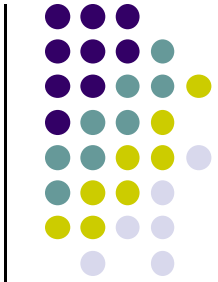
Figure 3: *Bidirectional Reflective Shadow Maps combine importance of direct light and its contribution to the framebuffer.*

Selecting VPLs from pVPLs



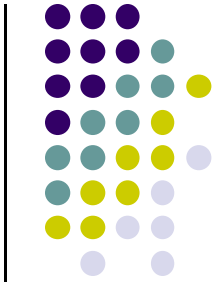
- Selection Distributions:
  - Uniform
    - May require large amount of samples
  - Based on outgoing irradiance
    - Performs better than uniform
    - VPL might not illuminate any view sample
  - Based on the influence of each pVPL on all view samples
    - Optimal, but too costly
    - Requires normalization to avoid bias
    - Simplifications can be made for approximation

Selecting VPLs from pVPLs (cont.)



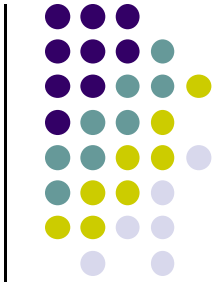
- Simplifications used:
  - A pVPL evaluates its impact on a few randomly-chosen view samples
    - In practice about 0.1% percent
  - Visibility is neglected when evaluating contributions from each pVPL
    - Removes the need for ray-scene intersections
    - Only light and view sample position are required
- pVPL contributions are stored in Bidirectional Reflective Shadow Map (BRSM)

Technical Details



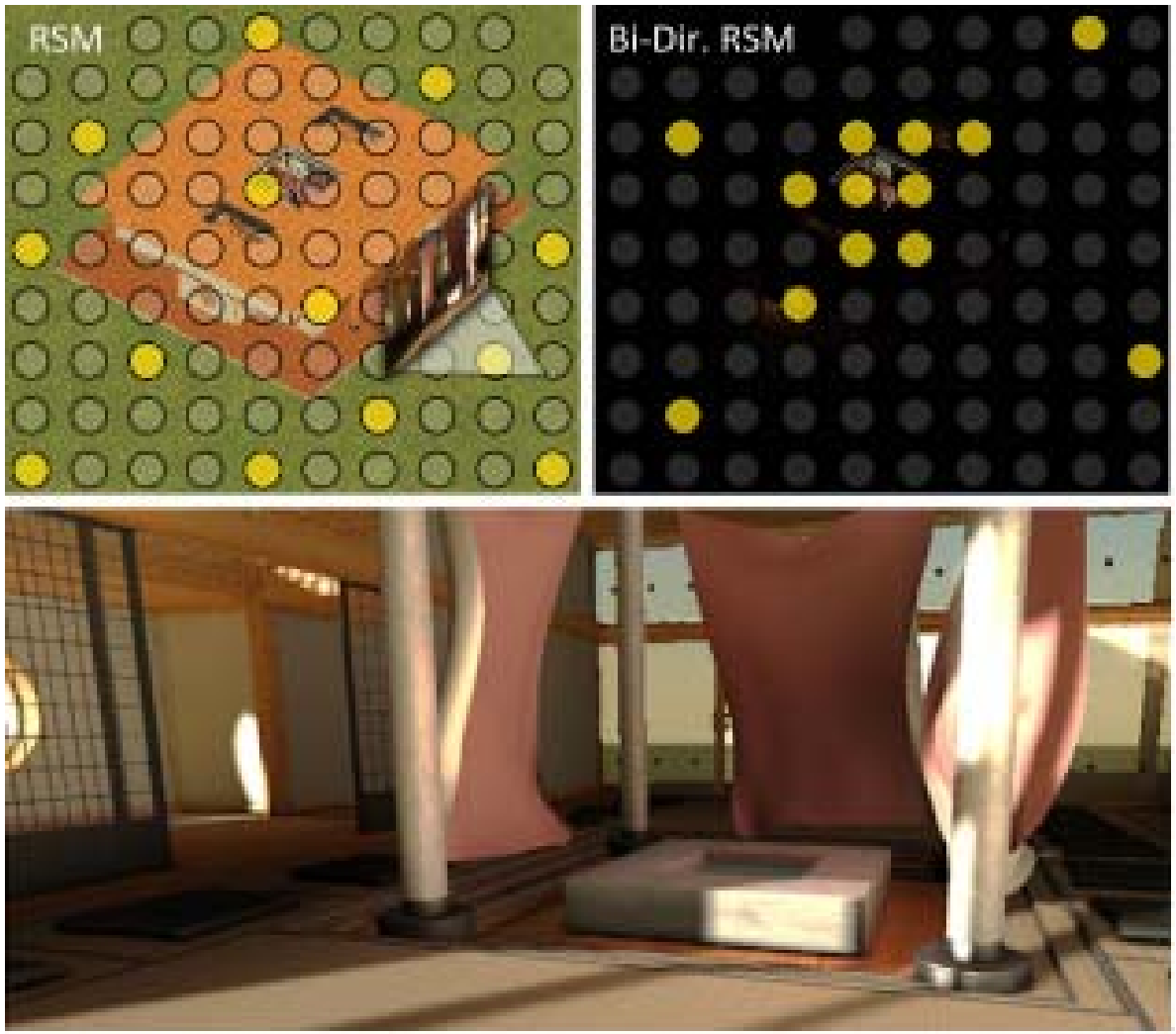
- Constructing the BRSM
  - Starts with a regular grid, but jitters lookup position
  - Each pVPL uses random value texture to create unique pattern

