LG Electronics

AXAA: Adaptive approXimate Anti-Aliasing

Jae-Ho Nah, Sunho Ki, Yeongkyu Lim, Jinhong Park, and Chulho Shin

SIGGRAPH2016
Render the Possibilities

Research Highlights

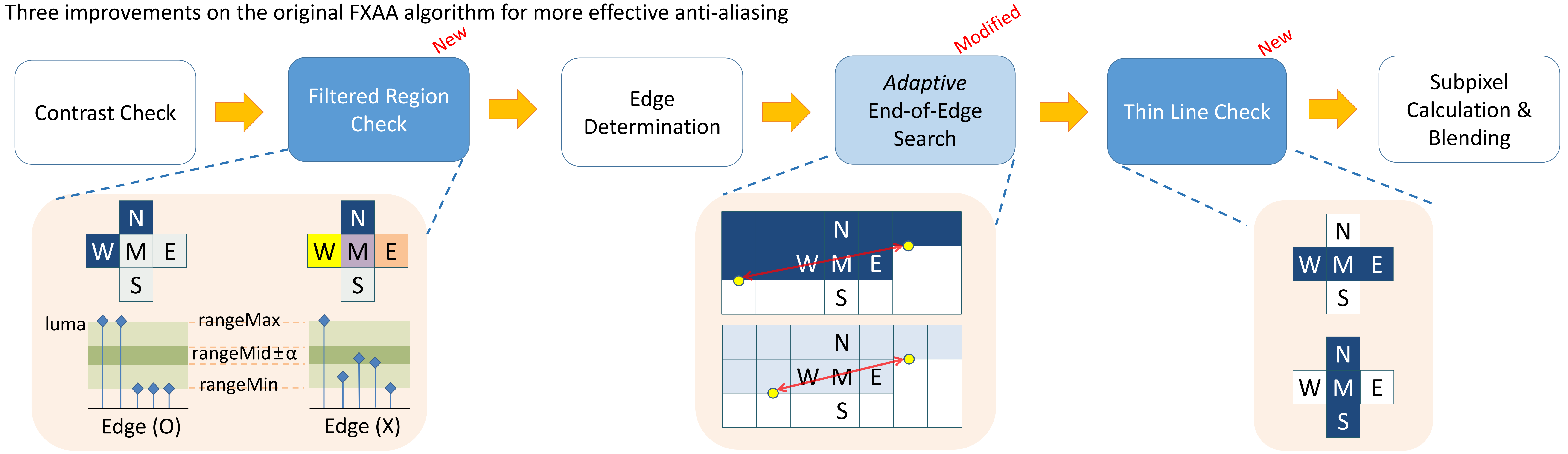
- A simple & effective method to solve excessive blurring of FXAA
- Similar performance to FXAA
- Comparable image quality to SMAA/CMAA

Post-Processing Anti-Aliasing

- Image-based AA that is independent of rendering pipelines
- Attractive alternative to multi-sample anti-aliasing (MSAA) due to its low overhead and suitability for deferred shading
- Morphological anti-aliasing (MLAA) [1]
  - Discontinuities detection, edge classification, and blending
  - Multi-pass approach
- Enhanced subpixel morphological anti-aliasing (SMAA) [2]
  - Improving the quality of MLAA in various ways
  - Accurate distance searches, local contrast adaption, extended patterns and geometric features detection, and combinations with temporal super-sampling and MSAA
- Conservative morphological anti-aliasing (CMAA) [3]
  - Separation between locally dominant edges and long shapes
  - Minimized shape distortion and smoothly anti-aliased edges
- Fast approximate anti-aliasing (FXAA) [4]
  - Single-pass approach: only a single shader kernel is required
  - Edge detection by checking luma contrast
  - The fastest method, but often excessively blurs pixels

Adaptive approXimate Anti-Aliasing (AXAA)

- Three improvements on the original FXAA algorithm for more effective anti-aliasing



