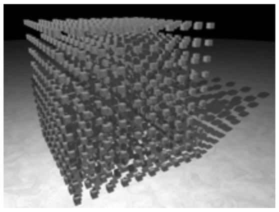
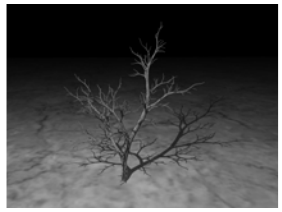
~~An Efficient Hybrid Shadow Rendering Algorithm~~

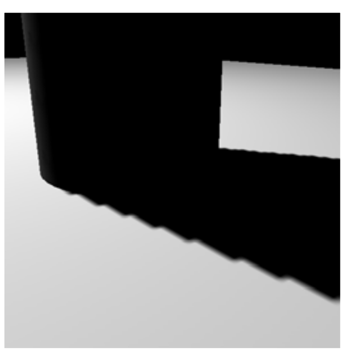

Eric Chan
Frédo Durand

Massachusetts Institute of Technology

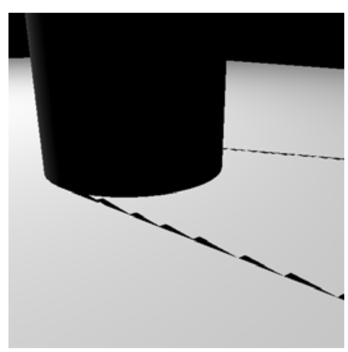
Not Another Talk on Shadows?!

Main ideas:

- combination of shadow maps + shadow volumes
- computation masks



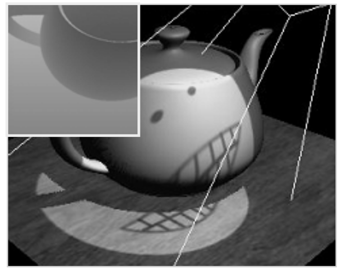
+



Classic Shadow Algorithms

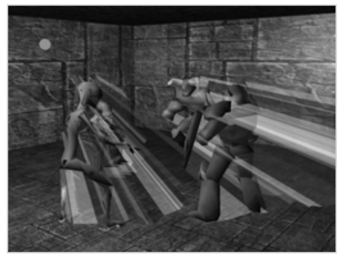
Shadow maps (Williams 1978)

- fast and simple
- undersampling artifacts
- lots of recent research!



Shadow volumes (Crow 1977)

- object-space
- accurate
- accelerated by stencil buffer
- high fillrate consumption!

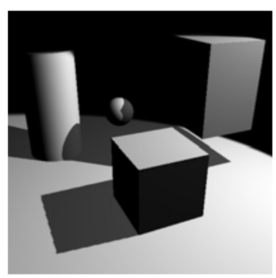


NVIDIA

Fillrate Problem

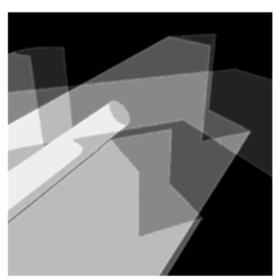
Lots and lots of fillrate!

- rasterization
- stencil updates



Why?

- polygons have large screen area
- polygons overlap



Fillrate Problem

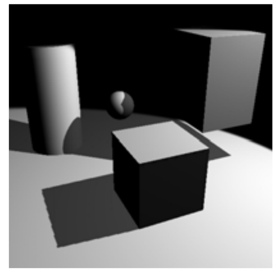
Lots and lots of fillrate!

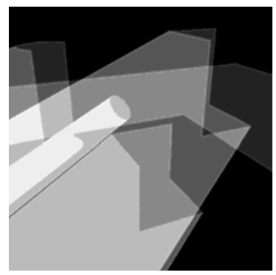
- **rasterization**
- **stencil updates**

Why?

- **polygons have large screen area**
- **polygons overlap**

But is this <u>really</u> a problem?





But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- **bump mapping**

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- **bump mapping**

- **per-pixel surface shading**

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- bump mapping
- per-pixel surface shading
- dynamic and projected lights

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- bump mapping
- per-pixel surface shading
- dynamic and projected lights
- atmospheric effects

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- bump mapping
- per-pixel surface shading
- dynamic and projected lights
- atmospheric effects
- particle effects

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- bump mapping
- per-pixel surface shading
- dynamic and projected lights
- atmospheric effects
- particle effects

- shadow volumes

But Is This *Really* A Problem?

Case study: Doom 3 engine (id Software)

- bump mapping
- per-pixel surface shading
- dynamic and projected lights
- atmospheric effects
- particle effects

} **50%**

- shadow volumes } **50%**

"Shadowing accounts for about half of the game's rendering time."

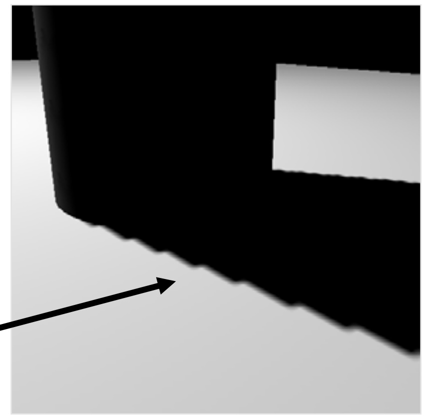
— John Carmack

Two Observations

Two Observations (shadow maps)

