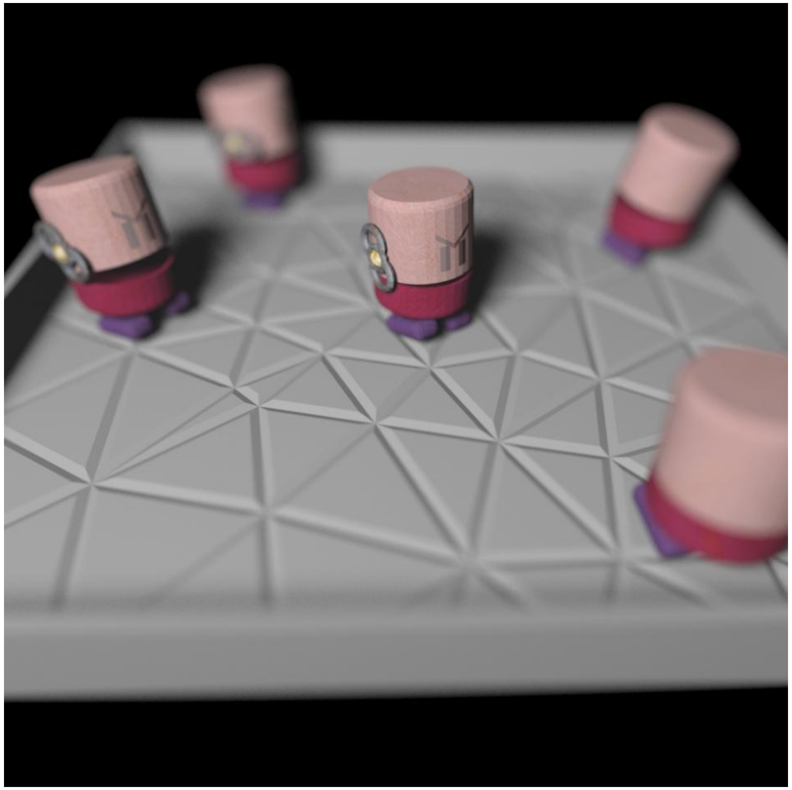
Adaptive Wavelet Rendering

Author: **Ryan Overbeck**
Craig Donner
Ravi Ramamoorthi

Presenter: Guillaume de Choulot

The Problem (combined effects)



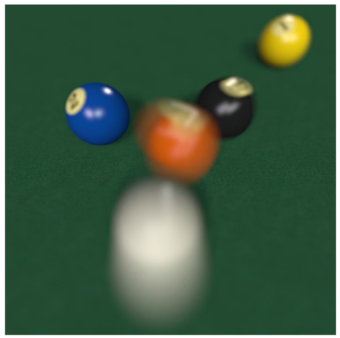
Pixel = $\int \int \int \ldots$ ⇨ 6D

Pixel Area, Camera Aperture, Area Light

General combinations of effects

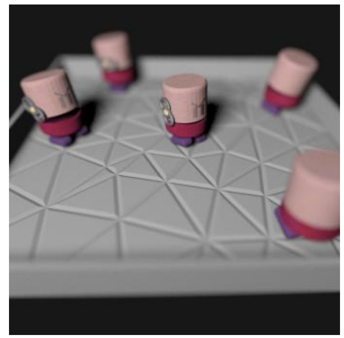
Antialiasing
Depth of field

4D



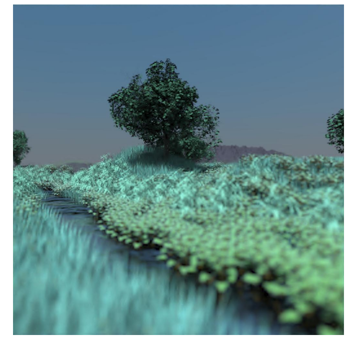
Antialiasing
Depth of field
Motion Blur

5D



Antialiasing
Depth of field
Area Lighting

6D



Antialiasing
Depth of field
Envir. Lighting

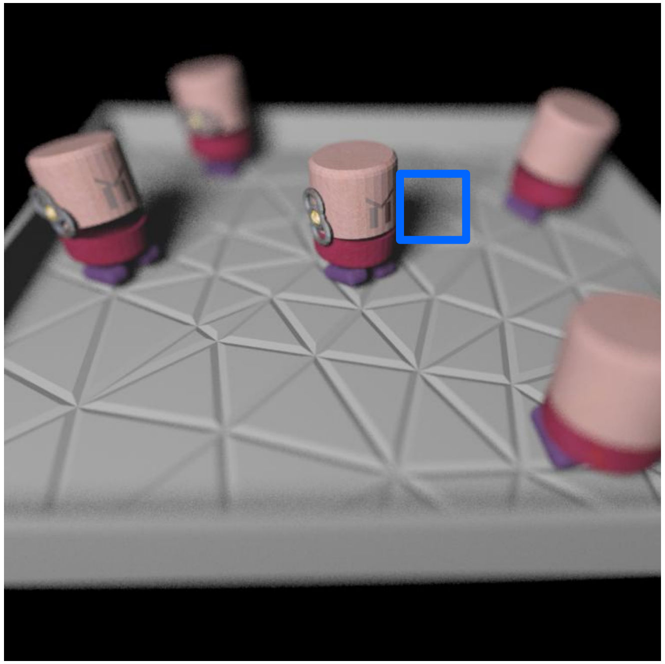
6D

Antialiasing
Depth of field
Area Lighting
1 Bounce GI

8D

Monte Carlo Problem (1/1)

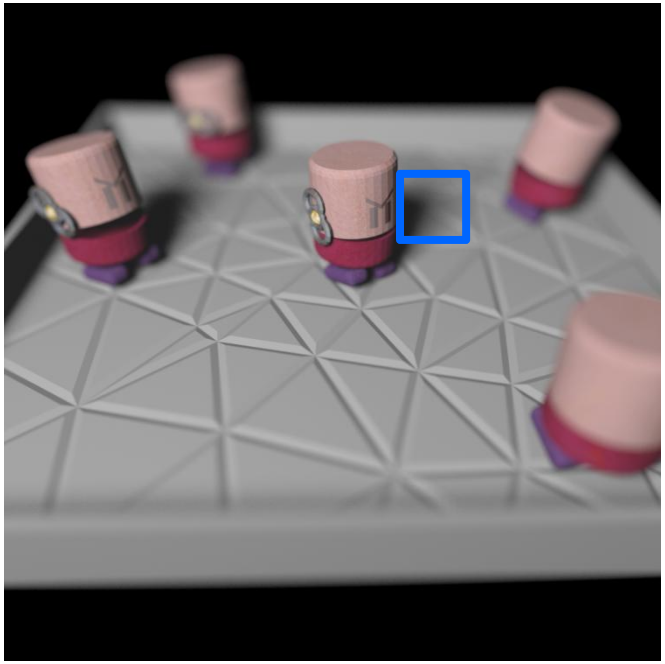
Noisy for low sample counts (smooth regions)



32 Samples Per Pixel

Monte Carlo Problem (2/2)

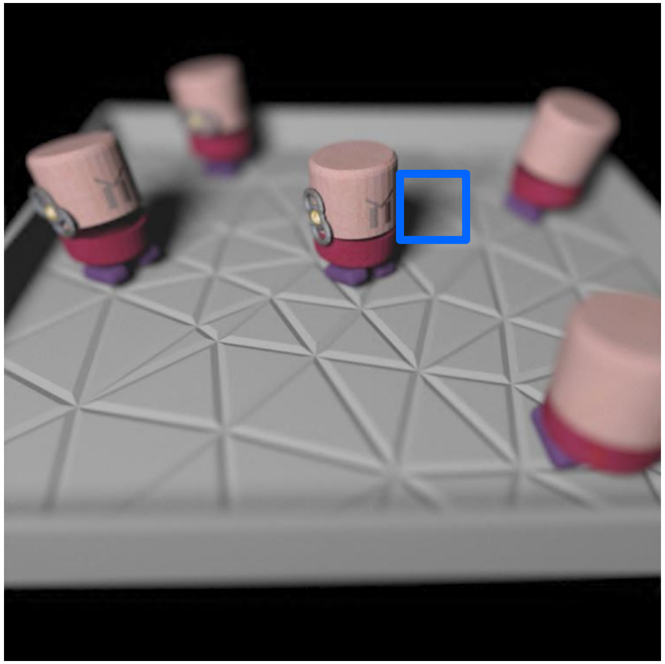
Requires hundreds to thousands of samples



512 Samples Per Pixel

Our Solution (important sampling)

Adaptive Wavelet Rendering



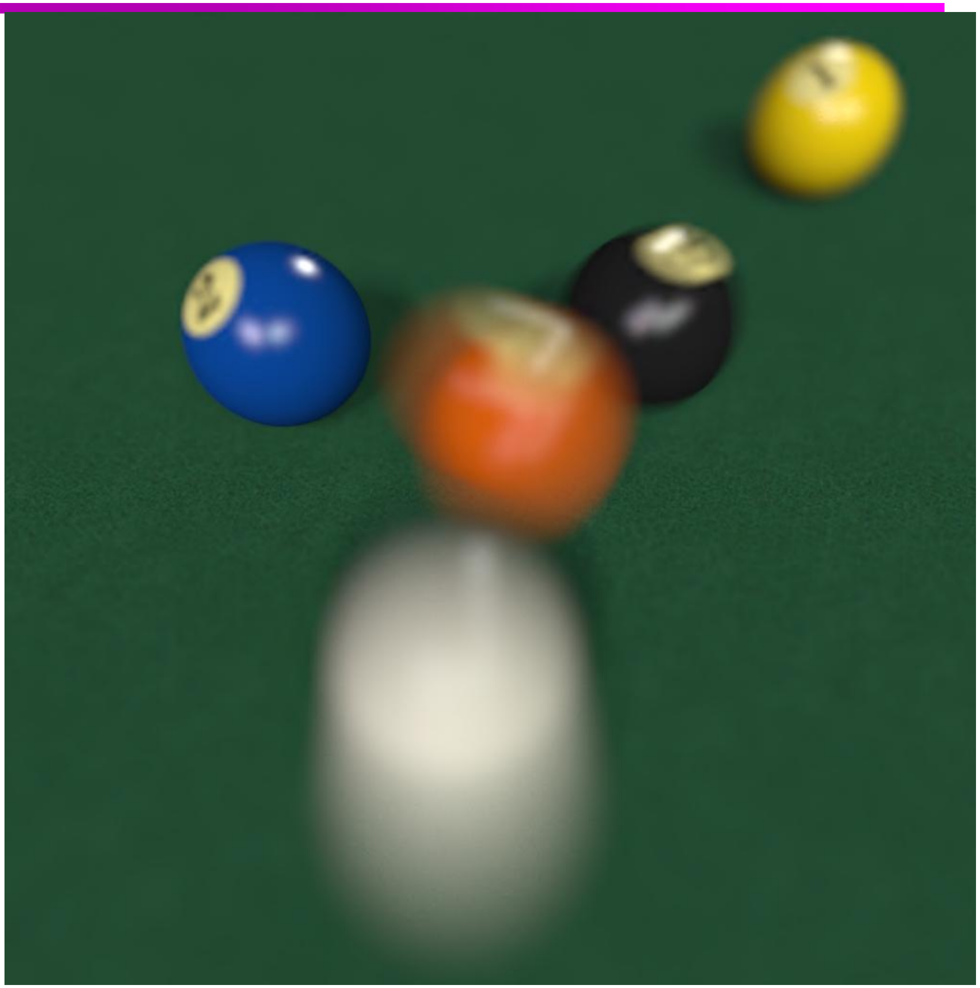
32 Samples Per Pixel
(average)