Technical Details (cont.)

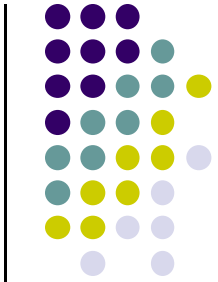


- Choosing VPLs
  - Uses several cumulative density functions (CDF)
  - Derive a CDF for each column $C_y[i]$ of the BRSM
  - Compute CDF C_x from the sums of values in each columns
  - For uniform sample: $[x, y]^T \in [1, \ldots, \text{width}] \times [1, \ldots, \text{height}]$
    - Column Position $i := C^{-1}{}_x[x]$
    - Row Position $j := C_y[i]^{-1}[y]$
  - Samples divided by probability by which it was chosen

Results Comparison



Improving View-Sample Selection for BRSM

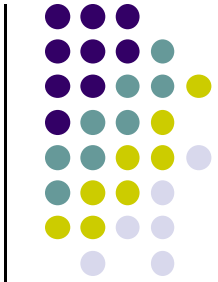


- So far, view samples are random
- Improvement: Selecting view samples that pVPL impacts strongly
  - Cannot involve information about view sample
  - Depends on rendering equation:

$$L(\mathbf{x},\omega_o) = L_e(\mathbf{x},\omega_r) + \int_\Omega f(\mathbf{x},\omega_i,\omega_r) L(\mathbf{y},-\omega_i)\cos\theta_i \, \mathrm{d}\omega_i$$

  - All terms depend on view sample values

Improving View-Sample Selection for BRSM



- Simplifying Approximation
  - Use the pixel position in frame buffer
  - Pixels close in real space will be close in frame buffer
  - Use distance falloff of $1/x^2$
  - Performs well in challenging situations
    - Corners will have many VPLs to avoid singularities
  - Allows for reduction of light blotches clamping