Shadow-map aliasing is ugly

But — only noticeable at shadow silhouettes

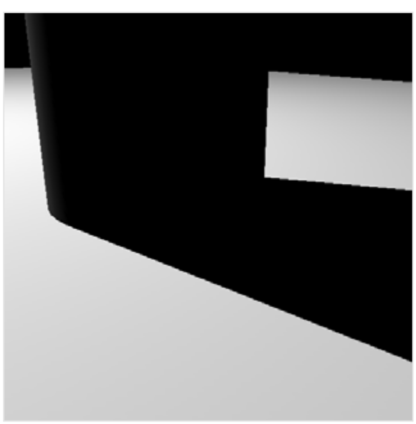


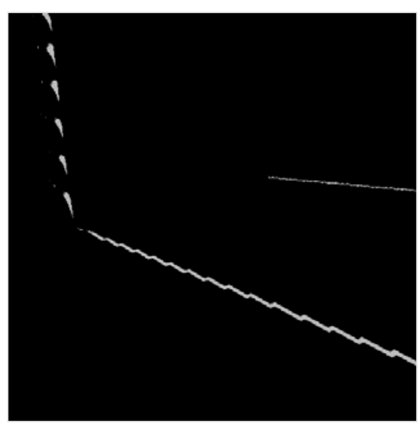
shadow silhouette

Two Observations (shadow volumes)

Shadow volumes are accurate everywhere

But — accuracy is only needed at silhouettes





Hybrid Approach

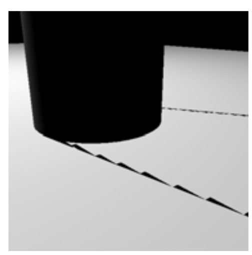
Decompose the problem:

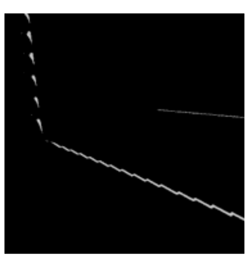
- use shadow volumes at silhouettes
- use shadow maps everywhere else

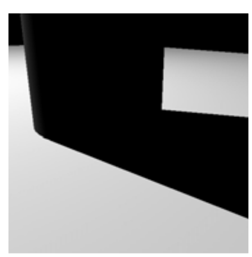
shadow map + shadow volume

Algorithm

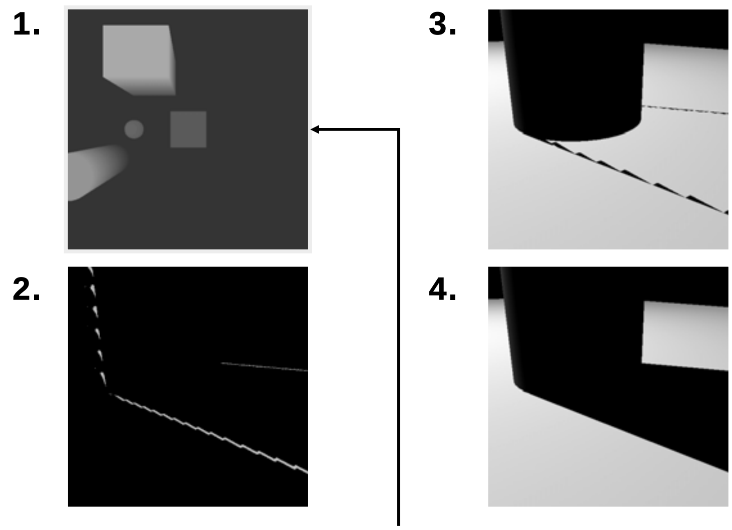
1. 

3. 

2. 

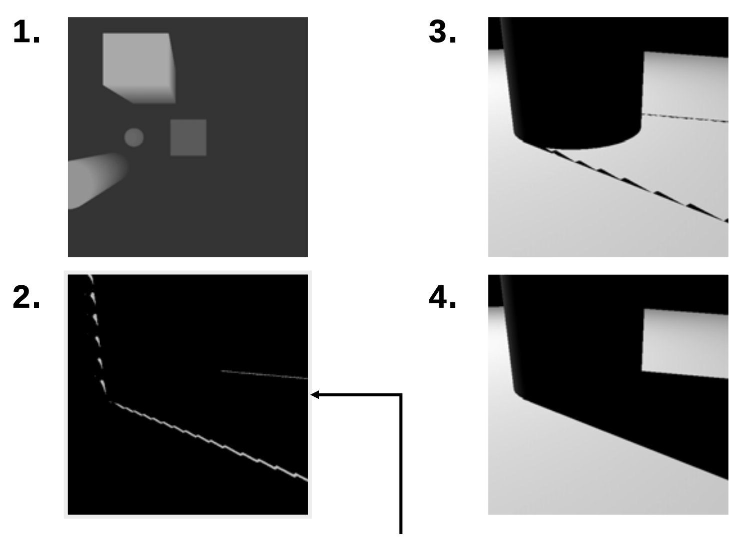
4. 

