High Quality Adaptive Soft Shadow Mapping

A paper by Gaël Guennebaud, Loïc Barthe and Mathias Paulin

Roy Triesscheijn

Overview

- Algorithms from 2 papers, initial 2006 version and improved version from 2007
  - Real-time soft shadow mapping by backprojection (2006)
  - High-Quality Adaptive Soft Shadow Mapping (2007)

- Authors Gaël Guennebaud, Loïc Barthe and Mathias Paulin

- Combination of shadow maps, back projecting and adaptive precision to create real time soft-shadow maps

Index

1. Visibility Computation

2. Performance & Adaptive Precision

3. Summary & Discussion

Visibility Computation (2006)

Visibility Pass

- First step is to compute a normal shadow map

- The shadow map is back projected onto the light source to compute the percentage visibility for a point **p** that needs to be shaded

- Computed by finding the occluded area of each point on the light by samples in the shadow map

Visibility Computation (2006)

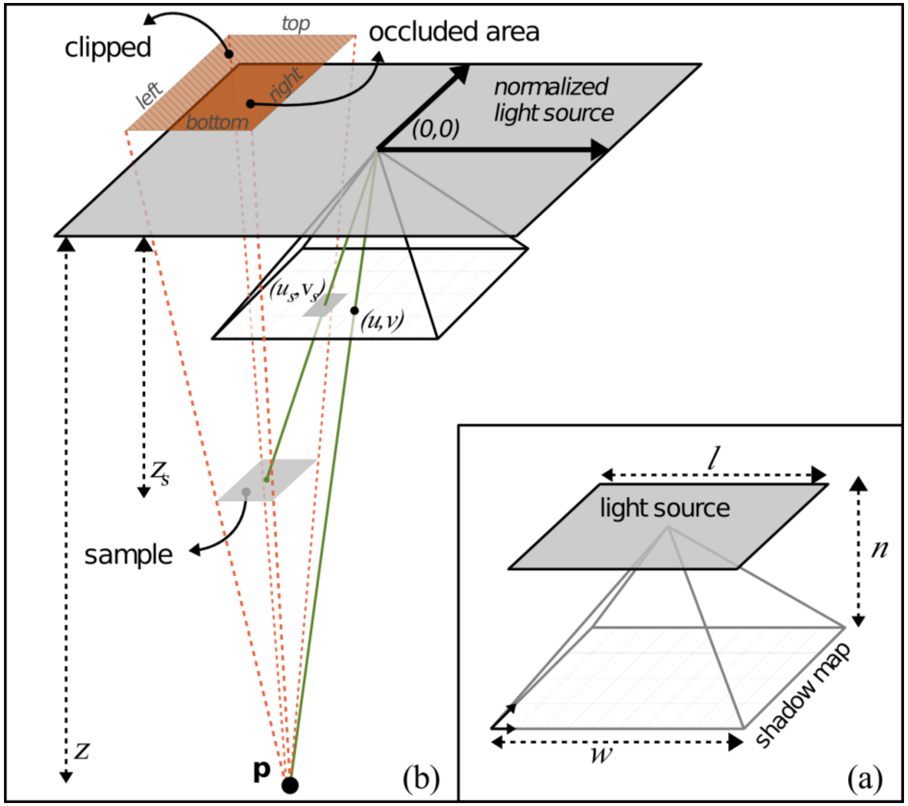
For point **p** in the scene

- Project **p** onto light, light space coordinates (u,v)

- Assume **p** is fully visible

- Remove from **p** the area occluded by every sample stored in the shadow map

(Search area is later optimized using Hierarchical Shadow Maps, HSM)