Improving View-Sample Selection for BRSM

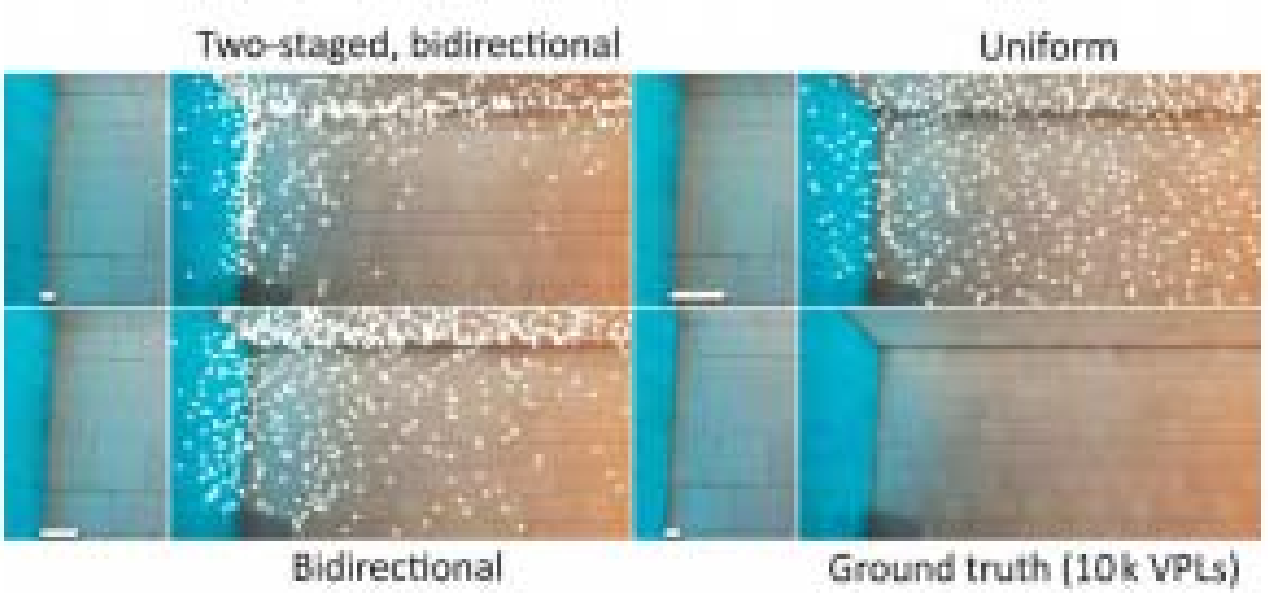
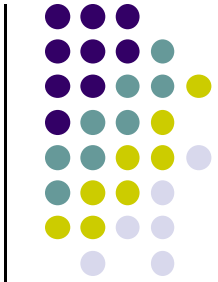
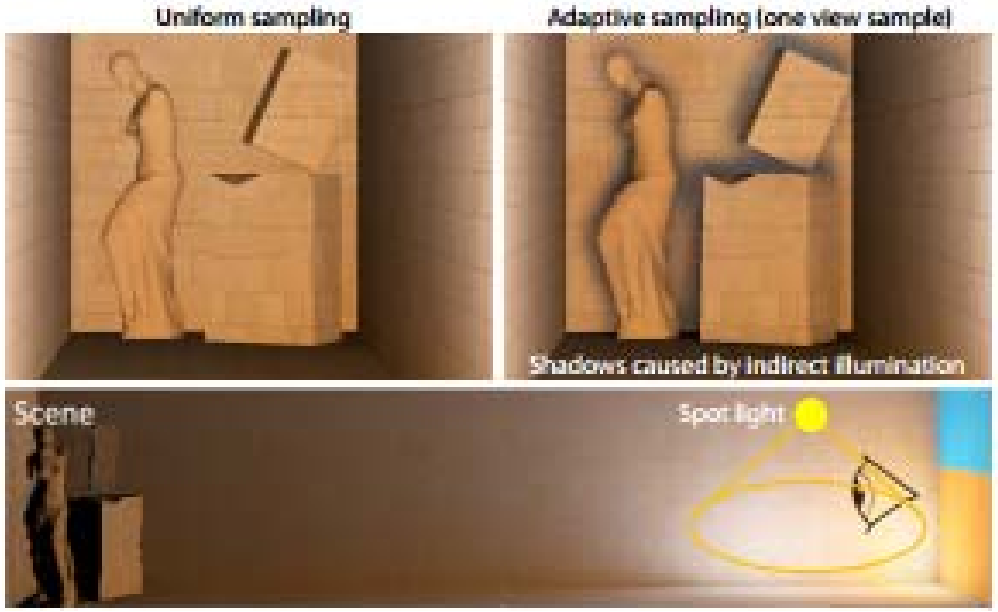


Figure 5: *A comparison of various estimation strategies for the VPL sampling. A comparison with a ground truth shows how better sampling strategies deliver more faithful results. The white bar denotes the bias due to clamping in each solution: Smaller is better.*

