High-Quality Parallel Depth-of-Field Using Line Samples

Stanley Tzeng, Anjul Patney, Andrew Davidson, Mohamed S. Ebeida, Scott A. Mitchell, John D. Owens

University of California, Davis  Sandia National Laboratories
Depth of Field is Beautiful

Pixar’s Toy Story 3
The Problem

Good noise-reducing Depth-of-Field effects takes a lot of point samples!
The Problem

Good noise-reducing Depth-of-Field effects takes a lot of point samples!
The Problem

Good noise-reducing Depth-of-Field effects takes a lot of point samples!
The Problem

Good noise-reducing Depth-of-Field effects takes a lot of point samples!
This Talk

• Instead of point samples, use **line samples**!
  – Heavier compute per sample, but need fewer samples for good results

• A tiled line sampling renderer for graphics hardware
  – How to make line sampling work within tight memory constraints?
Line Samples

A dimensional extension of point samples

Gribel et al.: High-Quality Spatio-Temporal Rendering using Semi-Analytical Visibility

Jones and Perry: Antialiasing with Line Samples
Point Samples in a pixel

Scene primitive
Point Samples in a pixel

Shoot samples

Pixel
Point Samples in a pixel

Intersect & Generate colored samples
Point Samples in a pixel

Composite

Pixel
Line Samples in a pixel
Line Samples in a pixel

Shoot samples

Pixel
Line Samples in a pixel

Intersect & Generate colored line segments
Line Samples in a pixel

Composite

Pixel
Side Note

Samples can be in any arbitrary orientation.
Depth of Field Sampling

• Point Sampling: Generate 4D point samples in \((x,y,u,v)\) space.
  – Point on screen \((x,y)\) +
  – Point on lens \((u,v)\)

• Line Sampling: Fix a point in screen space \((x,y)\), sample along the lens in \((u,v)\) space.
How to Sample Along the Lens?

Grid
- Good area coverage
- Uneven line lengths

Pinwheel
- Even Line Lengths
- Bias towards center
How to Sample Along the Lens?

Pinwheel

- Even Line Lengths
- Bias towards center

Reweigh the samples obtained
Step 1: Project scene geometry from screen space to lens space.
Line Samples for DoF

Step 2: Sample the lens using a wagonwheel pattern
Line Samples for DoF

Step 3: Record intersected segments
Line Samples for DoF

Step 4: Composite segments to pixel color
Line Samples using DoF

2 Line Samples
Line Samples using DoF

4 Samples
Line Samples using DoF

8 Samples
Line Samples using DoF

16 Samples
Comparison

16 Samples
Point Samples

16 Samples
Line Samples
Comparison

256 Samples
Point Samples

16 Samples
Line Samples

Metric of Comparison!
Line Sampling on the GPU

What’s really important to us:

• Easy parallel processing
• Bounded memory requirements
  – Shared memory is limited
  – Games require fixed storage
Parallel Implementation: 2 Kernels

Intersect line samples with scene geometry, & generate line segments

Global Line Segment Buffer

Composite and filter segments into final pixels
Parallel Implementation: 2 Kernels

Intersect line samples with scene geometry, & generate line segments

Global Line Segment Buffer

Composite and filter segments into final pixels

Problem Occurs Here!
Problem

800 x 600 x 16 x 16 x 24

width x height x # line samples x # line segments x Bytes per segment

![Application window with Out of memory message]
Binning

- Split screen into tiles – use shared memory instead of global memory
- Combine the two kernels into one
New Problem

• New problem: Shared memory is small.

Line Segments

Line Segment Buffer
Solution:
Keep only the important line segments!

“All animals are equal, but some animals are more equal than others.”

Animal Farm
Heuristics

Only keep segments that contribute the most to the color of the sample

- Green Segment Too Small
- Green Segment Occluded

UCDavison
Heuristics

Only keep segments that contribute the most to the color of the sample

- Green Segment Too Small
- Green Segment Occluded
- Blue Segment Merged
Heuristics

Rules:
• Discard if segment is too short
• Discard if segment is occluded
• Merge if segments are similar

A small amount of shared memory = small tiles
Another Problem!
Bins Overflow

4x4 bins, 256 max triangles
Bins Overflow

4x4 bins, 512 max triangles
Bins Overflow

4x4 bins, 1024 max triangles
Bins Overflow

Overflow because of depth

Overflow because of large CoC.
Bins Overflow

Overflow because of depth

Overflow because of large CoC.
Bins Overflow

Overflow because of depth

Overflow because of large CoC.
Bins Overflow

Overflow because of depth

Overflow because of large CoC.
Level of Detail

- Close and blurred: Low detail
- Close and sharp: High detail
- Far and blurred: Low detail
Level of Detail

c/a says, in addition to a traditional LoD scheme:

• If object is close but very blurred – choose lower LoD
• If object is far but still sharp – choose higher LoD

1/3 as many triangles, and
1.6 x improvement in speed
Final System Structure

- Vertex Shading
  - CoC Evaluation
  - 1 Vertex / Lane

- Binning
  - 1 Tri / Lane

- Sample
  - 1 Tile / Core
  - 1 Pixel / 16 Lanes
  - 1 Sample / Lane

- Composite
Stage 1: Vertex Transform and CoC computation.
Final System Structure

Stage 2: Triangle Binning into Tiles
Stage 3: Tile Sample, Composite, and Filter
Results

256 Point Samples
0.49 fps

16 Line Samples
2.04 fps
Results

256 Point Samples
0.64 fps

16 Line Samples
2.86 fps
Drawbacks

Several Problems to the current model

• Heavily dependent on shared memory.
• Not completely accurate – sometimes significant segments may be discarded.
• Shading is done once per line segment.
  – Can cause oversmoothing.
Artifacts

Point Samples
• Too few point samples causes noise

Line Samples
• Too few line samples causes ghosting
Convergence

Point Samples
• Takes many samples to converge

Line Samples
• Converges much faster than point samples

256 Samples

16 Samples
Contributions

• Depth-of-field using line sampling techniques
• Implementing tiled line sampler on GPUs
• Blur dependent level of detail
Future Work

• Line sampling in more aggressive stochastic rendering systems
  – Area lighting / Soft shadows
  – Global Illumination

• Better memory management
  – Better heuristics?
Thank you!

Special thanks to Intel’s ISTC-VC and NSF for funding!

Alduin from TES V: Skyrim