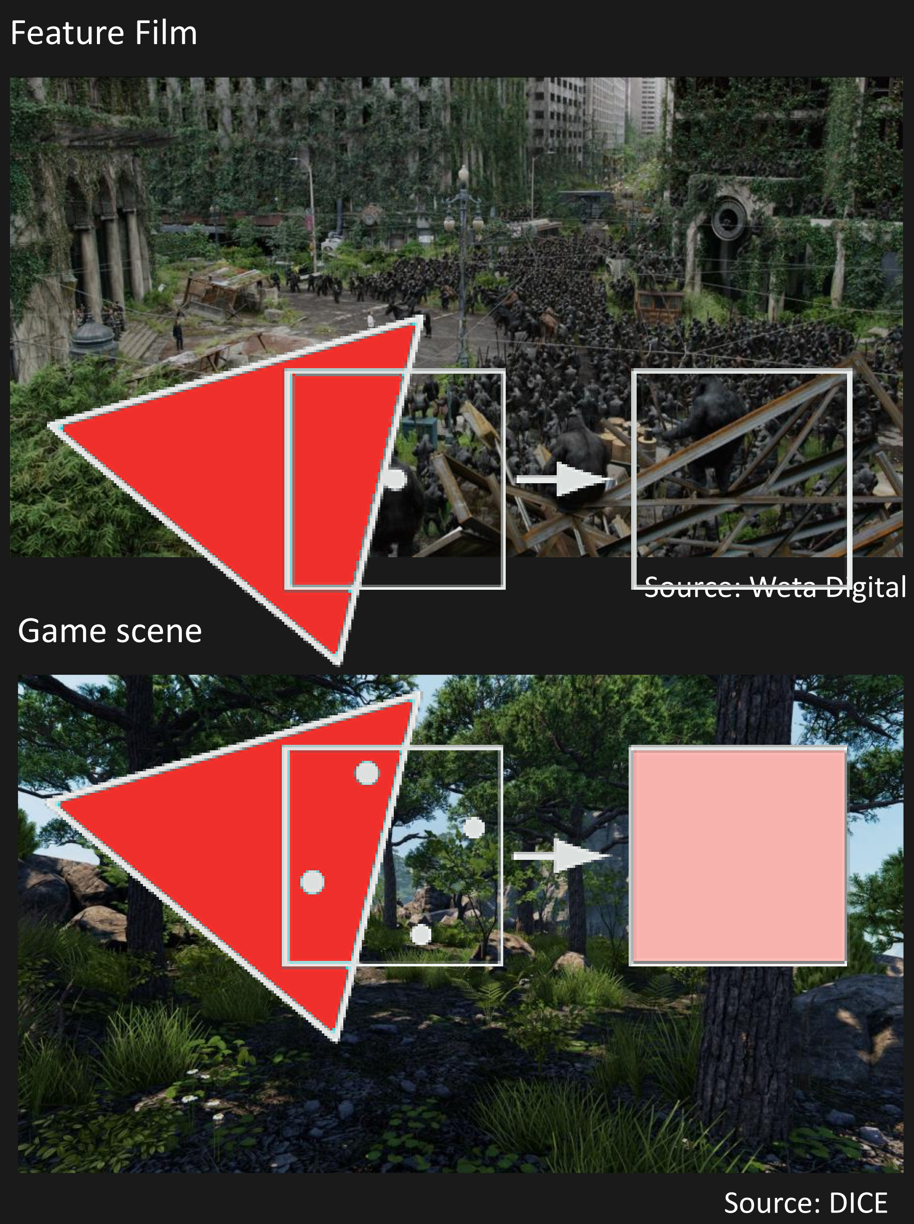
Decoupled Coverage Anti-Aliasing

Yuxiang Wang[1] Chris Wyman[2] Yong He[3] Pradeep Sen[1]

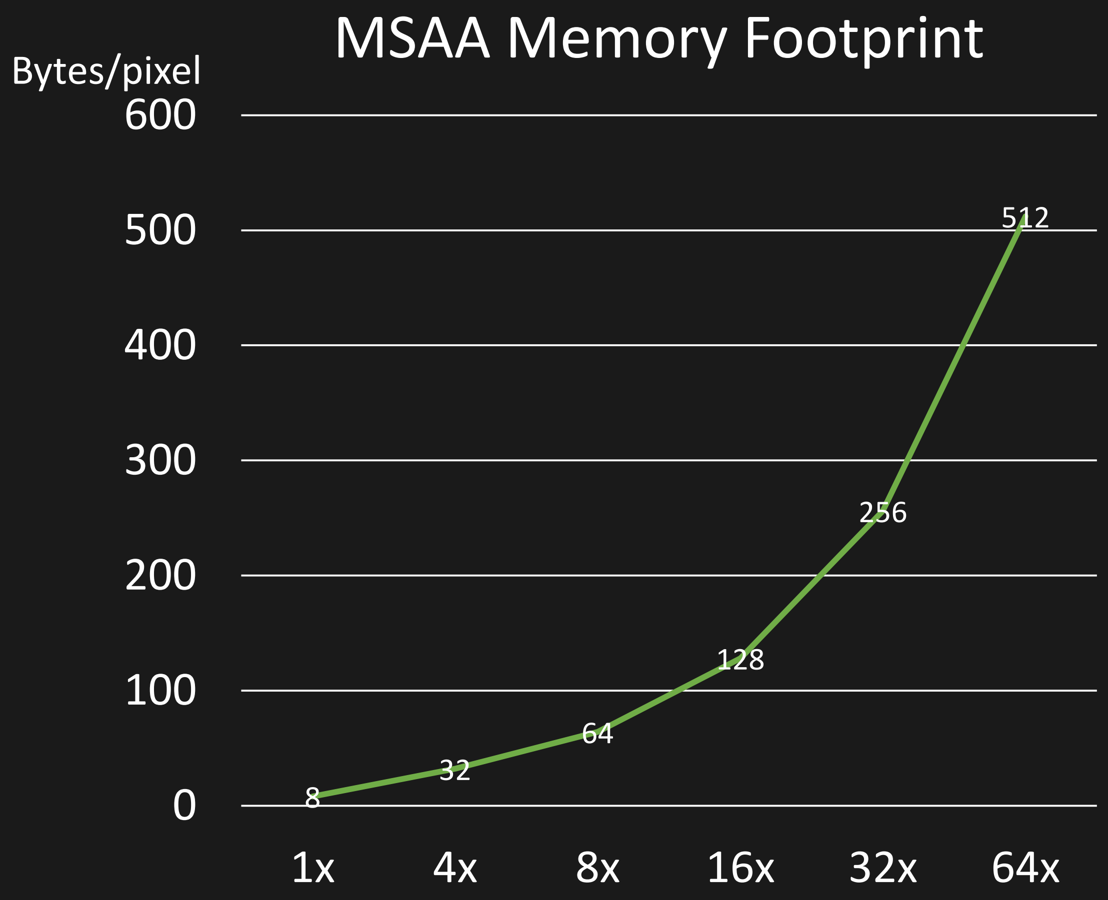
UC Santa Barbara[1] NVIDIA[2] Carnegie Mellon University[3]

Motivation

- Geometric anti-aliasing is a long standing problem
- MSAA as gold standard
  - Idea: decoupled **shading** and **visibility**
  - Reduce shading cost
- For high quality rendering, **storage** is costly



Motivation

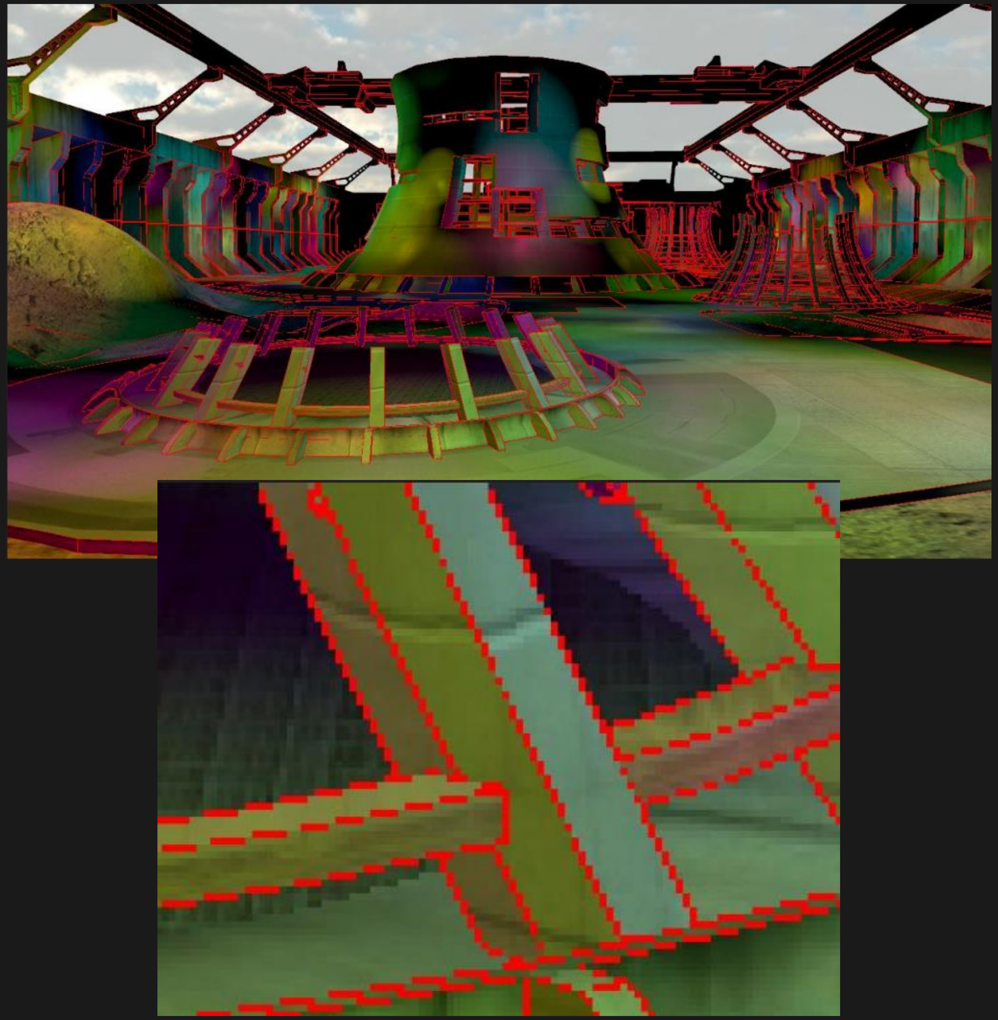


- Estimate:
  - 4 byte/sample color
  - 4 byte/sample depth
  - No compression
- Linear growth with # samples
- 64x MSAA 1080p:
  - ~1 GB for RGBA8
  - 2+ GB for G-Buffer

Related Work

- Simple/Complex [Lauritzen 2010]
  - Analyze planar features shared in G-Buffer
  - Amortize shading cost

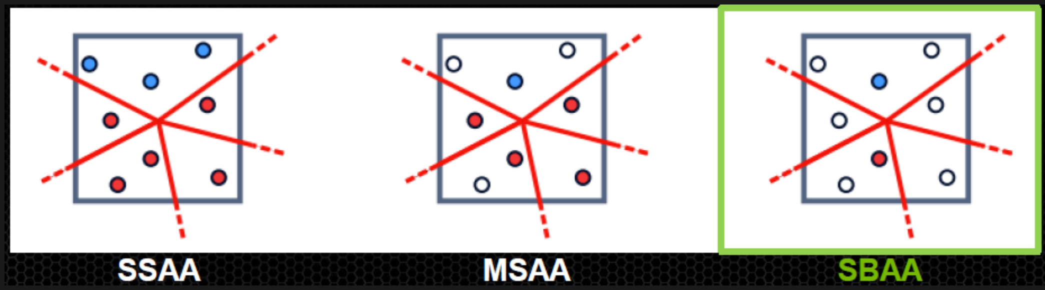
  - Large memory footprint with sizeable depth and color information



Source: [Lauritzen 2010]

Related Work

- Surface Based AA (SBAA)[Salvi & Vidimče 2012], Streaming G-Buffer [Kerzner & Salvi 2014]
  - Only store N important surfaces
- Aggregate Geometry AA (AGAA) [Crassin et al. 2015]
  - Filter & compression
- Rely on MSAA depth sampling -> large memory footprint @ high sample rates



Source: [Salvi & Vidimče 2012]

Source: [Crassin et al. 2015]

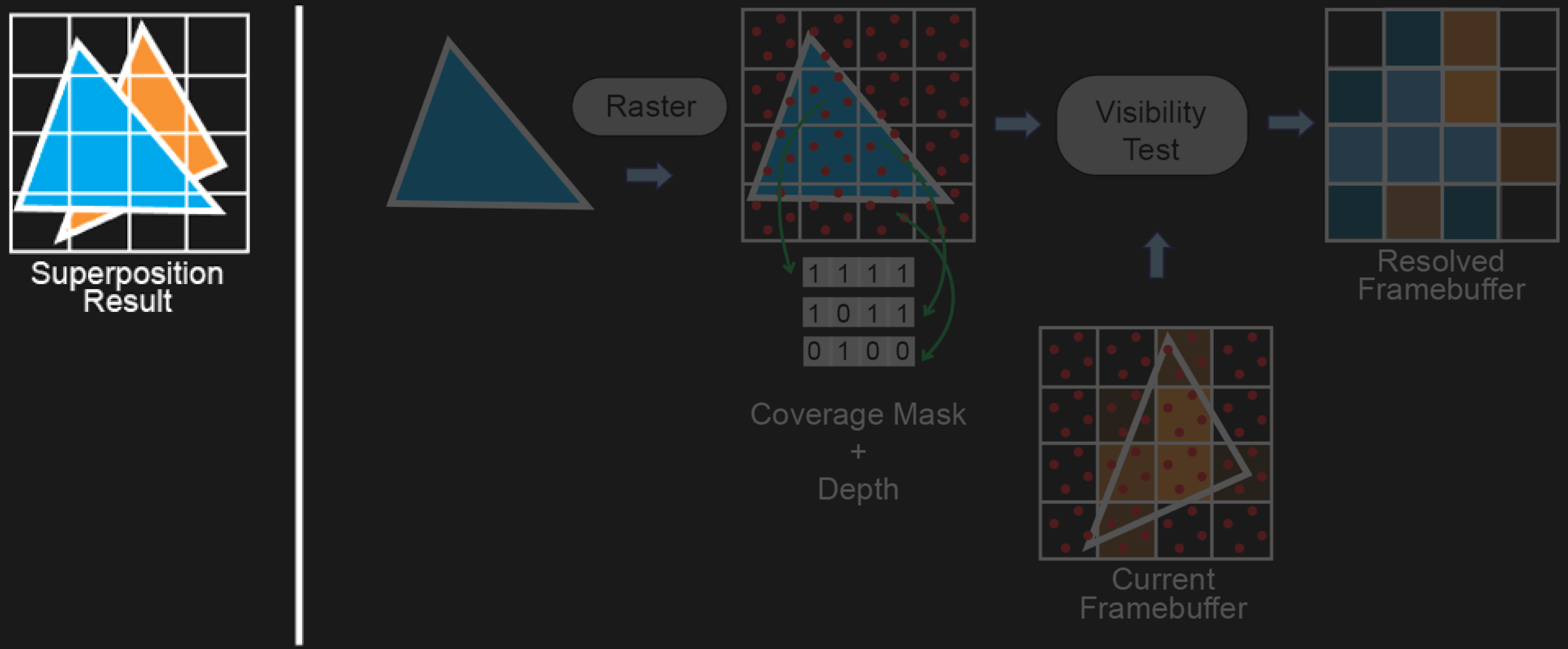
Motivation

- Observations from prior G-Buffer compression work [Salvi & Vidimče 2012]
  - 2-3 shading surfaces are enough for each pixel
- Can we use a higher fidelity coverage for compressed surfaces?
  - High fidelity coverage mask easy to get [Waller et al. 2000][Wyman et al. 2015]
  - Model contribution of each surface more precisely
- Or, in other words…
  - Can we decouple coverage from visibility?