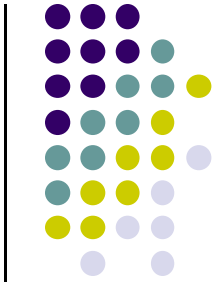
Adaptive Imperfect Shadow Maps



- Handles visibility in larger scenes than with ISM
- Blocker sampling is variable
  - Denser for closer geometries
  - Coarser for farther geometries

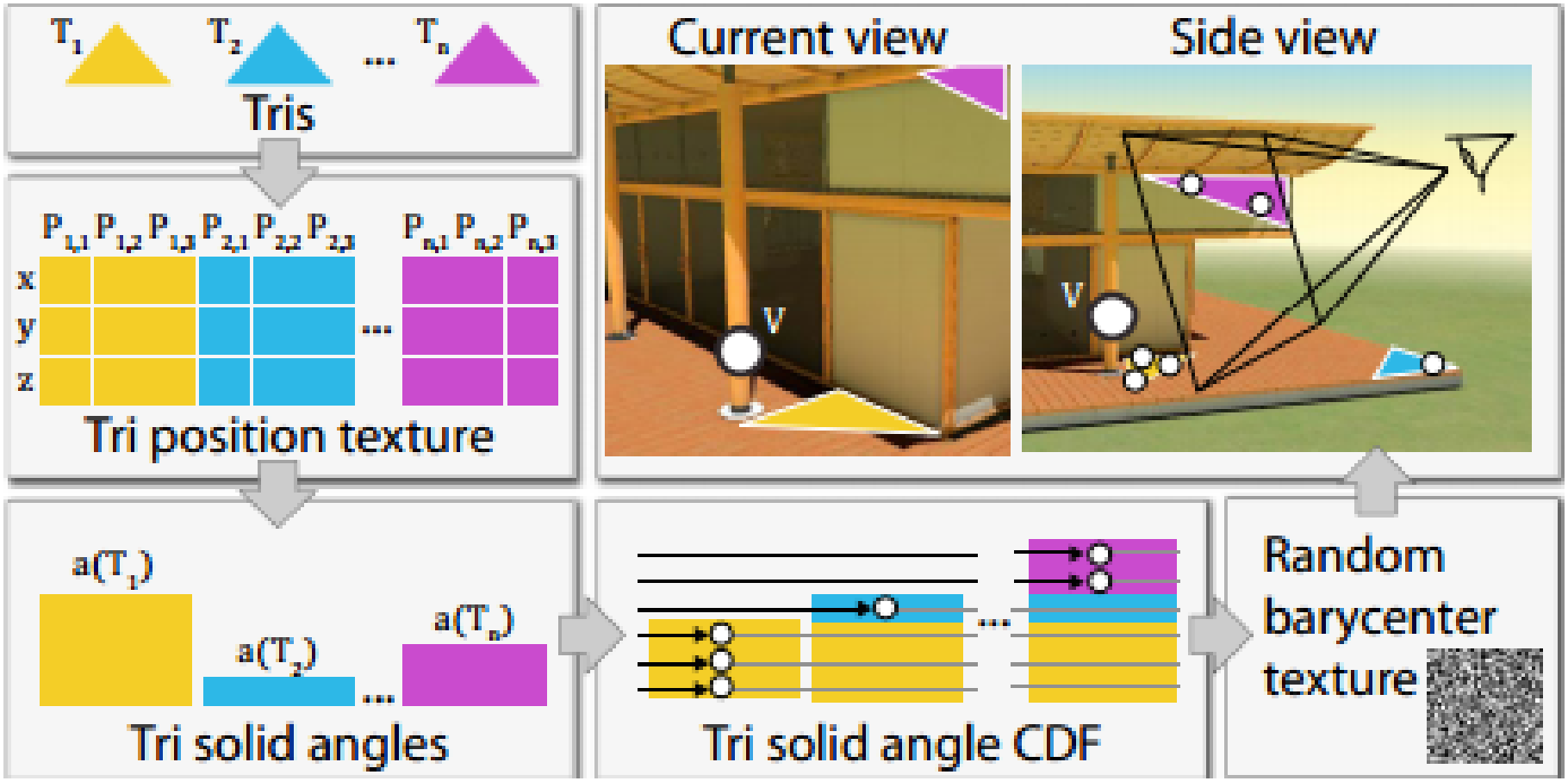


Adaptive Imperfect Shadow Maps

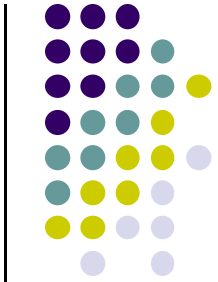


- Finding a point-based blocker representation
  - Ideally, consider all triangles and view samples
  - Efficient approximation:
    - Solid angles of all triangles are calculated
    - CDF is created based on solid angles
    - CDF is used to obtain N triangles
    - Random points in the triangles are determined
      - Using Barycentric coordinates