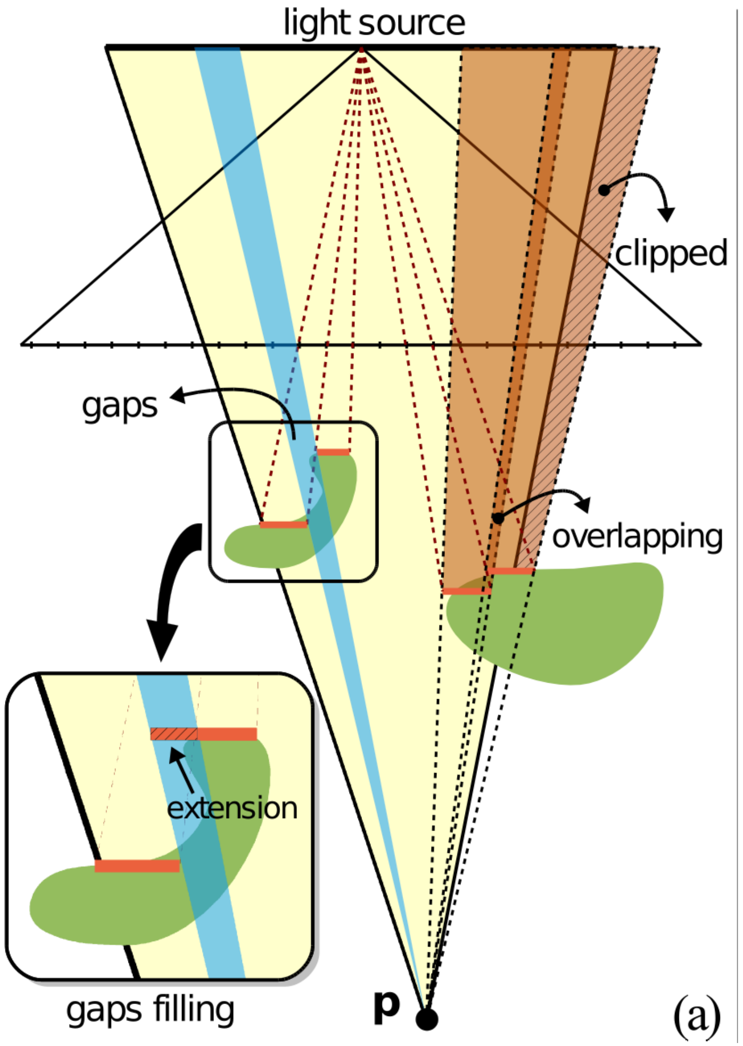
Visibility Computation (2006)

Gap filling

- Visibility is not correct due to gaps and overlaps when back projecting

- Overlaps cause darker penumbrae, not fixed

- Gaps fixed by extending occluders to neighbouring occluding samples in the shadow map



Visibility Computation (2006)

Results

- The 2006 algorithm already gives nice results, is geometry independent, works in real time

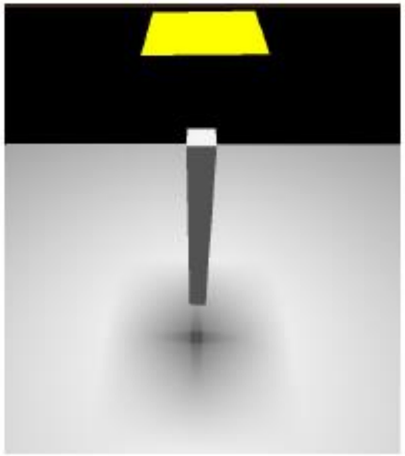
Ground Truth (2500ms per frame, 0.4fps)

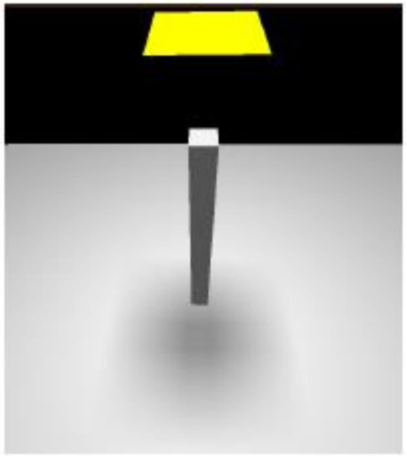
2006 algorithm (40ms per frame, 25fps)