Higher anti-aliasing quality in less storage by decoupling coverage and visibility rates

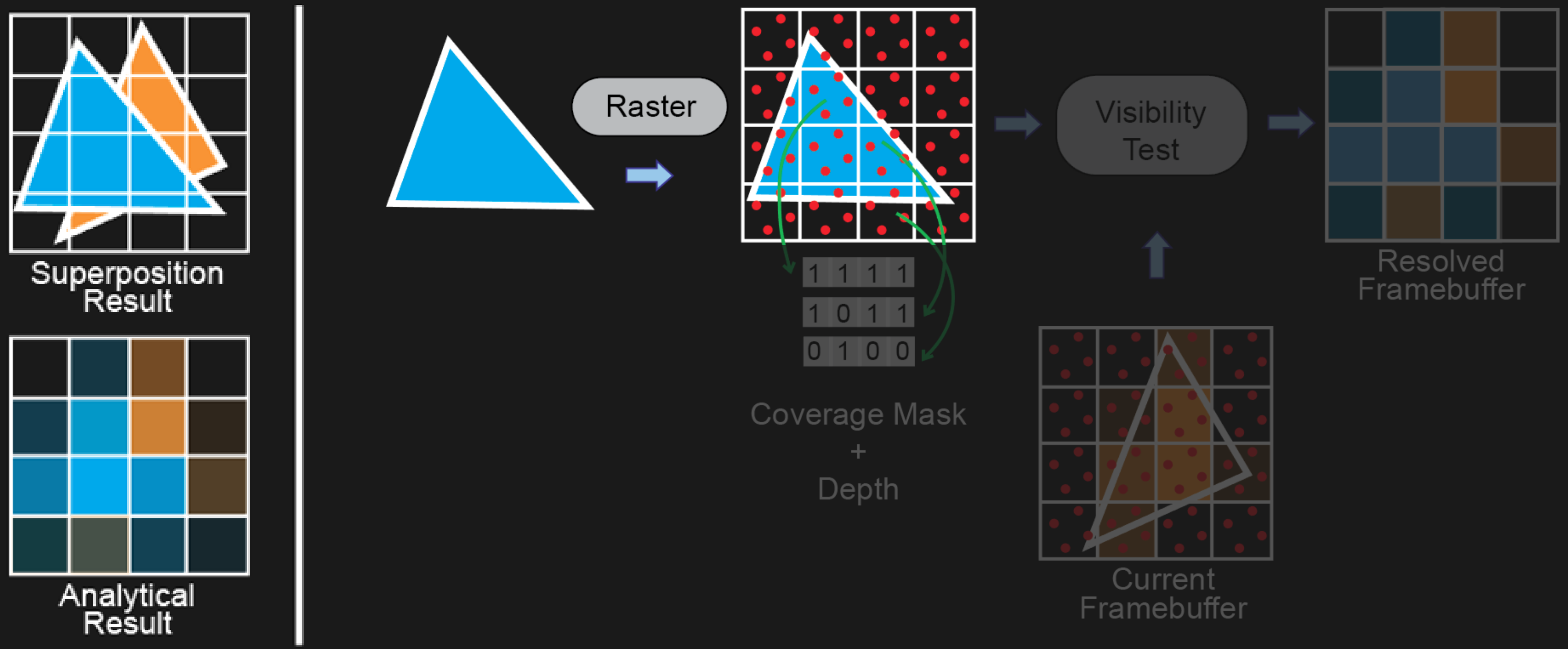
Algorithm Overview

4x MSAA



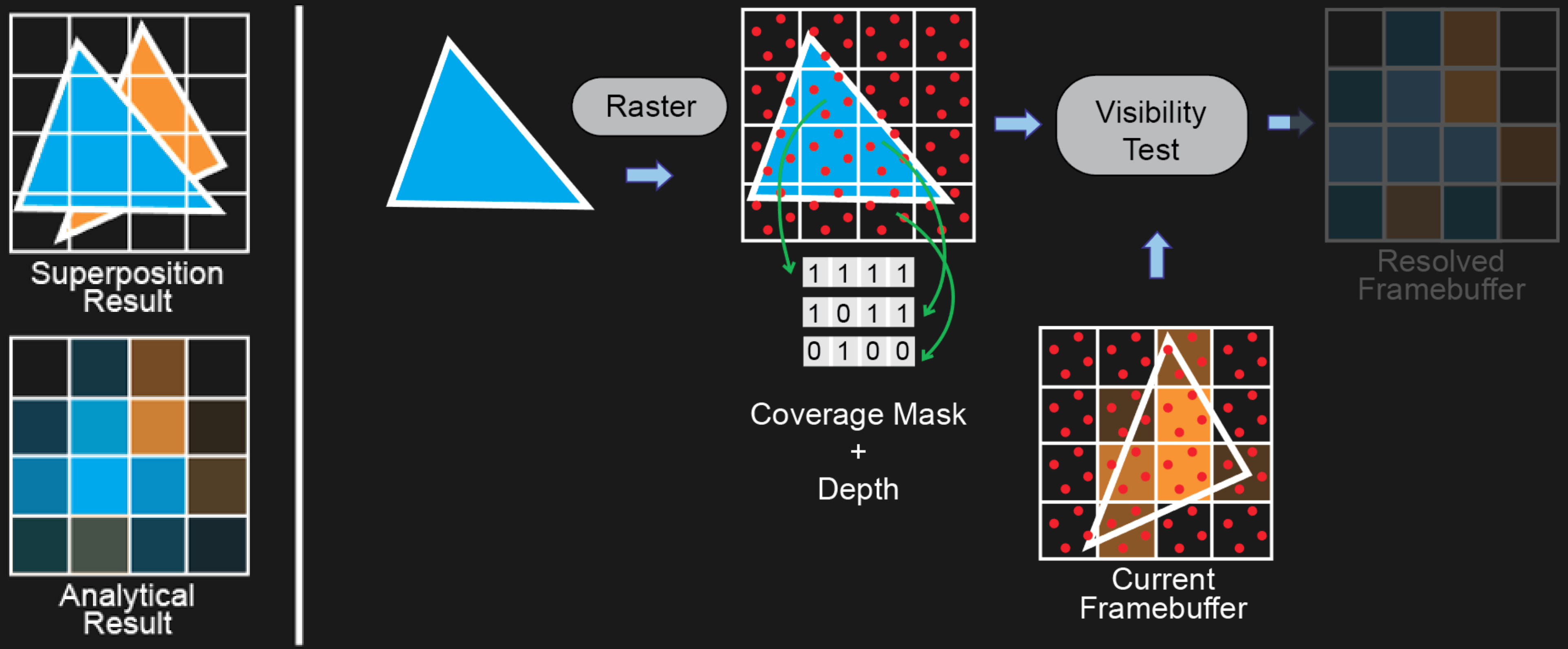
Algorithm Overview

4x MSAA



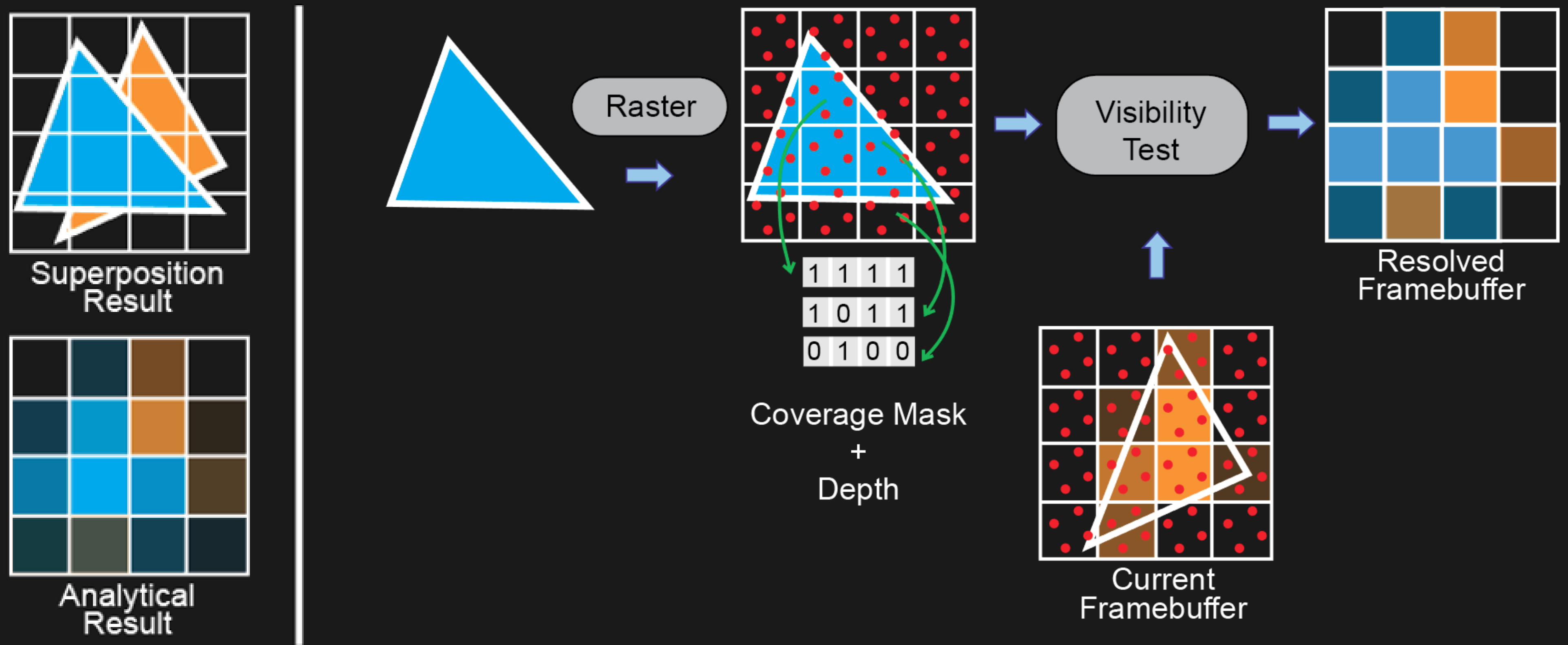
Algorithm Overview

4x MSAA



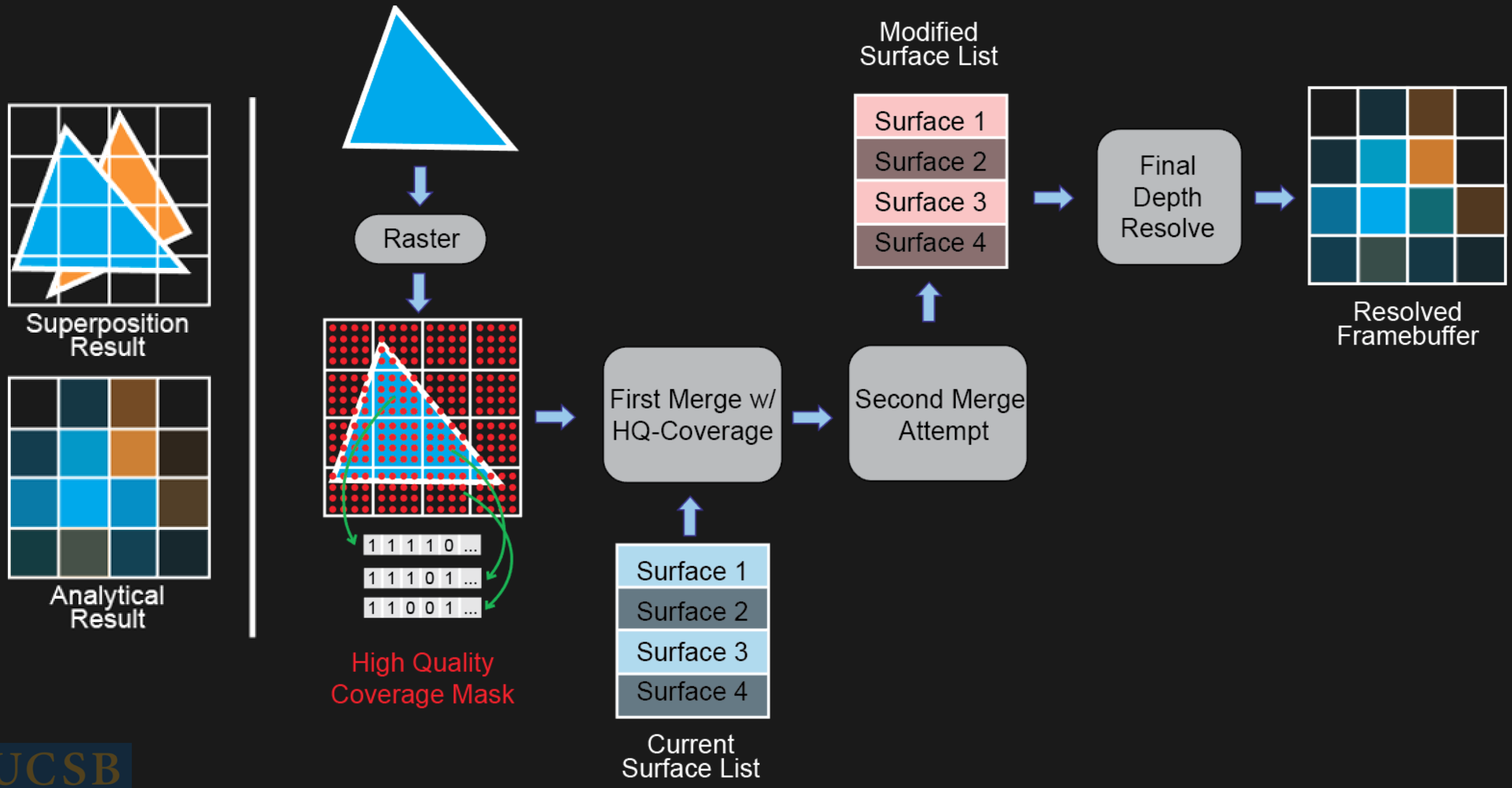
Algorithm Overview