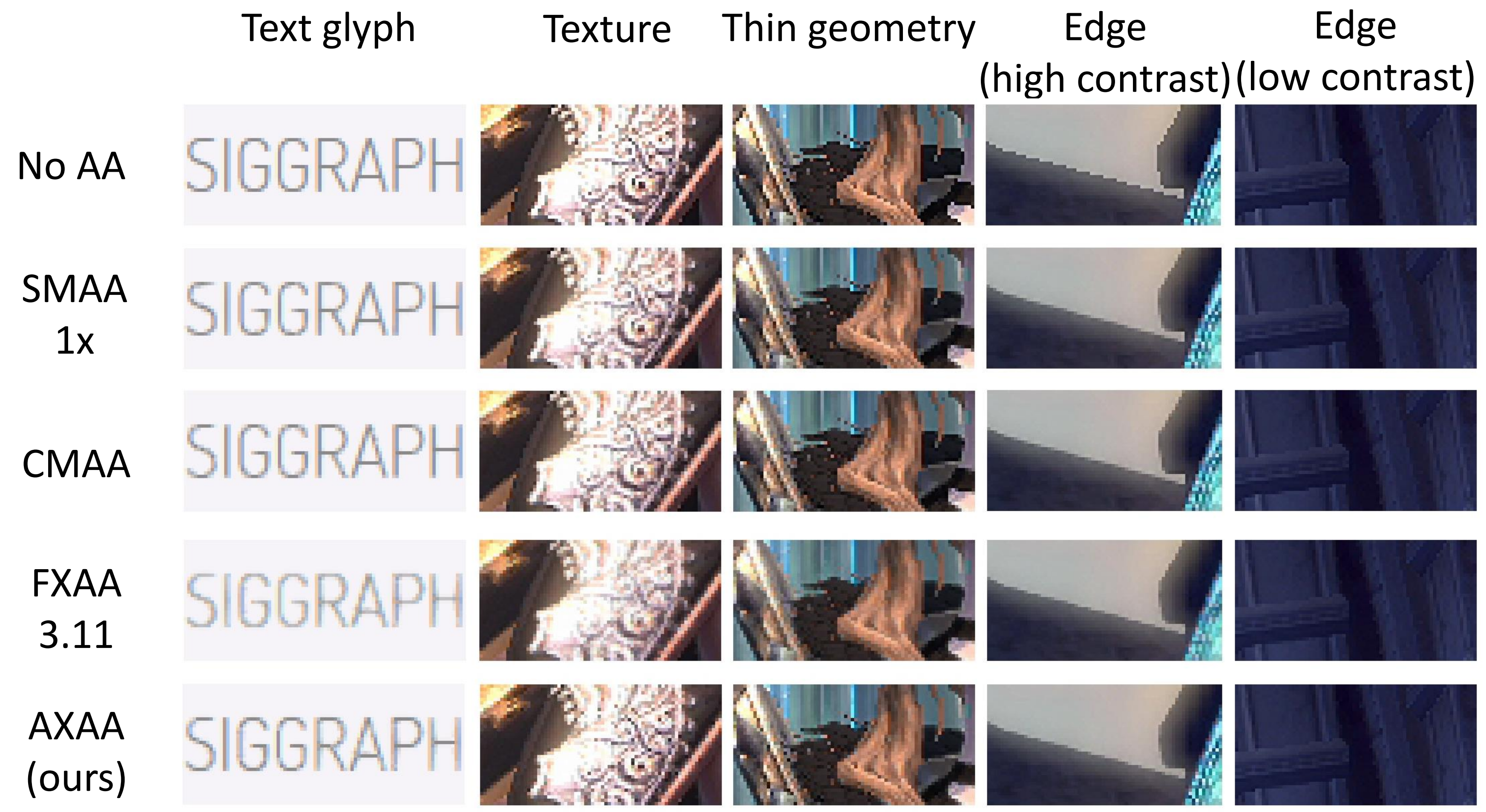
- Early exit on *already* filtered regions
- Image filtering blends neighbor pixels, so we can detect the regions by
  1) Calculating additional median luma values (*rangeMid*)
  2) Checking whether the luma values of M or its neighbors are within *rangeMid*$\pm\alpha$ (current $\alpha = 10\%$)
- Increases visibility of textures and true-type fonts