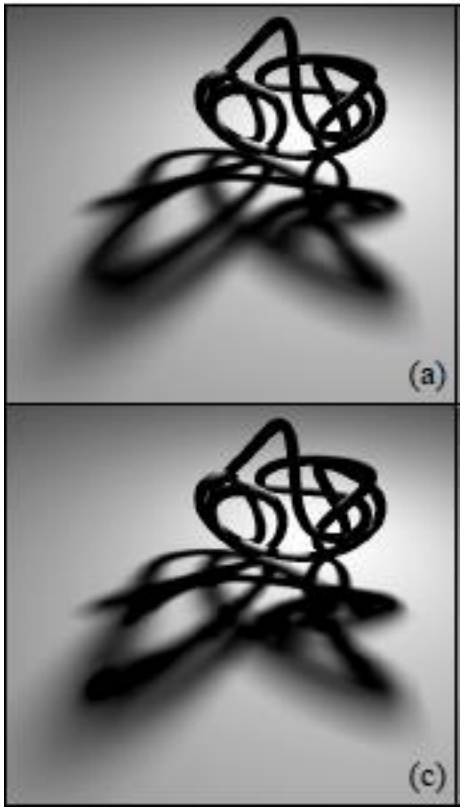
Visibility Computation (2006)

Results

- But, gap filling and overlap cause shadow overestimation, there is the single light artifact, shadow contours can be rough







Ground truth

2006 algorithm

Top: ground truth
Bottom: 2006 algorithm

Visibility Computation (2007)

Goals for 2007 algorithm

- Fix the gap and overlap problems
- Do not overestimate shadows
- Produce smoother shadow edges
- Keep the performance equal

- The 2007 algorithm will not solve the single light artifact

Visibility Computation (2007)

Visibility Pass

- Still compute a normal shadow map first

- From this a Hierarchical Shadow Map is computed, which stores the lowest and highest depth values for each cell

- From the HSM a kernel (Search Area in the 2006 paper) is computed in which we can find possible occluders when back projecting

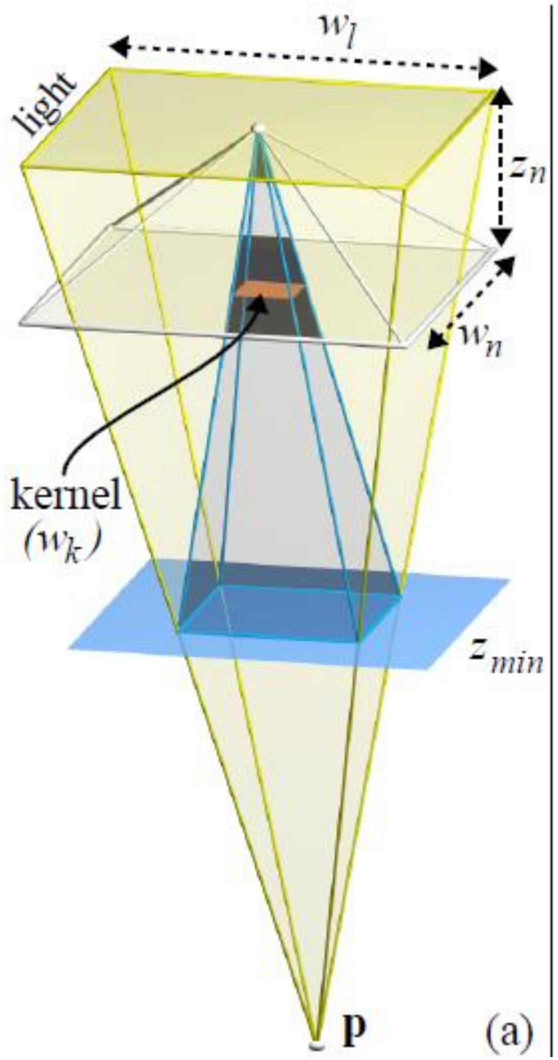
Visibility Computation (2007)

Kernel Computation

- In the HSM find the pyramid defined by the light quadrilateral and the point **p** that needs to be shaded

- Refine by projecting the intersection of the pyramid and the z_{min} plane defined by the top level of the HSM

- Iteratively refine by traversing the HSM



Visibility Computation (2007)

Visibility Computation

- Now we know where to search for occluders

- Instead of back projecting all occluders onto the light to compute the visibility percentage we only send the contour edges of the occluders as seen from **p**