Features of Adaptive Wavelet Rendering

Low Sample Counts

Converges from smooth

Features of Adaptive Wavelet Rendering



Average of 32 Samples Per Pixel

Low Sample Counts

Converges from smooth

Near Reference:
32 samples per pixel

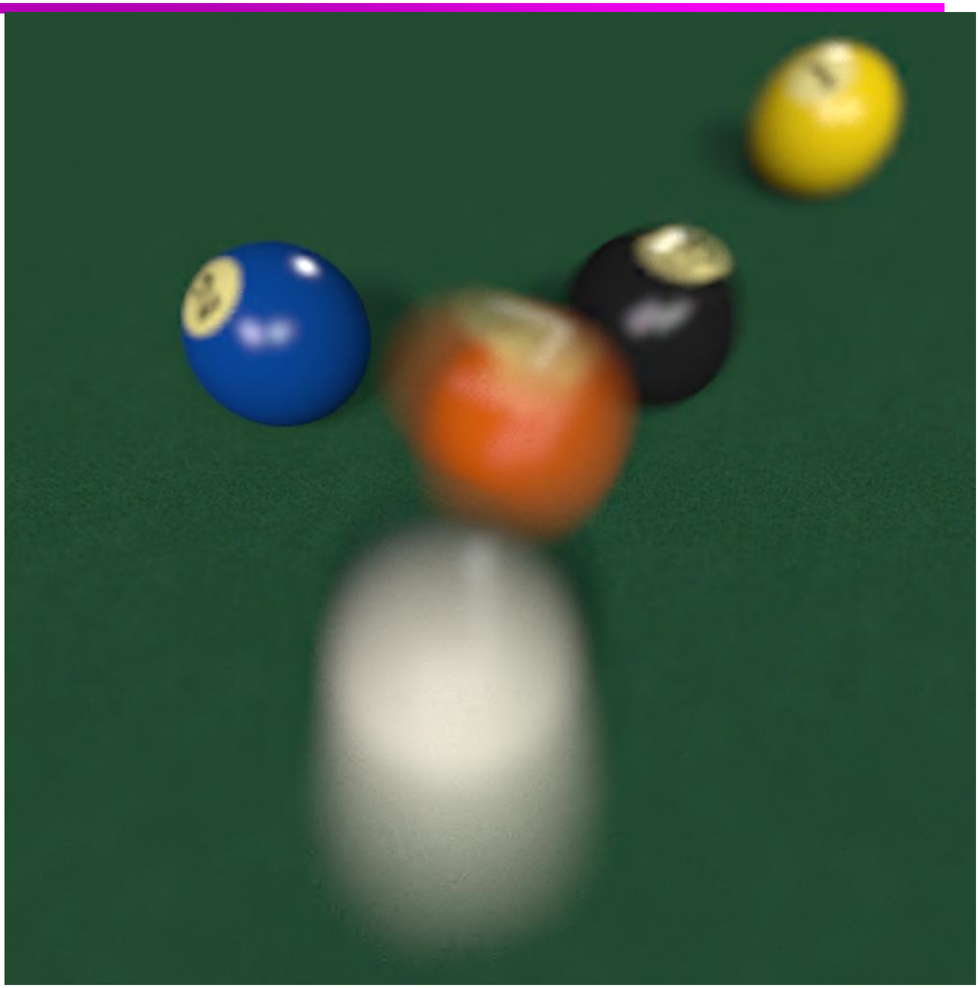
Features of Adaptive Wavelet Rendering

Low Sample Counts

Converges from smooth

Near Reference:
32 samples per pixel

Smooth Preview Quality:
8 samples per pixel



Average of 8 Samples Per Pixel

Features of Adaptive Wavelet Rendering

Low Sample Counts

Efficient

- Less samples gives
  - Faster render times

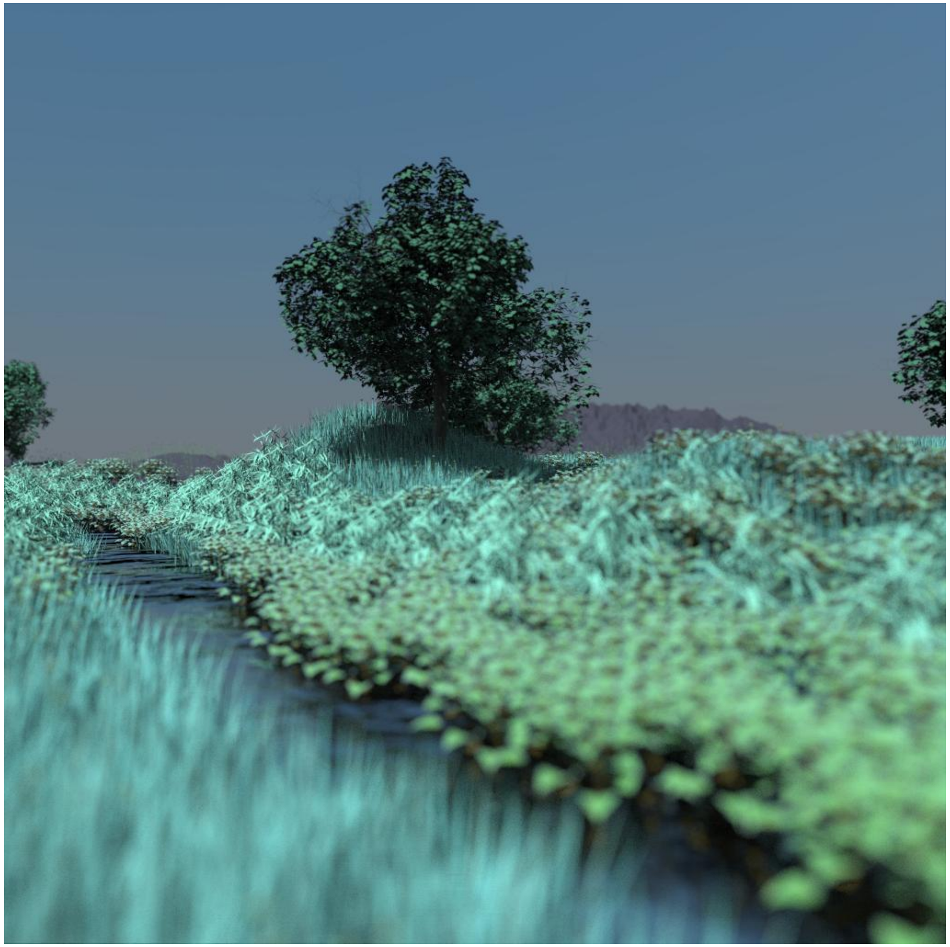
Features of Adaptive Wavelet Rendering

Low Sample Counts

Efficient

**Less samples gives
Faster render times**

Monte Carlo (512 spp)
>6 Hours



> 6 Hours
Monte Carlo
512 Samples Per Pixel

KAIST

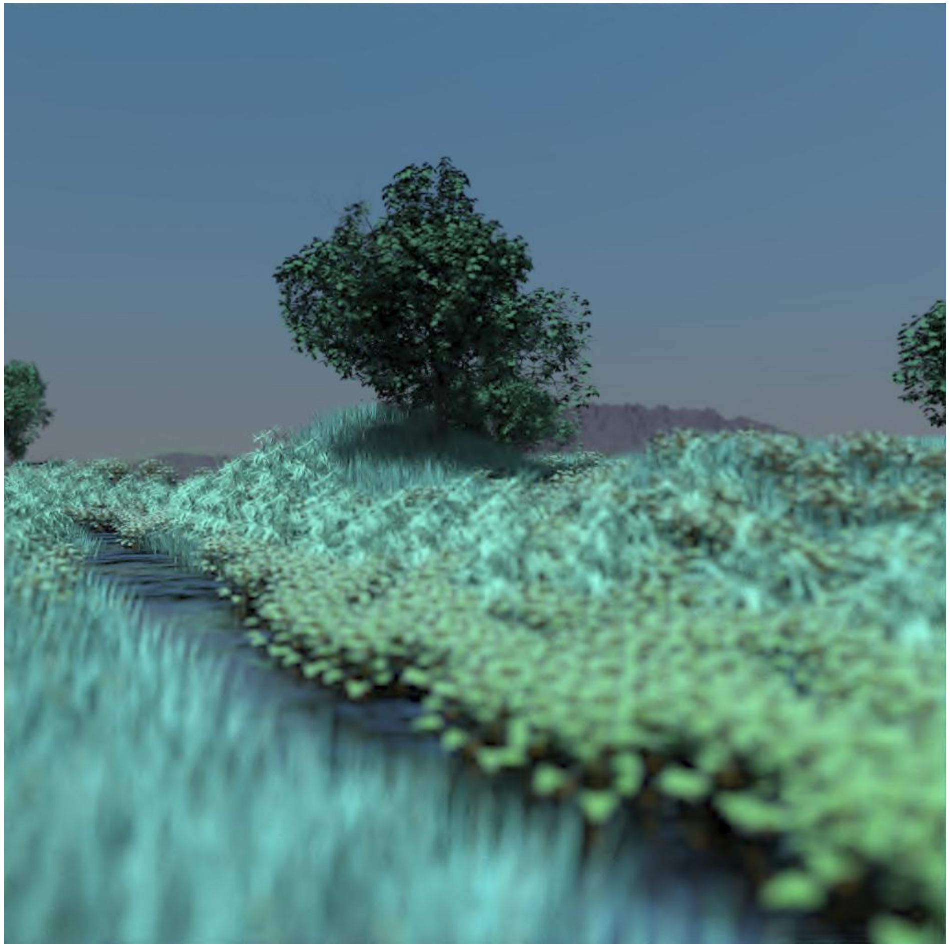
Features of Adaptive Wavelet Rendering

Low Sample Counts

Efficient

Less samples gives Faster render times

Monte Carlo (512 spp)	Our Method (32 spp)
>6 Hours	34 minutes



34 Minutes

Adaptive Wavelet Rendering

32 Samples Per Pixel (average)

Features of Adaptive Wavelet Rendering

Low Sample Counts

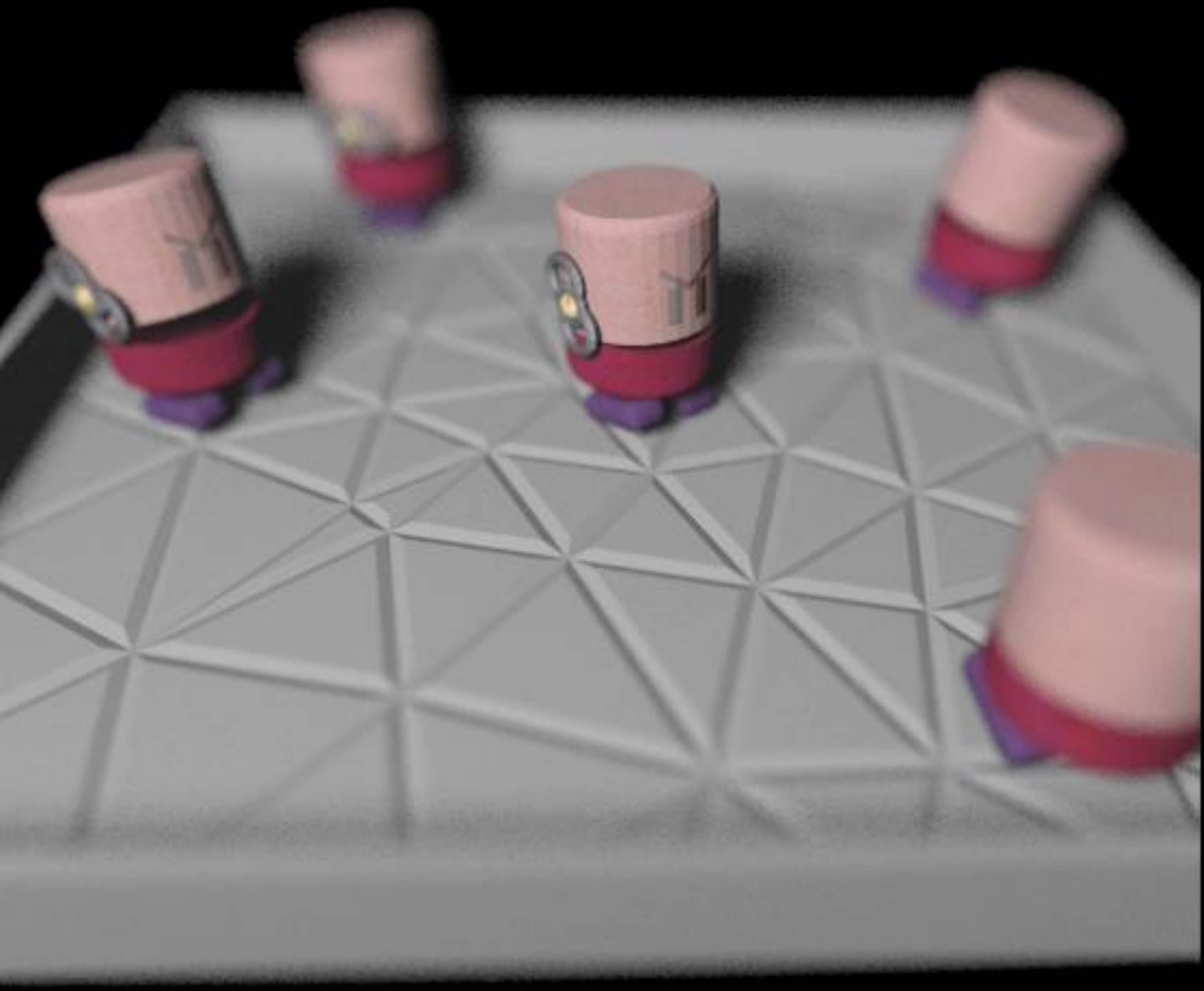
Efficient

General

- **Insensitive to problem dimensionality**
- **General combinations of effects**

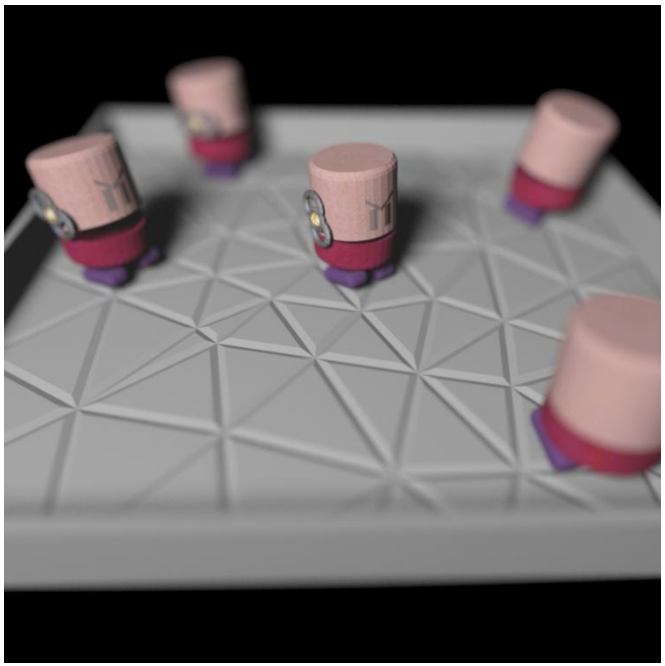
32 Samples per pixel (average)
15 minutes

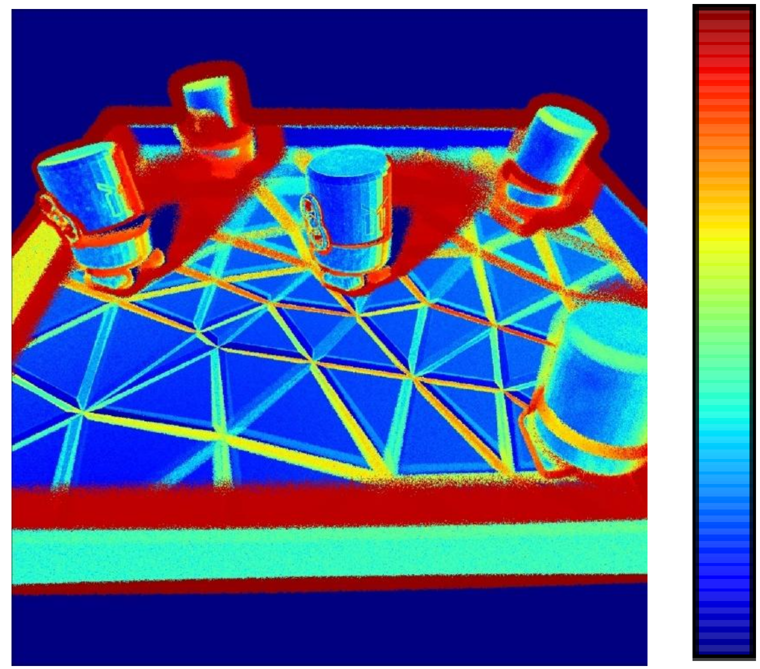
$\int$ Pixel Area $\int$ Camera Aperture $\int$ Area Light $\int$ Hemi-sphere … ⇨ 8D



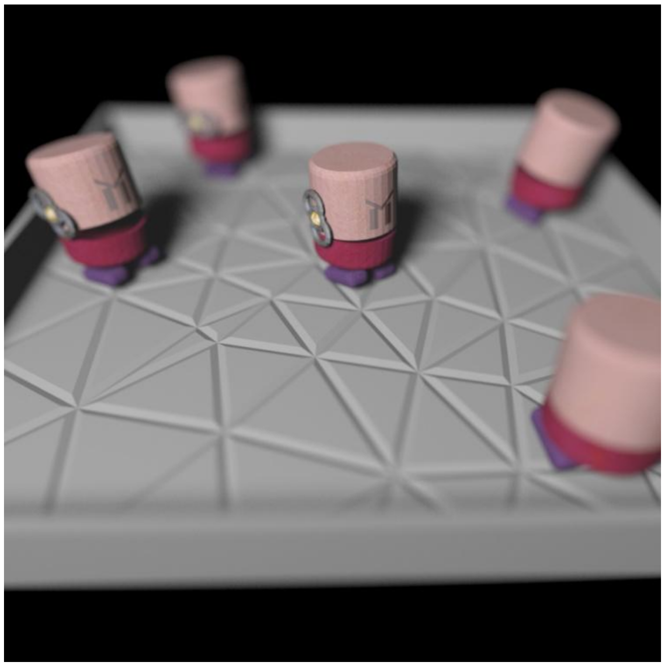
The Insight (Important sampling)

Variance is often local

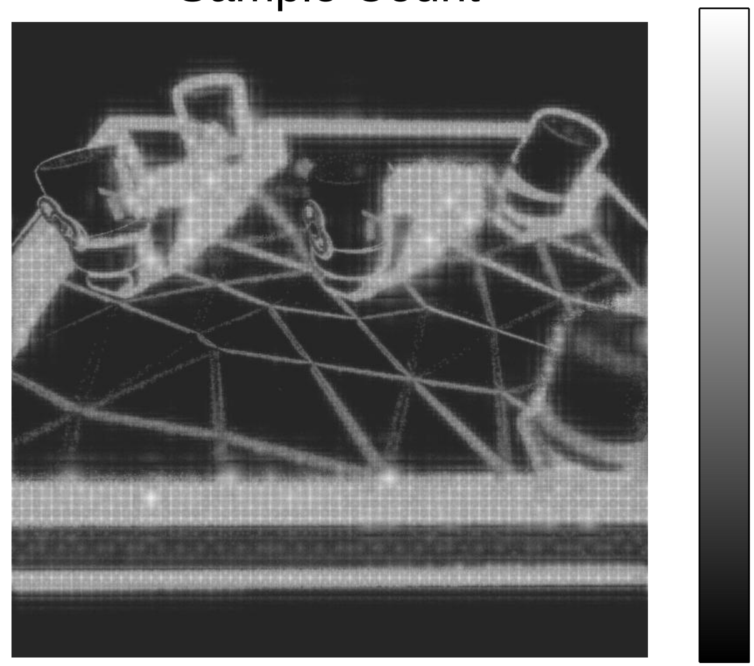




Send more samples to high variance



Sample Count

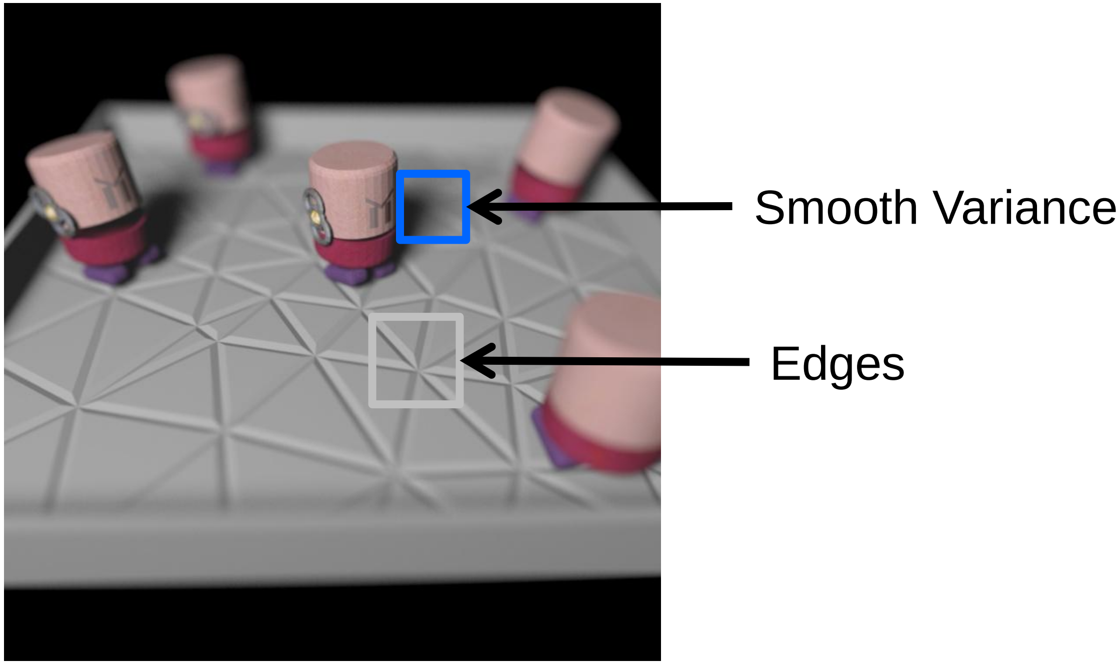


Our method

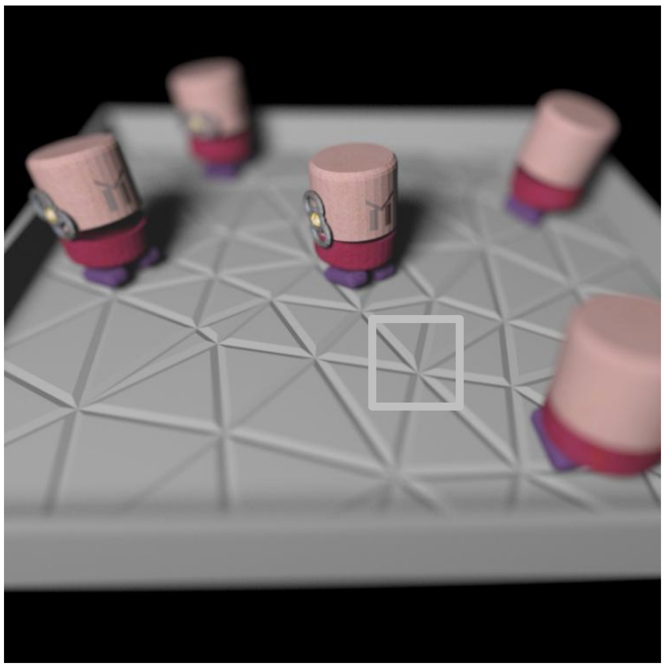
32 Samples Per Pixel
(average)

