Antialiasing with Line Samples

Thouis R. Jones, Ronald N. Perry
MERL - Mitsubishi Electric Research Laboratory

Antialiasing

- Fundamentally a sampled convolution:

$$S(P_x, P_y) = \int\int I(x, y) F(x - P_x, y - P_y) dxdy$$

$S(P_x, P_y)$ – Sampled Image (pixel at x, y)

$I(x, y)$ – Continuous Data

$F(x, y)$ – Bandlimiting Filter

Analytic Antialiasing

- Analytic antialiasing requires solving visibility to give a continuous 2D image
  - Visible polygons tessellate image plane
  - Arbitrarily complex shapes
  - Efficient methods exist for evaluating the integral from 2D tessellation (Duff 1989, McCool 1995)

Reduce Dimensionality - Point Sampling

- Point sampling reduces the dimension of the visibility calculation to 0D in image plane
- Pixel's value is a weighted sum of values at sample points in the image plane
- Most widespread and well studied method for antialiasing geometry

Reduce Dimensionality - 1D Sampling

- Another option is to reduce the dimension by 1, and sample along 1D elements
- Prior Art:
  - Max 1990 - Antialiasing Scan-Line Data
  - Guenter & Tumblin 1996 - Quadrature Prefiltering for High Quality Antialiasing
  - Tanaka & Takahashi 1990 - Cross Scanline Algorithm

Prior Art - Max

- Sample along scanlines
- Analytic antialiasing in scanline direction, supersampling in other direction
- Extended in same paper to use edge slopes to better approximate 2D image before 2D filtering

Prior Art - Guenter & Tumblin

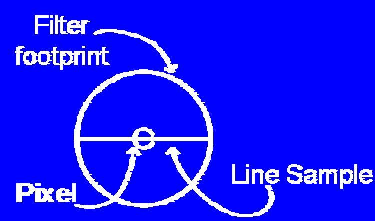
- Quadrature prefiltering - accurate numerical approximation of antialiasing integral
- Assumes existing 2D visibility solution
  - Phrased as an efficient computation of the antialiasing integral, not as a sampling method
  - As in Max 1990, unidirectional sampling

Prior Art - Tanaka & Takahashi

- Uses horizontal scanlines and vertical "sub-scanlines" to find 2D visibility solution
- Filters 2D image
- Again, not really phrased as a sampling method

Line Sampling

- Small 1D samples - “line samples”
  - Centered at pixel, spanning filter footprint
- Multiple line samples and sampling directions per pixel



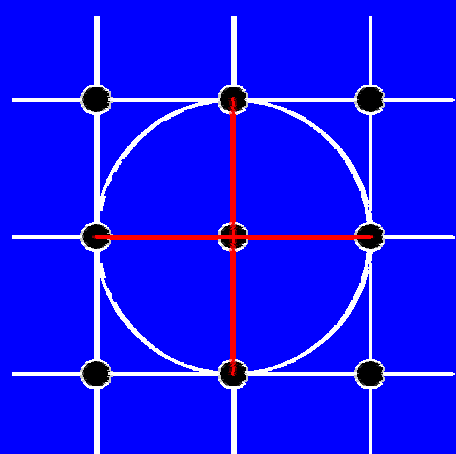
Line Sampling (continued)

- 1D filtering only - Cheaper/Faster
  - 1D tables
  - Edge slopes ignored in filtering
- Blending of samples based on image features
  - Does use edge slopes...
  - …but separates blending from filtering, keeping both simple

Theory and Practice

- Theory:
  - Arbitrary number of line samples per pixel in arbitrary directions
- Practice:
  - 2 line samples per pixel, horizontal and vertical
  - Line samples are subsegments of horizontal and vertical “scanlines”

Practice (continued)



Line Sampling Algorithm

- Determine visible segments along line samples at each pixel
- Keep sum of weights at each pixel (from edge crossings)
- Apply 1D table-based filter
- Blend values from vertical and horizontal line samples

Determining Visible Segments

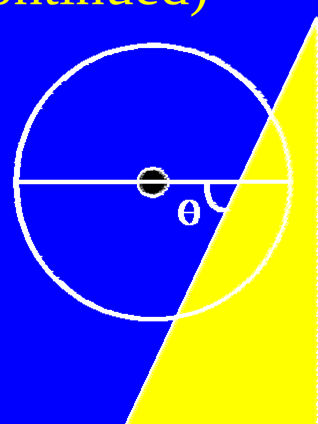
- Horizontal and vertical line samples are subsegments of scanlines
  - Use scanline methods for visibility
- Less efficient methods for arbitrary sampling directions (see paper)