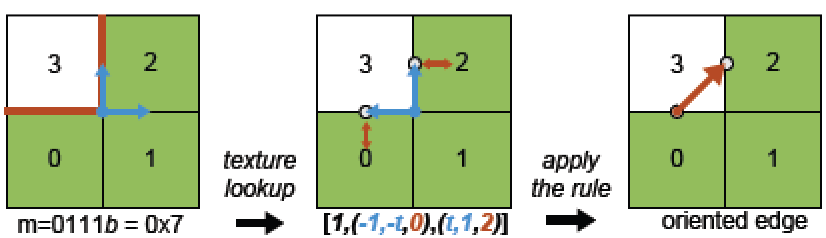
- The contours are filled by radially integration, this solves both the gap and overlap problems

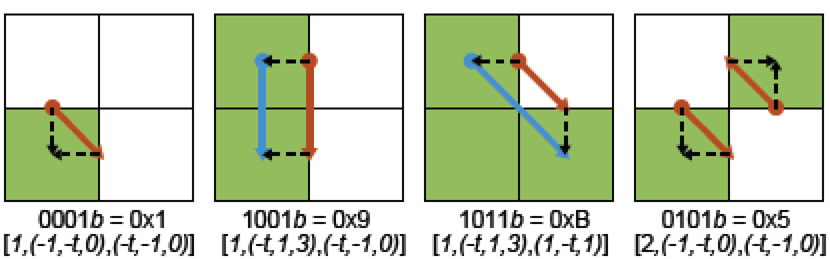
Visibility Computation (2007)

Smooth Contour Detection

- Contour detection algorithm based on 2x2 blocks of samples from the kernel

- Contours are precomputed (left image) shader only needs look up in a table (right image) based on a mask calculated from the occluded pixels



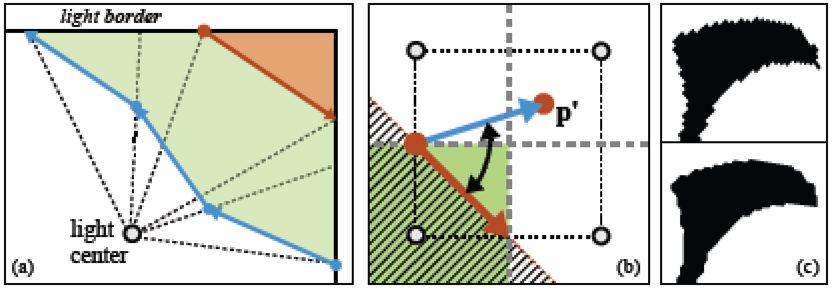


Visibility Computation (2007)

Radial Area Integration

- Accumulate the signed area covered from each contour edge (a)

- Similar to hard-shadow computations with normal shadow map, but we have to check the hard shadows are inside the contours (b) else we get an aliased result (c)



Visibility Computation (2007)

Results

No more shadow overestimation, smoother shadows, equal performance as 2006 algorithm

Ground truth

2006 algorithm

2007 algorithm

Visibility Computation (2007)

Results

But, still discontinuities in difficult shadows

Ground truth

2006 algorithm

2007 algorithm

Visibility Computation (2007)

Results

But, still discontinuities in difficult shadows

(Exaggerated)

Ground truth   2006 algorithm   2007 algorithm

Adaptive Precision (2006)

Problem Description

- Speed of the algorithm mainly depends on the size in pixels of the search areas

- For very large penumbra this can cause performance problems

- But very large penumbra require less detail than thinner ones (due to lower frequency)

Adaptive Precision (2006)