4x MSAA



Algorithm Overview

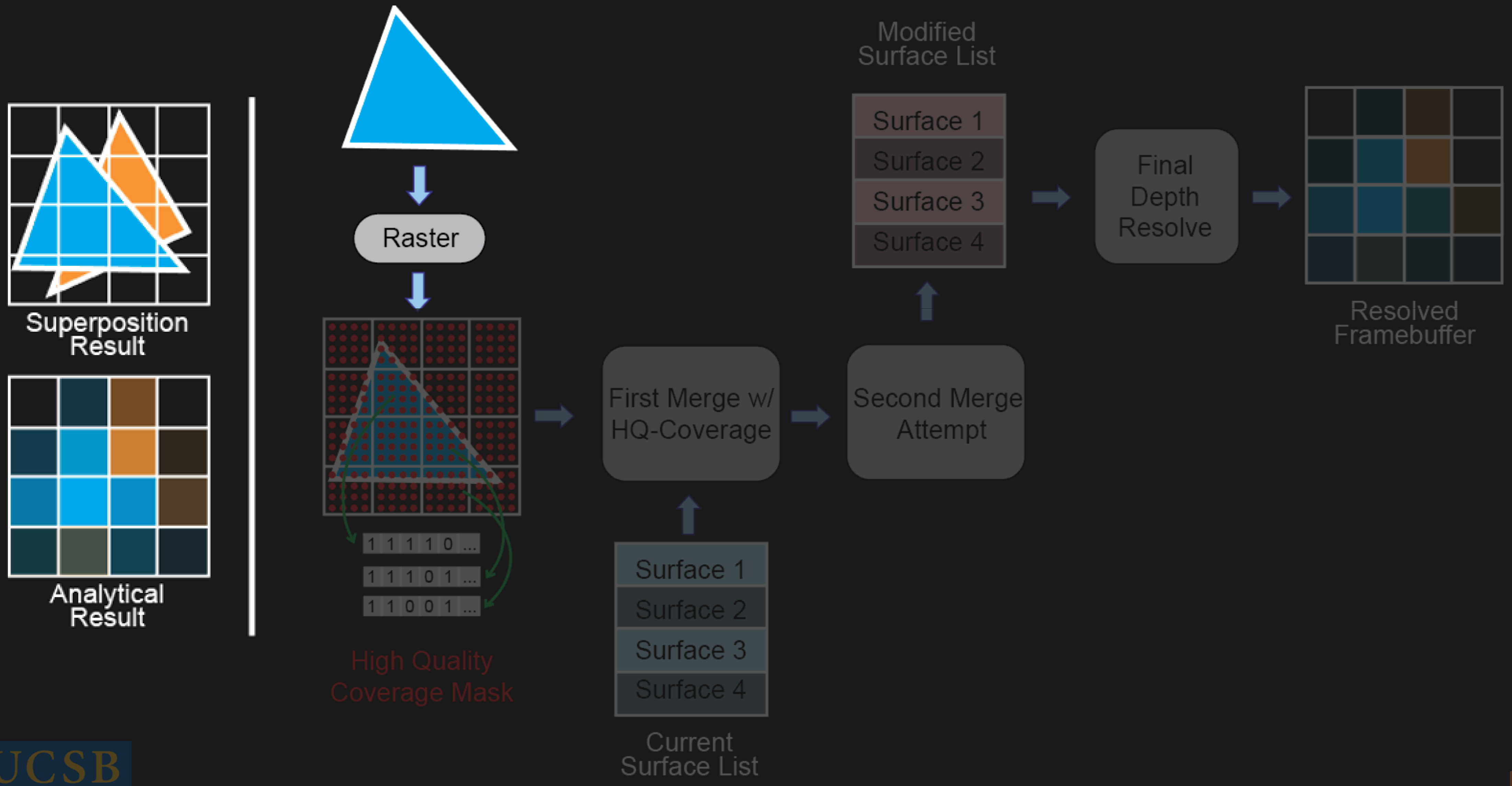
Decoupled Coverage AA



UCSB

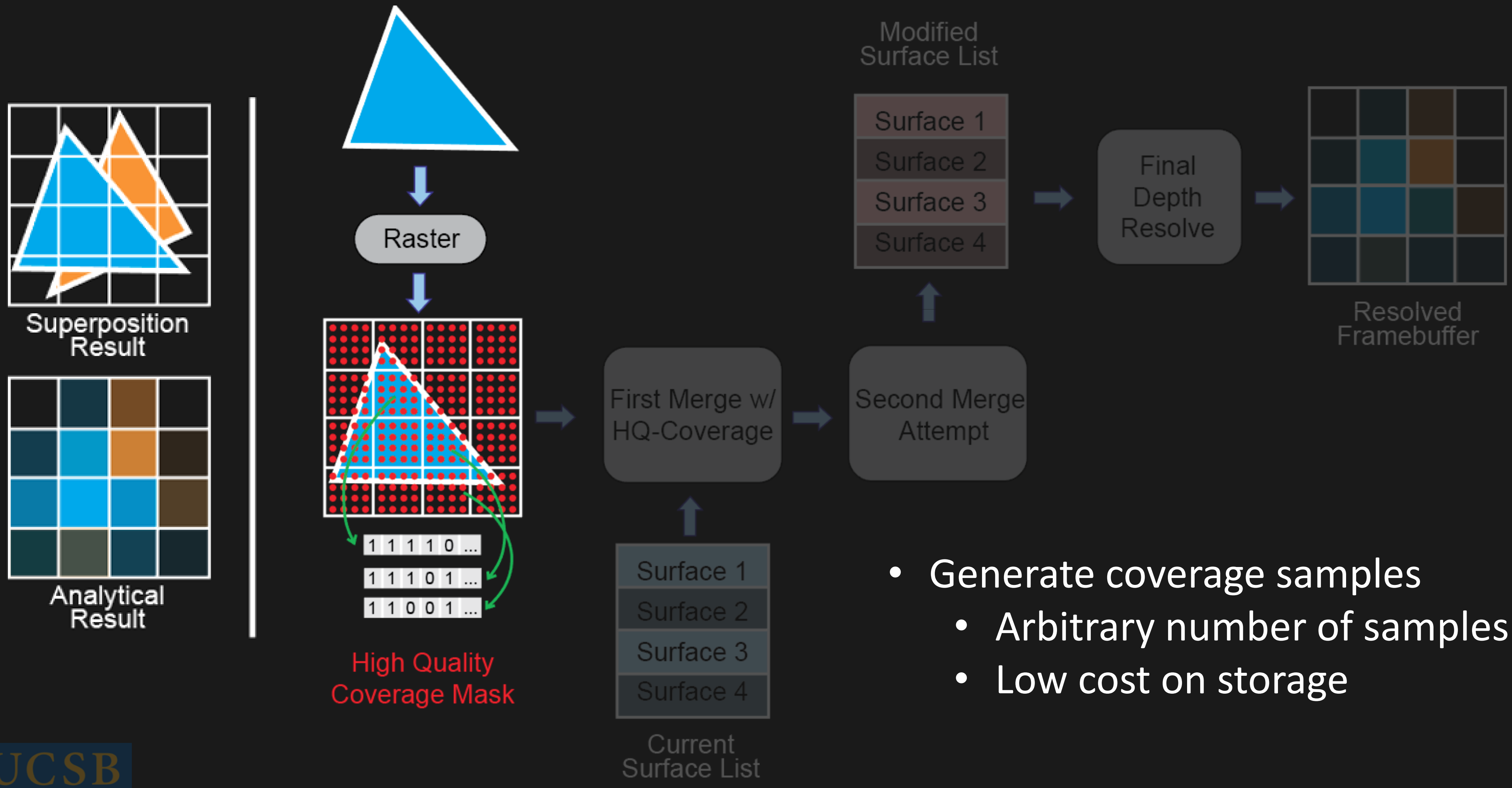
Algorithm Overview

Decoupled Coverage AA



Algorithm Overview

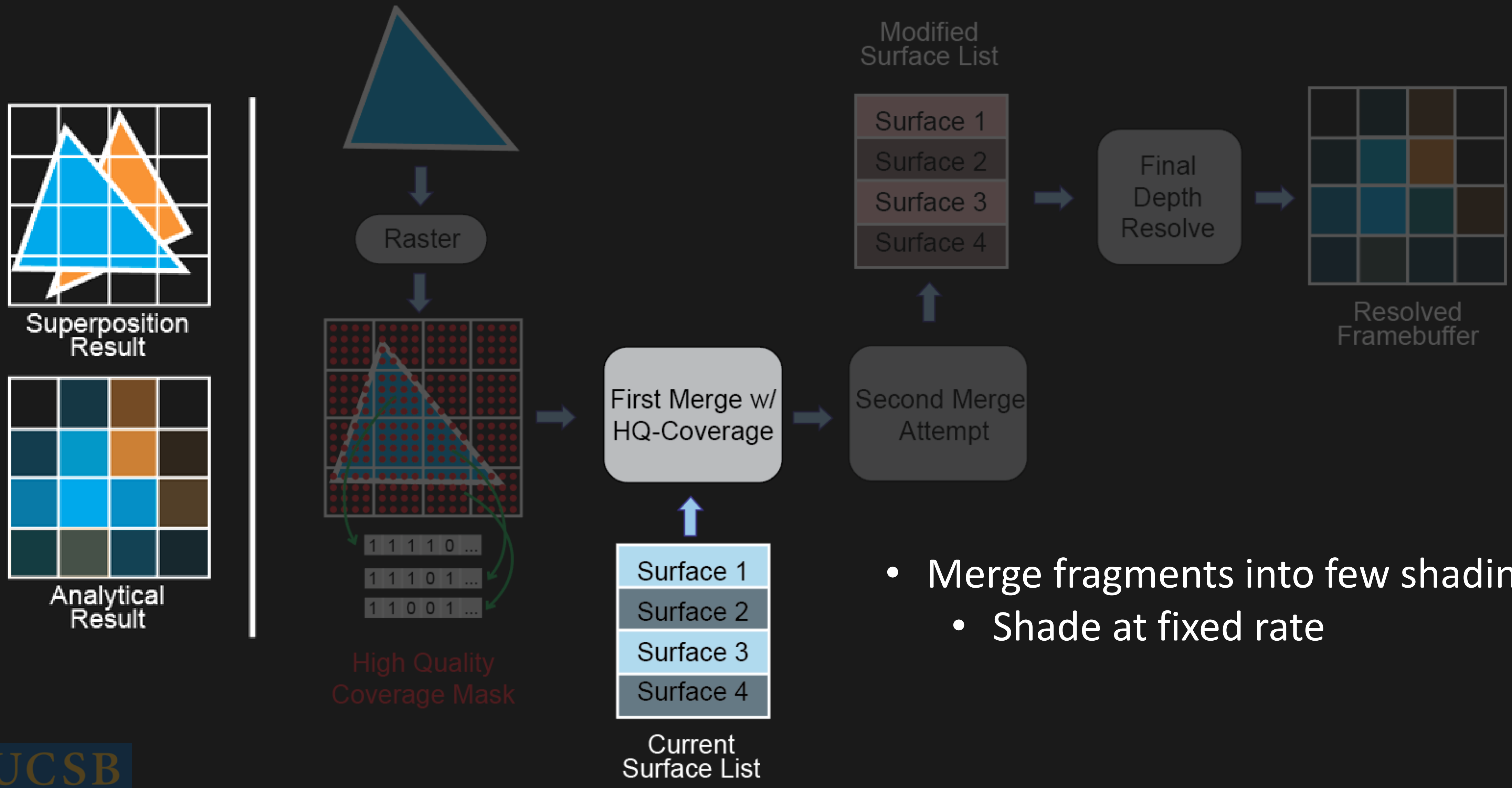
Decoupled Coverage AA



- Generate coverage samples
  - Arbitrary number of samples per pixel
  - Low cost on storage