Algorithm



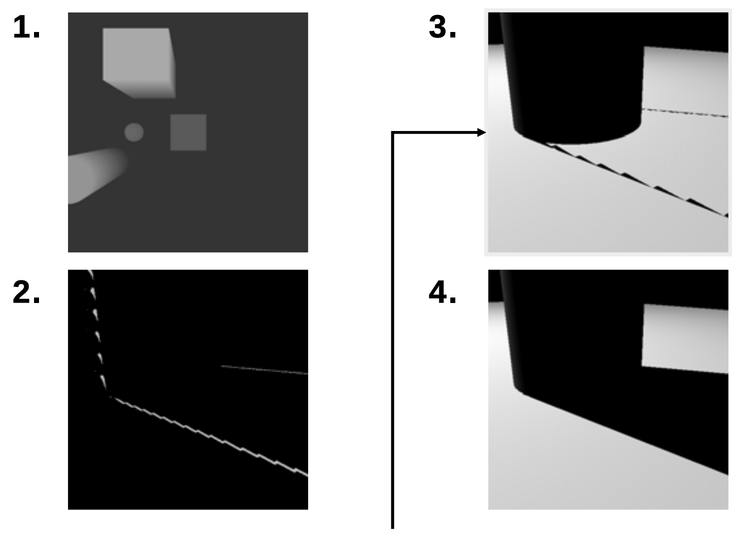
create a shadow map

Algorithm



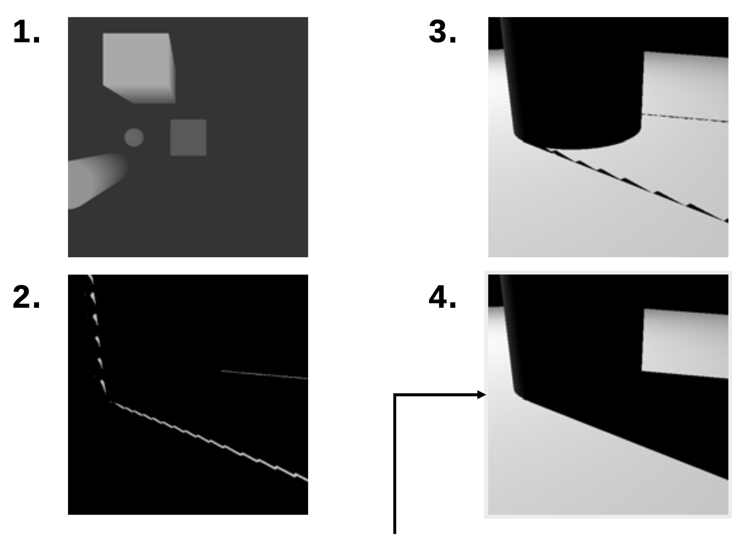
find silhouette pixels

Algorithm



apply shadow volumes <u>only at silhouette pixels</u>

Algorithm

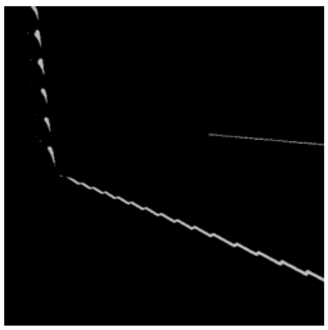


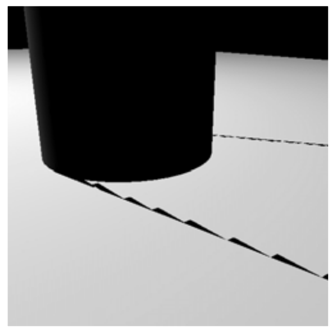
apply shadow maps everywhere else

Algorithm Details

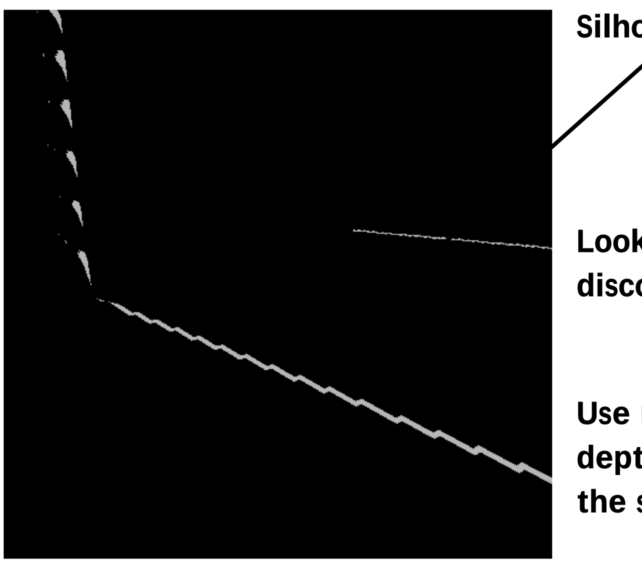
Questions:

- **how to find silhouette pixels?**
- **how to rasterize only silhouette pixels?**





Find Silhouette Pixels

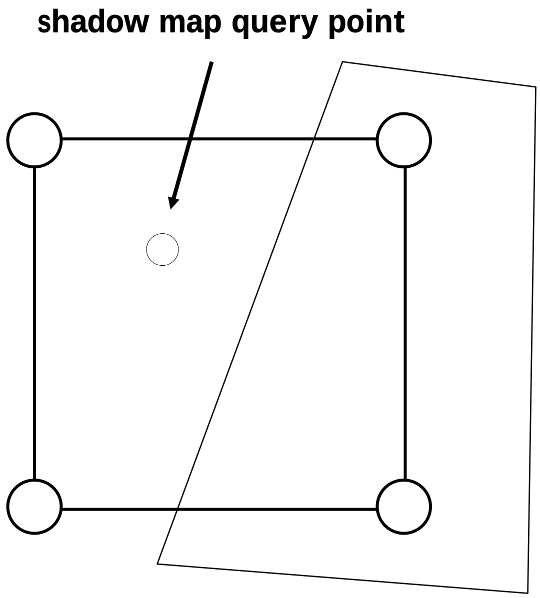


Silhouette pixels

Look for depth discontinuities

Use nearest 2x2 depth samples of the shadow map

Find Silhouette Pixels (example)



Check results:

- 2 in shadow
- 2 visible

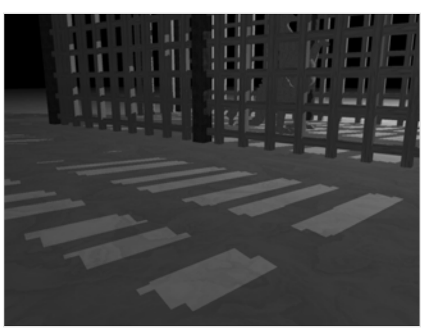
Disagreement!