Solution

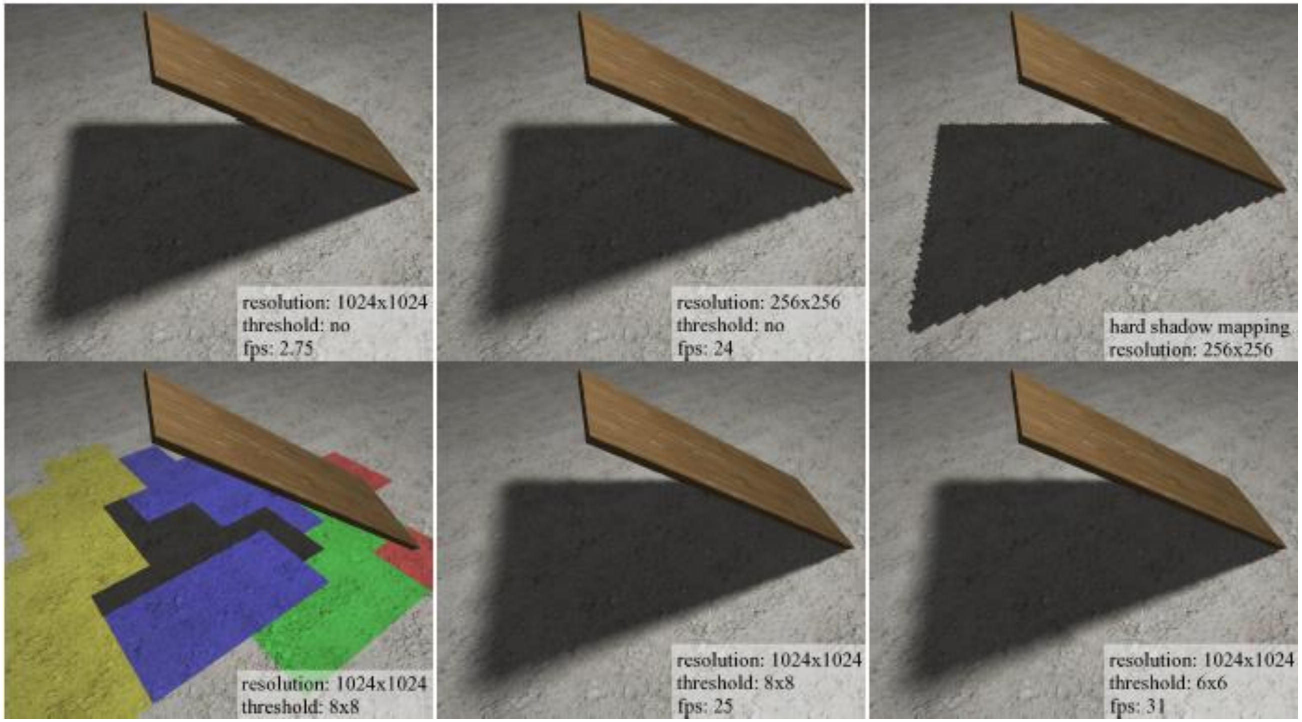
- Introduce a Hierarchical Shadow Map (HSM), similar to mip-maps or quadtrees

- Sample very large penumbra on a less detailed level in the HSM defined by a maximum search area threshold, guarantees a level of performance

- Leads to small visual quality degradations where levels change

Adaptive Precision (2006)

Results



Different settings for the threshold and shadow map resolution

Degradation

Adaptive Precision (2007)

Goals for 2007 algorithm

- Fix the degradation where different levels of the HSM are sampled

- Also provide a second optimization which works in screen space and reduces the output resolution, as opposed to the input resolution of the above light space solution

Adaptive Precision (2007)