Algorithm Overview

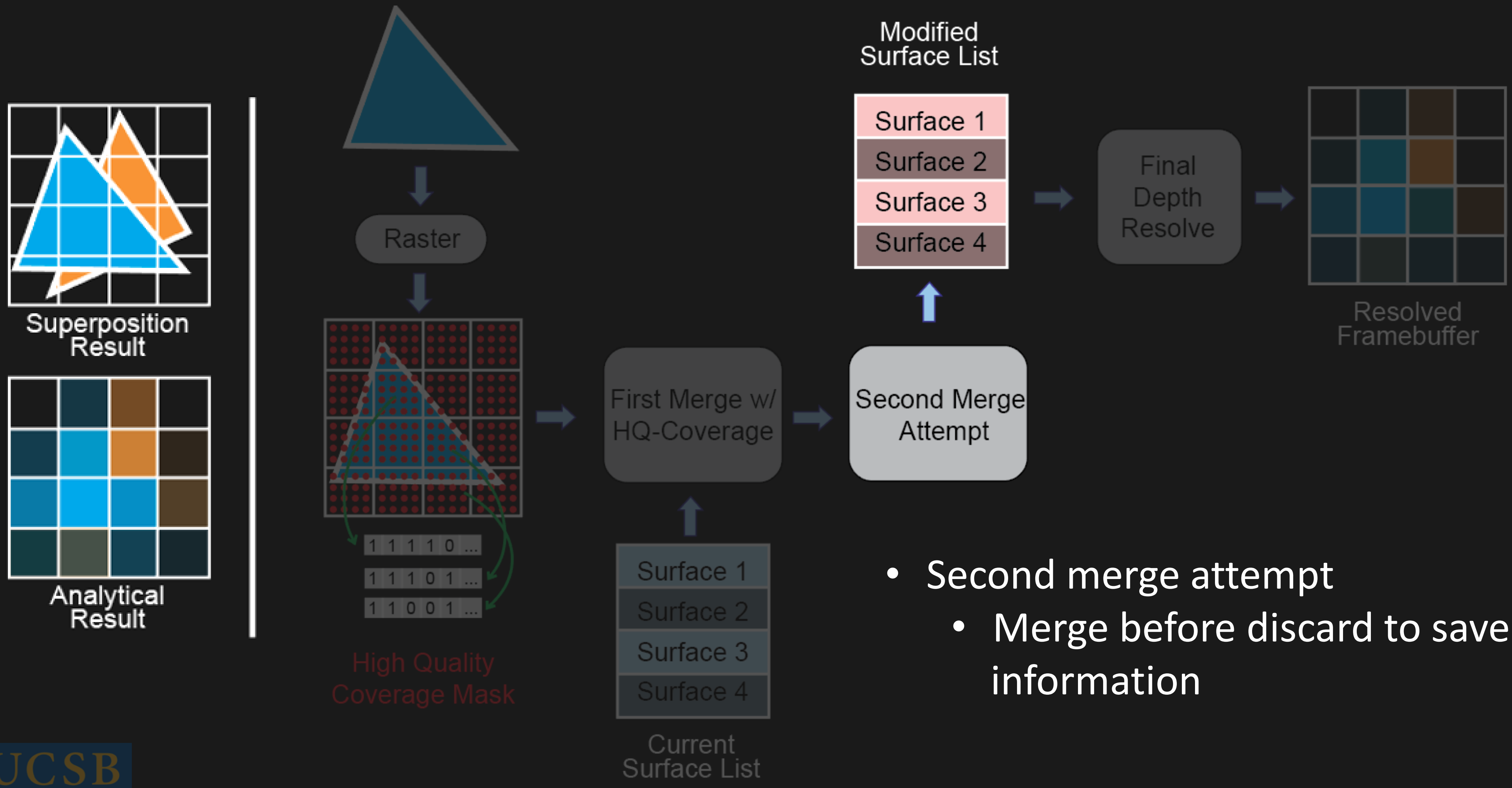
Decoupled Coverage AA



- Merge fragments into few shading surfaces
  - Shade at fixed rate

Algorithm Overview

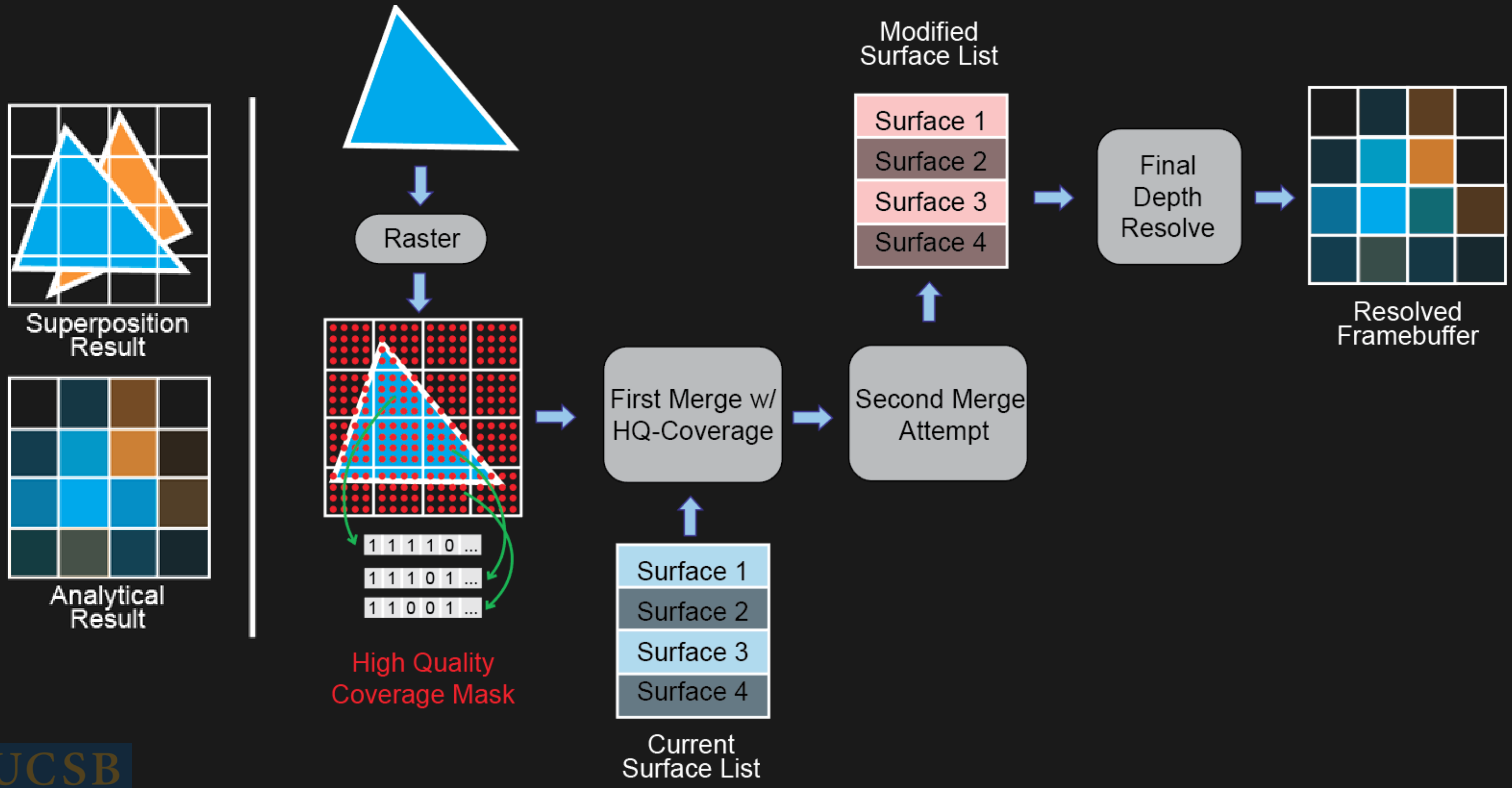
Decoupled Coverage AA



- Second merge attempt
  - Merge before discard to save more information

Algorithm Overview

Decoupled Coverage AA

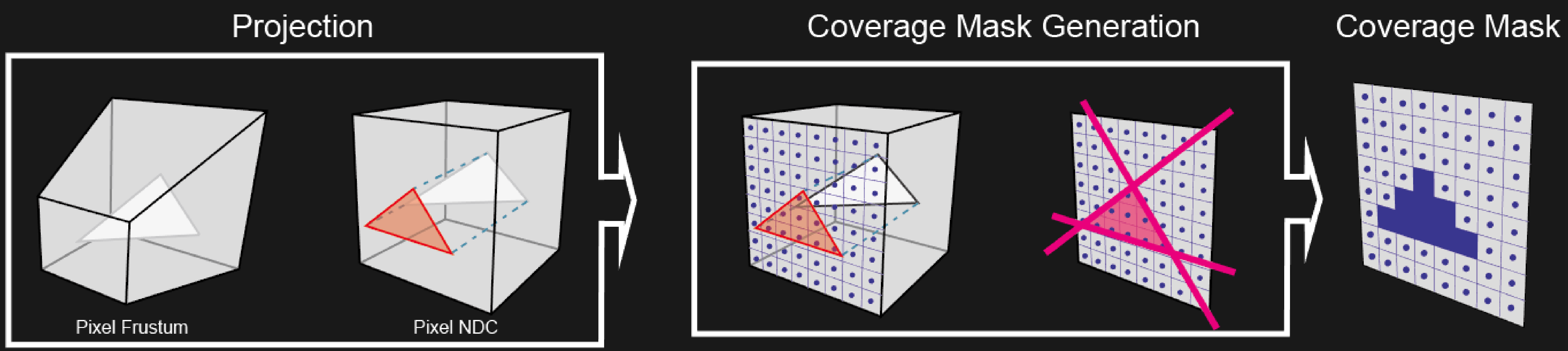


UCSB

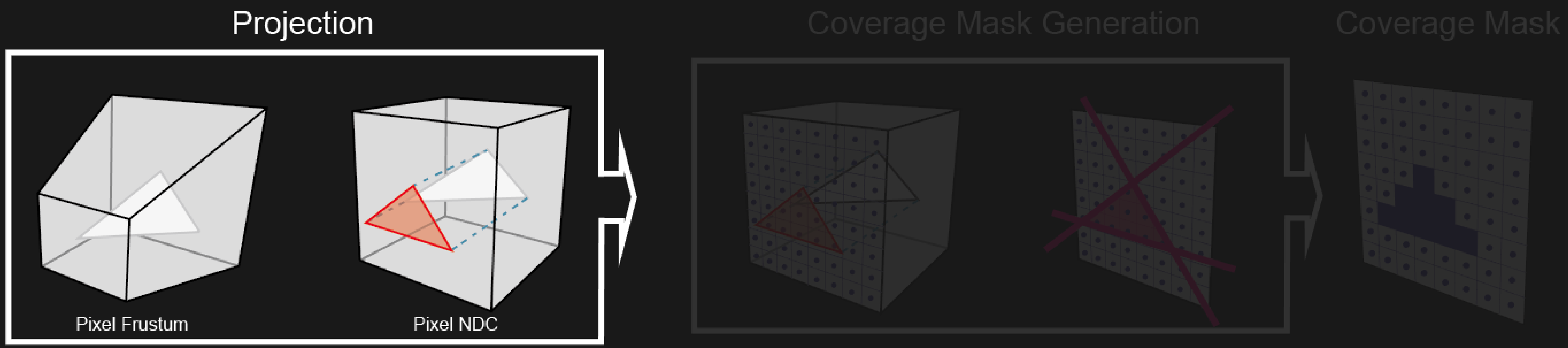
Coverage map generation

- MSAA: <= 8 sample/pixel
  - Depth + colors replicated per sample (e.g., 8 bytes/sample RGBA8 and 16-20 bytes/sample deferred)
- Coverage mask is cheaper than MSAA sample (depth+coverage)
  - 1 coverage sample -> 1 bit
  - High sampling rate supported

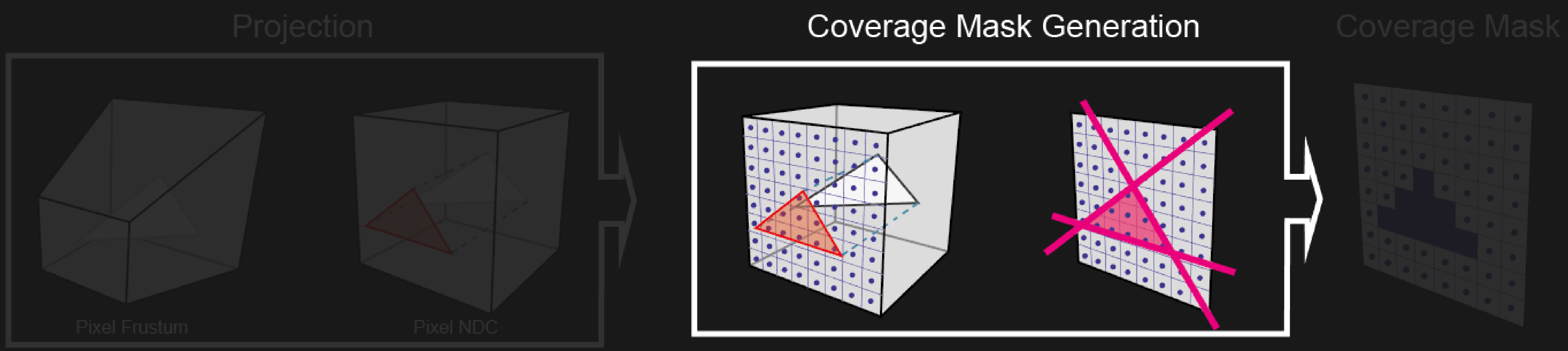
Coverage map generation



Coverage map generation

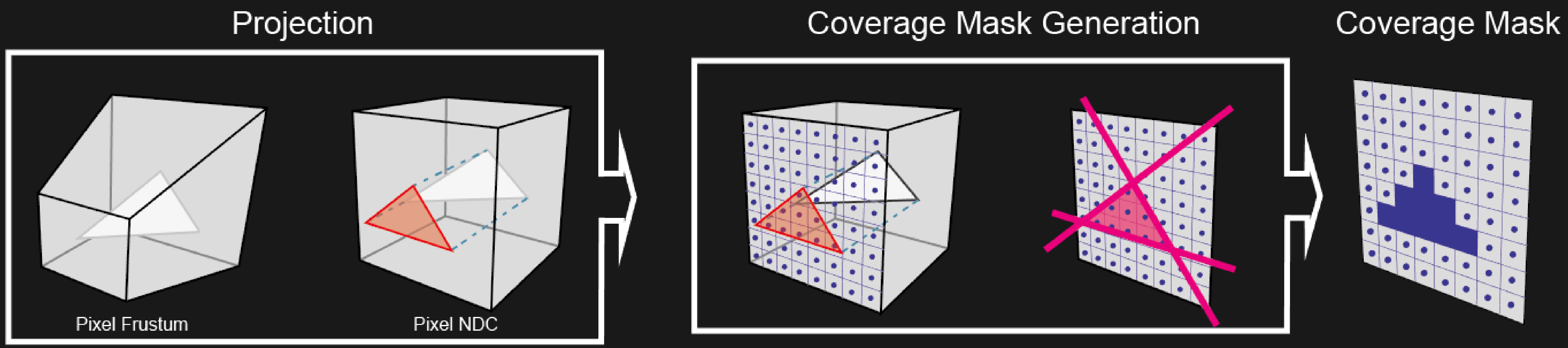


Coverage map generation



- Look-up table for per-edge coverage

Coverage map generation



- Look-up table for per-edge coverage

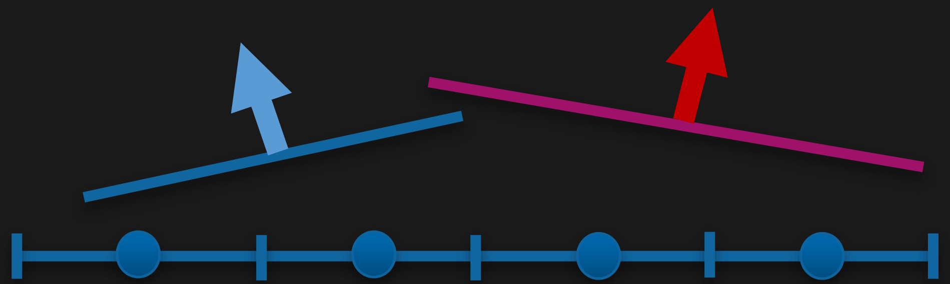
Fragment Merging

- Try to merge fragment into existing shading surfaces
  - Satisfy merge rules
  - Combine the coverage mask
  - Weighted average normal, depth, etc., based on coverage bits
  - Fail: add fragment into list

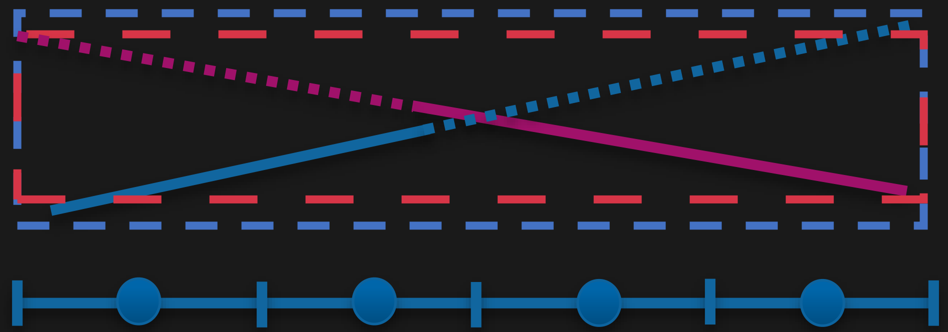
- Aggregates geometry information
- Without losing high fidelity coverage information