- silhouette pixel

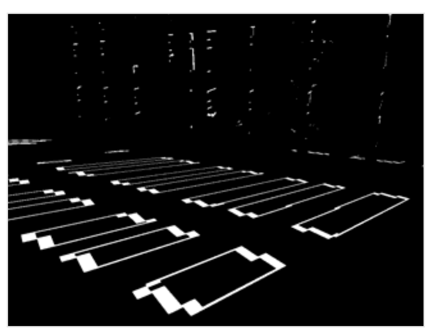
Restricted Rasterization

Use a mask to limit rasterization:

- tag silhouette pixels in framebuffer
- mask off all other pixels



example scene

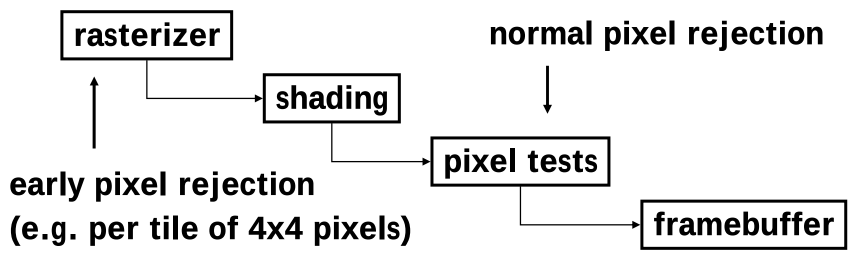


mask

Computation Mask

We need a <u>computation mask</u>

- user-specified mask
- hardware early pixel rejection
- reduces rasterization, shading, memory bandwidth



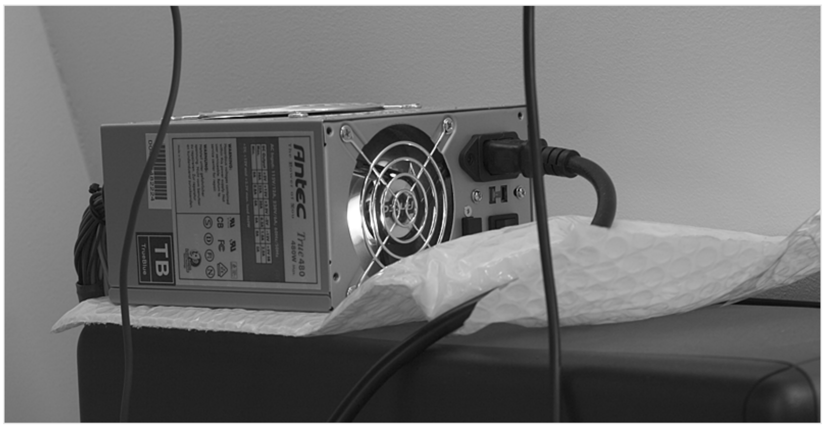
Hardware Support

Current hardware doesn't have computation mask

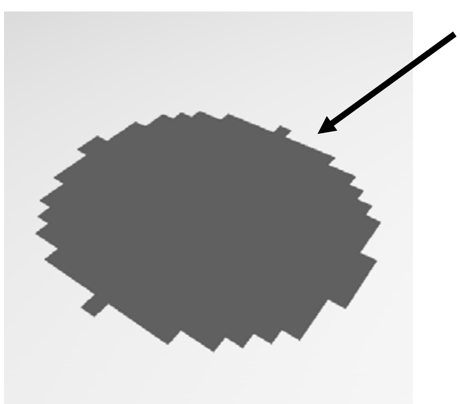
- but — hardware already has early z culling!
- minimal changes needed for native mask support
- our implementation uses a <u>simulated</u> mask