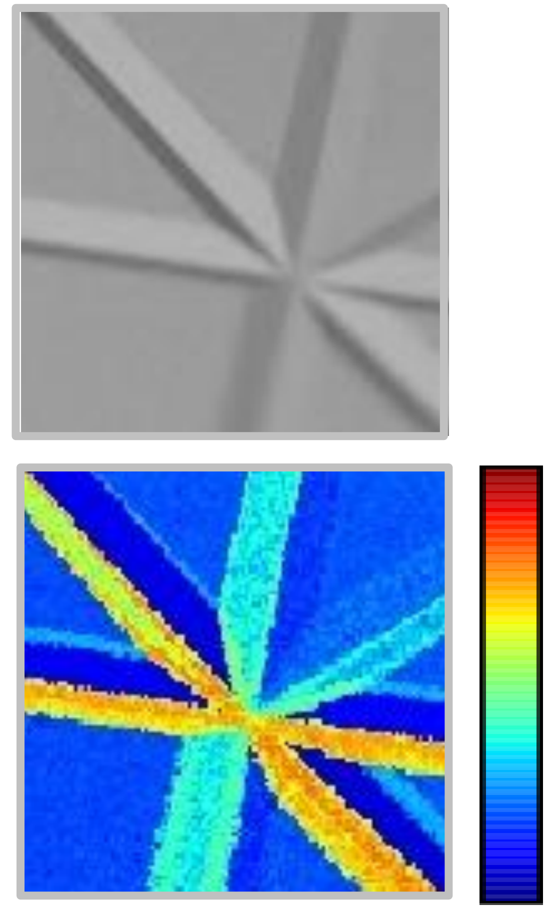
KAIST

Two forms of variance



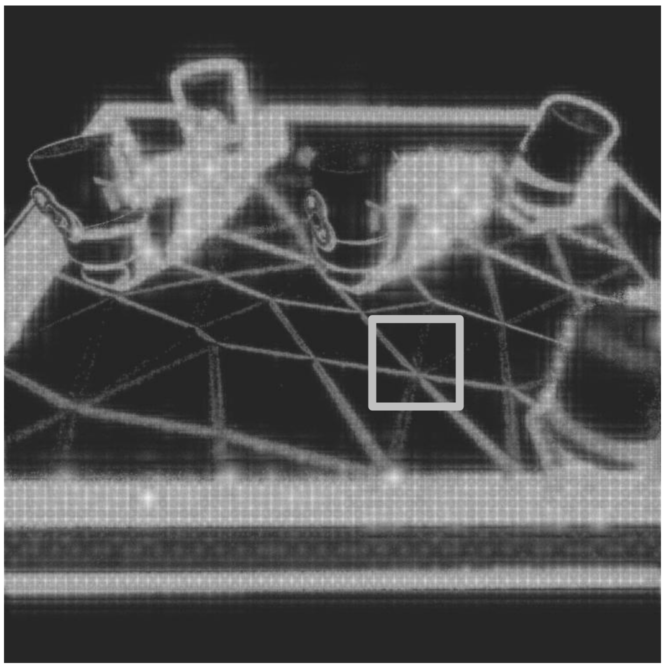
Variance from image space: edges



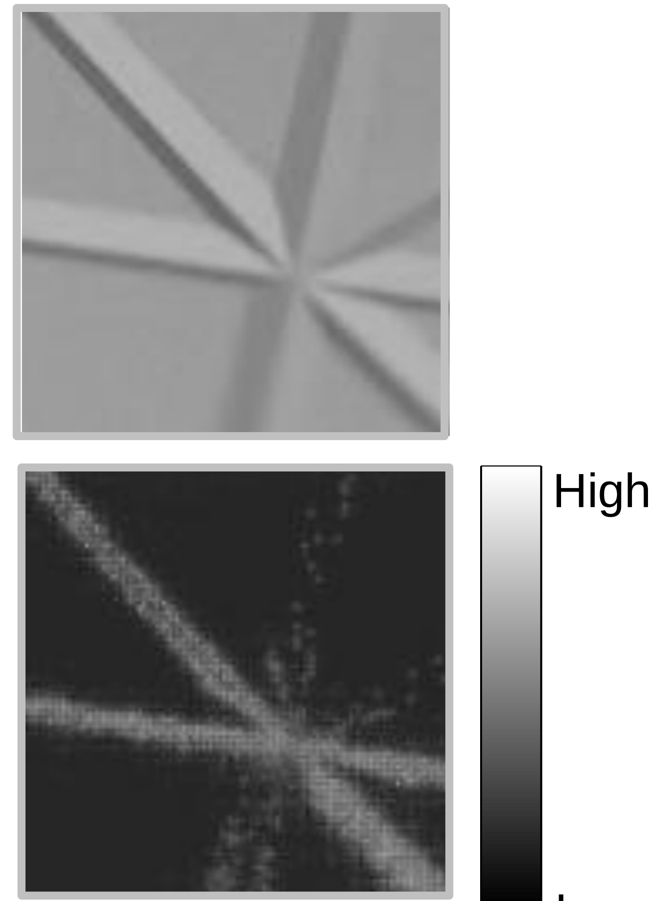


Variance

Focused samples to image edges



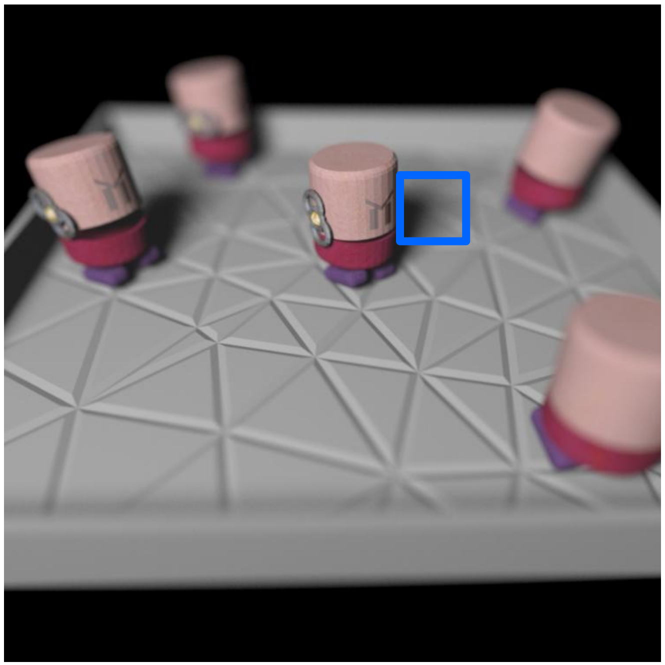
Final Result

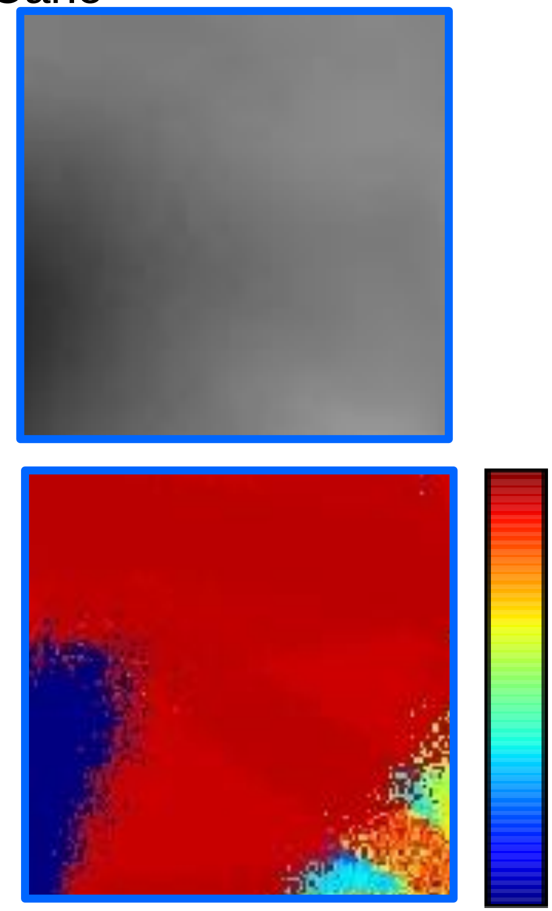


Samples

Variance from other dims: smooth

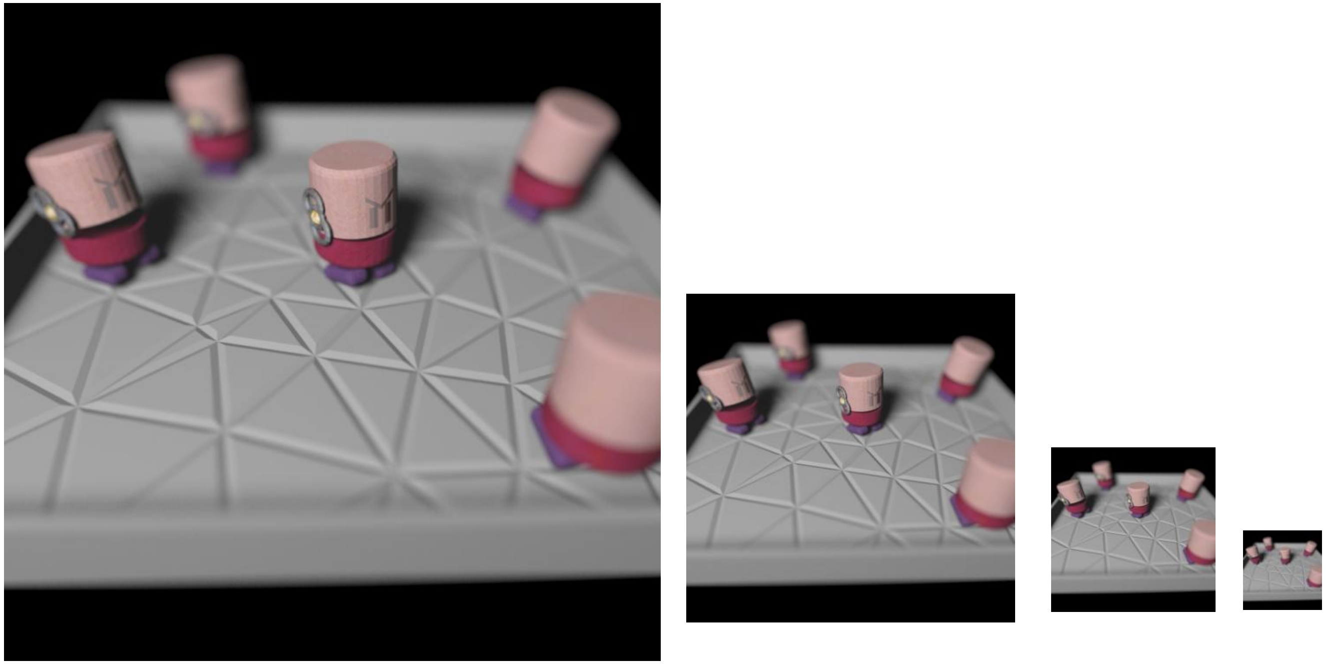
Difficult for Monte Carlo



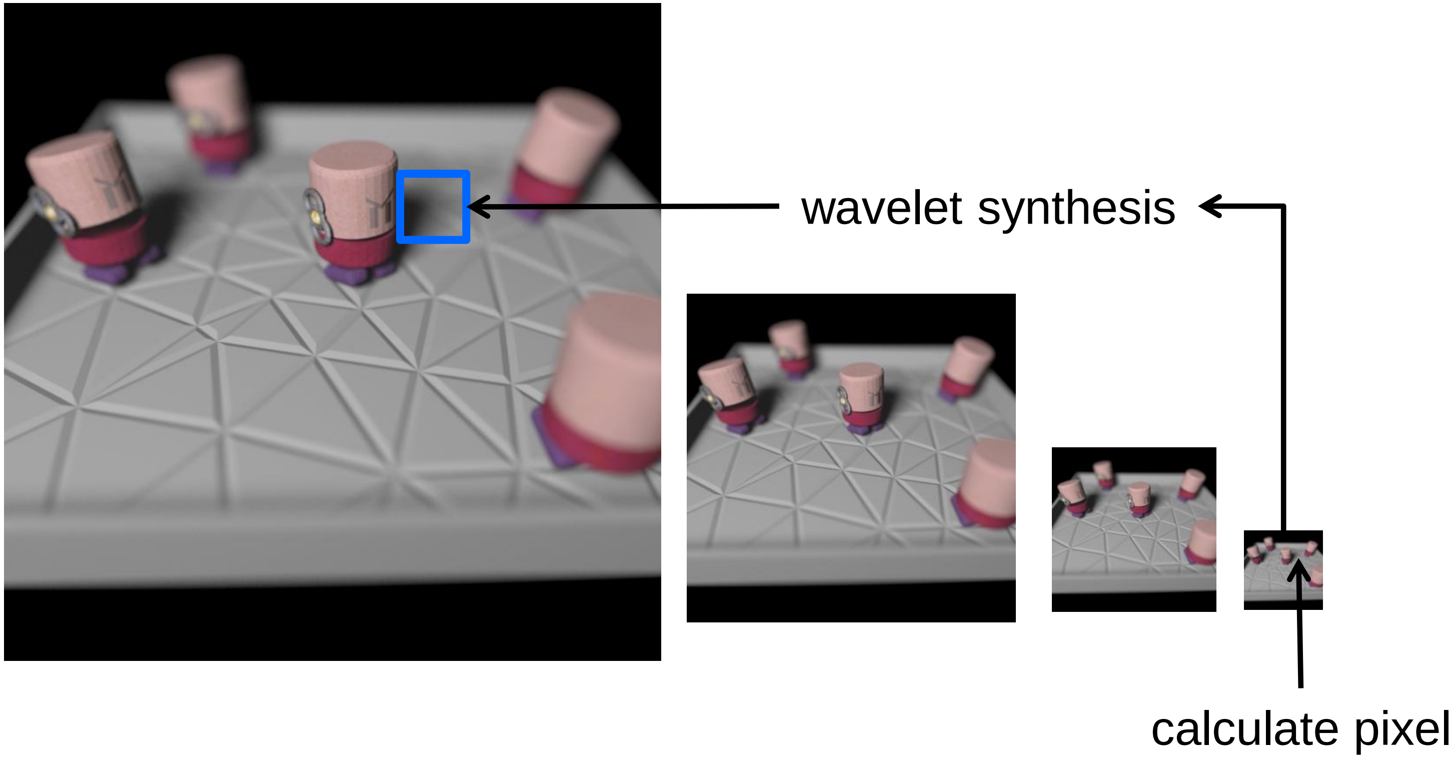


Variance

Smooth is easier in multi-scale

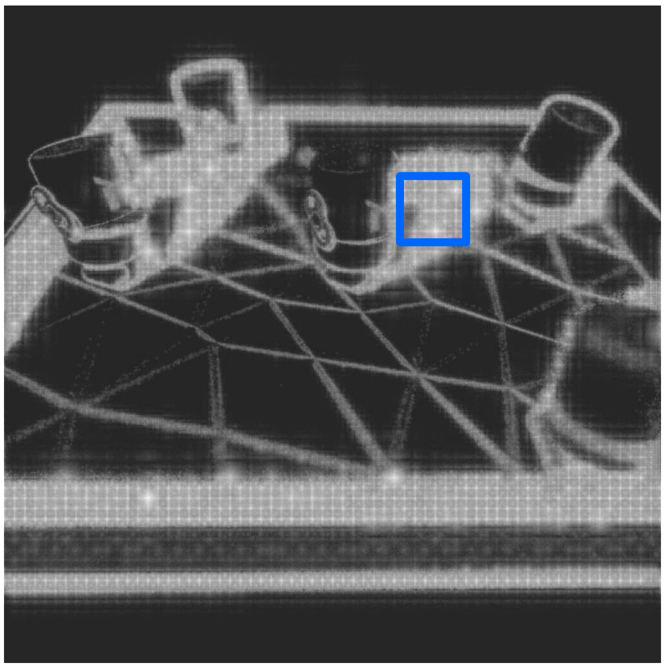


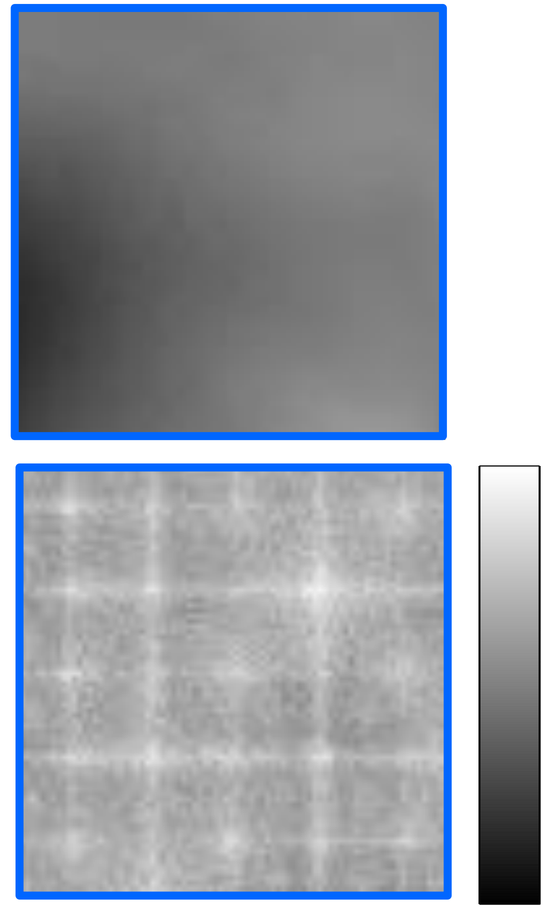
Smooth is easier for wavelets



KAIST

Coarse Sampling of Smooth Regions





Algorithm Outline

Start: 4 Samples per Pixel

Adaptive Sampling

Reconstruction

Related Work

Adaptive sampling

- Bolin & Meyer 1998, Whitted 1980, Mitchell 1987, Veach and Guibas 1997, Walter et al. 2006 (Multidim Lightcuts)

Multi-scale

- Keller 2001 (Hierarchical MC), Heinrich and Sindambiwe 1999, Guo 1998, Bala et al. 2003, Walter et al. 2005 (Lightcuts), Perona and Malik 1990

Wavelet sampling and reconstruction

- Our method

works well for both edges and smooth regions

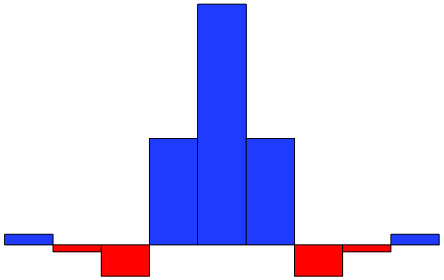
KAIST

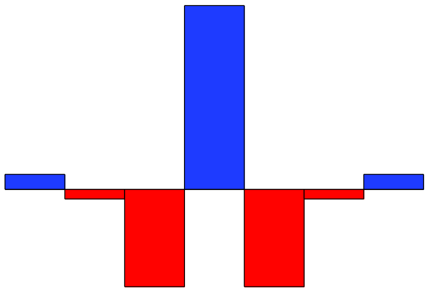
Background: Wavelets

Wavelets made of 2 functions

ϕ Scale

ψ Wavelet



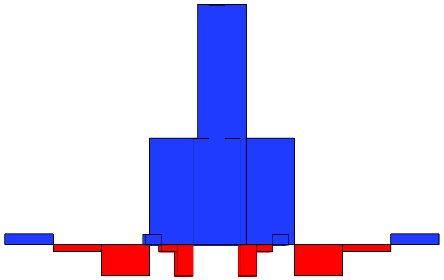


Low Pass
(smooth)

Band Pass
(edges)

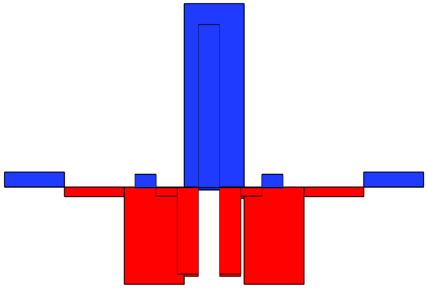
Wavelet Hierarchy (1D)

Scale



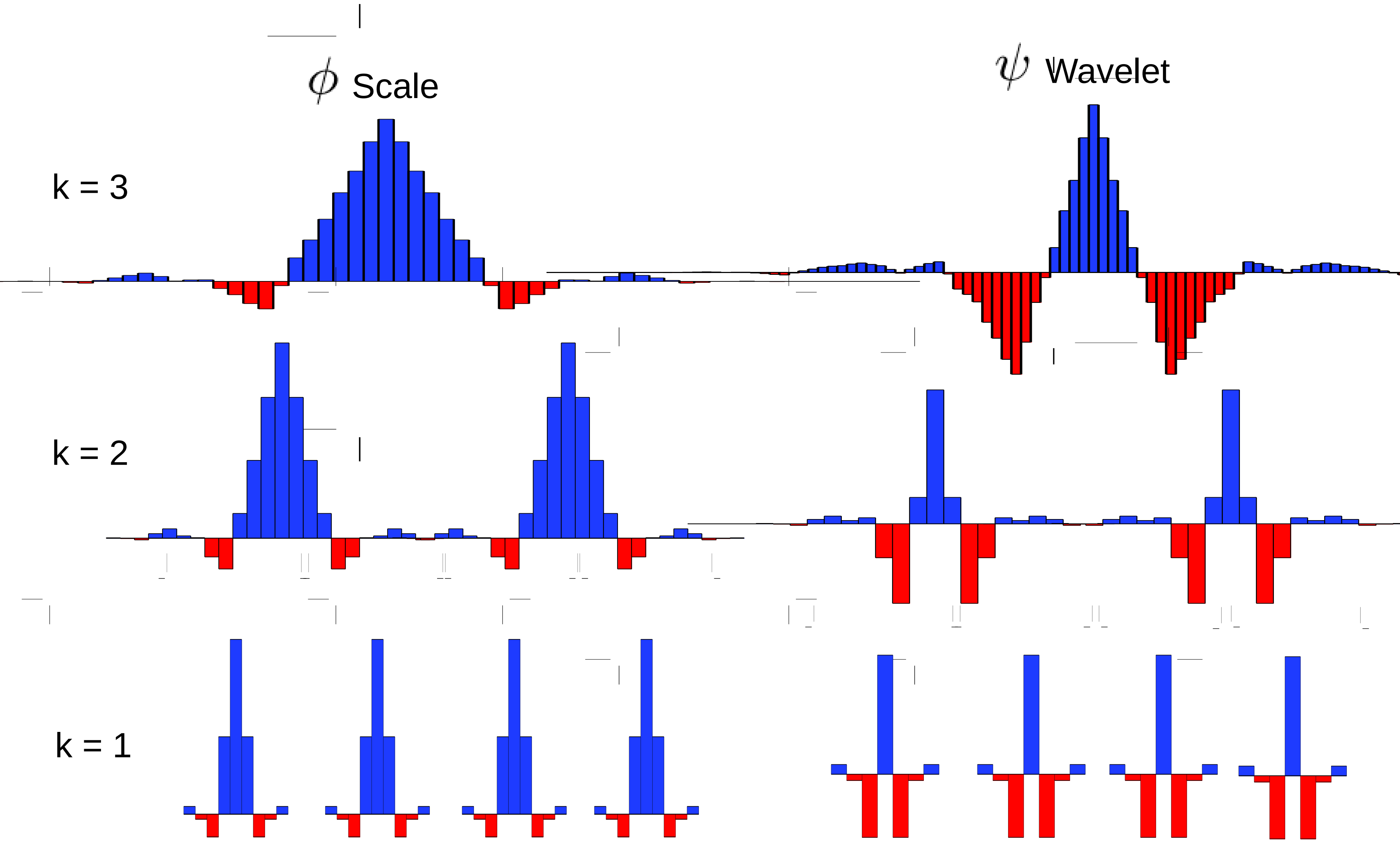
Low Pass
(smooth)

Wavelet



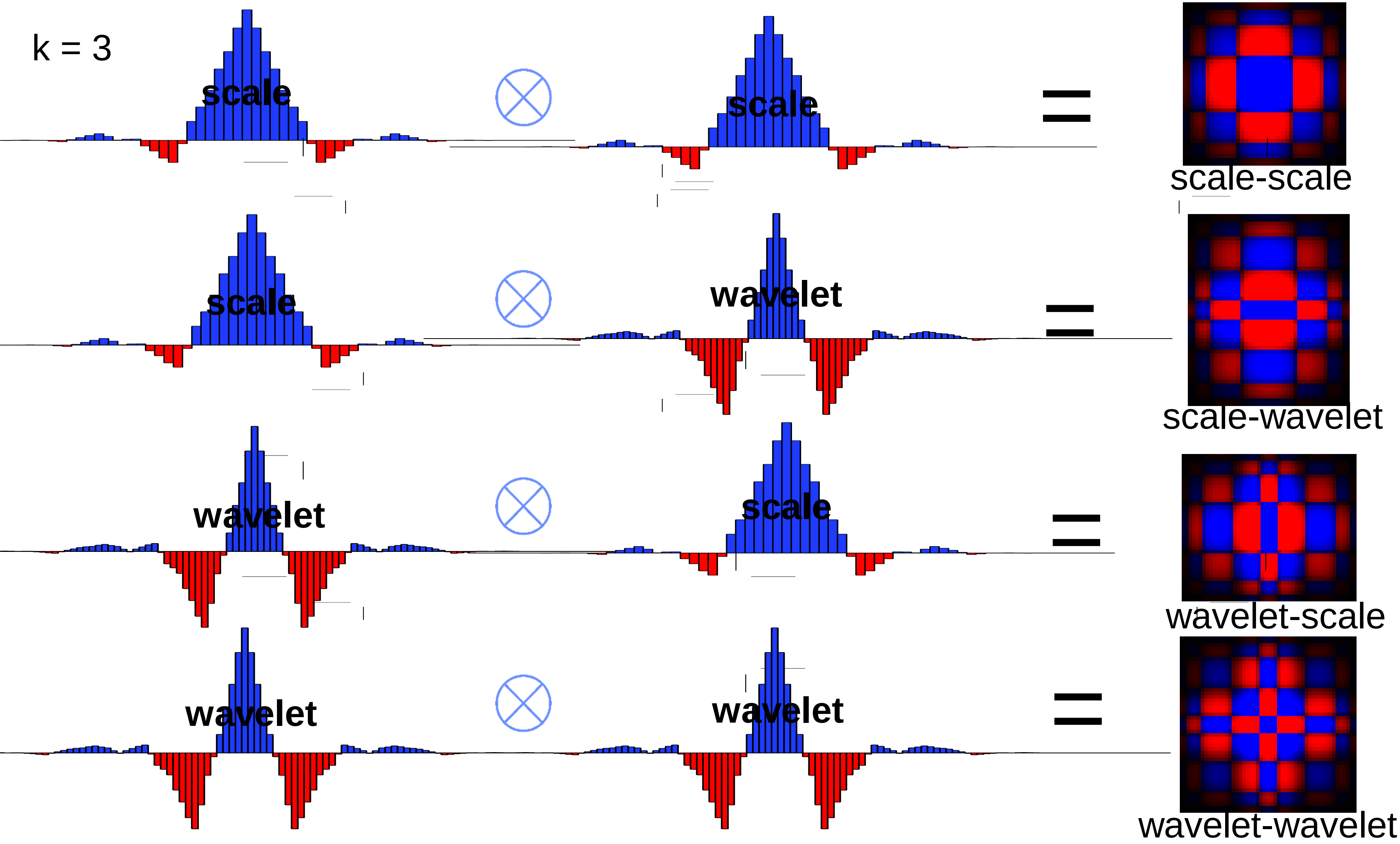
Band Pass
(edges)