Merge heuristics

- Merge rules
  - Aligned normal



  - Overlapping depth intervals

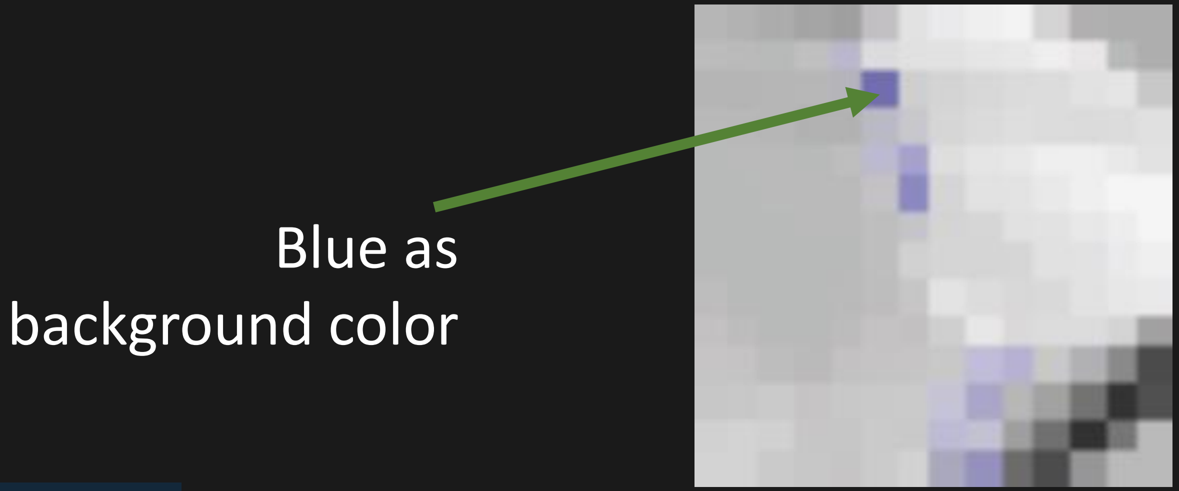


Fragment Merging

- Keep 4 surfaces at most
  - 2-3 surfaces for Streaming G-Buffer [Kerzner & Salvi 2014]
  - 2 surfaces for AGAA [Crassin et al. 2015]
- Sufficient to handle sophisticated scenes
  - High fidelity coverage mask catches small geometry
  - Discard rules vs. Clustering approach

Second merge attempt

- When the surface list is full, we need to discard
  - Discard the one with smallest visible coverage
- Discard loses information…
  - Leaking to background

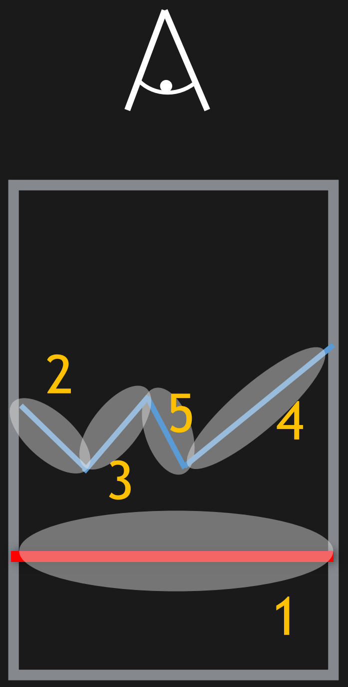


Only merge once

Reference

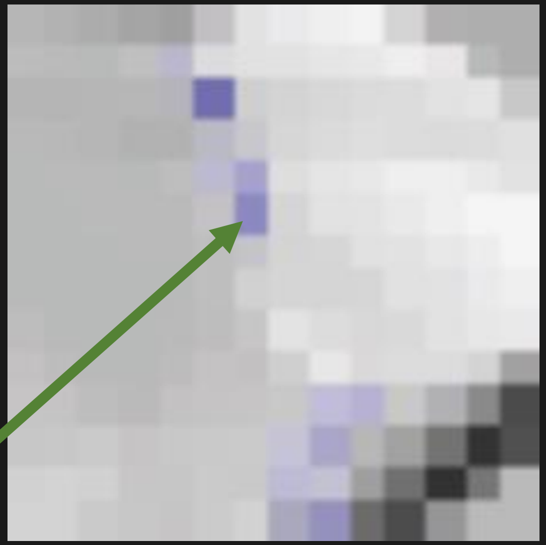
Second merge attempt

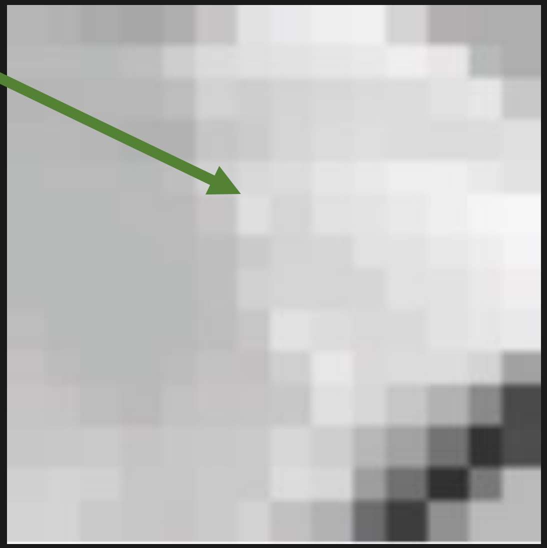
- How does the leaking happen with single merge?
- Consider this complex pixel:
  - The eye should see only the blue surfaces
  - Consider this primitive order
  - Large derivatives result in big bounding box
  - No accurate coverage determination…
  - But only have room for 4 surfaces



Second merge attempt

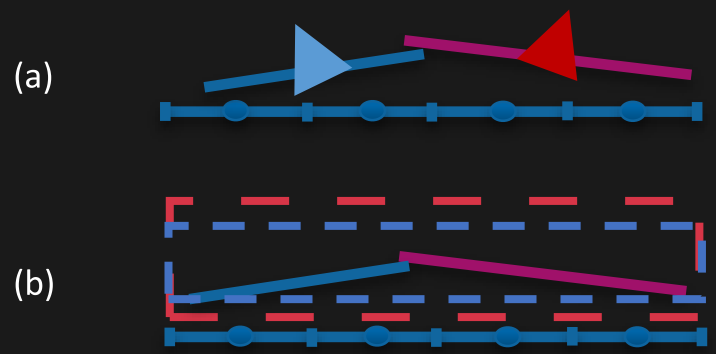
- How does the leaking happen with single merge?
  - Discarding small, nearby surfaces likely to cause leaks
- Prefer to avoid discarding important geometry
  - Prevent loss of nearby sub-pixel geometry
  - Potential cost of blurring color on small surfaces





Second merge attempt

- Give the smallest surface a second chance!
- Merge before discard:
  - Select the smallest coverage surface after first merge
    - Never try discard the front most one
  - Try to merge it with others using relaxed rule
    - Apply only overlapping depth interval rule
    - If mergeable, average all attributes as usual



Implementation

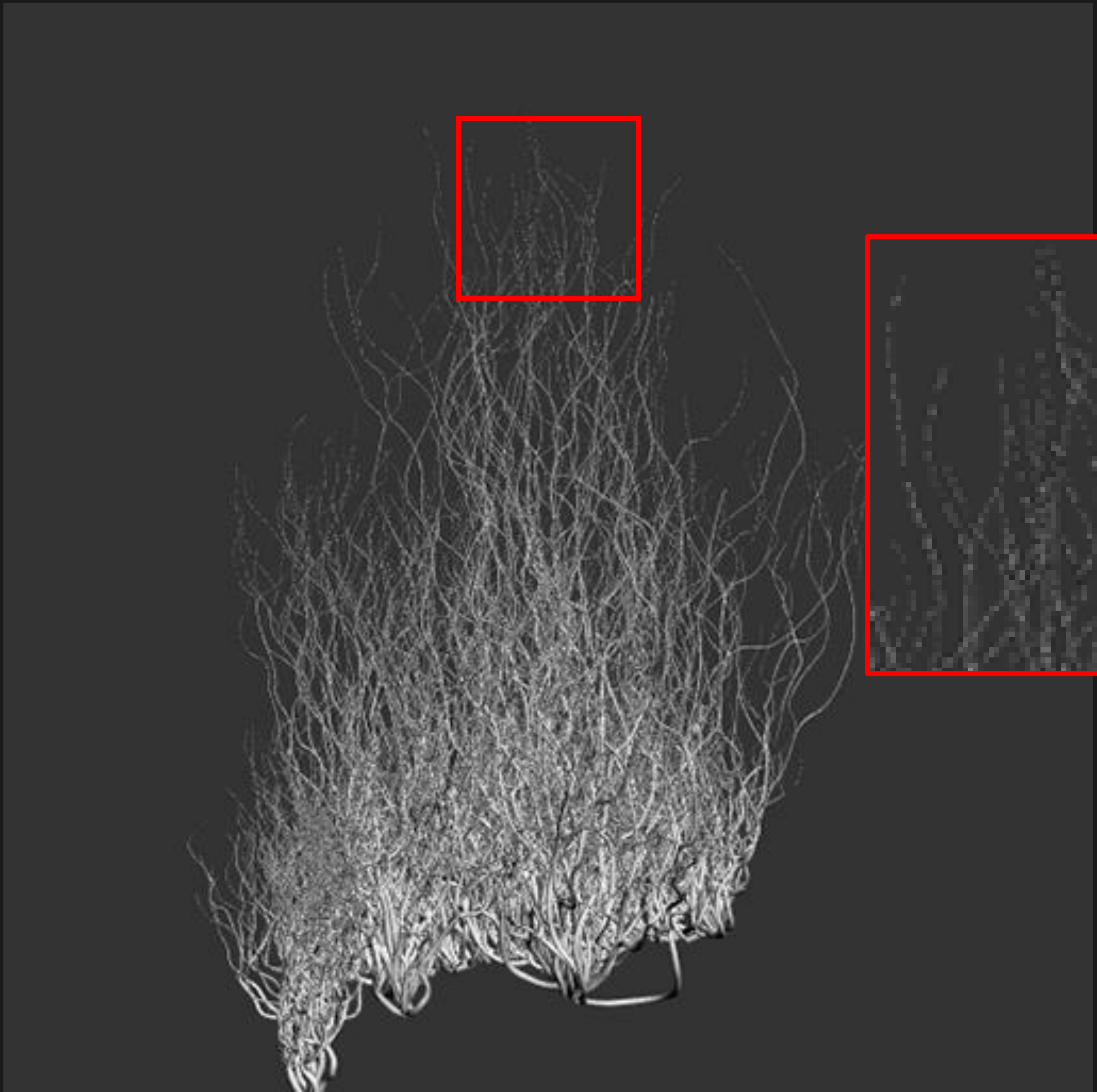
- Conservative rasterizer
  - Process partially covered fragments
- Pixel shader interlock
  - Ensure primitive ordering
  - Fragment shader lock
  - Resolve discard & temporal artifact
- Z-prepass