Weights from Edge Crossings

- Why edge crossings?
  - A line sample's accuracy depends on its orientation relative to image features
  - If a line sample intersects an edge, its filtering accuracy is highest when perpendicular, lowest when parallel

Weights (continued)

- Use $\sin^2 \theta$ as weight
  - Normalized: weights for horizontal and vertical line samples sum to one

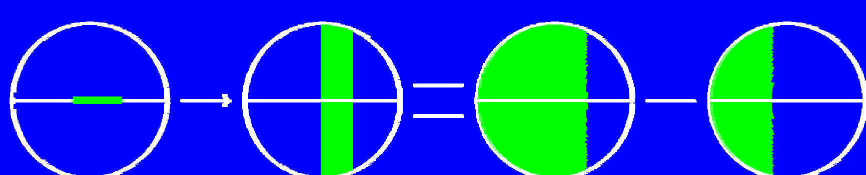


Weights (continued)

- Sum weights at each pixel (post-visibility)
- Intersecting triangles - use cross product of normals to find slope of created edge
- Edge weights should be adjusted by color change across the edge

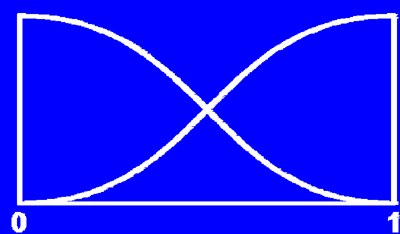
1D Table-based Filter

- Stretch 1D to 2D, then filter
  - Perpendicular, not according to edge slope
- Combine stretch and filter
  - Use summed filter table

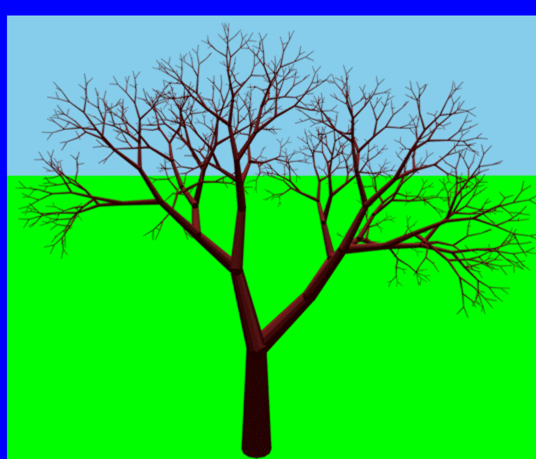


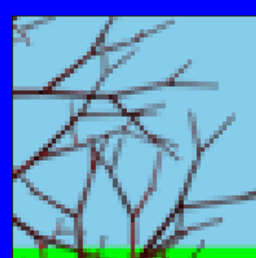
Blend Values from Line Samples

- Sample weights are $\sum \sin^2\theta$
- Good results using step function for blending, but discontinuity can cause aliasing
- Use cubic blending (Hermite)

