Decoupled Sampling for Real-Time Graphics Pipelines

Jonathan Ragan-Kelley, MIT CSAIL
Jaakko Lehtinen, Jiawen Chen, MIT CSAIL
Michael Doggett, Lund University
Frédo Durand, MIT CSAIL

Complex:
Geometry
Visibility
Shading
“Virtually indistinguishable from live action...as visually rich as real scenes.”

“This goal...force[d] us to completely rethink the entire rendering process.”

Cook et al., 1987
The Reyes Image Rendering Architecture
Rendering:
Compute what’s visible.
Compute what color it is.
Complex visibility
many stochastic point samples
in 5D (space, time, lens aperture)

Complex shading
expensive evaluation
can be prefiltered
Design goals

1. Scale to large numbers of stochastic visibility samples.
2. Shade at the lowest frequency possible.
3. Only shade visible points.
Design goals

1. Scale to large numbers of stochastic visibility samples.
2. Shade at the lowest frequency possible.
3. Only shade visible points.
A simple graphics pipeline

transformed primitives

covered pixels, shading samples

visible pixels, shading samples

colored pixels
Supersampling

transformed primitives

covered subpixels, shading samples

visible subpixels, shading samples

colored subpixels

Xform → Rast → Depth-Stencil → Shade → FB

transformed primitives

covered subpixels, shading samples

visible subpixels, shading samples

colored subpixels

Beyond Programmable Shading 2010
Supersampling generalizes to blur

transformed primitives

covered subpixels, shading samples

visible subpixels, shading samples

colored subpixels

Xform → Rast → Depth-Stencil → Shade → FB

covered subpixels, shading samples

visible subpixels, shading samples

colored subpixels

Beyond Programmable Shading 2010
Multisampling (MSAA)

- Transformed primitives
- Covered subpixels, shading samples
- Visible subpixels, shading samples
- Colored subpixels

Xform → Rast → Depth-Stencil → Shade → FB

Beyond Programmable Shading 2010
MSAA breaks with blur: attributes move relative to pixels
MSAA breaks with blur: attributes move relative to pixels
Reyes

transformed primitives → diced micropolygons → shaded micropolygons

colored subpixels → visible colored subpixels

Xform → Split-Dice → Shade → Rast → Depth → FB

Beyond Programmable Shading 2010
Reyes generalizes to blur

- Transformed primitives
- Diced micropolygons
- Shaded micropolygons
- Colored subpixels
- Visible colored subpixels

Xform → Split-Dice → Shade → Rast → Depth → FB
<table>
<thead>
<tr>
<th>Feature</th>
<th>Super-sampling</th>
<th>MSAA</th>
<th>Reyes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stochastic visibility sampling</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Shade at independent (lower) frequency</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Shade only what’s visible</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Method</td>
<td>Super-sampling</td>
<td>MSAA</td>
<td>Reyes</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>Stochastic visibility sampling</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Shade at independent (lower) frequency</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Shade only what’s visible</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>

Beyond Programmable Shading 2010
Our technique:

Post-visibility Decoupled Sampling

1. Separate visibility from shading samples.

2. Define an explicit mapping from visibility to shading space.

3. Use a cache to manage irregular shading-visibility relationships, without precomputation.
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples
(screen space)
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples
(screen space)
Decoupled Sampling with motion blur

Visibility samples (screen space)

t = 0.0

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

$t = 0.0$

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

\[ t = 0.0 \]

Visibility samples (screen space)

Shading grid

Foreach primitive:
  Foreach vis sample:
    Skip if not visible
    Map to shading sample
    If not in cache:
      Shade and cache
    Else:
      Use cached value

Beyond Programmable Shading 2010
t = 0.0

Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples (screen space)
Decoupled Sampling with motion blur

visibility samples (screen space)

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples
(screen space)
Decoupled Sampling with motion blur

t = 0.0

visibility samples (screen space)

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

\[ t = 0.75 \]

visibility samples
(screen space)

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

Visibility samples (screen space)

$t = 0.0$

Shading grid

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

$$t = 0.5$$

visibility samples (screen space)

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples
(screen space)

shading grid

t = 0.5
Decoupled Sampling with motion blur

\[ t = 0.75 \]

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples (screen space)

shading grid

t = …
Decoupled Sampling with motion blur

Visibility samples (screen space)

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
      use cached value
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

Visibility samples (screen space)
Decoupled Sampling with motion blur

foreach primitive:
  foreach vis sample:
    skip if not visible
    map to shading sample
    if not in cache:
      shade and cache
    else:
      use cached value

visibility samples (screen space)
shading grid
Decoupled Sampling with motion blur

foreach primitive:
    foreach vis sample:
        skip if not visible
        map to shading sample
        if not in cache:
            shade and cache
        else:
            use cached value
Decoupled Sampling

Transformed primitives → Rast → Depth-Stencil → Map → FB

- Covered subpixels
- Visible subpixels

Cache
Shade

Beyond Programmable Shading 2010
Decoupled Sampling

- Transformed primitives
- Covered subpixels
- Visible subpixels
- Map
- DepthStencil
- Rast
- Xform
- FB

Shading requests
Cache
Shade

Beyond Programmable Shading 2010
Decoupled Sampling

transformed primitives

covered subpixels

visible subpixels

Xform → Rast

Depth-Stencil

Map

FB

shaded results

shading requests

Cache

Shade
Decoupled Sampling

- Transformed primitives
- Covered subpixels
- Visible subpixels
- Colored subpixels

Diagram:
- Xform → Rast → Depth-Stencil → Map → FB
- Shading requests → Shade → Shaded results
- Cache
Decoupled Sampling

Xform → Rast → Depth-Stencil → Map → FB

- Transformed primitives
- Covered subpixels
- Visible subpixels
- Colored subpixels

Shading requests → Shaded results

Beyond Programmable Shading 2010
Blur vs. shading rate: defocus

Half-Life 2, Episode 2
1280x720, 27 samples/pixel

4.5-43x less shading than ideal supersampling
Blur vs. shading rate: motion

Team Fortress 2
1280x720, 27 samples/pixel

3-40x less shading than ideal supersampling
Blur vs. Shading Rate

Decoupled Sampling
64 visibility samples
Decoupled Sampling:

✓ Scales to large numbers of stochastic visibility samples.
✓ Shades at the lowest frequency possible.
✓ Only shades visible points.
We can map to more than to pixels:
blur-adaptive shading rate

Adaptive shading rate:
37% less shading

max pixel error <5%
We could map to object-space

Burns et al. 2010
A Lazy Object-Space Shading Architecture with Decoupled Sampling.
Integration with a fragment shading pipeline

Quads for derivatives
Minimum granularity of shading requests are *quads*, not individual samples. Decoupling gives high quad occupancy (unlike stochastic supersampling).

Wide SIMD shader execution
*Dynamic* quad coalescing hardware just works. Larger shading request batches trade overshade for *static* SIMD coherence.

Texture coherence is essentially unaffected
Simulation of texture cache hit rate shows no measurable difference.
Summary

It is possible to decouple shading from visibility sampling in a fragment shading system.
Explicit mapping from shading to visibility samples decouples rates.
Caching naturally manages irregular communication.

Decoupled Sampling makes GPU-style pipelines scalable to rendering with stochastic blur at modest shading cost.

Stochastic motion and defocus blur are becoming feasible for GPUs!
Thank you

Special thanks:
- Jason Mitchell - Valve content
- NVIDIA, Intel, Singapore-MIT Gambit - Funding
- Kayvon Fatahalian, Jeremy Sugerman, Solomon Boulos - Talk feedback