- *Adaptively* sets the search range according to luma contrast
  1) max(dmin, dmax) ≤ 0.1 : only 1 iteration
  2) min (dmin, dmax) > 0.1 : 2+ iterations are available
  3) min (dmin, dmax) > 0.3 : 3+ iterations are available,
  where dmin and dmax are the differences between the luma of M and the min/max luma in the 3x3 region
- Improves performance with non-distinguishable quality deterioration in low-contrast regions
- Compensates for overheads of the other two modifications

- Conserves thin one-pixel-width lines
- If *gradientN* > β and *gradientS* > β, then we bypass further bilinear filtering (current $\beta = 0.3$)
- Increases visibility of small fonts and thin geometry

Results

- Quality comparison



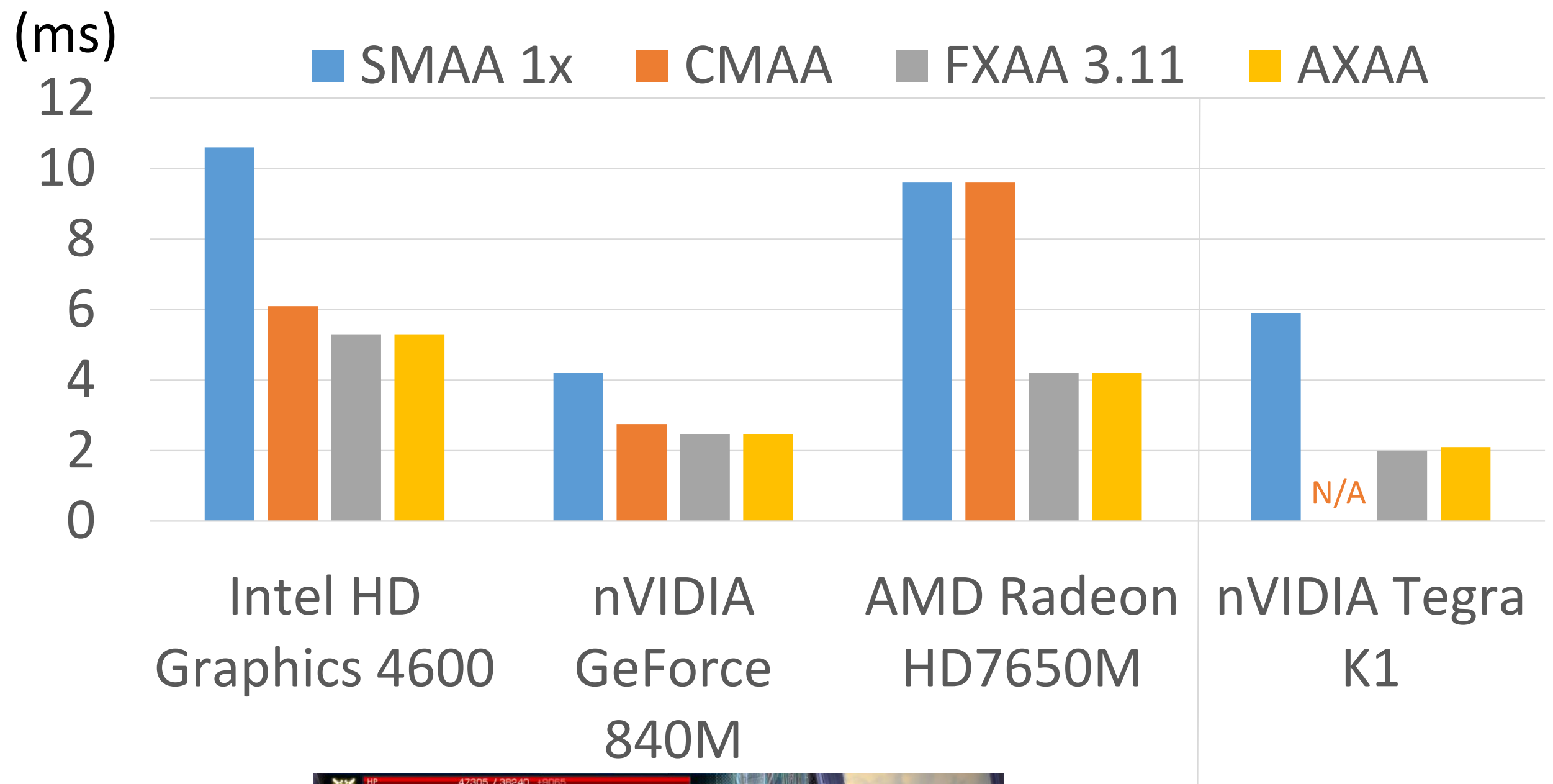
Better visibility than FXAA

Preserved details contrary to FXAA

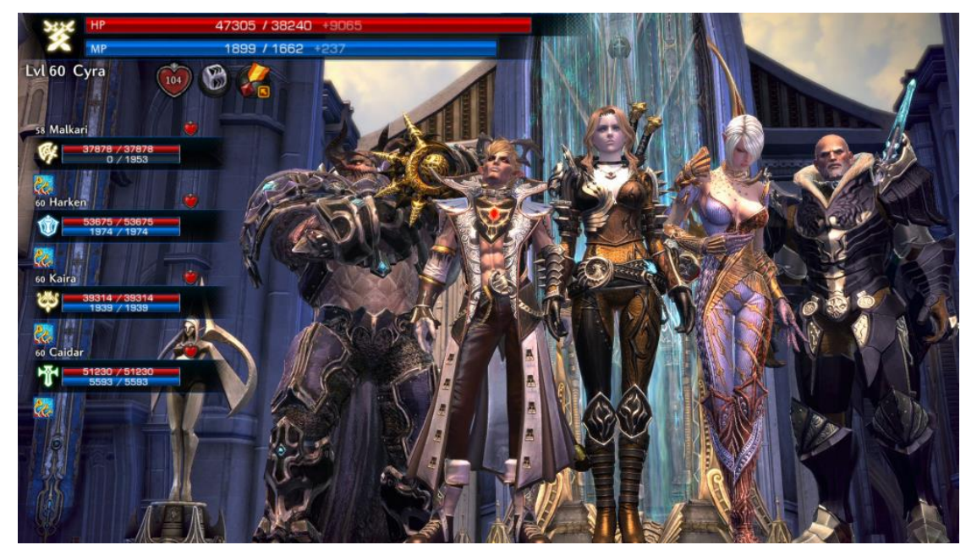
Edge smoothing similar to the others

Less search steps than FXAA, but hardly recognizable differences

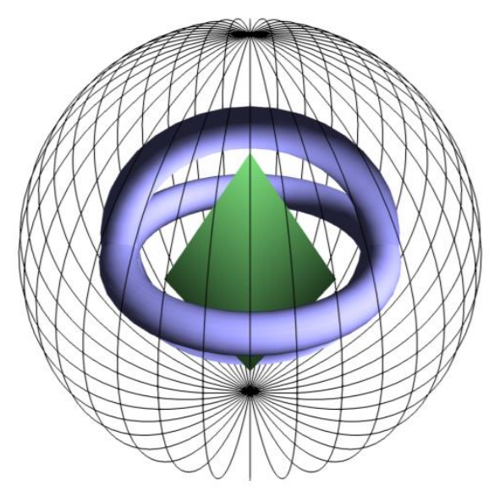
- Performance comparison @ FHD (lower is better)



Scene (API)



Tera (DirectX)
© Bluehole Studio



Gameworks FXAA (OpenGL)
© nVIDIA

- 1.7×~2.8× faster than SMAA 1x
- 1.1×~2.3× faster than CMAA
- Roughly same performance compared to FXAA

Limitations and Future Work

- The current parameters are optimal for our experimental scenes
  - AXAA may either miss jagged edges or blur non-edges in other scenes
  - We would like to continuously investigate these difficult cases
- Single sample per pixel
  - AXAA does not properly handle subpixel problems and temporal aliasing
  - A combination with spatial multi-sampling and temporal super-sampling can be a solution, as with SMAA 4x

References

[1] Reshetov, A. 2009. Morphological antialiasing. *High Performance Graphics 2009*.

[2] Jimenez, J. et al. 2012. SMAA: enhanced subpixel morphological antialiasing. *EUROGRAPHICS 2012*.

[3] Davies, L. 2014. Conservative morphological antialiasing (CMAA) - March 2014 update. *Intel Technical Report*.

[4] Lottes, T. 2009. FXAA. *NVIDIA White Paper.*

Email : nahjaeho@gmail.com