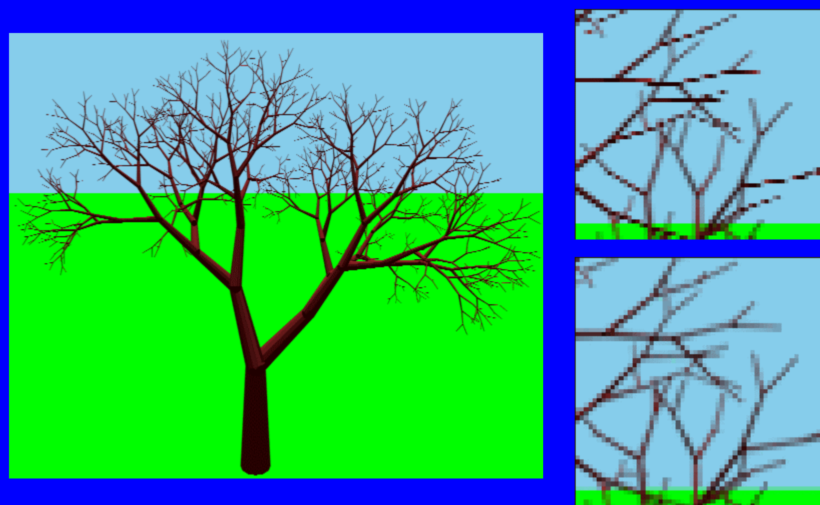


Results

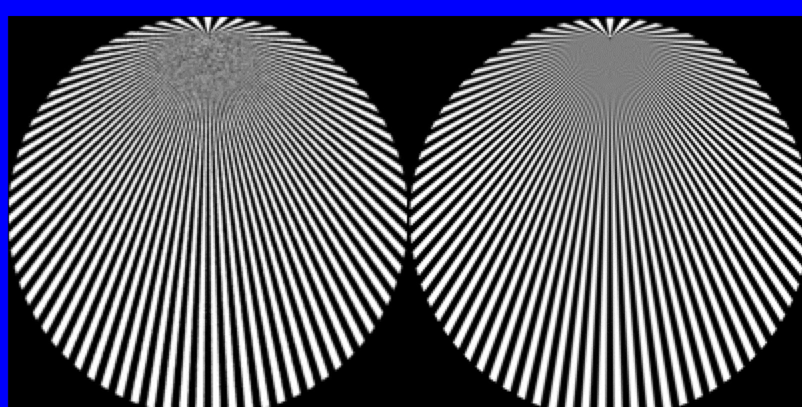




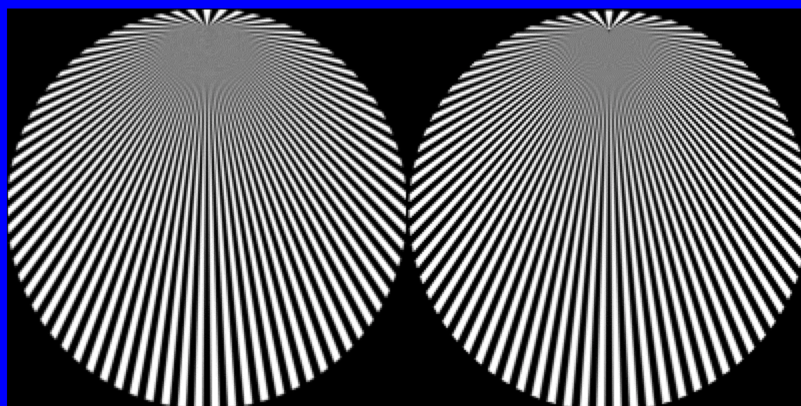
Horizontal Filtering Only



Comparison - Radial Triangles 16x Supersampling

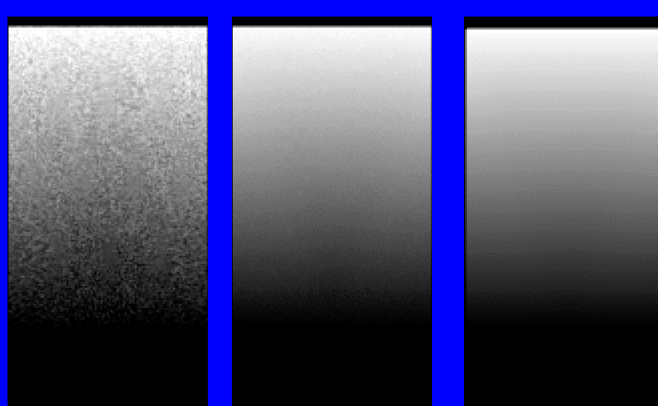


Comparison - Radial Triangles
256x Supersampling



Comparison - Triangle Comb

16x 256x Line Sampling



Comparison - Animation

Benefits of Line Sampling

- High quality
- Near analytic for substantially vertical or horizontal edges
- Low variance near lone edges
- Efficient
  - 2 scanline passes + 1D filtering + blending

Failure Cases

- Areas with high frequency content in two directions
  - Small features can be missed
  - Corners
- Non-trivial to extend to curved surfaces

Conclusions and Future Work

- Line Sampling can provide near-analytic quality antialiasing at substantially lower cost
- Future work:
  - Implement in realtime scanline renderer
  - Integration with texture mapping
  - Stochastic line sampling
  - Extension to motion blur
  - Reduced memory requirements

Acknowledgements

- Nelson Max
- Rob Kotredes
- Richard Coffey
- David Hart
- Peter-Pike Sloan
- MERL: Hanspeter Pfister, Larry Seiler, Joe Marks