Light Space Optimizations

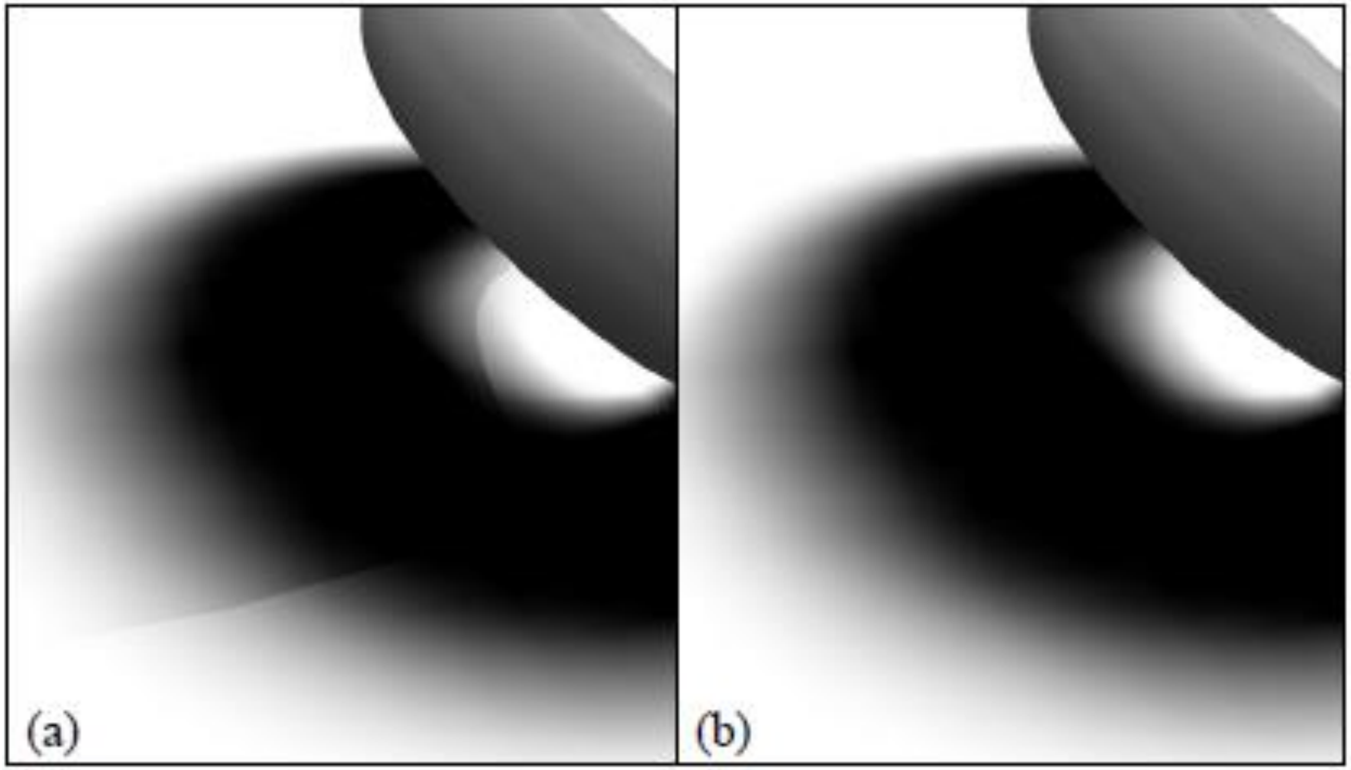
- Degradation is caused because of sampling level differences in the HSM

- The solution is to blend between different sampling levels using an algorithm akin to trilinear mip-map sampling

- Blending is done between the two closest HSM levels so overlaps are always smooth

Adaptive Precision (2007)

Light Space Optimizations Results



Adaptive Precision (2007)

Screen Space Optimizations, idea

- The idea is to adjust the screen resolution according to the screen space size of the penumbrae

- This is done by cancelling the visibility computations of some screen pixels

- The missing information is reconstructed using a pull-push algorithm

Adaptive Precision (2007)