Results

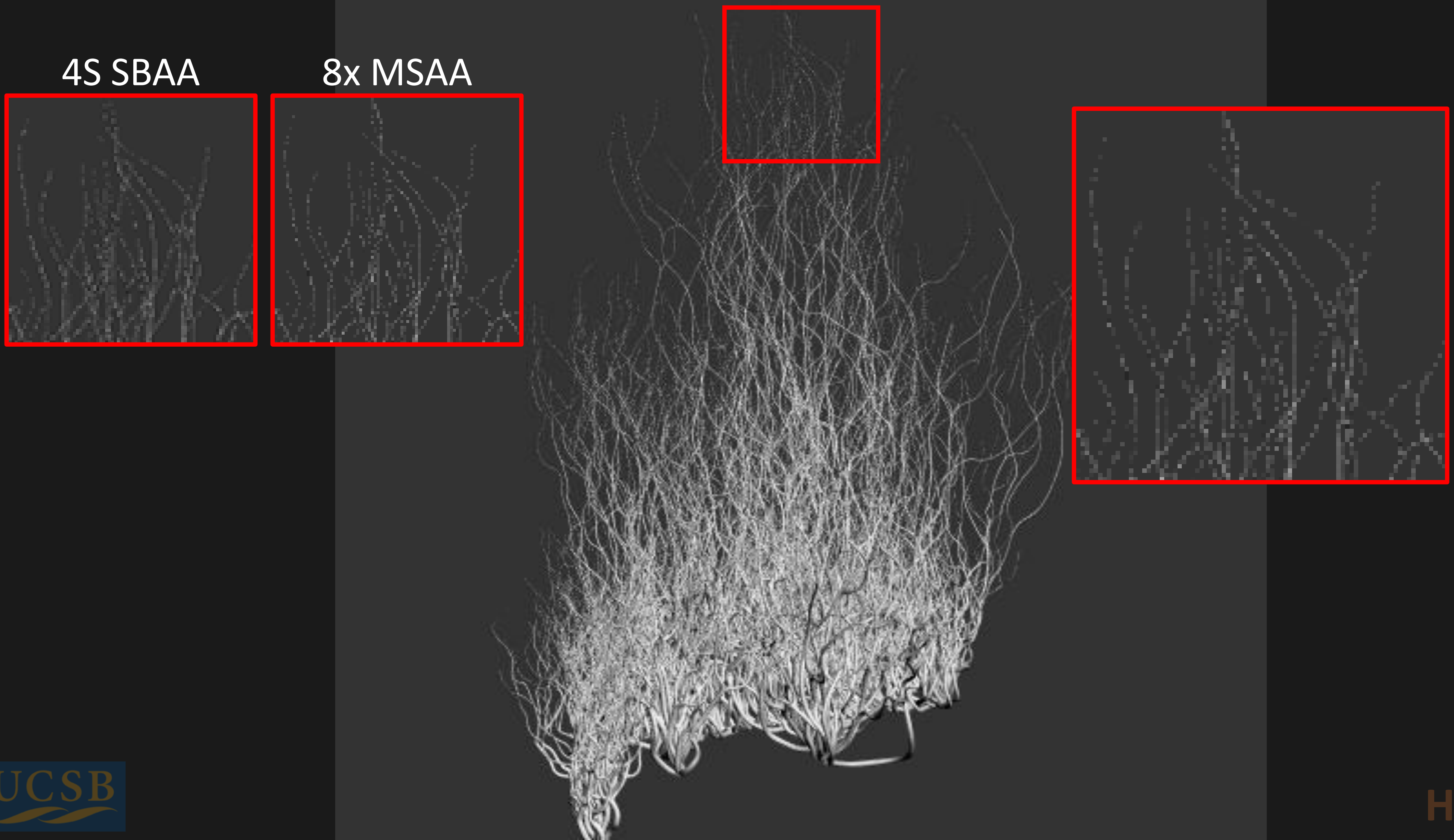
4S DCAA

4S SBAA

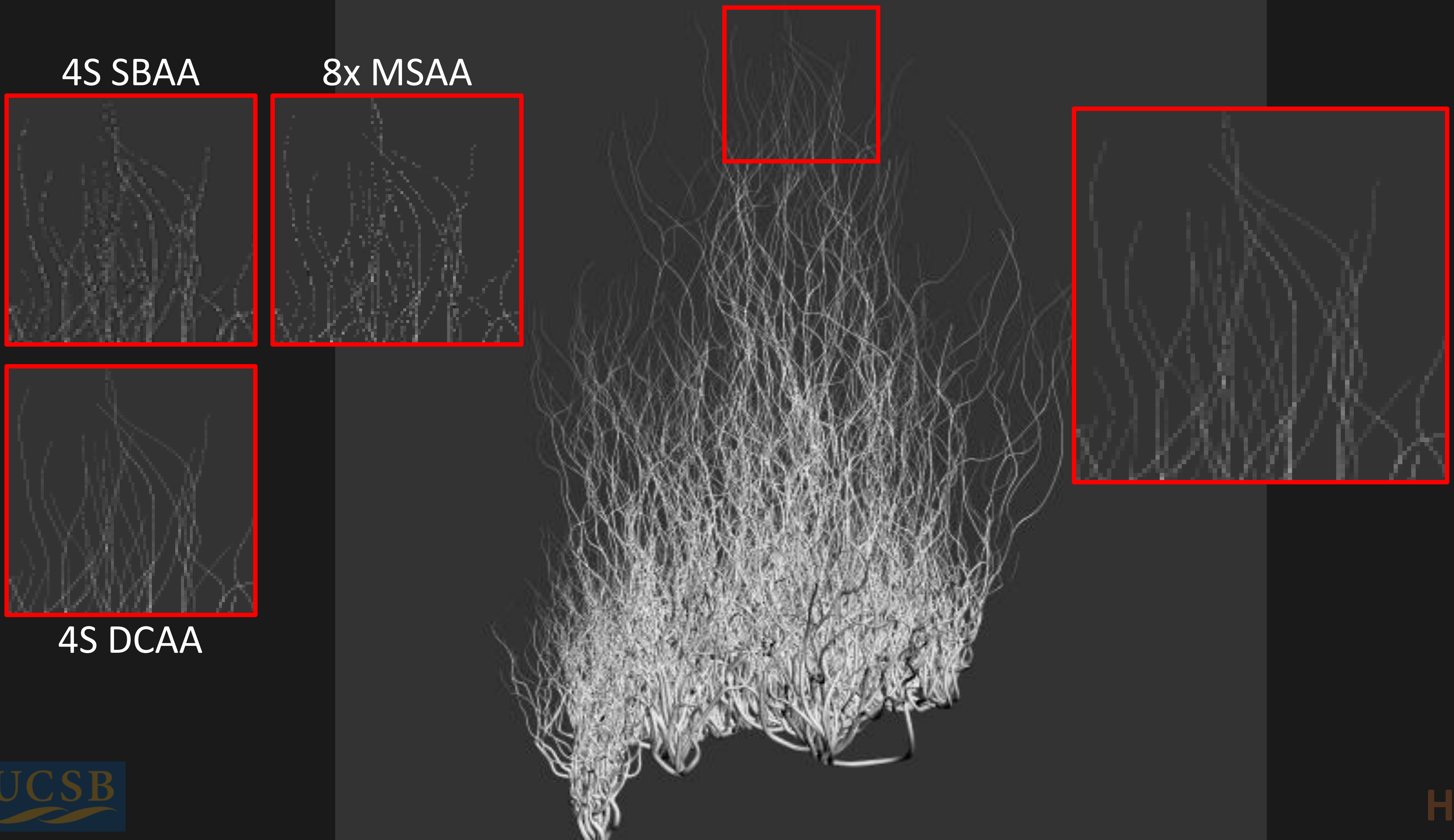
4S SBAA



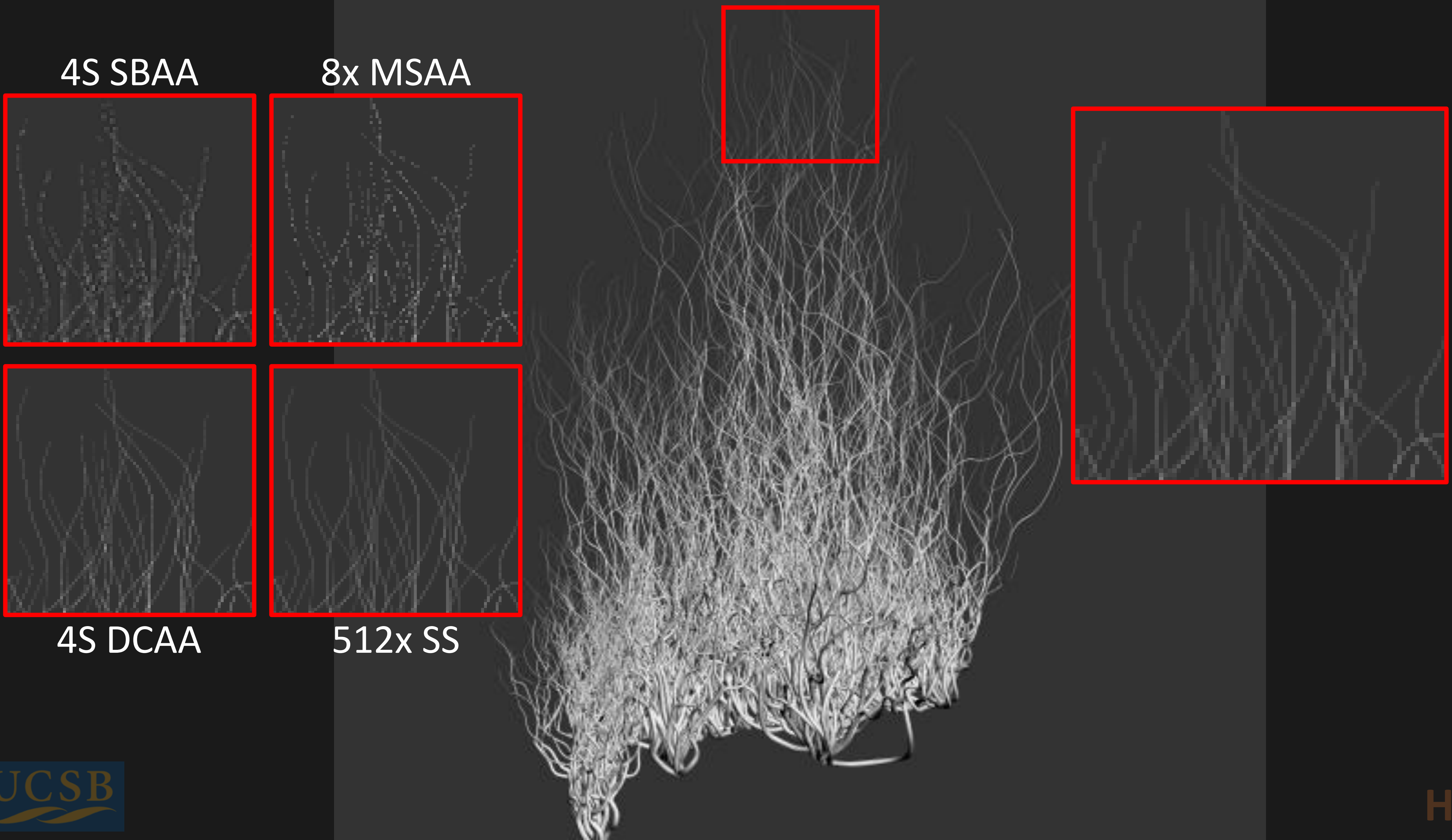
8x MSAA



4S DCAA



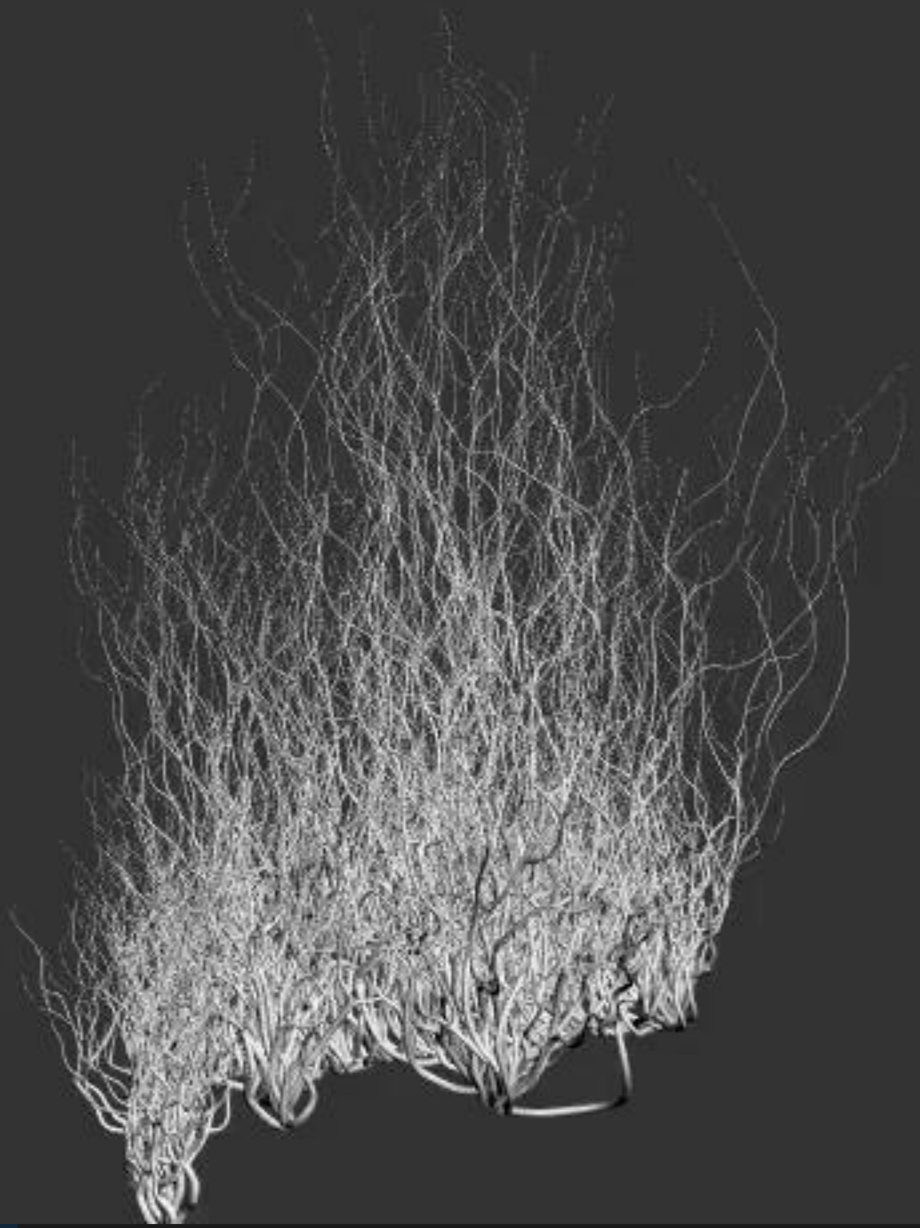
512x Supersampling



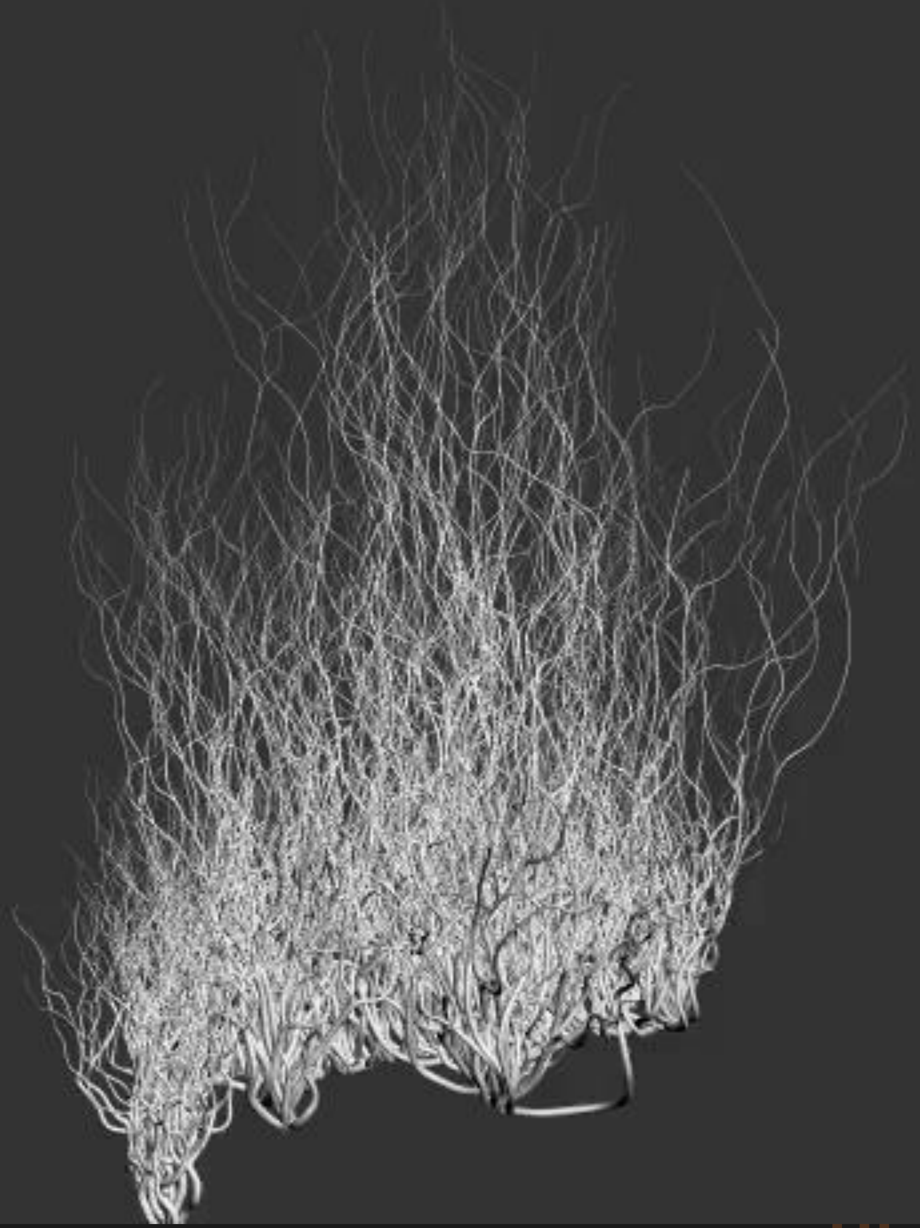
UCSB

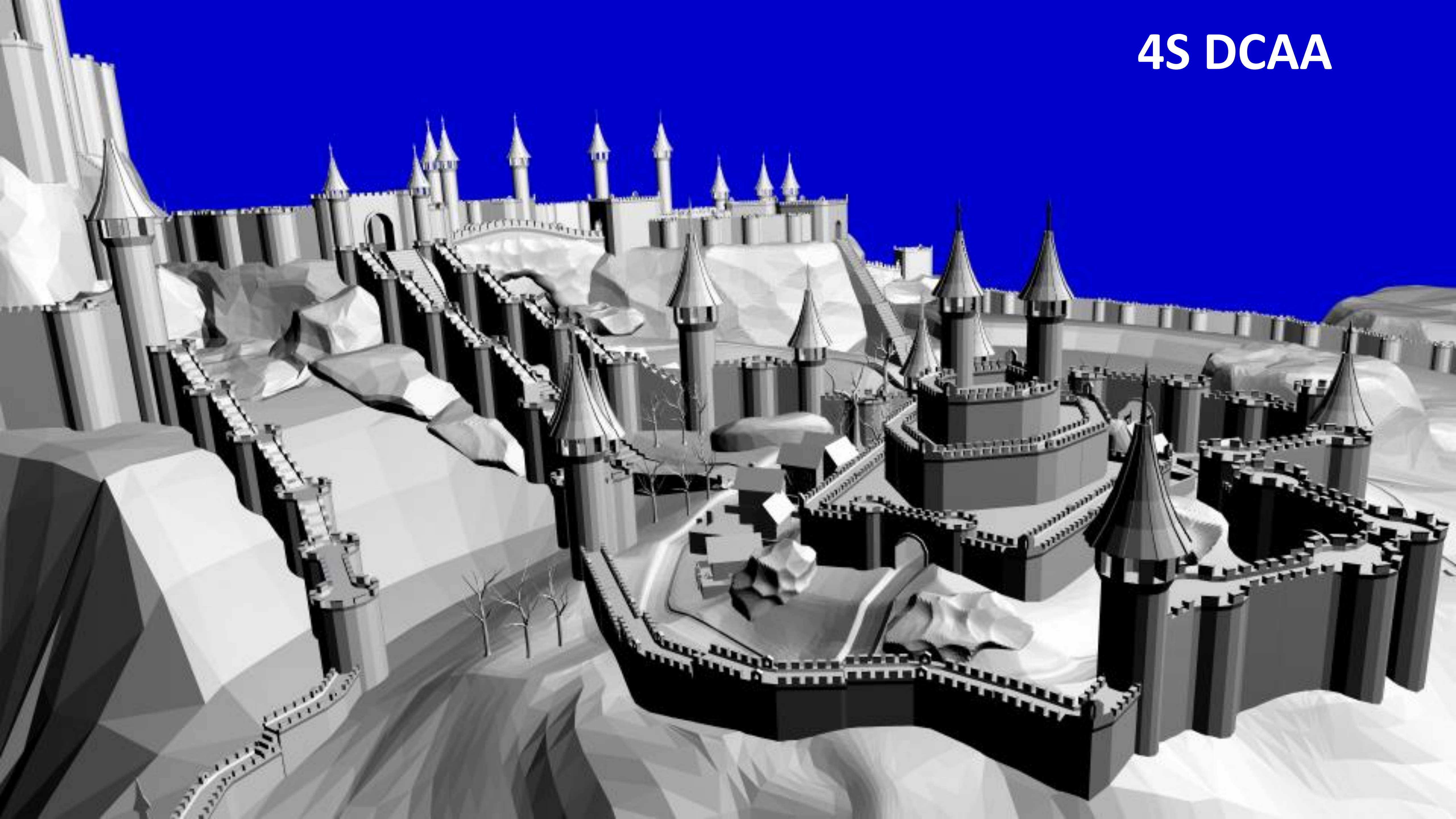
HPG 2015

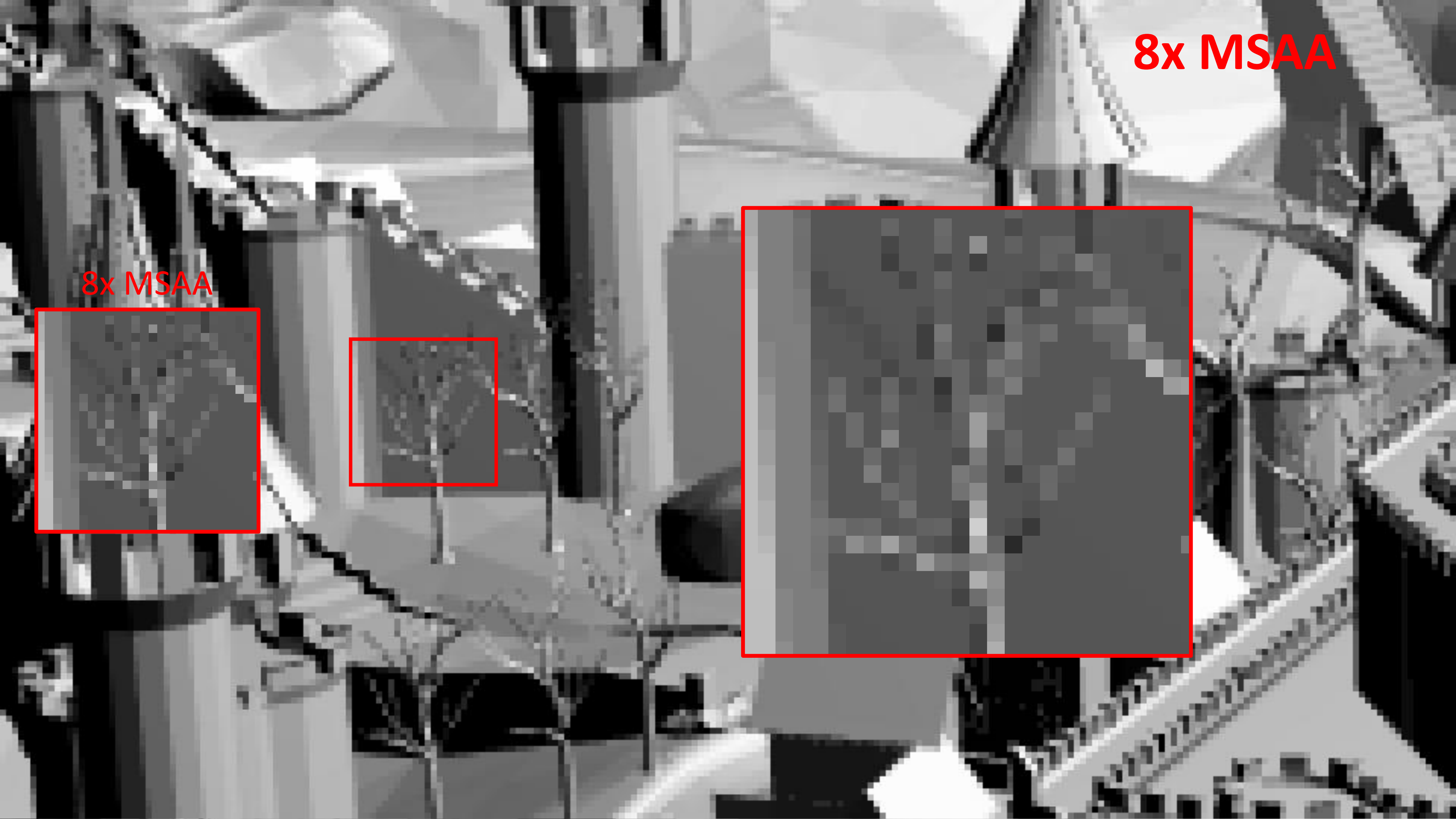
8x MSAA

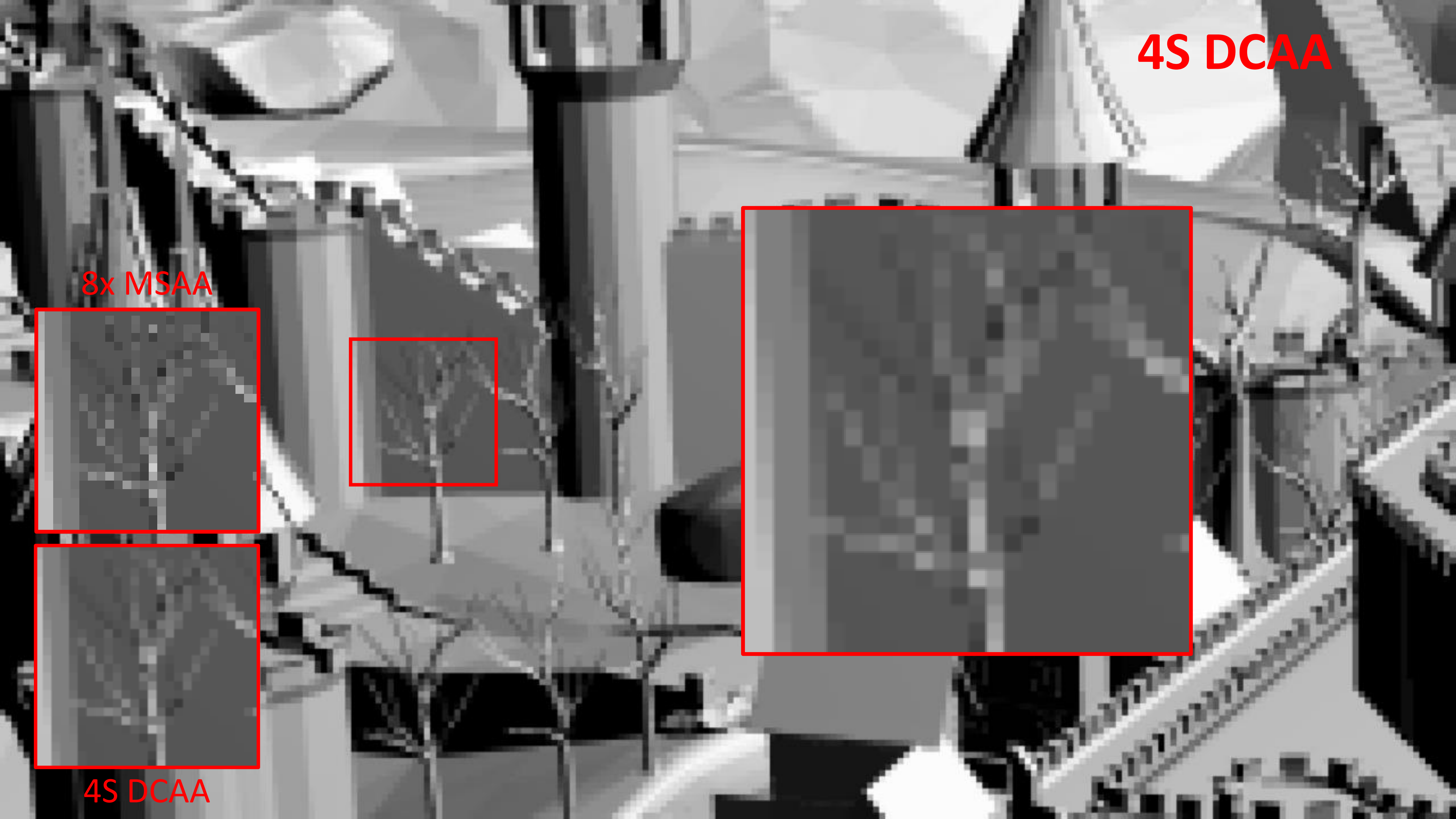


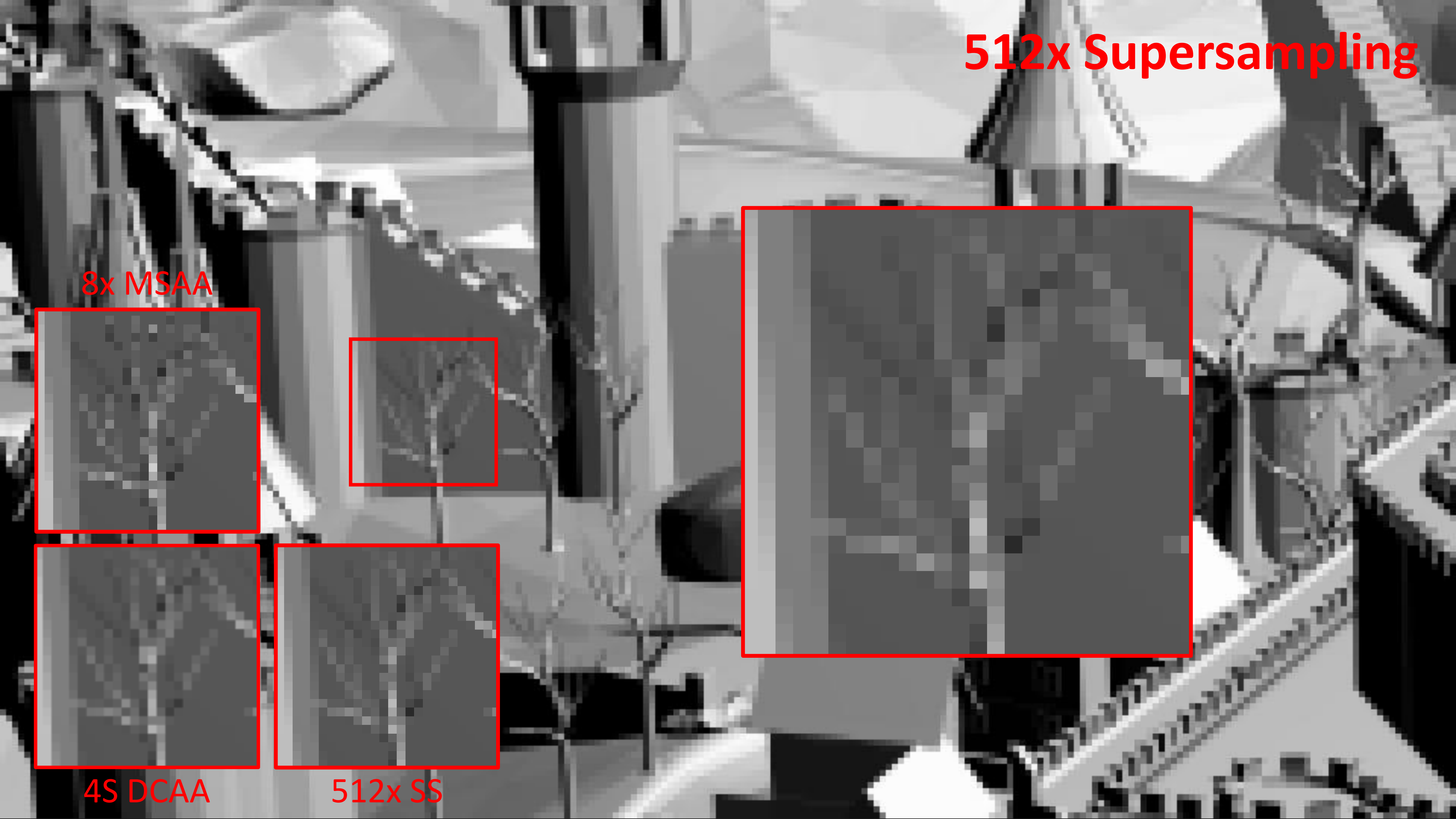
4S DCAA



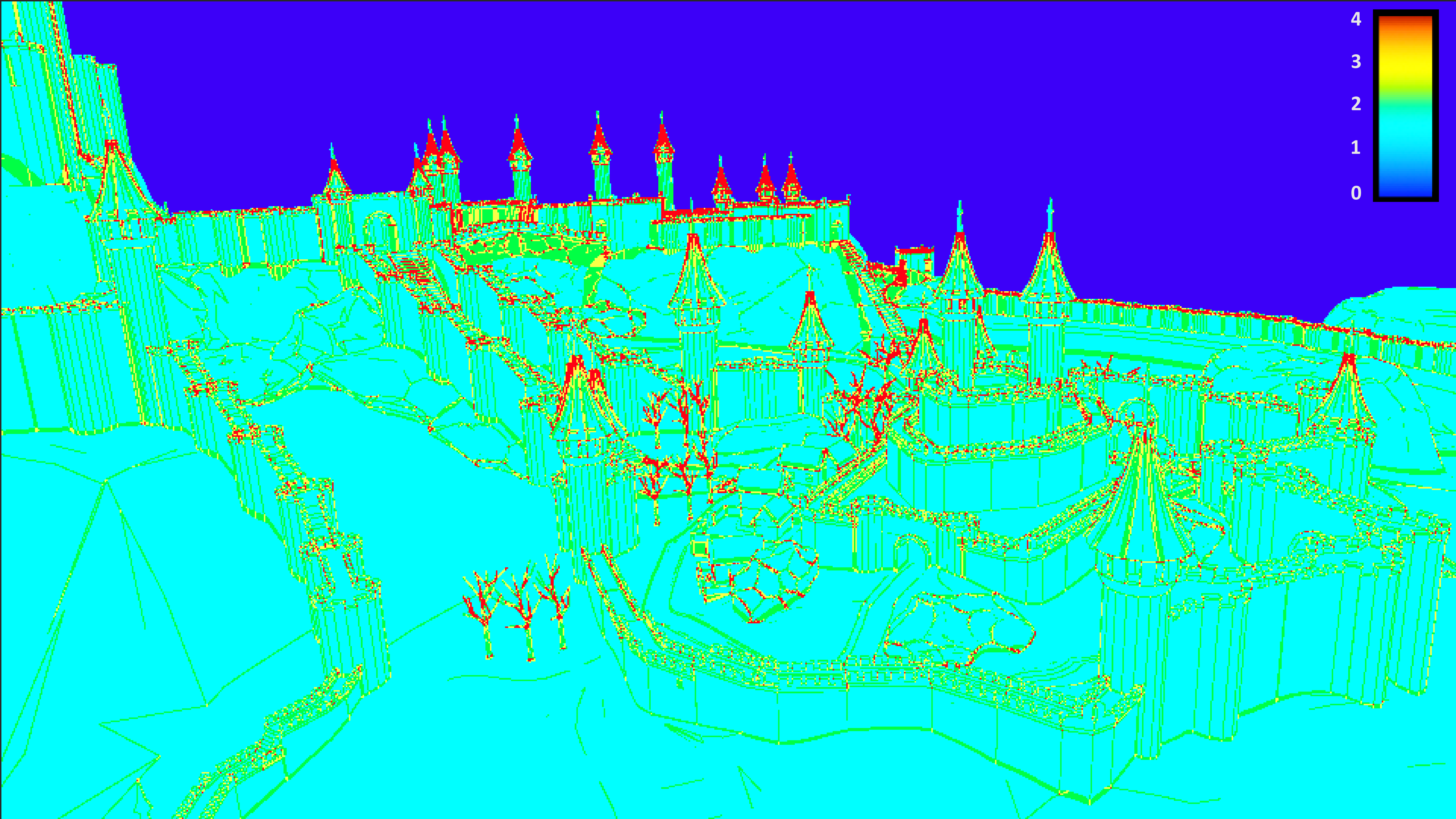

4S DCAA


8x MSAA
8x MSAA


4S DCAA
8x MSAA
4S DCAA



512x Supersampling
8x MSAA
4S DCAA
512x SS


4
3
2
1
0


4S DCAA

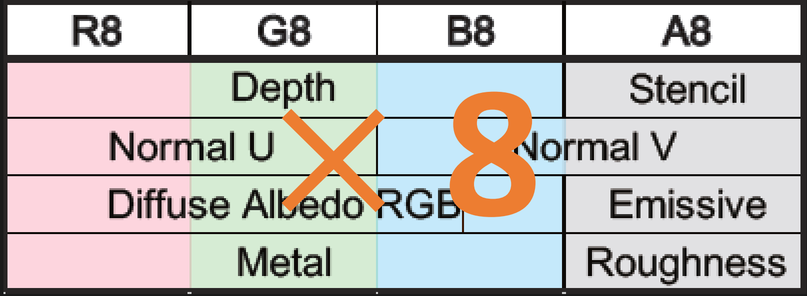

4S SBAA
4S SBAA

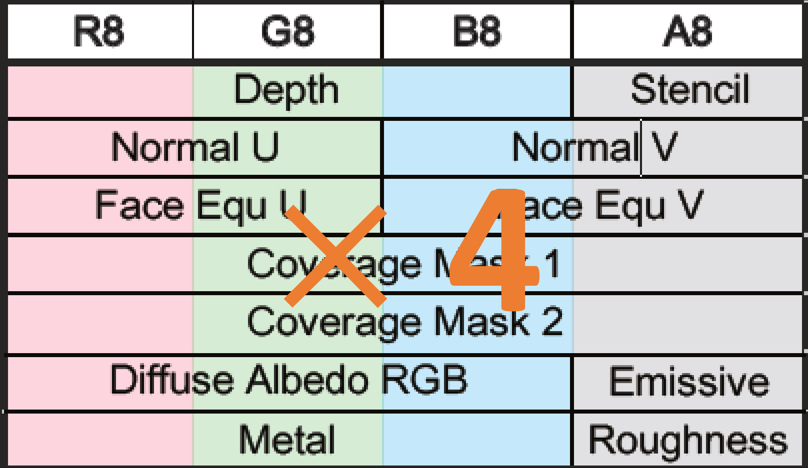

8x MSAA
4S SBAA
8x MSAA


4S DCAA
4S SBAA
8x MSAA
4S DCAA

512x Supersampling

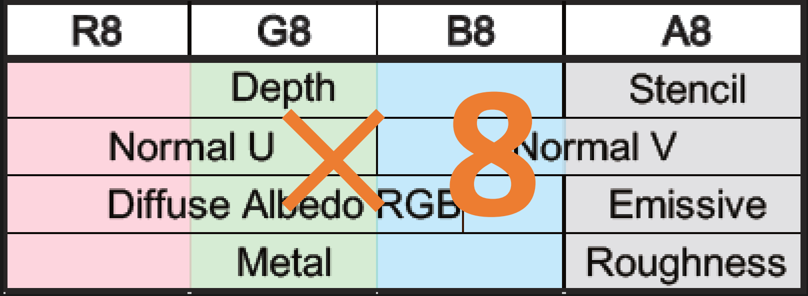
Evaluation

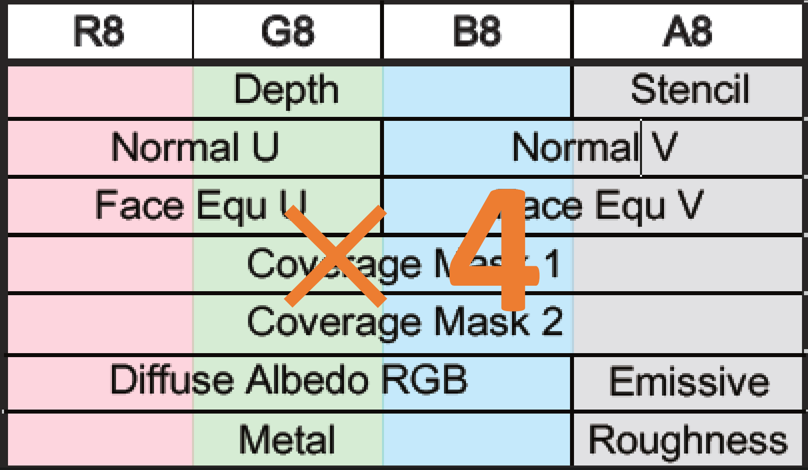