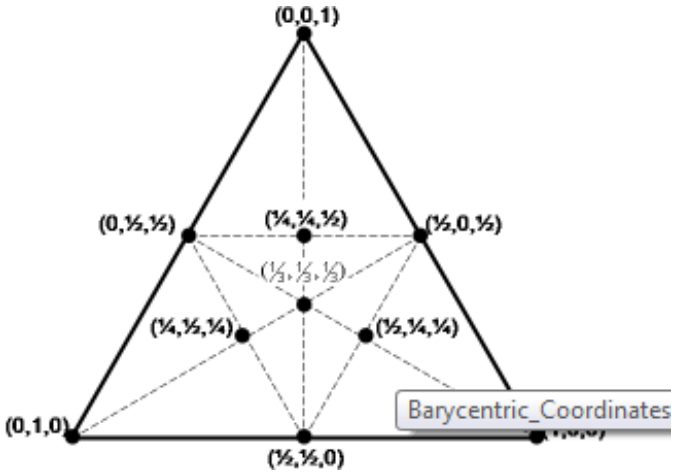
Sample-based Adaptivity



Using Barycentric Coordinates

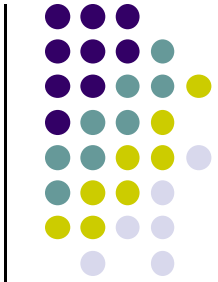


- Using a texture to get r_1, r_2
  - $r_l \in [0,1]$
  - r1 + r2 ≤ 1.
  - r3 = 1- r1 - r2
- Get coordinates of corners
- Use equations to find point
  - $x = r_1x_1 + r_2x_2 + r_3x_3$
  - $y = r_1y_1 + r_2y_2 + r_3y_3$



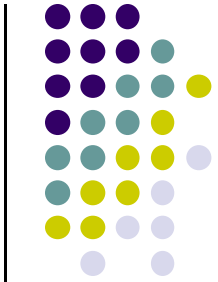
[courtesy of hairrendering.wordpress.com]

Making the Algorithm Run-Time



- Computing all triangle's blocking for all view samples is too slow
- Instead, randomly choose a set of view samples per triangle
  - Eight view samples per triangle
  - Increasing number with area not necessary