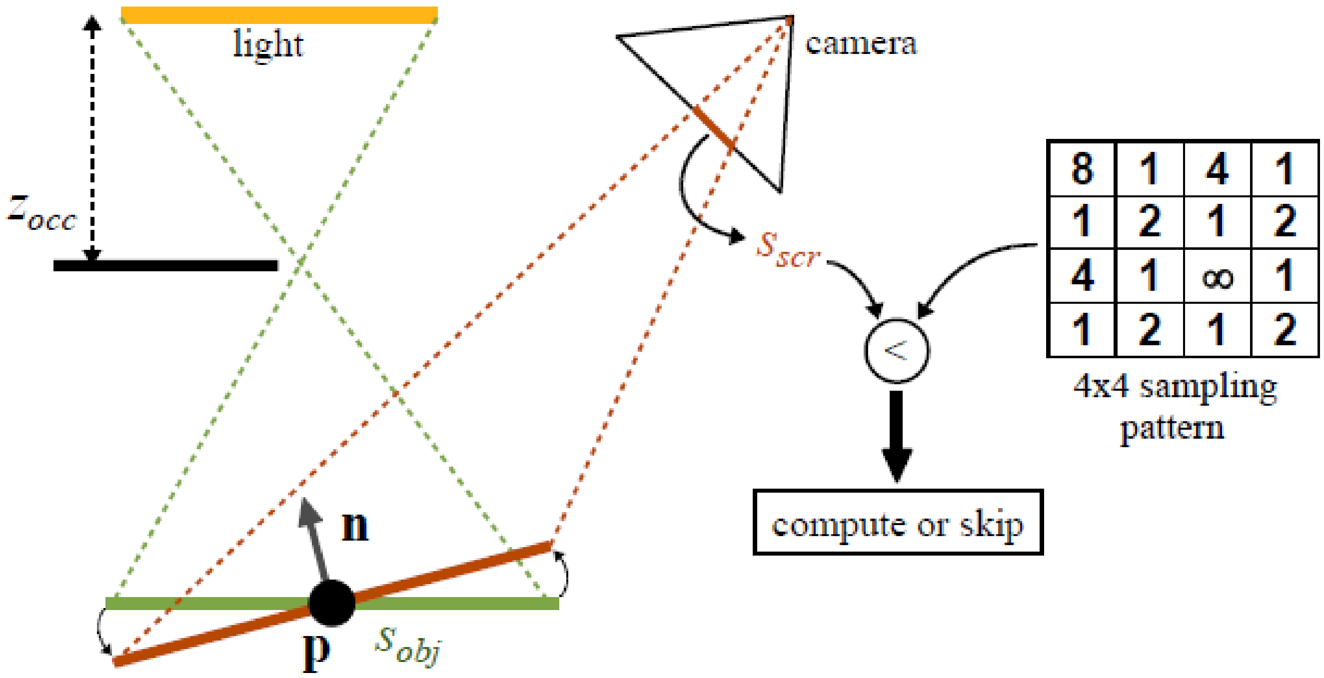
Screen Space Optimizations, skipping

- To do this correctly the penumbrae size is conservatively estimated as the smallest diameter of a projection of a disk with the object space radius of the penumbrae onto screen space

- The density of selected pixels is then adjusted for this screen space size of the penumbrae

Adaptive Precision (2007)

Screen Space Optimizations, skipping



Adaptive Precision (2007)

Screen Space Optimizations, Pull-Push

- This leads to a sparse unstructured visibility buffer that contains gaps

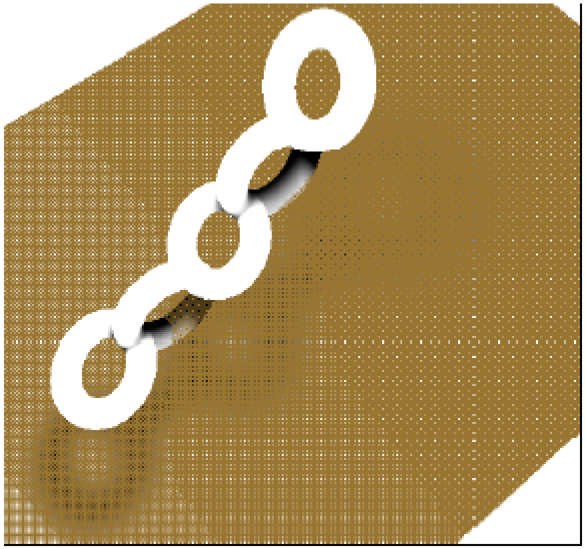
- A weight buffer is created with 1’s for the computed pixels and 0’s for the gaps

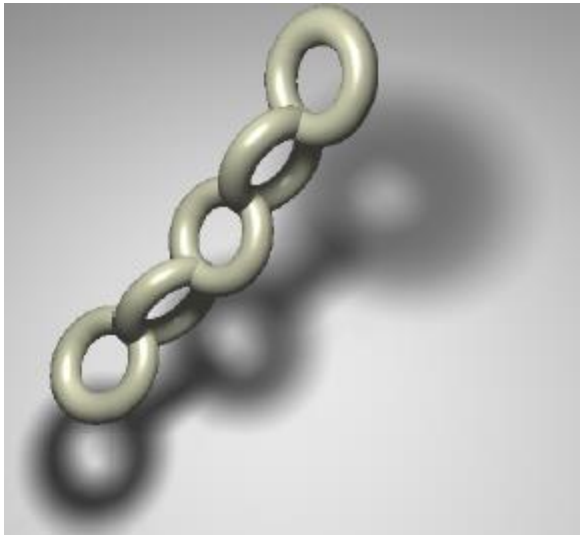
- Pull: the weight and visibility buffers are reduced by accumulating and averaging