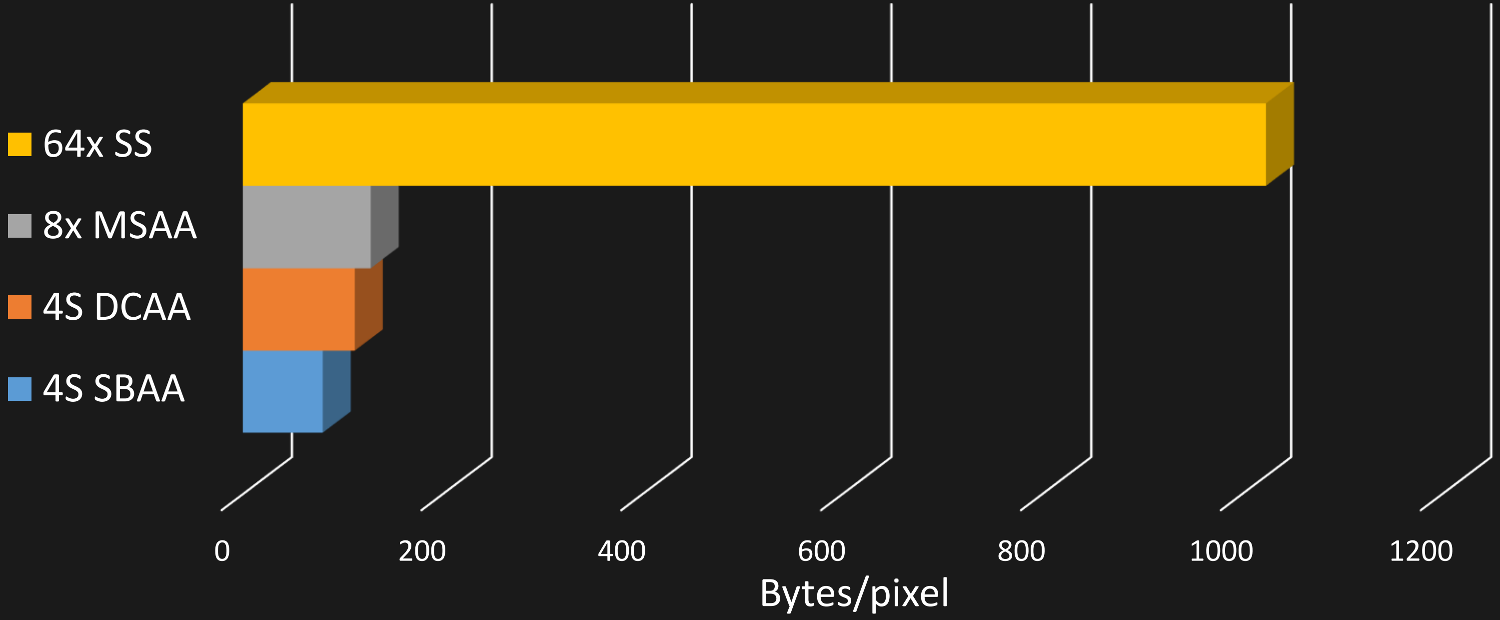


- Memory consumption
  - 28 bytes/surface × 4 surfaces/pixel = **112 bytes/pixel**
  - 8x MSAA: 16 bytes/sample × 8 = 128 bytes/pixel (1.14x of DCAA)
  - 64x SuperSampling: 16 bytes/sample × 64 = 1024 bytes/pixel (9.14x of DCAA)

Evaluation







	Z-Prepass	Merge	Resolve & Render	Total
Citadel	1.3 ms	23.2 ms	6.4 ms	30.9 ms
Tentacles	1.3 ms	574.5 ms	6.2 ms	582.0 ms

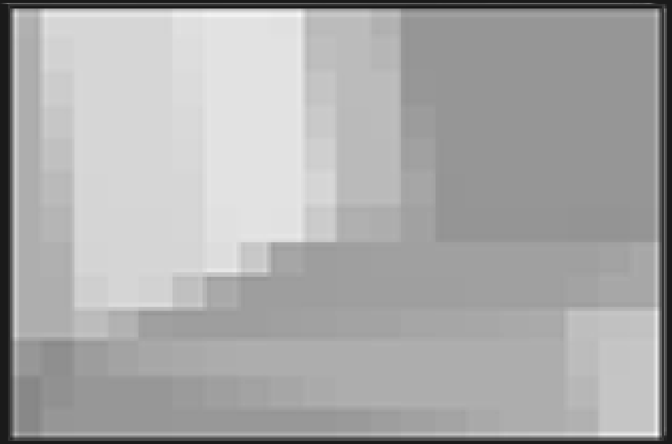
MSE		
4S SBAA	8x MSAA	DCAA
$2.47*10^{-4}$	$1.32*10^{-4}$	**$6.40*10^{-5}$**
$2.28*10^{-3}$	$6.05*10^{-4}$	**$5.65*10^{-4}$**

Limitation

- Rendering speed
  - Pixel Shader Interlock with Conservative Rasterizer
  - Better synchronization would help
- Merging artifacts

DCAA | **512x Supersampling**

Incorrectly merge



Limitation

- Rendering speed
  - Pixel Shader Interlock with Conservative Rasterizer
  - Better synchronization would help