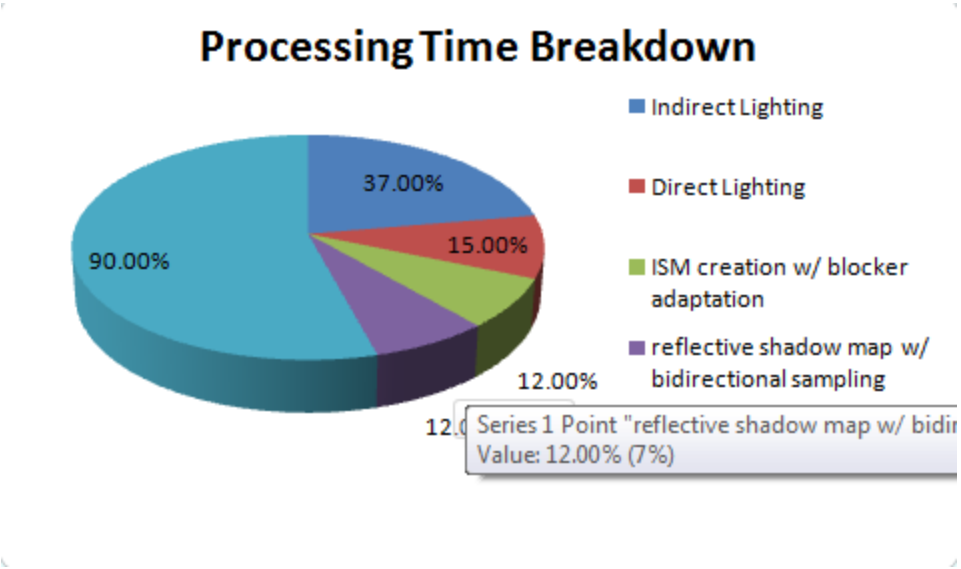
Dealing with Dynamism



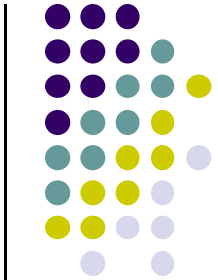
- High level of detail can damage temporal stability
- Lazy update scheme
  - Not all scene points are updated each frame
  - Performs better than uniform sampling
  - Human observer perceives fewer detail in motion
  - Speeds up algorithm
  - In practice, $1/8^{th}$ of all points are updated each scene

Results

- Run on Nvidia GTX 480 at 1600 x 800
- Interactive performance (around 15 fps)
- Algorithm adds 14ms GPU overhead for ISM



Results



- Bidirectional instant radiosity
  - Does not depend on ray-tracing, making it real-time
- Limitations
  - Lazy adaptation improves temporal incoherence, but adds lag.
  - Rasterized images are discrete, which could cause loss of information smaller than a pixel.
  - Approach is not independent of scene size