Wavelet Hierarchy (1D)



1D Tensor Product → 2D Basis

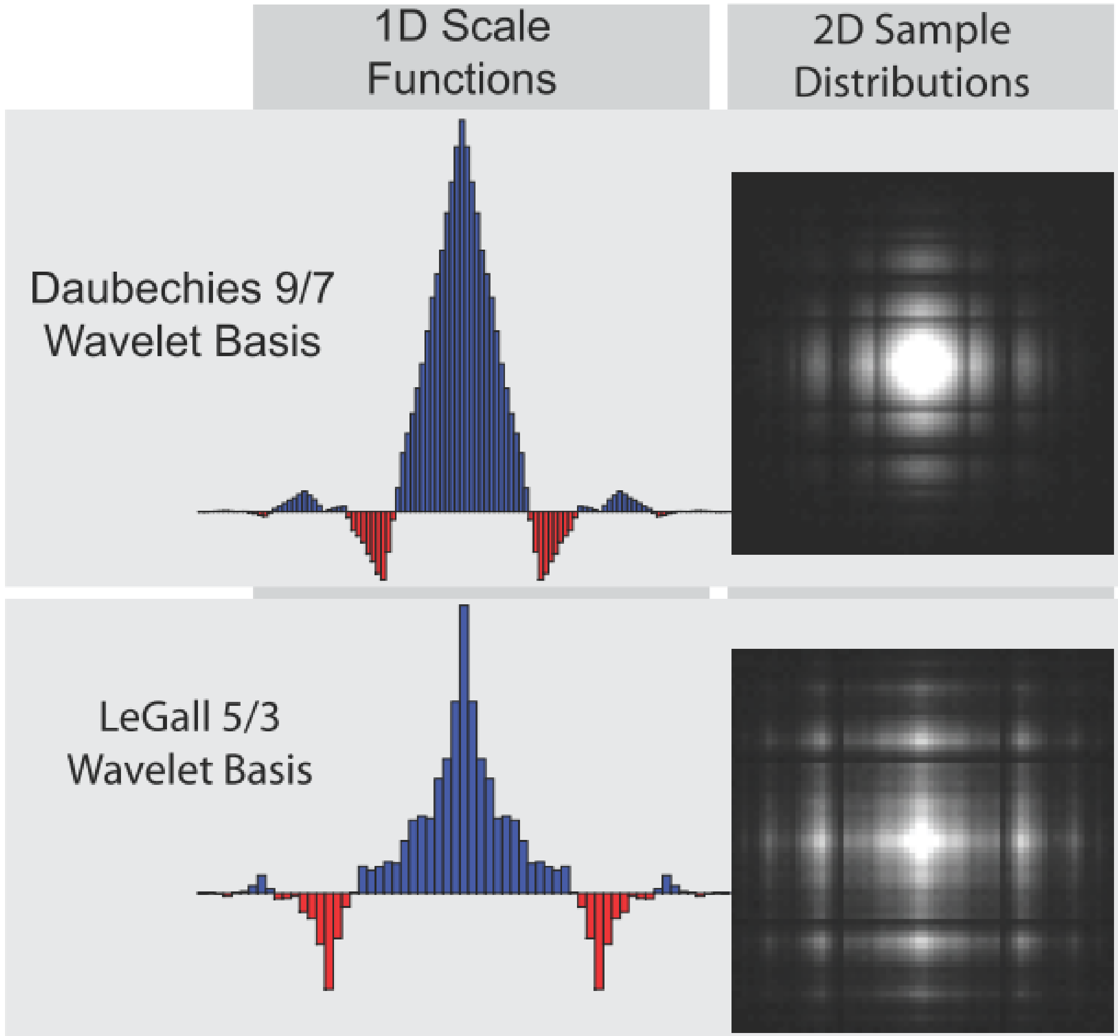
$$\Phi = \phi \otimes \phi^T, \Psi^0 = \phi \otimes \psi^T, \Psi^1 = \psi \otimes \phi^T, \text{ and } \Psi^2 = \psi \otimes \psi^T.$$



Wavelet used

k = 4

JPEG
2000
compression



Wavelets Basis (& DWT)

Wavelets (Multi-scale basis) VS Pixel Basis

Multi scale: coefficient/wavelet expressed in different scale

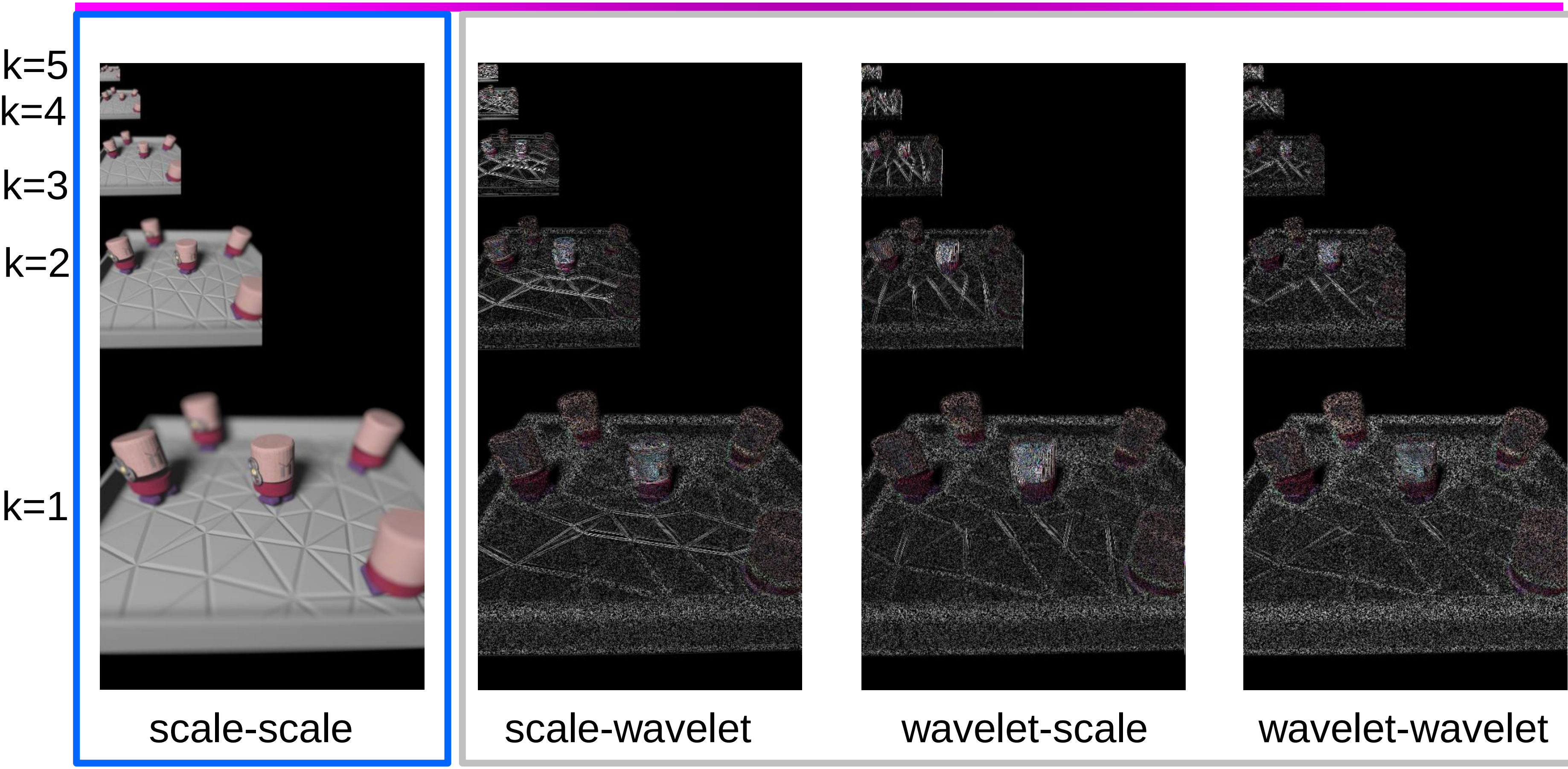
Discret Wavelet Transform (DWT):

1) Pixels ⇨ Wavelets coefficient (Analysis)

$$W_{k,ij}^{\alpha} = \langle B, \Psi_{k,ij}^{\alpha} \rangle = \int\int B \cdot \Psi_{k,ij}^{\alpha} dxdy.$$

2) Wavelets ⇨ Pixels (Synthesis)

Wavelet Hierarchy



Smooth regions

Edges

$$\Phi = \phi \otimes \phi^T, \Psi^0 = \phi \otimes \psi^T, \Psi^1 = \psi \otimes \phi^T, \text{ and } \Psi^2 = \psi \otimes \psi^T.$$

III) Algorithm Outline

0) Start: 4 Samples per Pixel (skipped)

1) Adaptive Sampling

2) Reconstruction

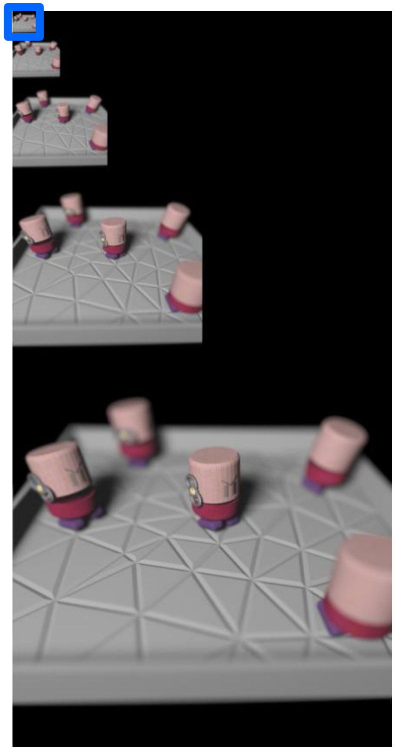
1) Adaptive Sampling

- Insert all scale coefficients into a priority queue $S_{k,ij} = \langle B, \Phi_{k,ij} \rangle = \int\int B \cdot \Phi_{k,ij} dxdy,$
- While more samples:
  - Send samples to highest priority coefficient
  - Update priority queue

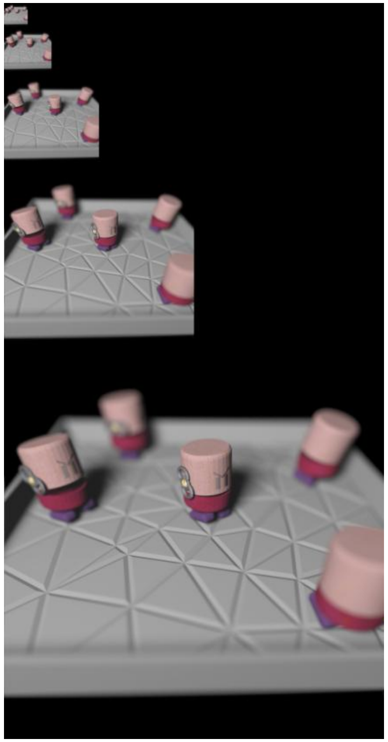
The problem:

- How to compute priority for each scale coefficient?