Results

- 2.6 GHz Pentium 4
- NVIDIA GeForce 6 (NV40) + crazy blue power supply



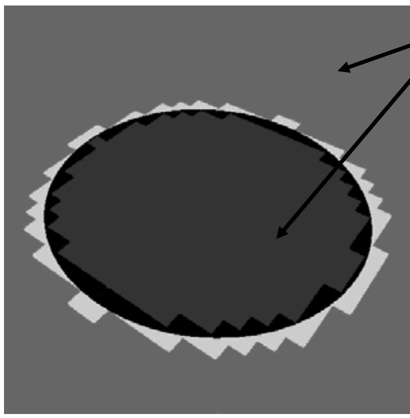
Hybrid Algorithm Example



Aliased shadow of a ball

standard shadow map

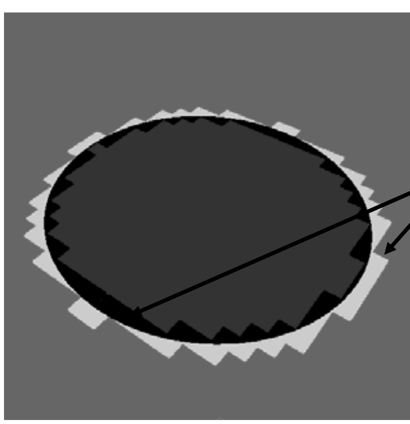
Hybrid Algorithm Example



Blue and red regions
handled by shadow maps

visualization

Hybrid Algorithm Example

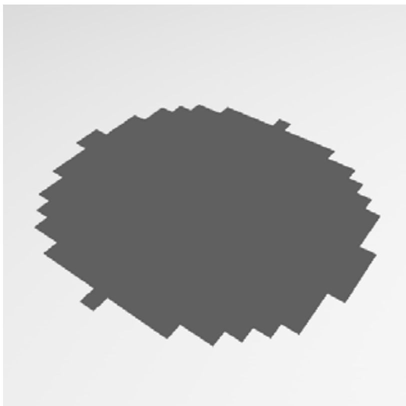


Blue and red regions
handled by shadow maps

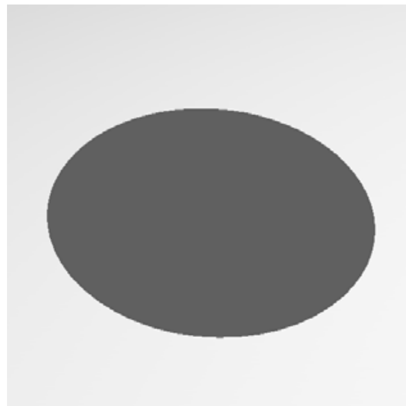
Black and green regions
handled by shadow volumes