- Push: the gaps are iteratively filled, from lowest to highest resolution by linear blending

Adaptive Precision (2007)

Screen Space Optimizations, Pull-Push

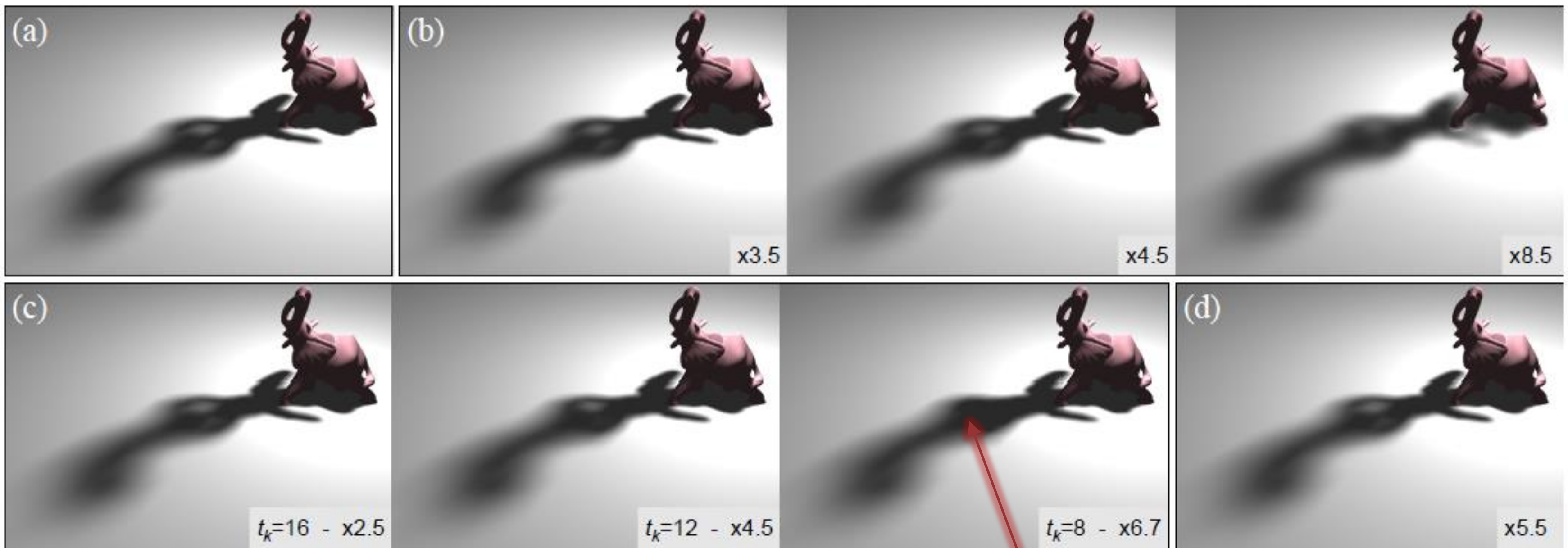




Before and after Pull-Push reconstruction, orange pixels are skipped pixels (gaps)

Adaptive Precision (2007)

Screen Space Optimizations, Results



a. Raw algorithm
b. View dependent selection (from high to low quality)
c. Light space adaptive precision (from hight to low quality
d. View depedent selection (high quality) + light space adaptive precision (high quality)

The numbers indicate the speed up of the algorithm, total performance went from a: 31fps to d: 71fps

Video

Summary & Discussion

Pros

- Geometry independent
- Realtime, good performance
- Nice results
- Easy to bias towards performance or quality
- Even wrong (higher performance) shadows look good
- All pros of shadow maps

Summary & Discussion

Cons

- Discontinuities
- Single light artifact
- All cons of shadow maps
- Aggressive reconstruction in light space can lead to blurring penumbrae
- Aggressive reconstruction in screen space can lead to filling in lit areas with shadows

Questions

?