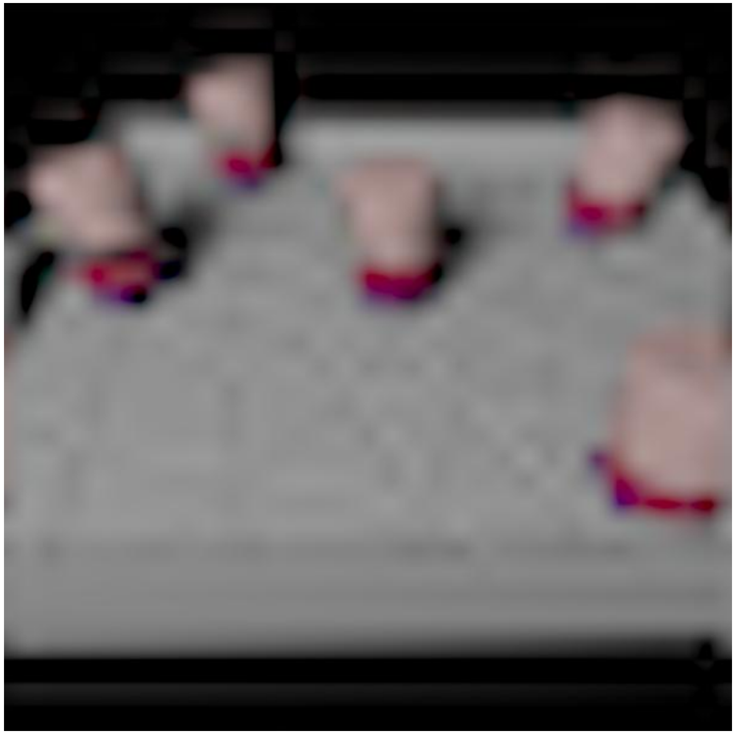
KAIST



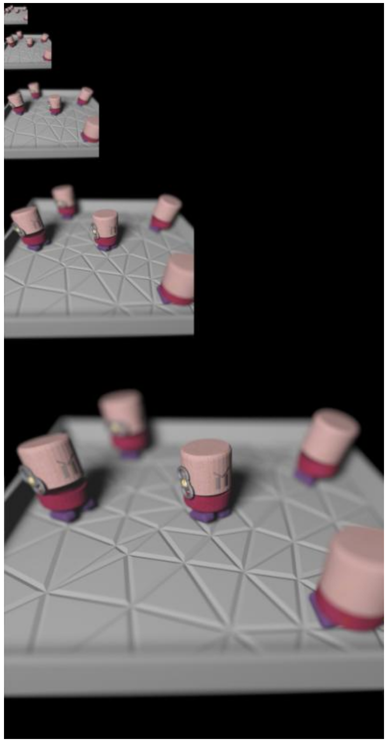
scale-scale



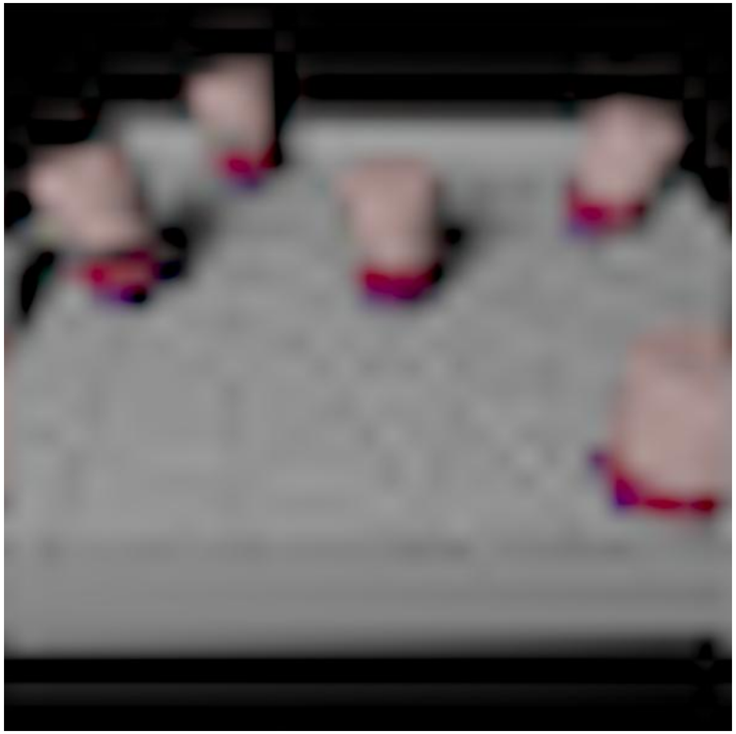
scale-scale



Wavelet synthesis is smoothing

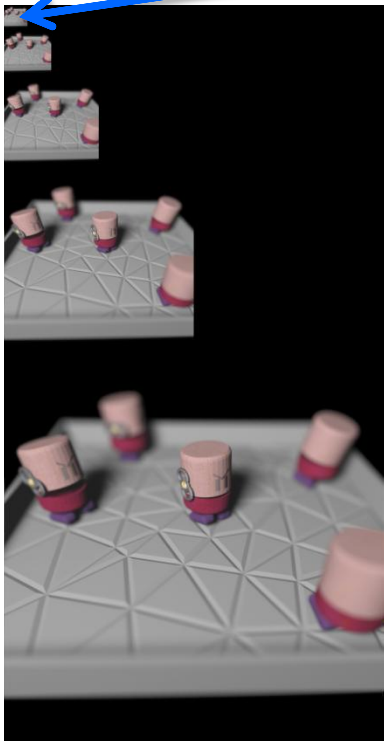


scale-scale

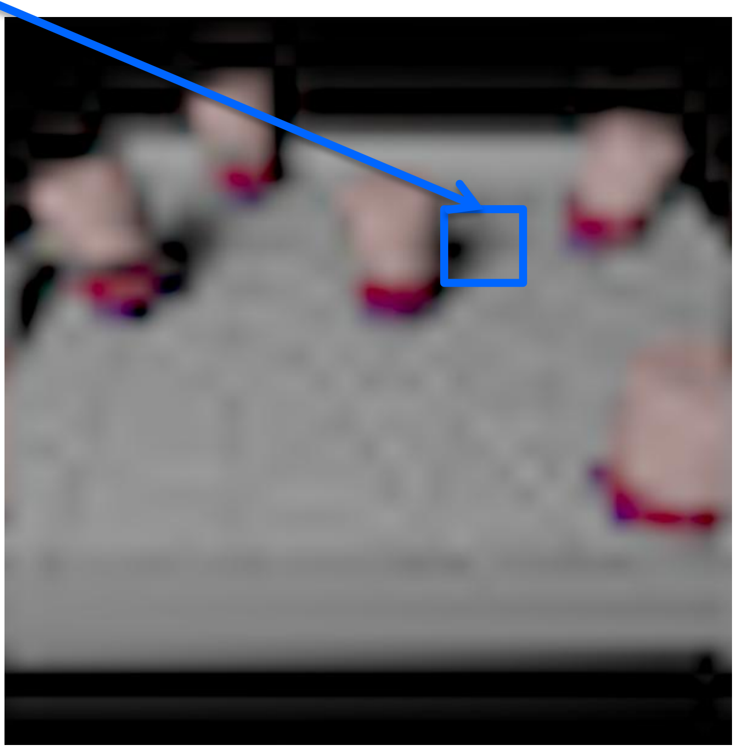


Coarse scale captures smoothness

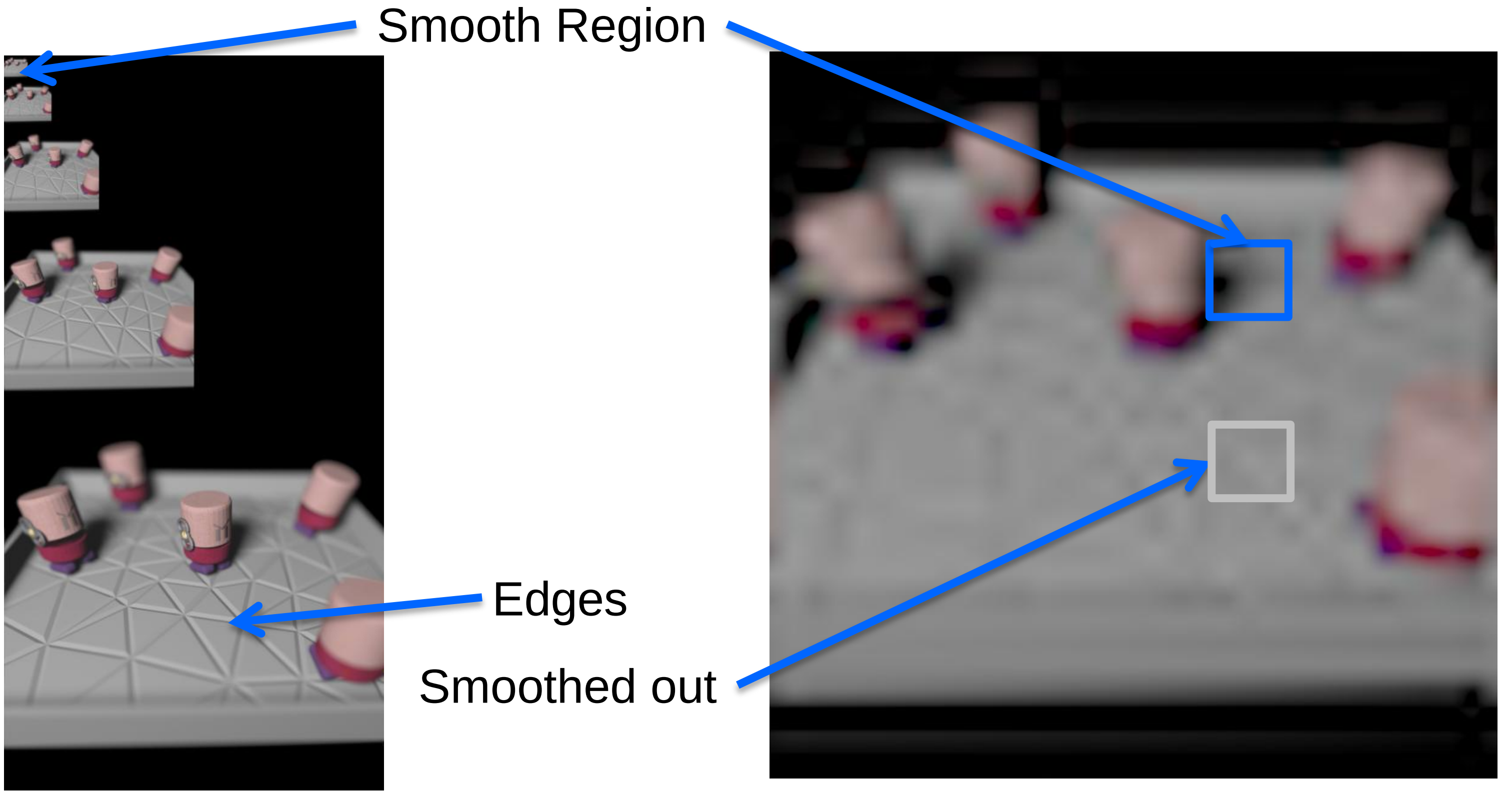
Smooth Region



scale-scale



Edges are in the fine scale



scale-scale

Adaptive Sampling

Goals:

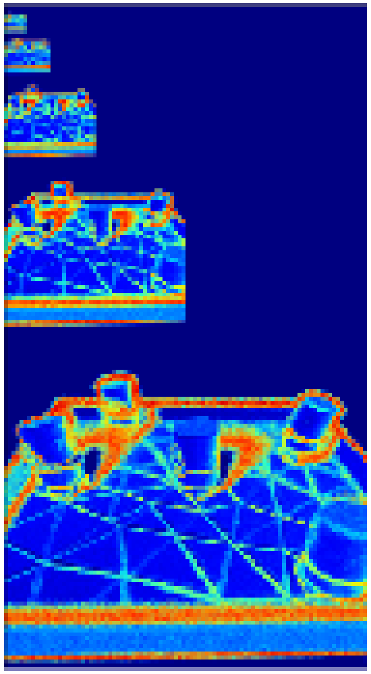
- In smooth regions, more samples to coarse coefficients
- Near edges, more samples to fine coefficients

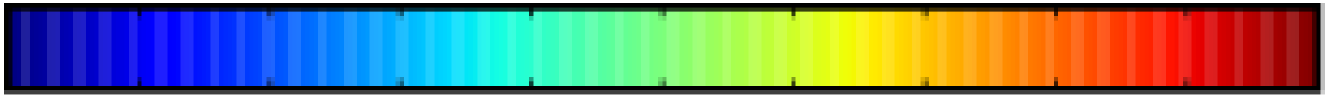
Solution:

- Compute priority based on variance and smoothness

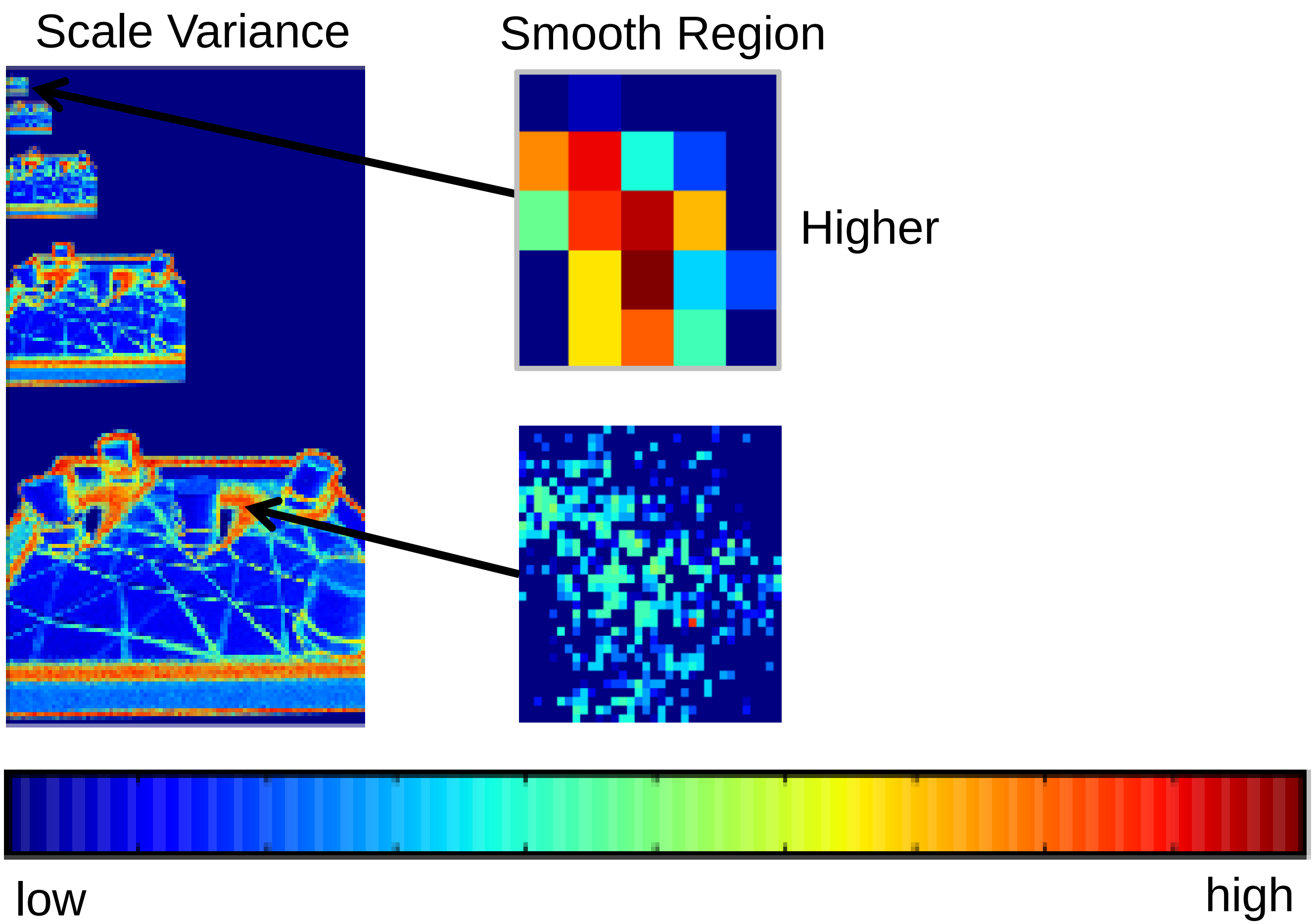
Start with Scale Coefficients' Variance

Scale Variance

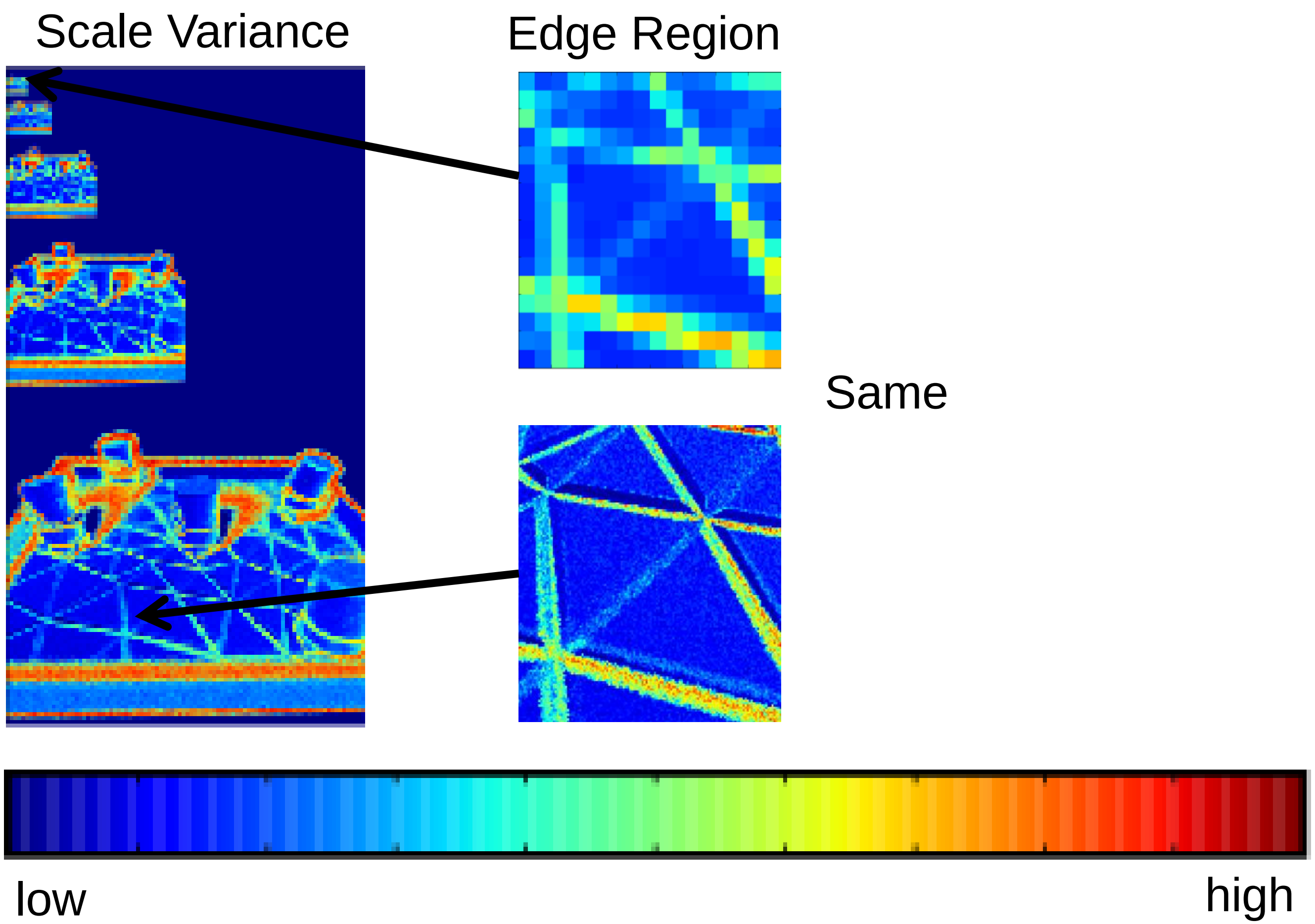




Smooth variance grows fine to coarse

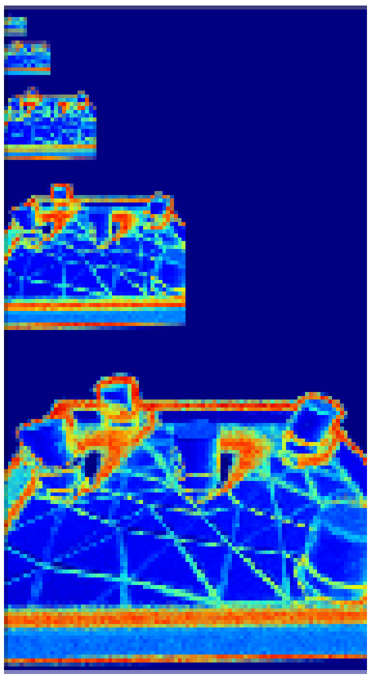


Edge variance stays the same

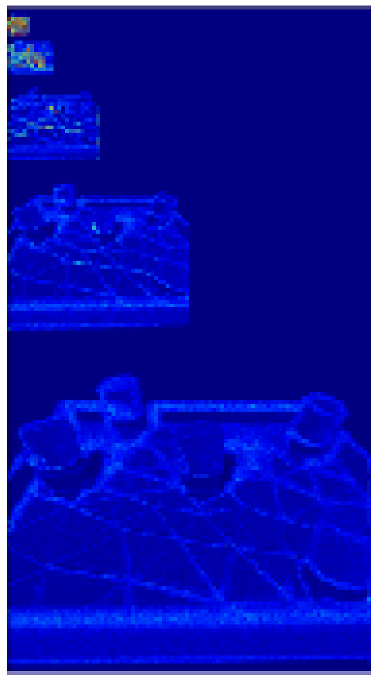


Squared wavelet magnitudes

Scale Variance

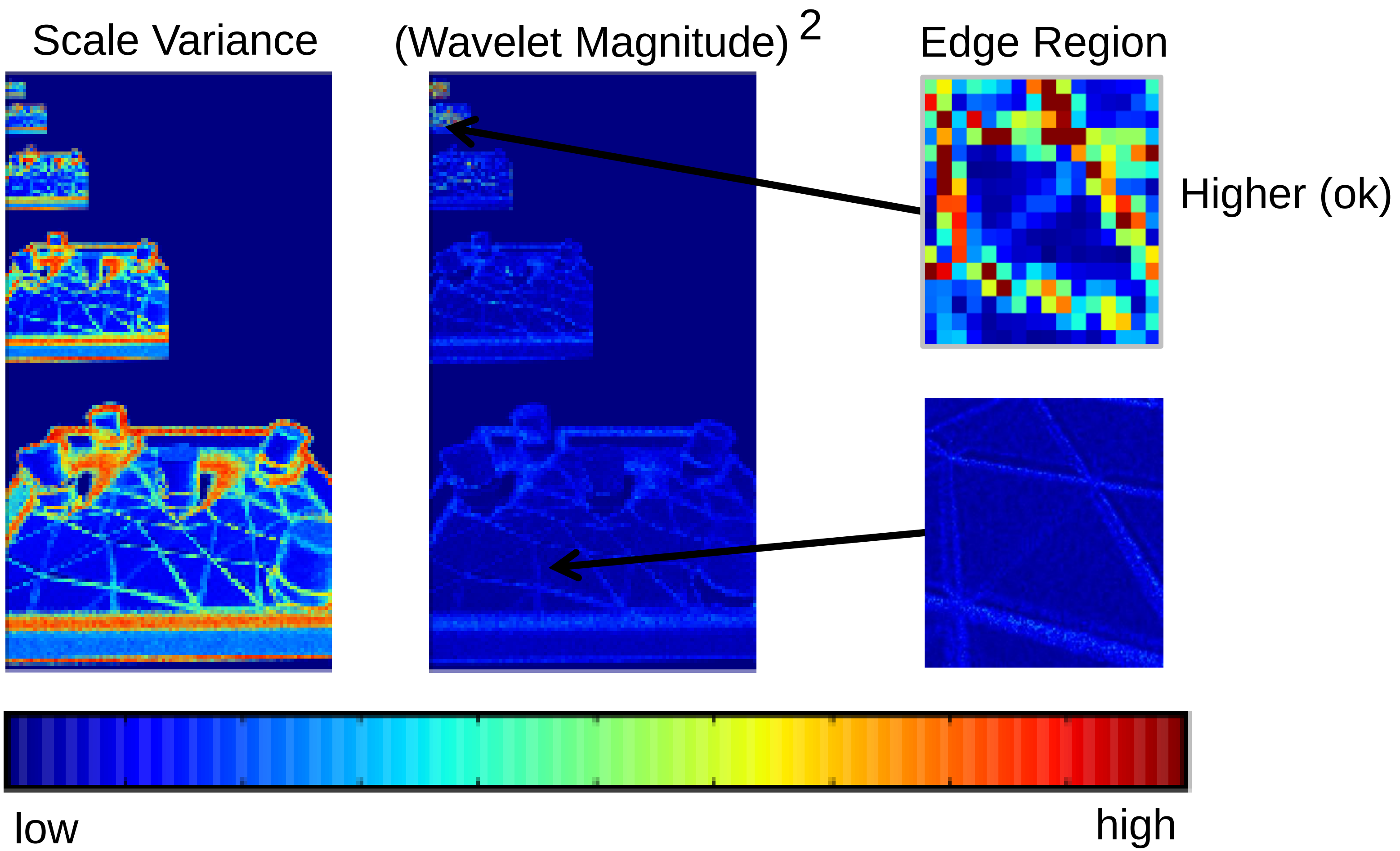


(Wavelet Magnitude) ^{2}

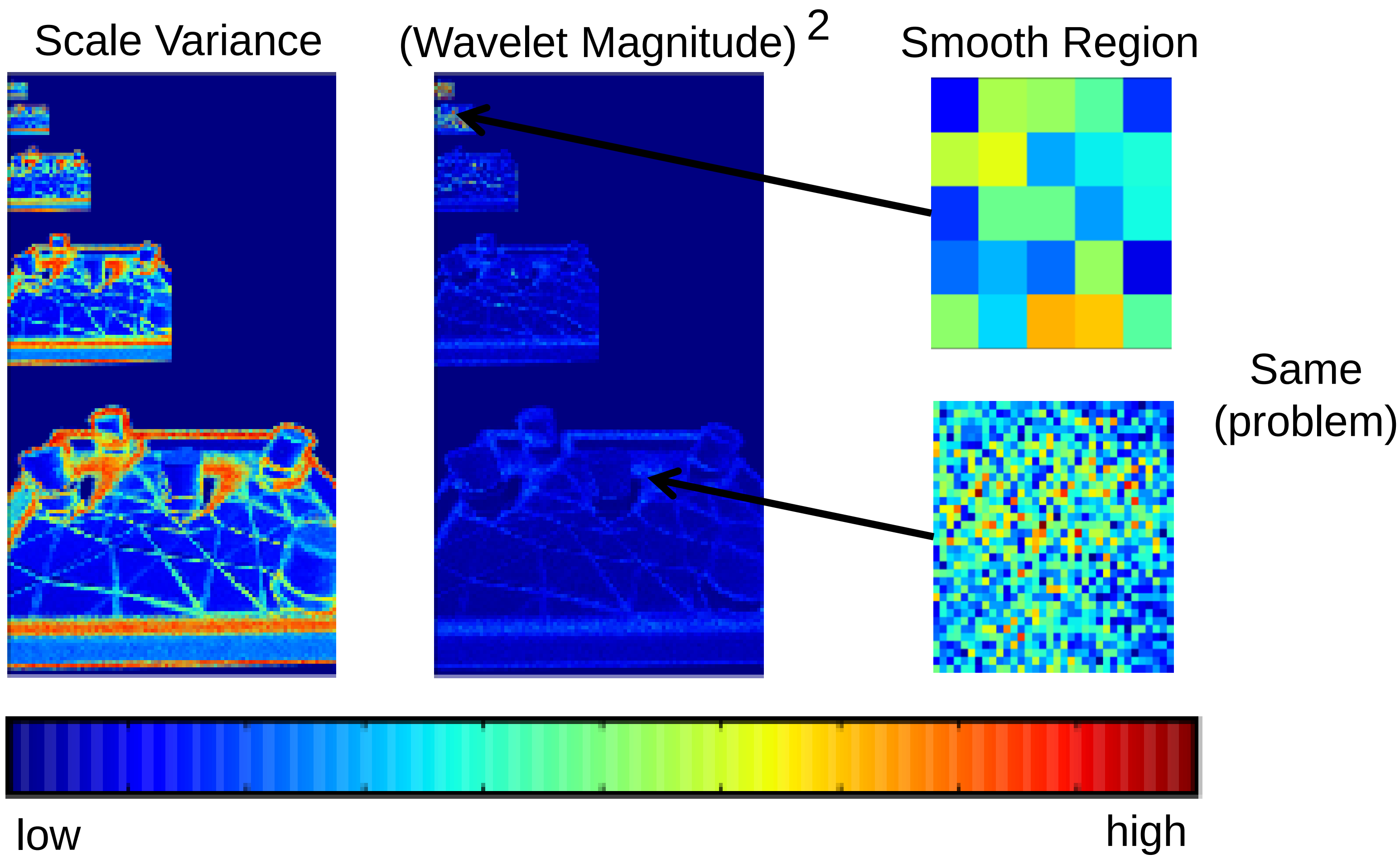


low high

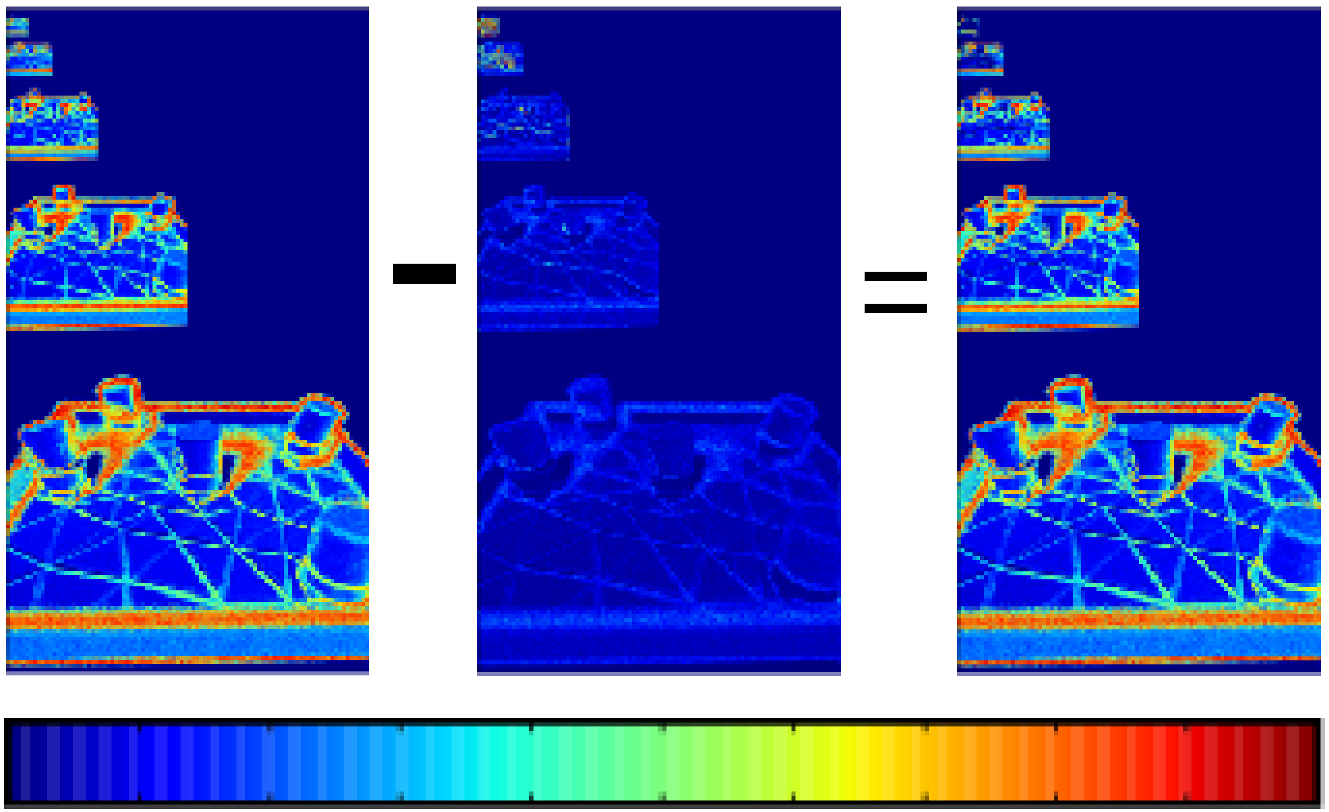
Edge wavelets grow fine to coarse



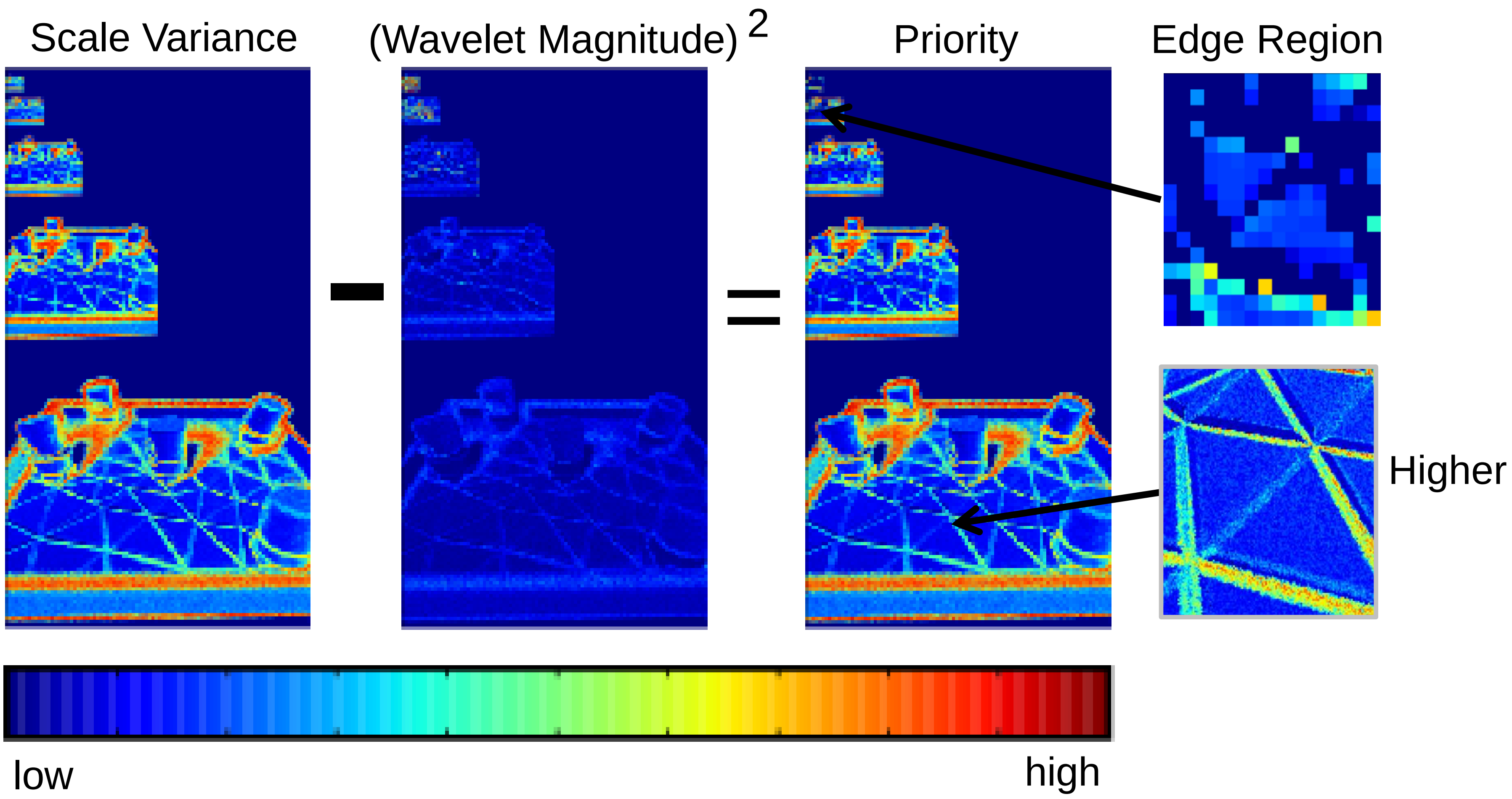
Smooth wavelets stay the same



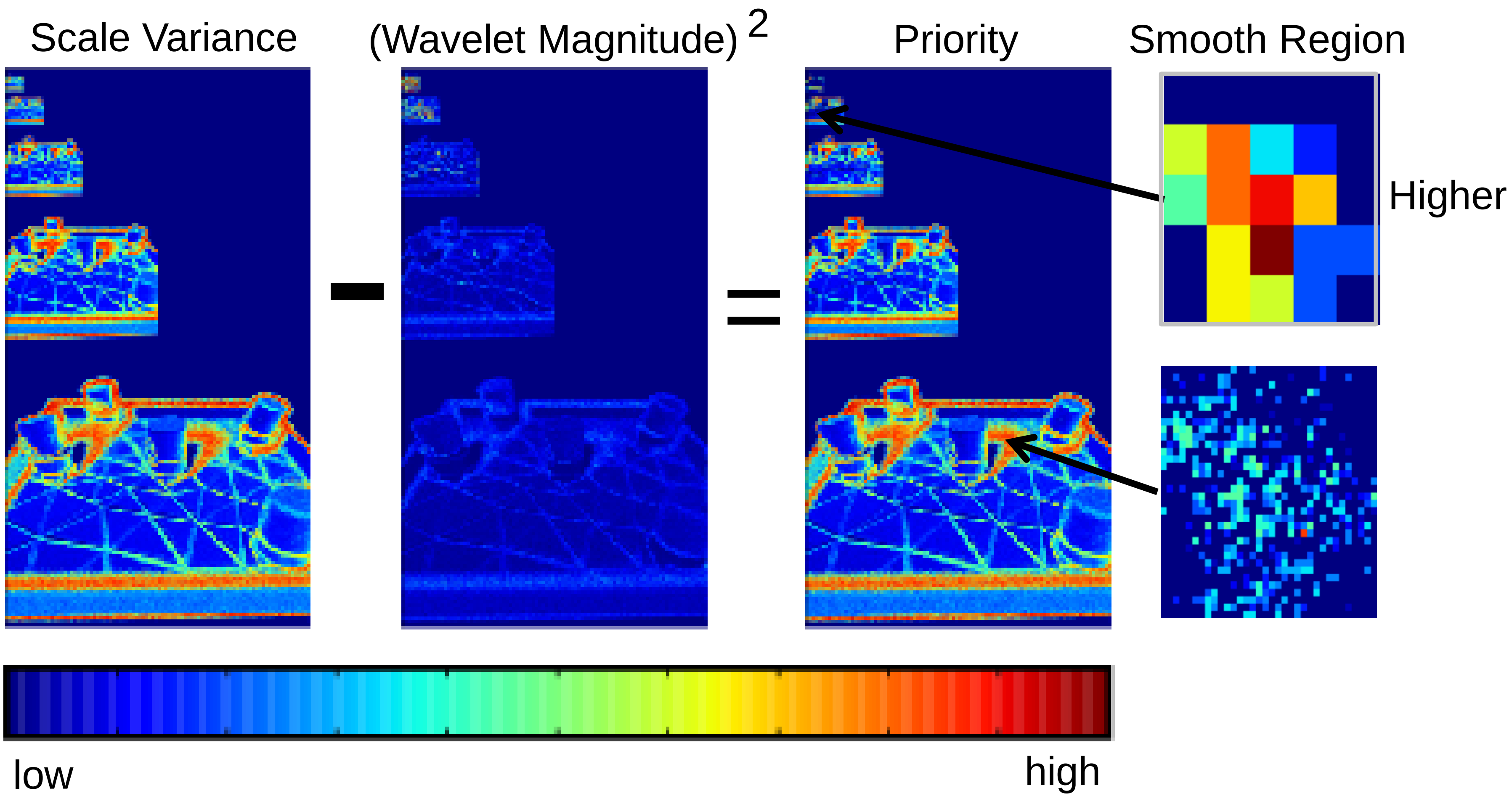
Priority equals the difference



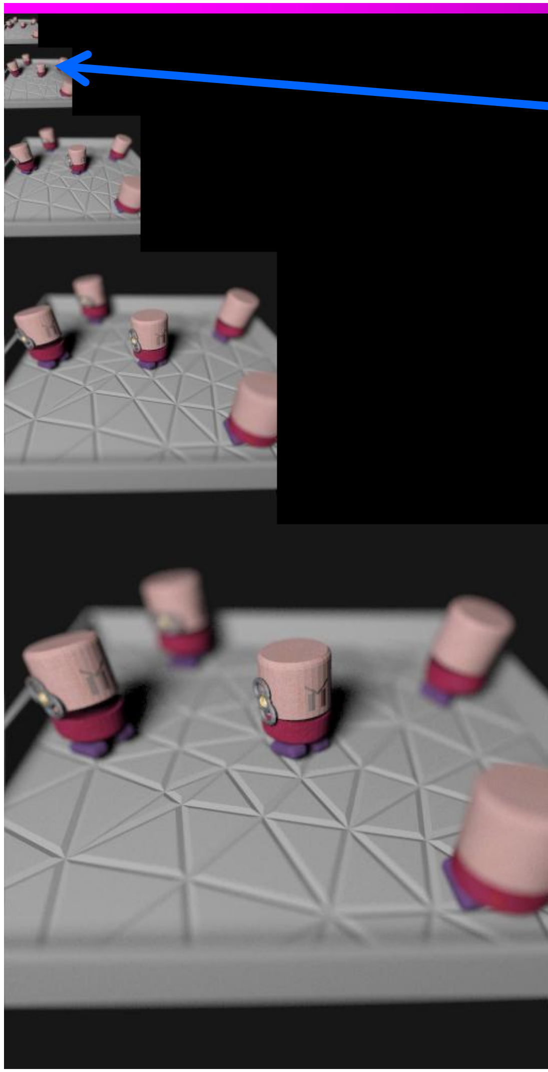
Edges: Higher priority at fine scales



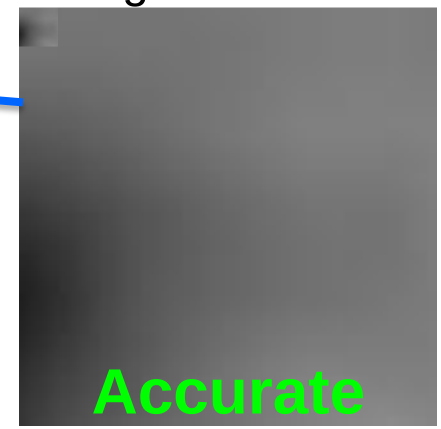
Smooth: Higher priority at coarse scales