visualization

Hybrid Algorithm Example

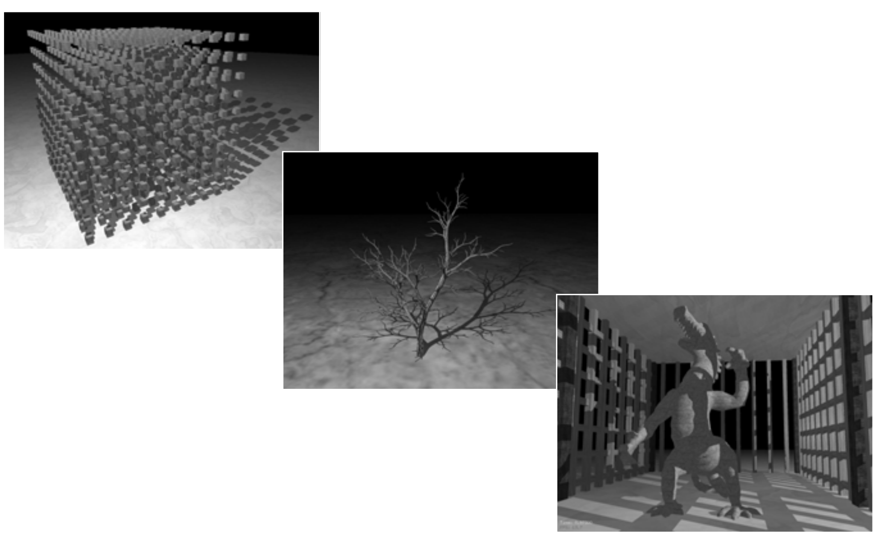


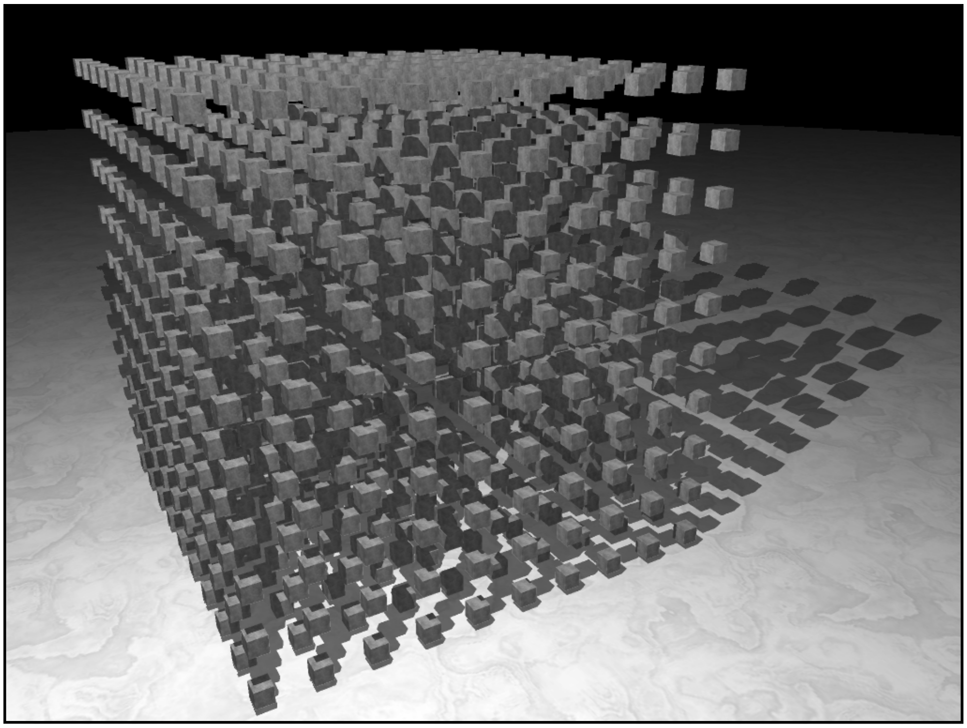
standard shadow map

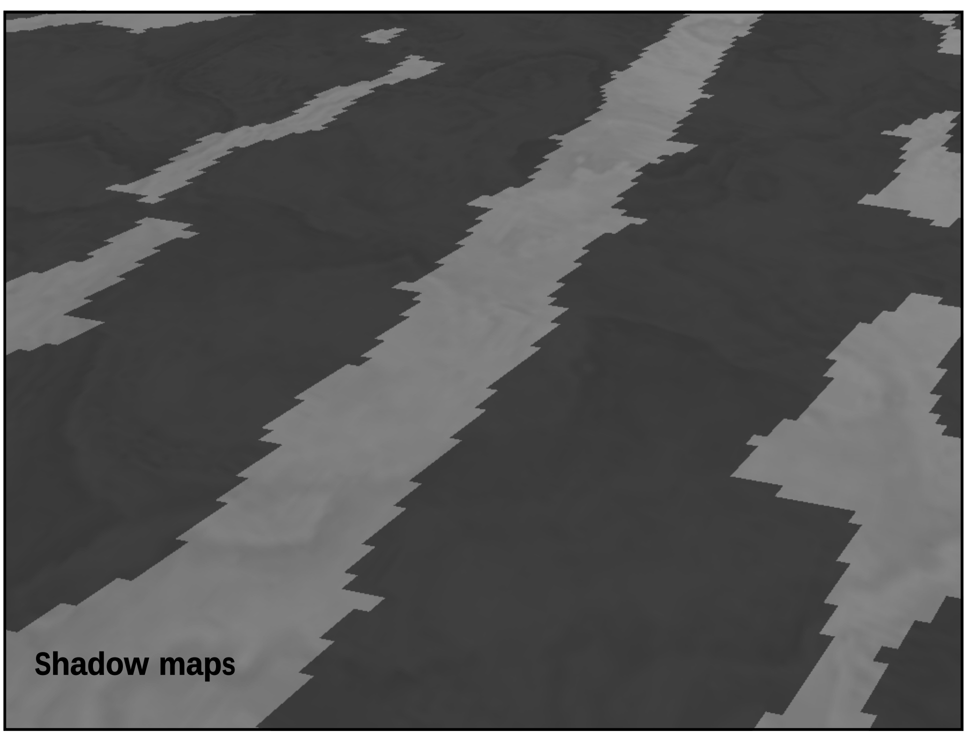


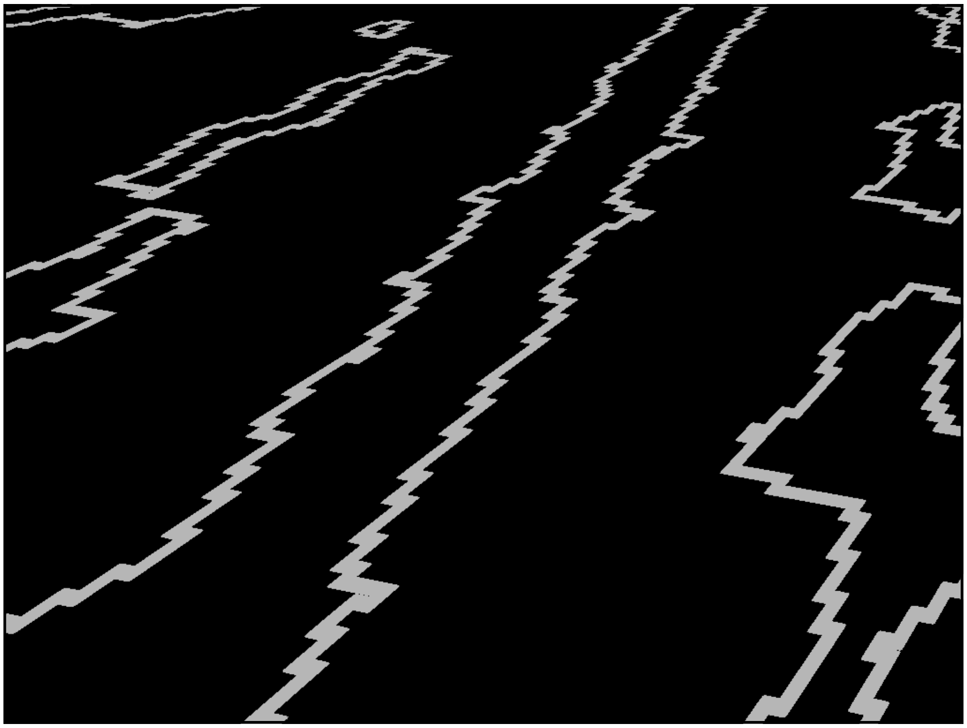
hybrid algorithm

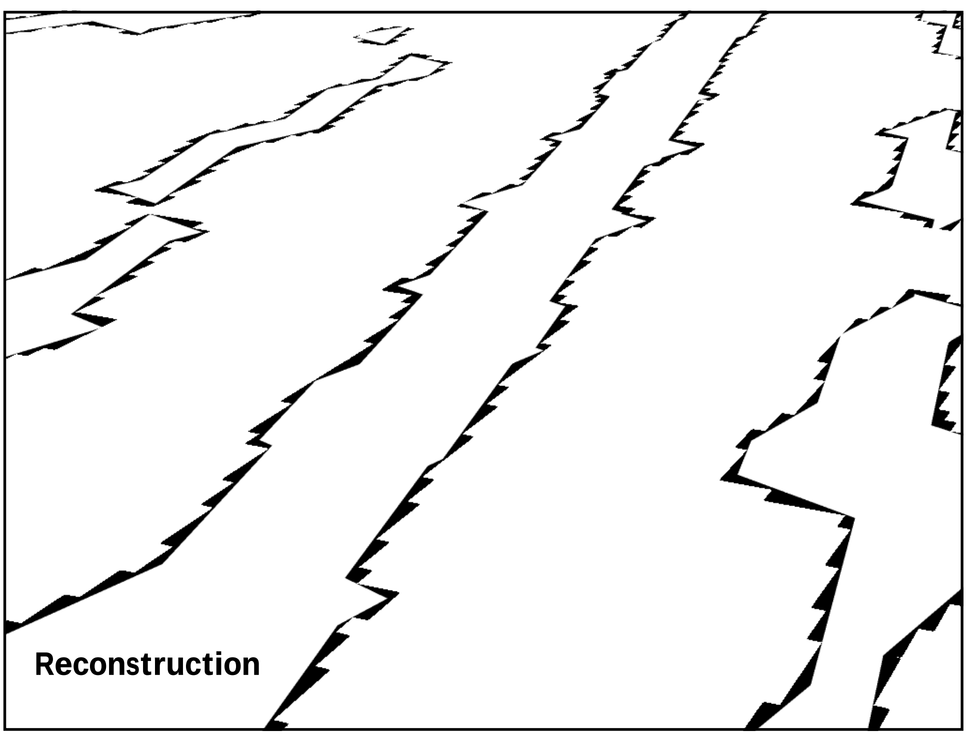
Test Scenes

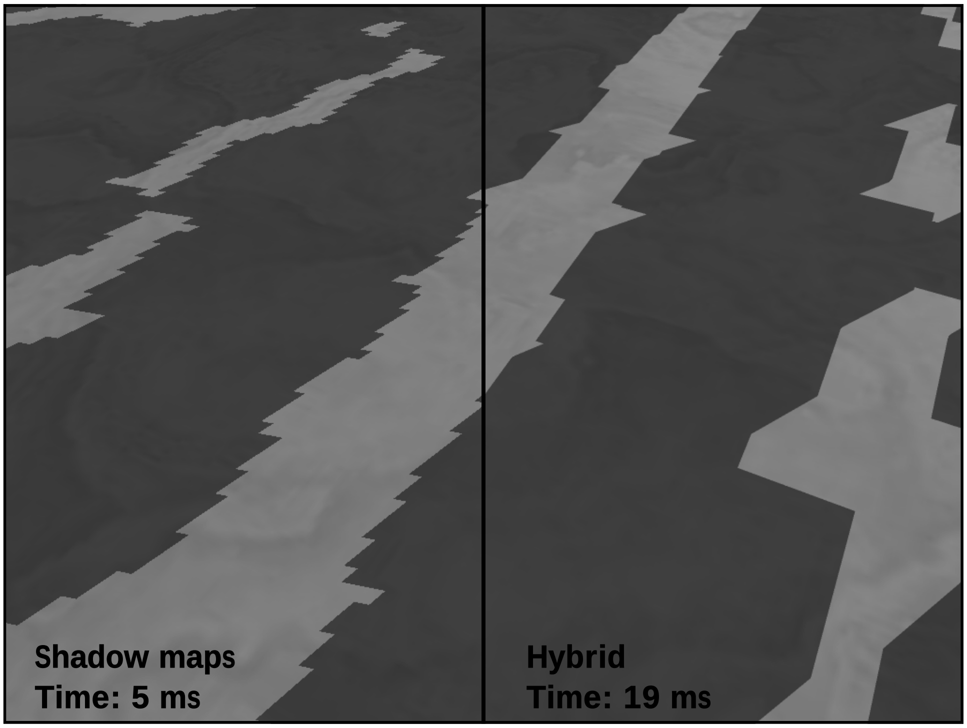


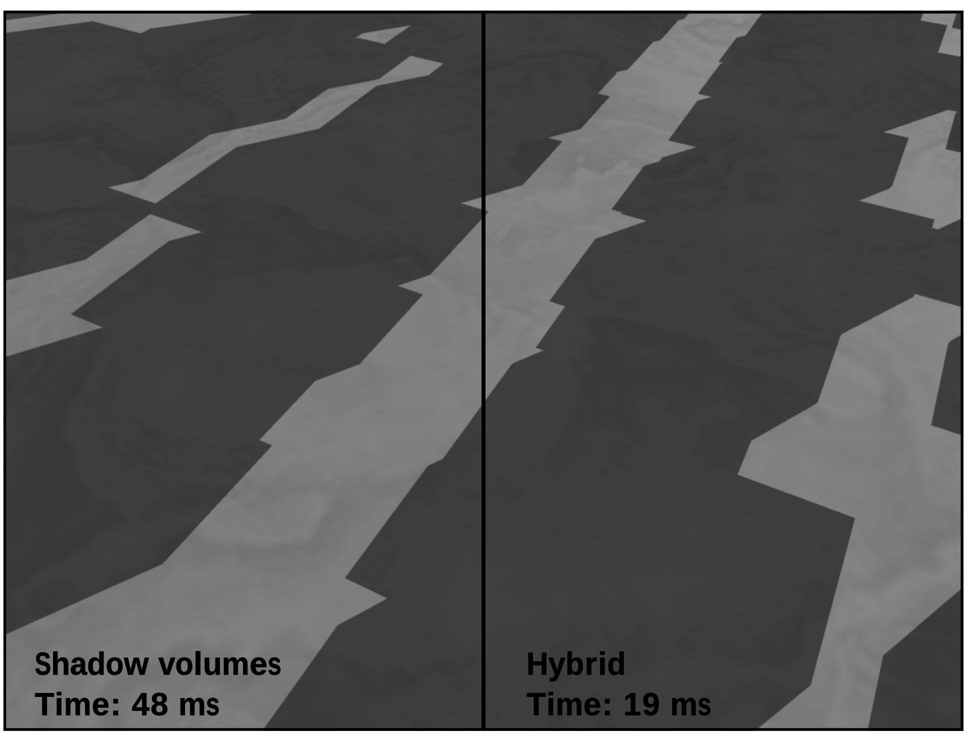



Shadow maps



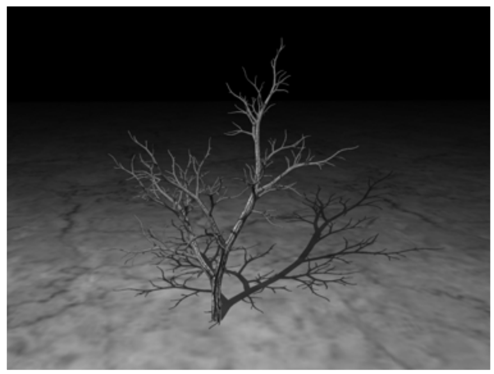

Reconstruction

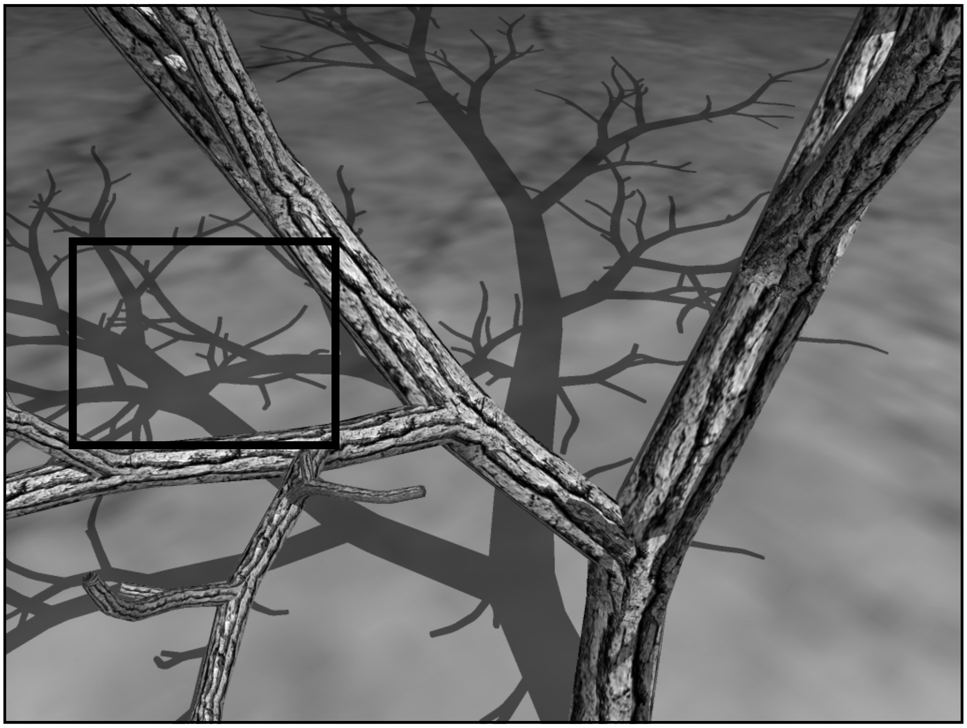

Shadow maps
Time: 5 ms
Hybrid
Time: 19 ms


Shadow volumes
Time: 48 ms
Hybrid
Time: 19 ms

Artifacts