- Merging artifacts

DCAA | **512x Supersampling**

Incorrectly discard

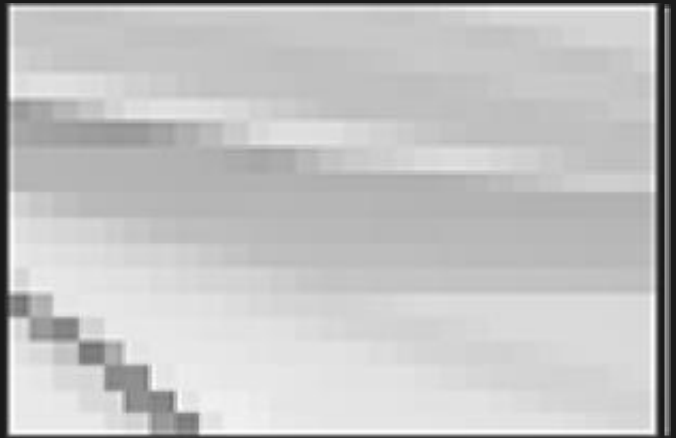
Limitation

- Rendering speed
  - Pixel Shader Interlock with Conservative Rasterizer
  - Better synchronization would help
- Merging artifacts

DCAA

512x Supersampling

Z-prepass leak



Conclusion

- A streaming compression algorithm for geometric anti-aliasing
- Achieves close to 512x SS result with storage of 8x MSAA

- Decouple visibility into depth and coverage
  - Higher sample rates in reasonable memory footprint
  - Other applications

- Performance limitation

Acknowledgments

- Anonymous reviewers
- NVIDIA hardware donation
- Aaron Lefohn and Anjul Patney for helpful discussions
- Zi Wang, Chieh-Chi Kao, Ekta Prashnani, and Abhishek Badki for help with the paper

- Funding:
    - National Science Foundation (IIS-1342931 and IIS-1321168)
    - NVIDIA internship

Thank you!