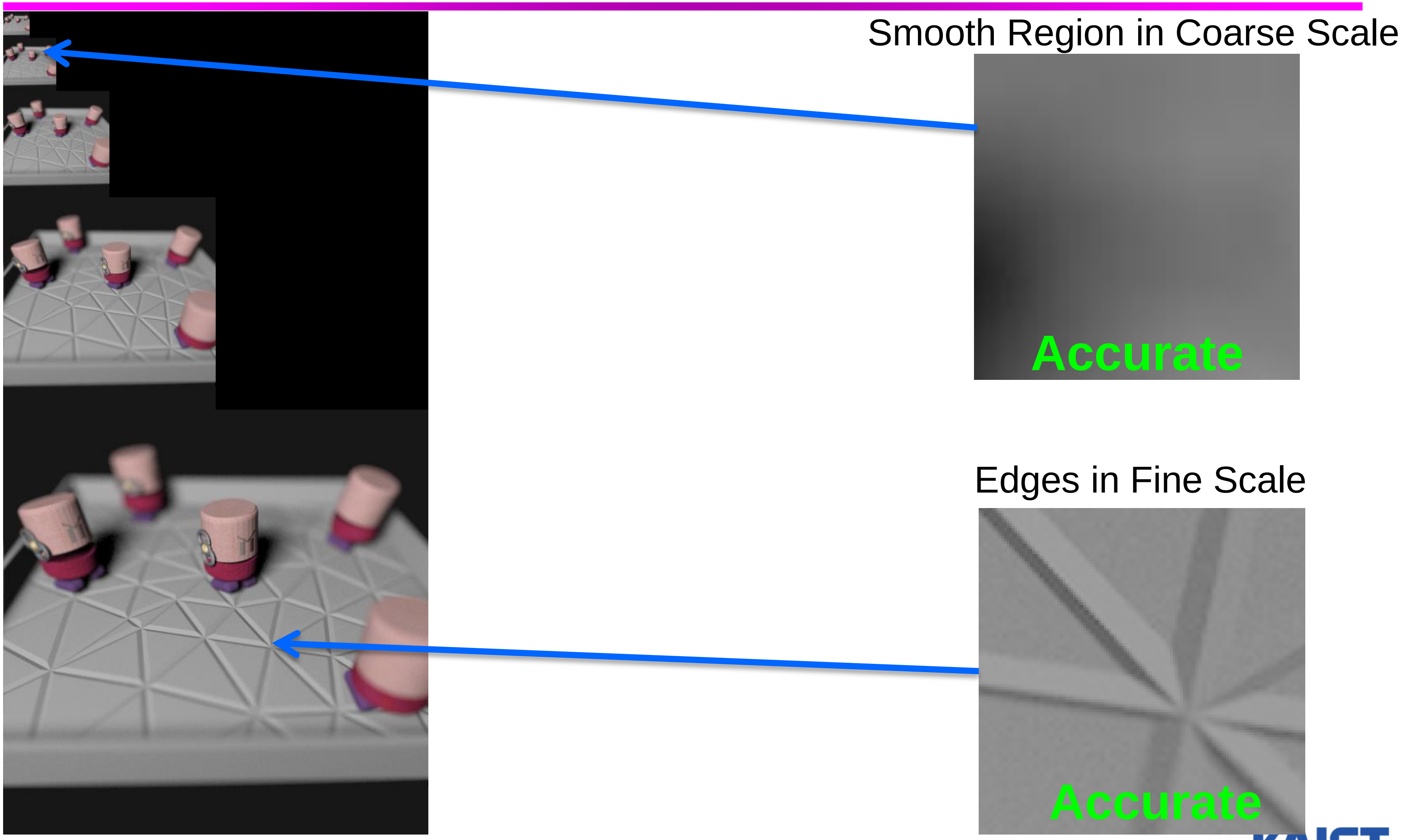
After adaptive sampling



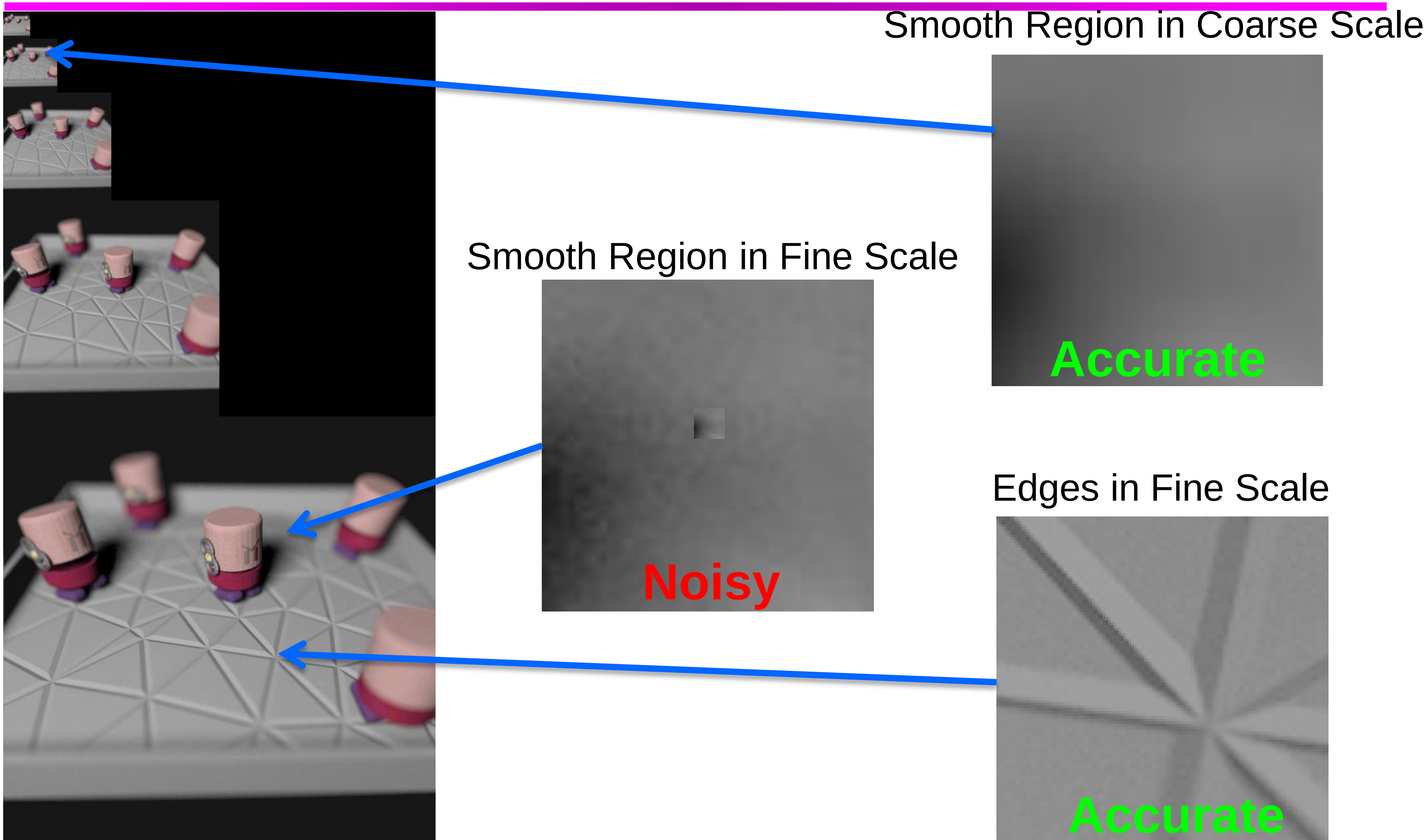
Smooth Region in Coarse Scale



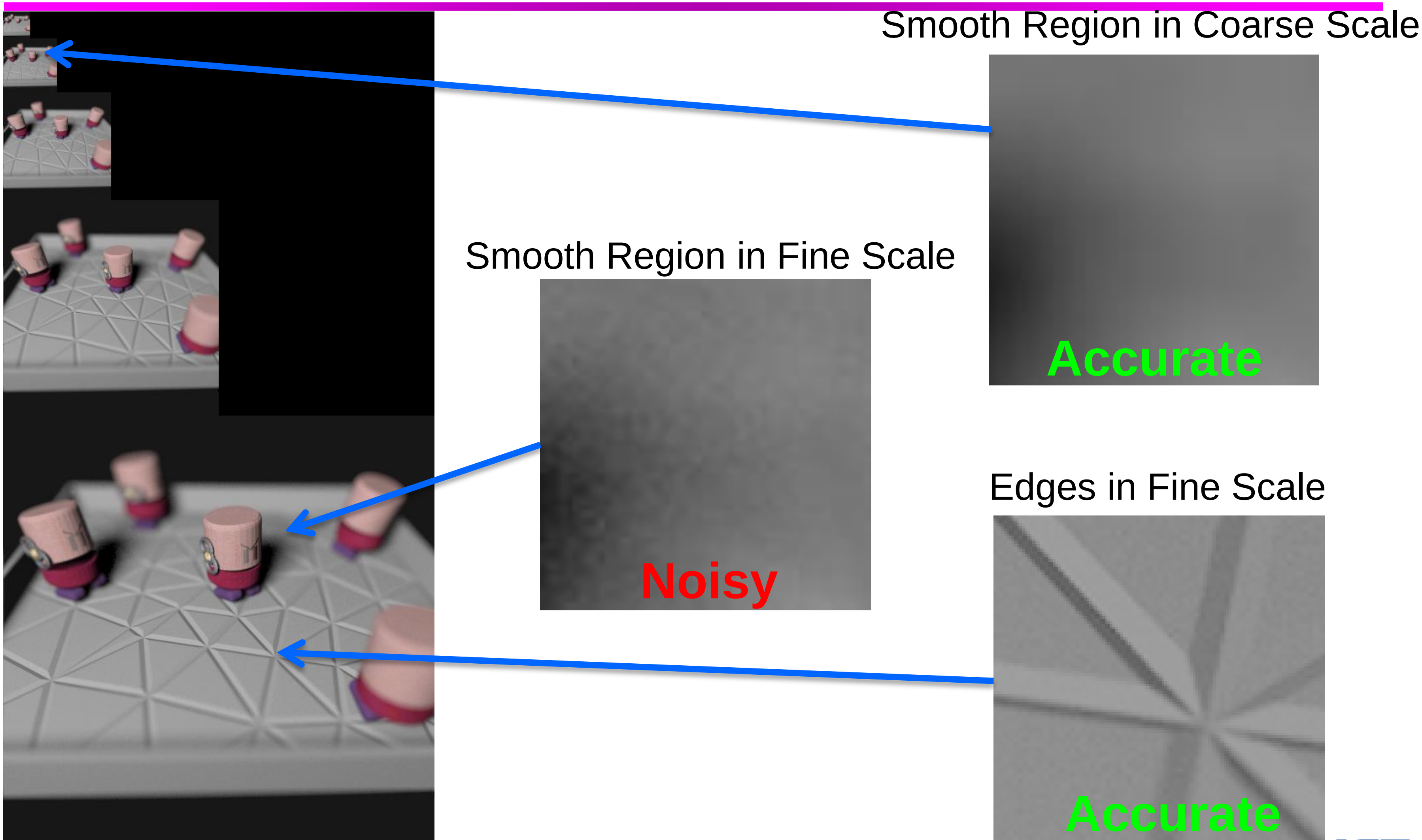
After adaptive sampling



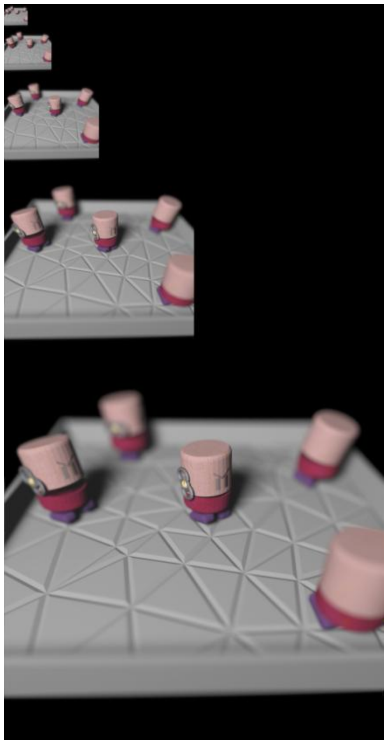
After adaptive sampling



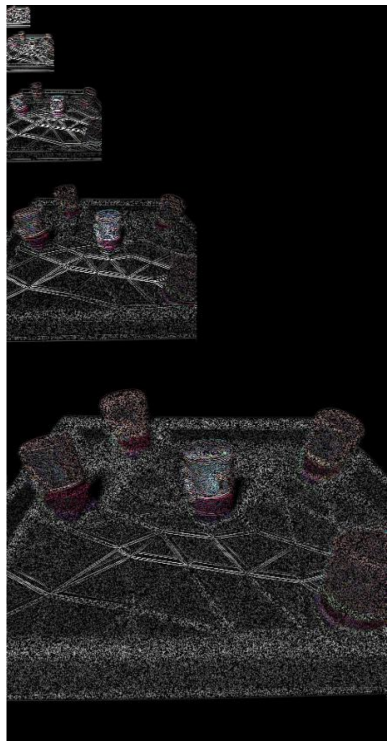
Reconstruction: smooth away fine scale noise



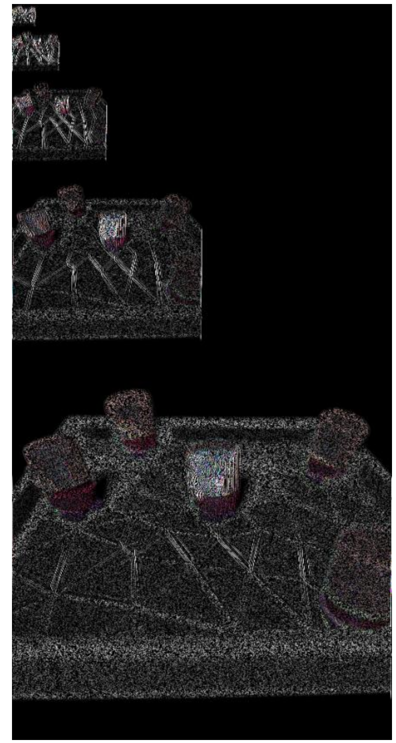
Wavelets capture edges



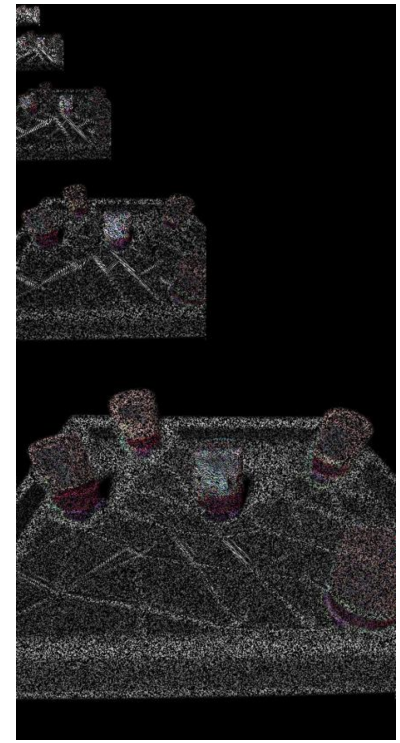
scale-scale



scale-wavelet



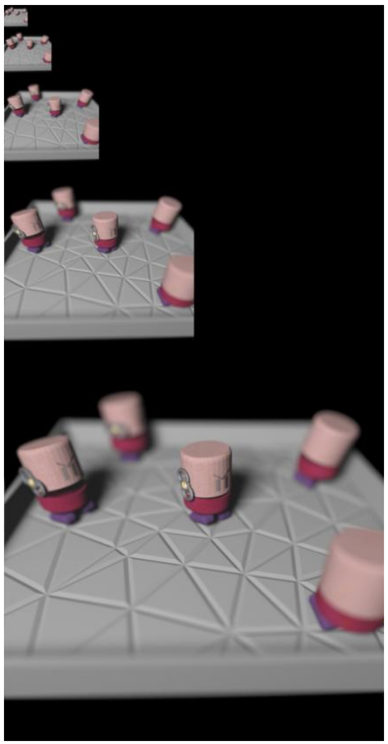
wavelet-scale

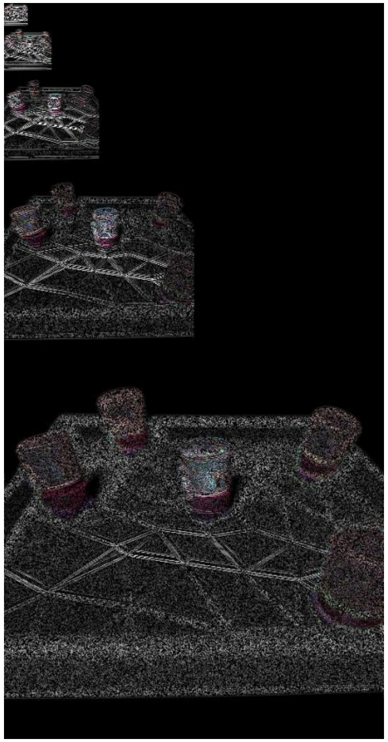


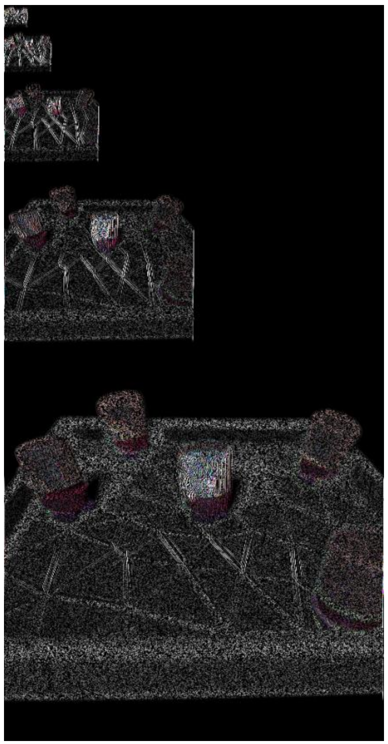
wavelet-wavelet

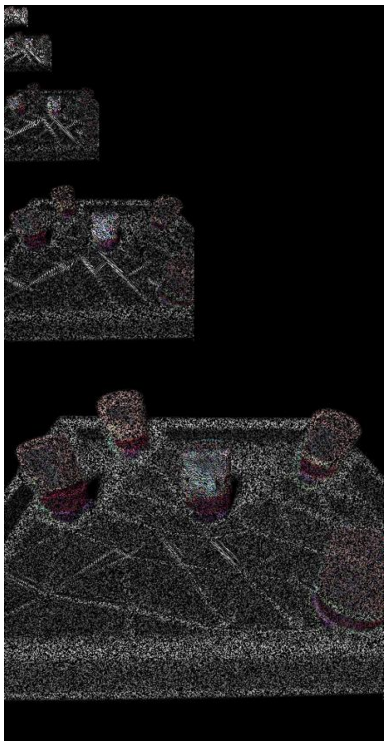
Edges

Wavelets capture edges and Noise


scale-scale


scale-wavelet


wavelet-scale


wavelet-wavelet

Edges and Noise

Algorithm Outline

0) Start: 4 Samples per Pixel

1) Adaptive Sampling

-> 2) Reconstruction

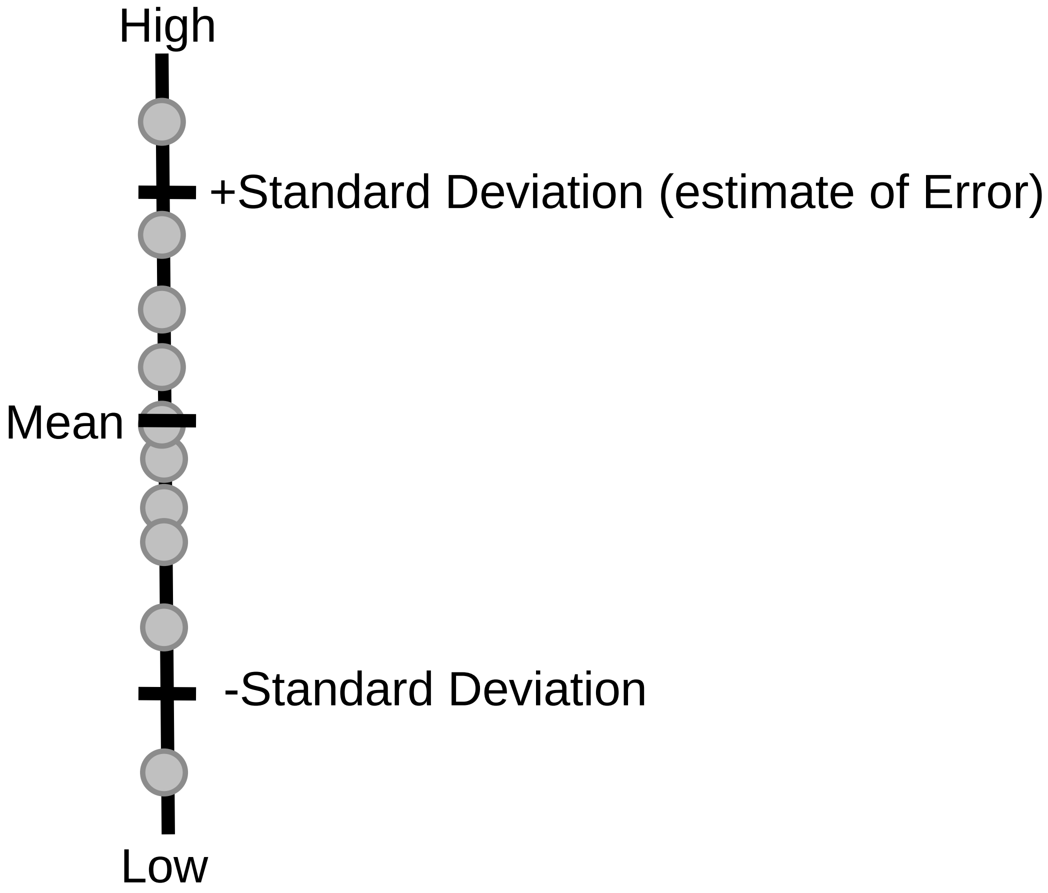
Wavelet Reconstruction

Remove noise by suppressing wavelet magnitudes

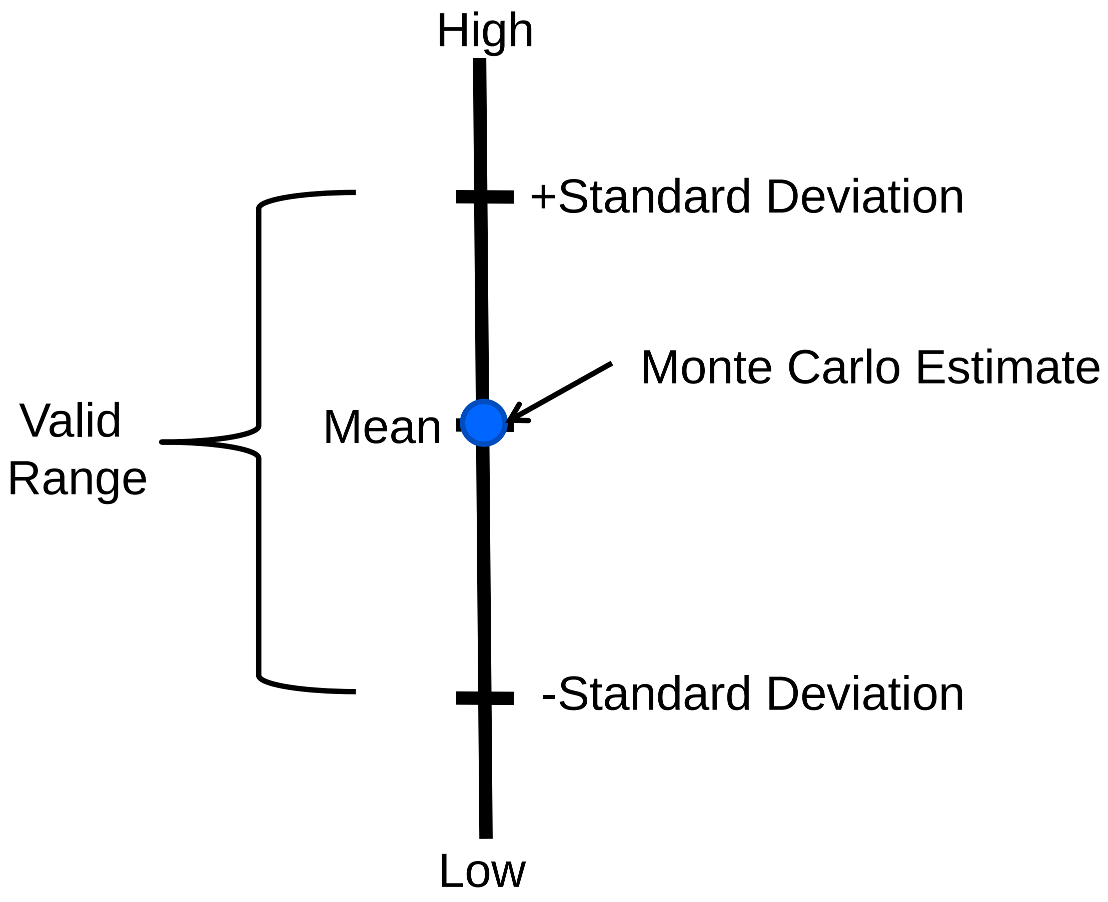
How?