- Low-resolution shadow map ⟶ discretization errors
- Misclassified silhouette pixels ⟶ missing features
- Difficult cases: fine geometry



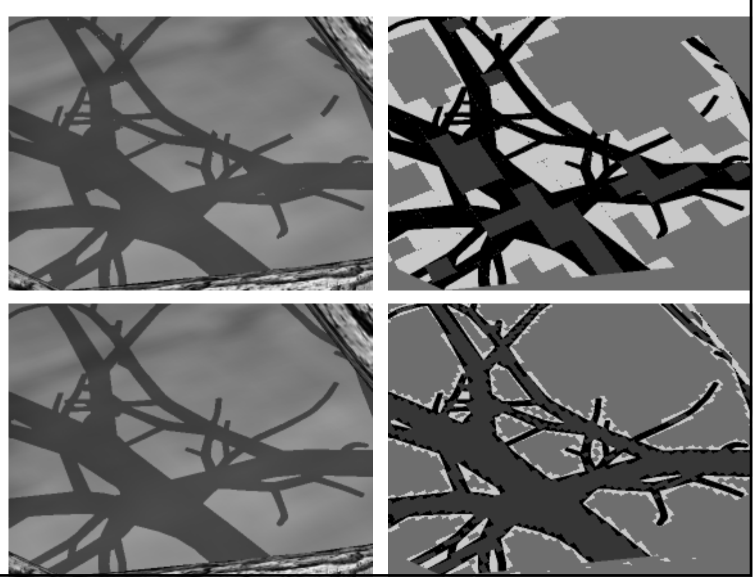


Example of Missing Features

result | visualization

256x256

1024x1024



Discussion

Algorithm designed to help fillrate-bound applications:

- requires an extra rendering pass
- 30% to 100% speedup in our test scenes
- performance depends a lot on culling hardware

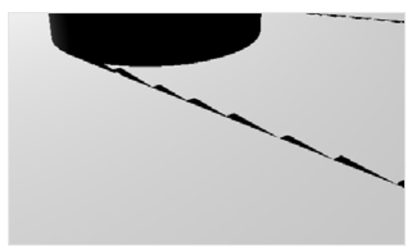
More details in the paper and web page …

- tradeoff analysis
- comparison to related work
- implementation details
- more performance and image comparisons

Summary

Hybrid shadow algorithm

+



Screen-space decomposition:

- most pixels use fast (but inexact) algorithm
- a few pixels use accurate (but expensive) algorithm

Computation Masks

Why?

- **pixels are not created equal**
- **programmer marks "interesting" pixels**
- **fast reject all other pixels**
- **not just for shadows!**
- **useful in general for multipass algorithms**
- **hardware is (mostly) already there**

Acknowledgments

Nick Triantos and Mark Kilgard (NVIDIA)

Jan Kautz and Addy Ngan (MIT)

Timo Aila

ASEE NDSEG Fellowship