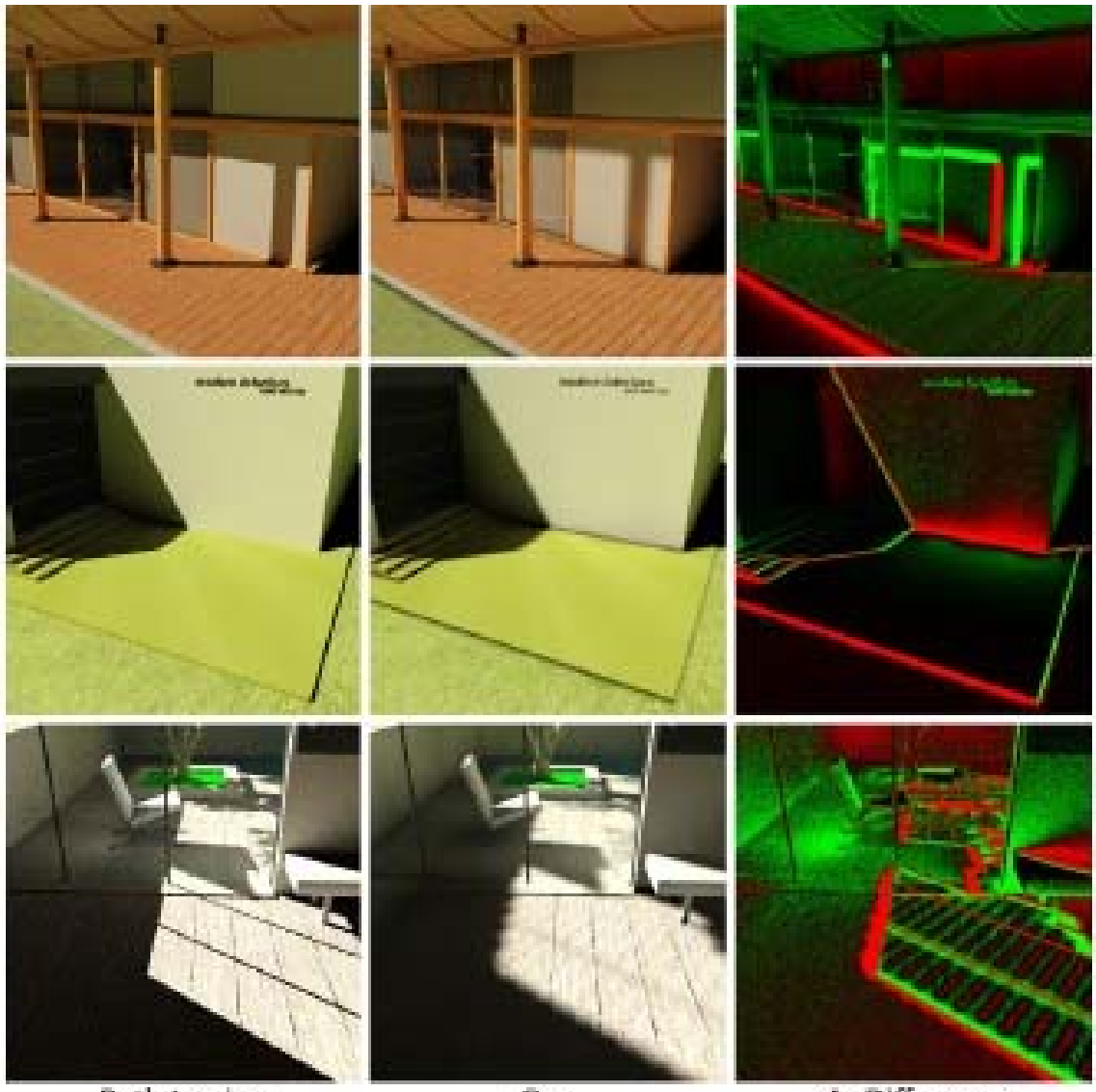
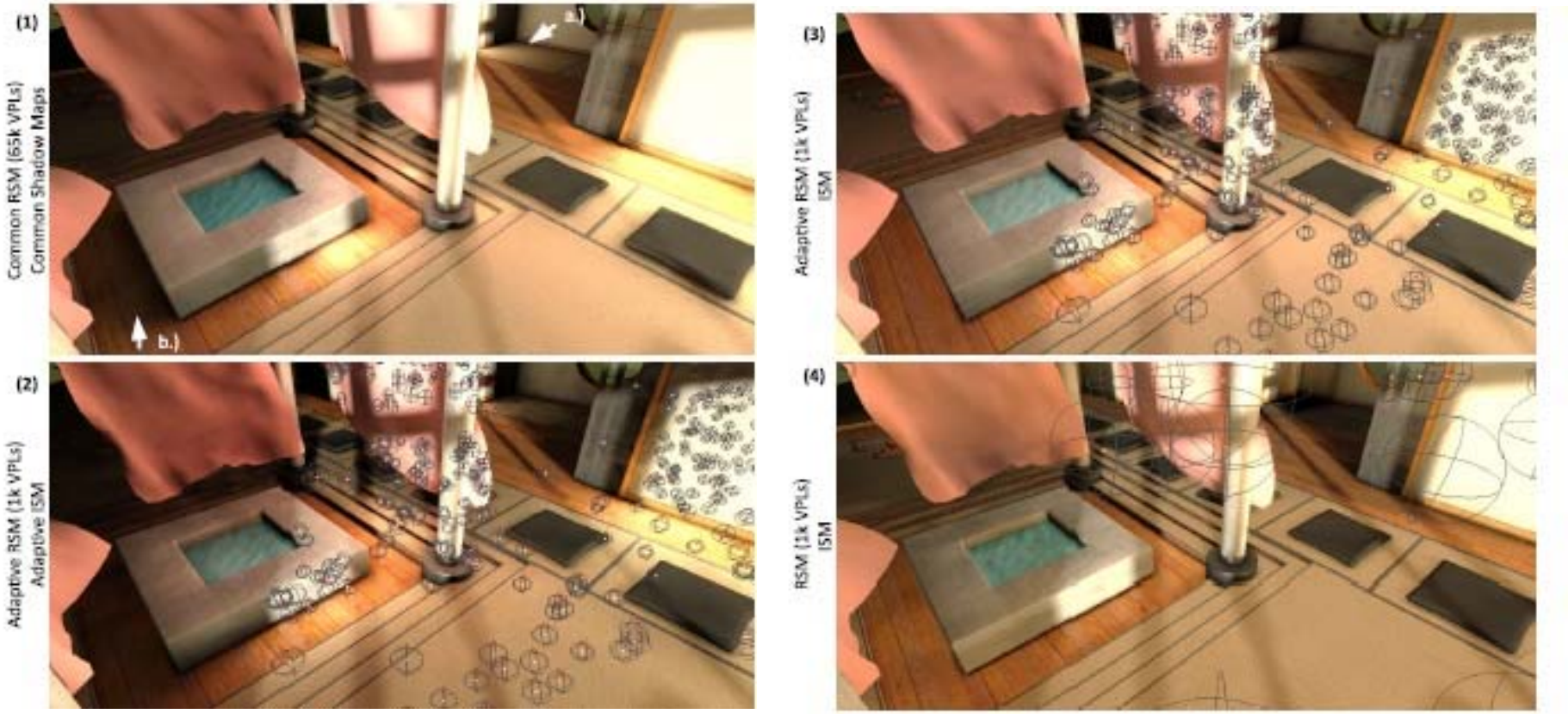
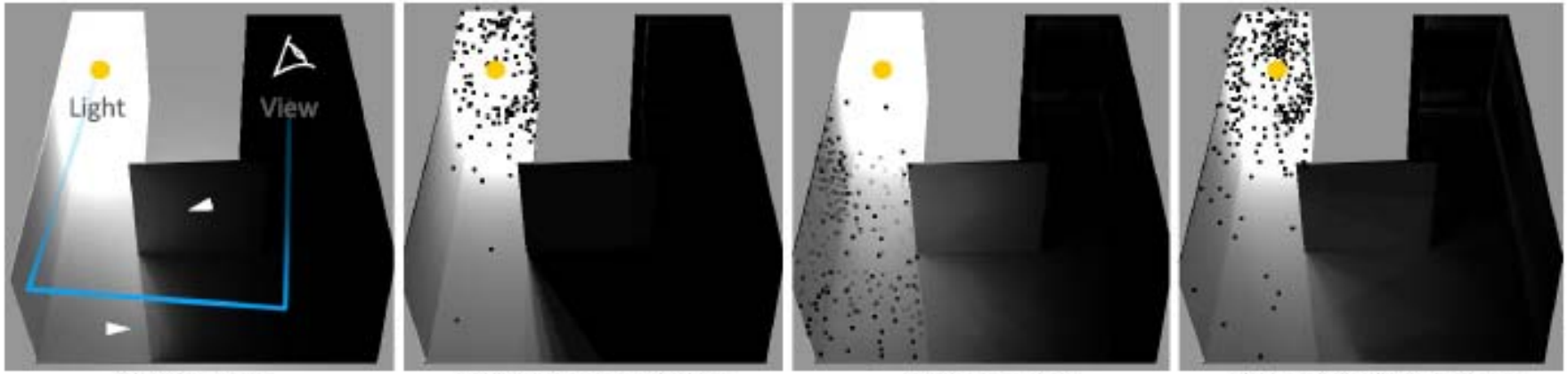


Figure 8: *Although many orders of magnitude faster (20 Hz vs. 20 min), our solution is close to the reference image. Some color deviations are due to the usage of 8 bit textures for our interactive result and 32 bit textures for the path tracing.*

Results

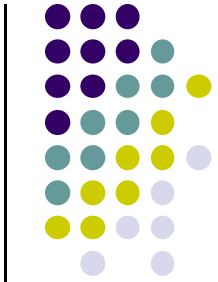

(1)
Common RSM (65k VPLs)
Common Shadow Maps
a.)
b.)
(2)
Adaptive RSM (1k VPLs)
Adaptive ISM
(3)
Adaptive RSM (1k VPLs)
ISM
(4)
RSM (1k VPLs)
ISM

Results



(1) Path tracing (2) Classic Instant Radiosity (3) Bi-directional (4) Bi-directional w/o visibility

References



- T. Ritschel, E. Eisemann, I. Han, J. D. K Kim, H.-P. Seidel
  Making Imperfect Shadow Maps View-Adaptive: High-Quality Global Illumination in Large Dynamic Scenes
  in Proceedings Eurographics Symposium on Rendering 2011