- Choose smoothest image which fits samples

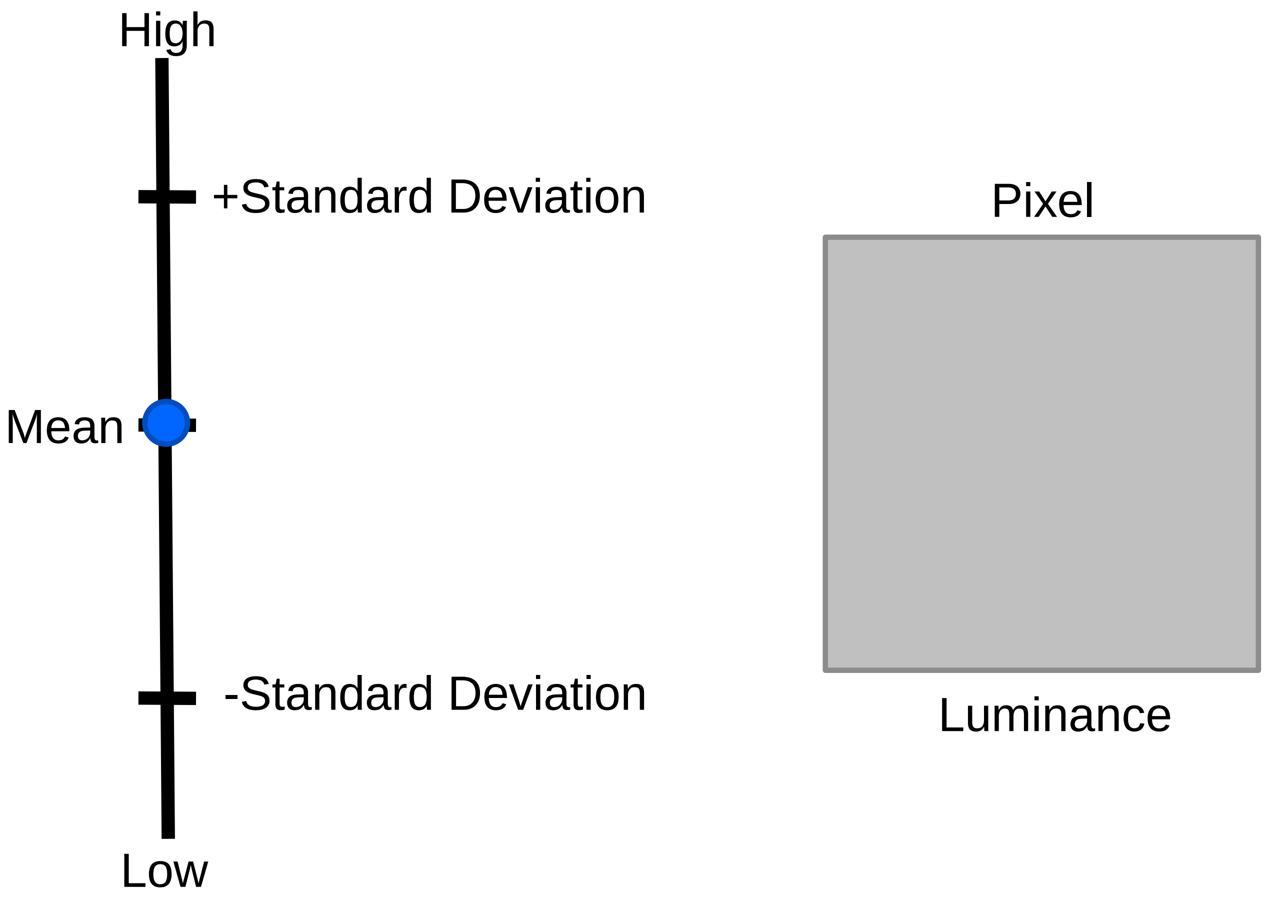
Monte Carlo: statistics



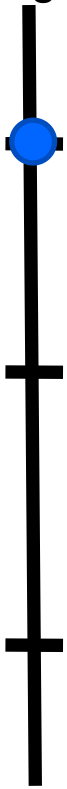
Monte Carlo: statistics



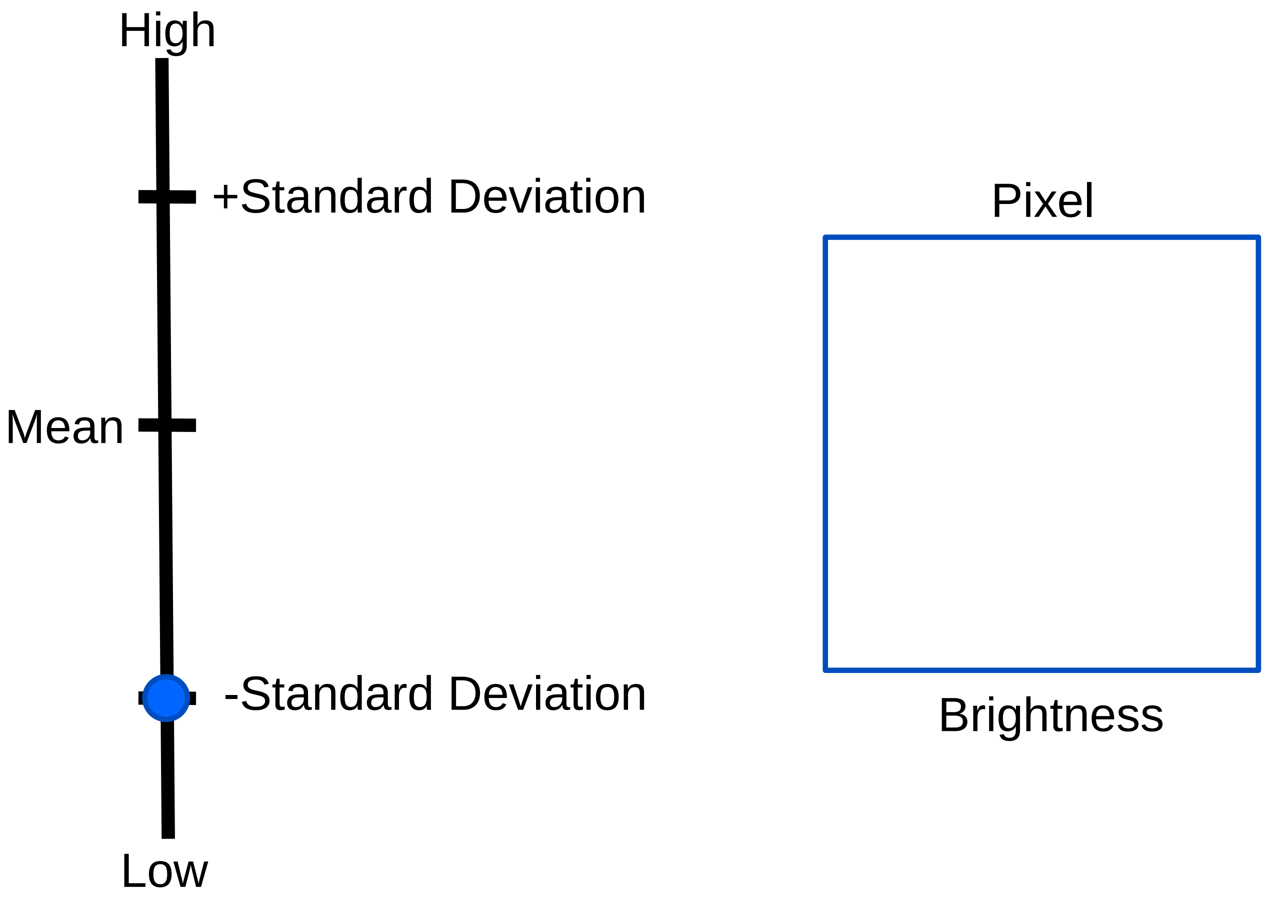
For pixel:


High
+Standard Deviation
Mean
-Standard Deviation
Low
Pixel
Luminance

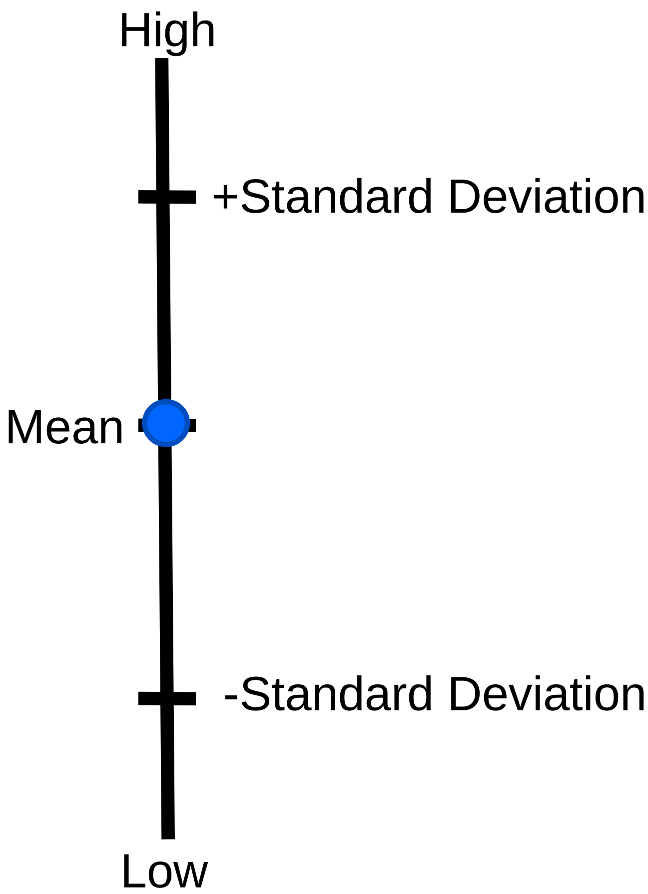
For pixel: luminance

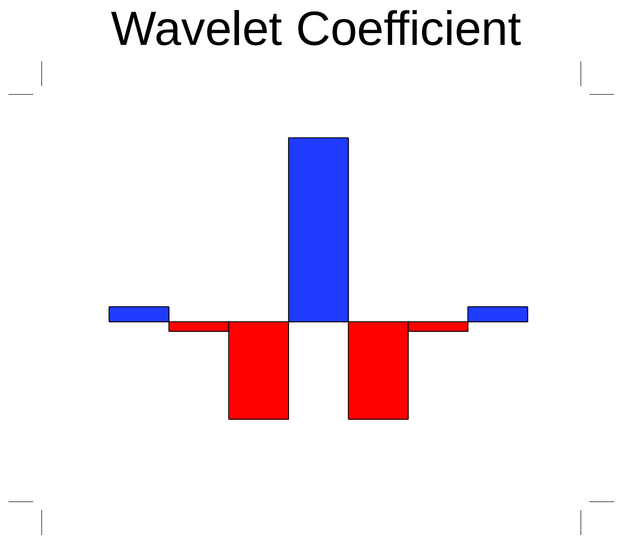


For pixel: luminance

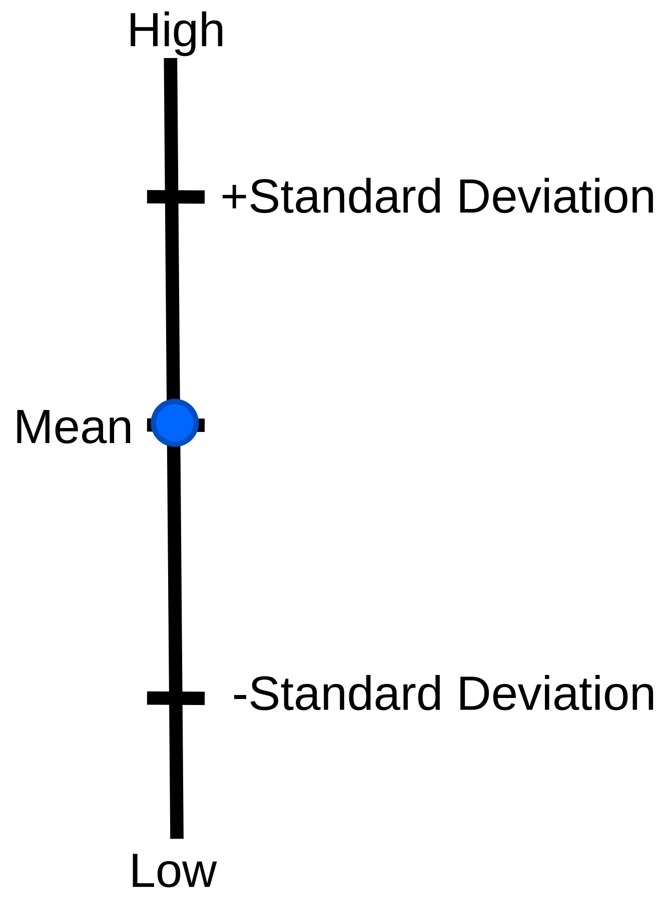


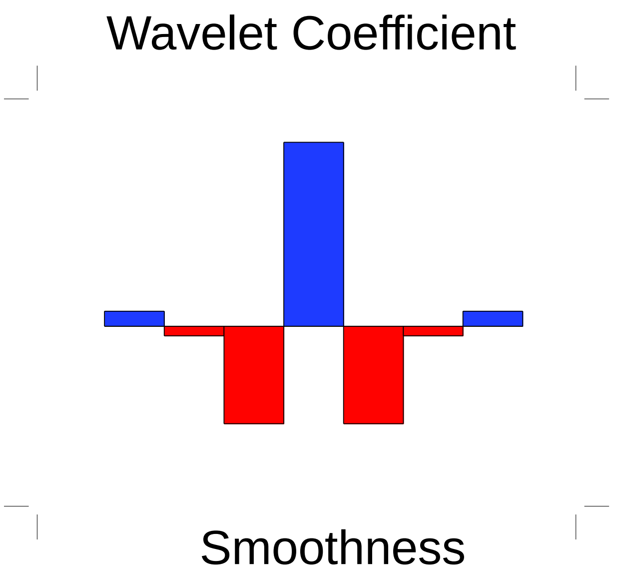
For wavelet:


High
+Standard Deviation
Mean
-Standard Deviation
Low

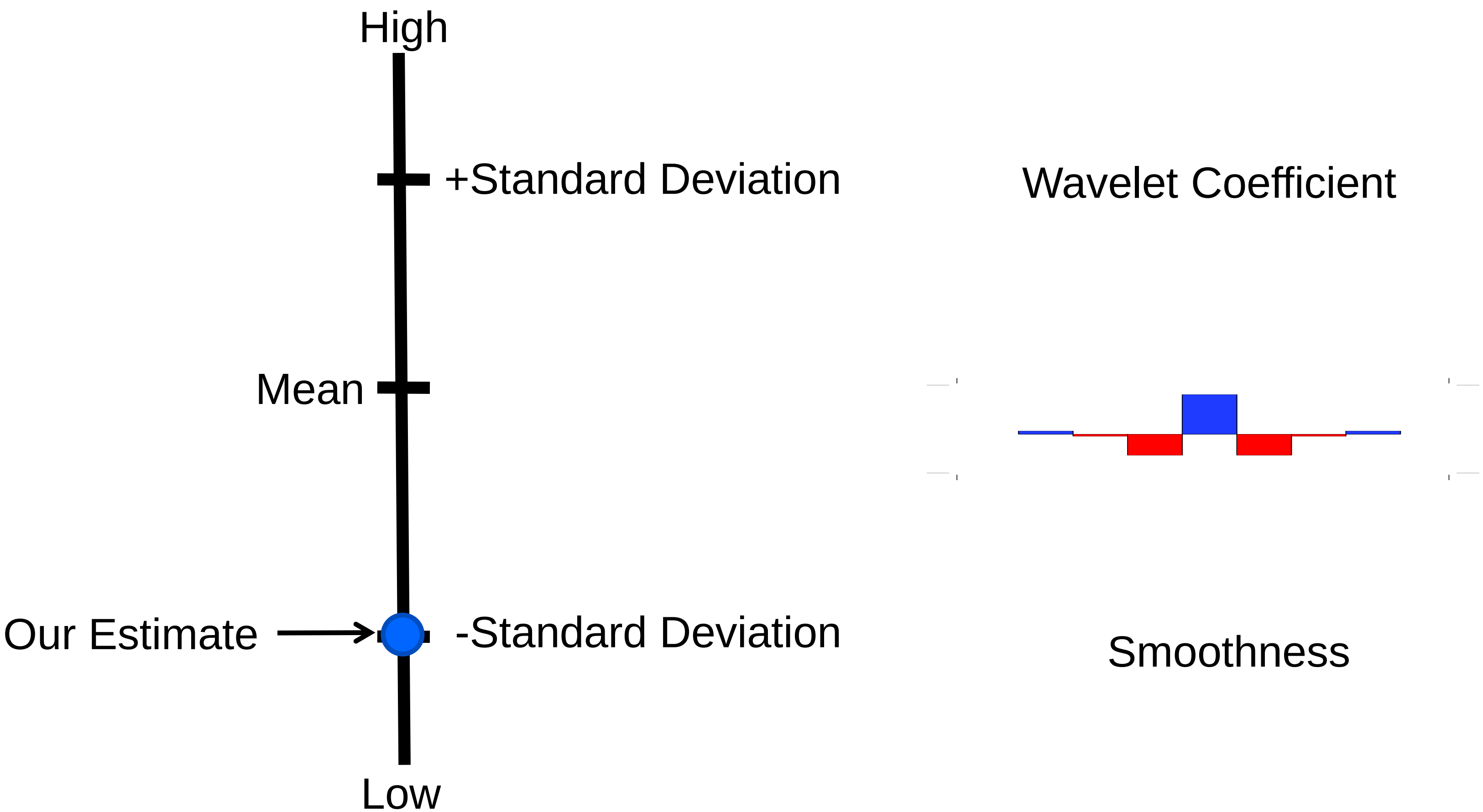

Wavelet Coefficient

For wavelet: Smoothness





Take the smoothest value



Reconstruction computation?!

standard deviation error

$$\Delta^{\alpha}_{k,ij} = \sqrt{\left\langle \sigma^2(\widetilde{B}), \left(\Psi^{\alpha}_{k,ij}\right)^2 \right\rangle}.$$

Subtracting the standard deviation from the magnitude of the wavelet coefficients gives this result:

$$W^{\alpha}_{k,ij} = \text{sign}(\widetilde{W}^{\alpha}_{k,ij}) \cdot \max\left(0, |\widetilde{W}^{\alpha}_{k,ij}| - c_s \cdot \Delta^{\alpha}_{k,ij}\right), \qquad (11)$$

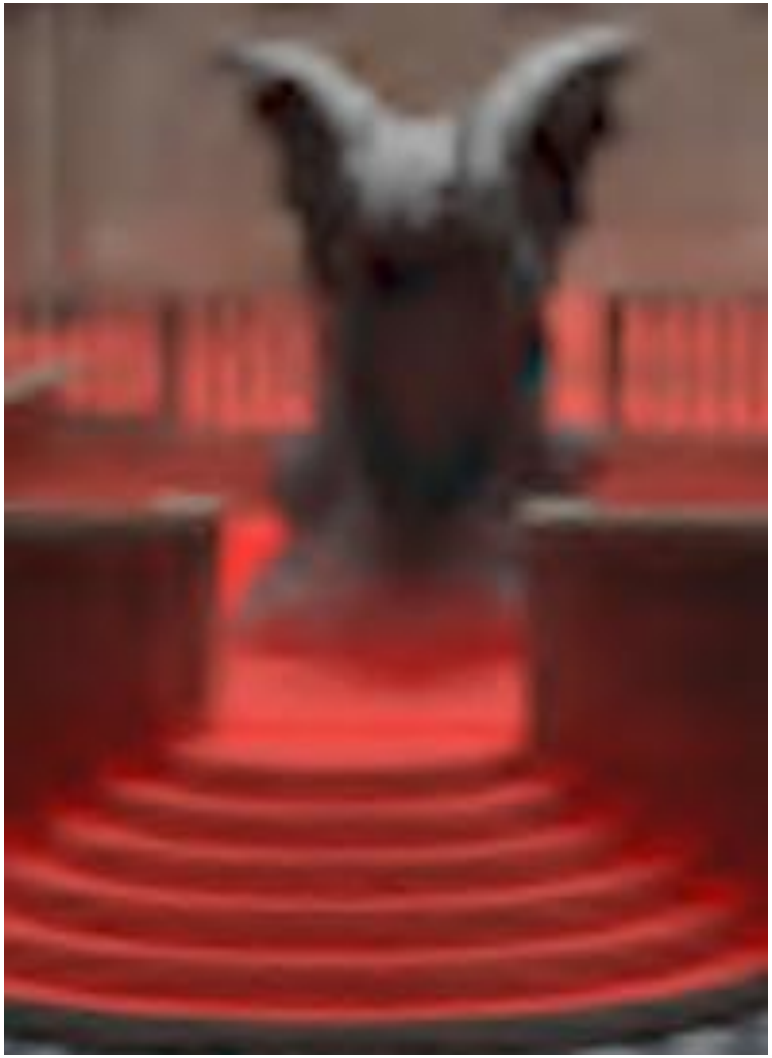
where $\widetilde{W}$ are the wavelet coefficients from the pixel means,

c_s (the smoothing constant) is a user-supplied constant

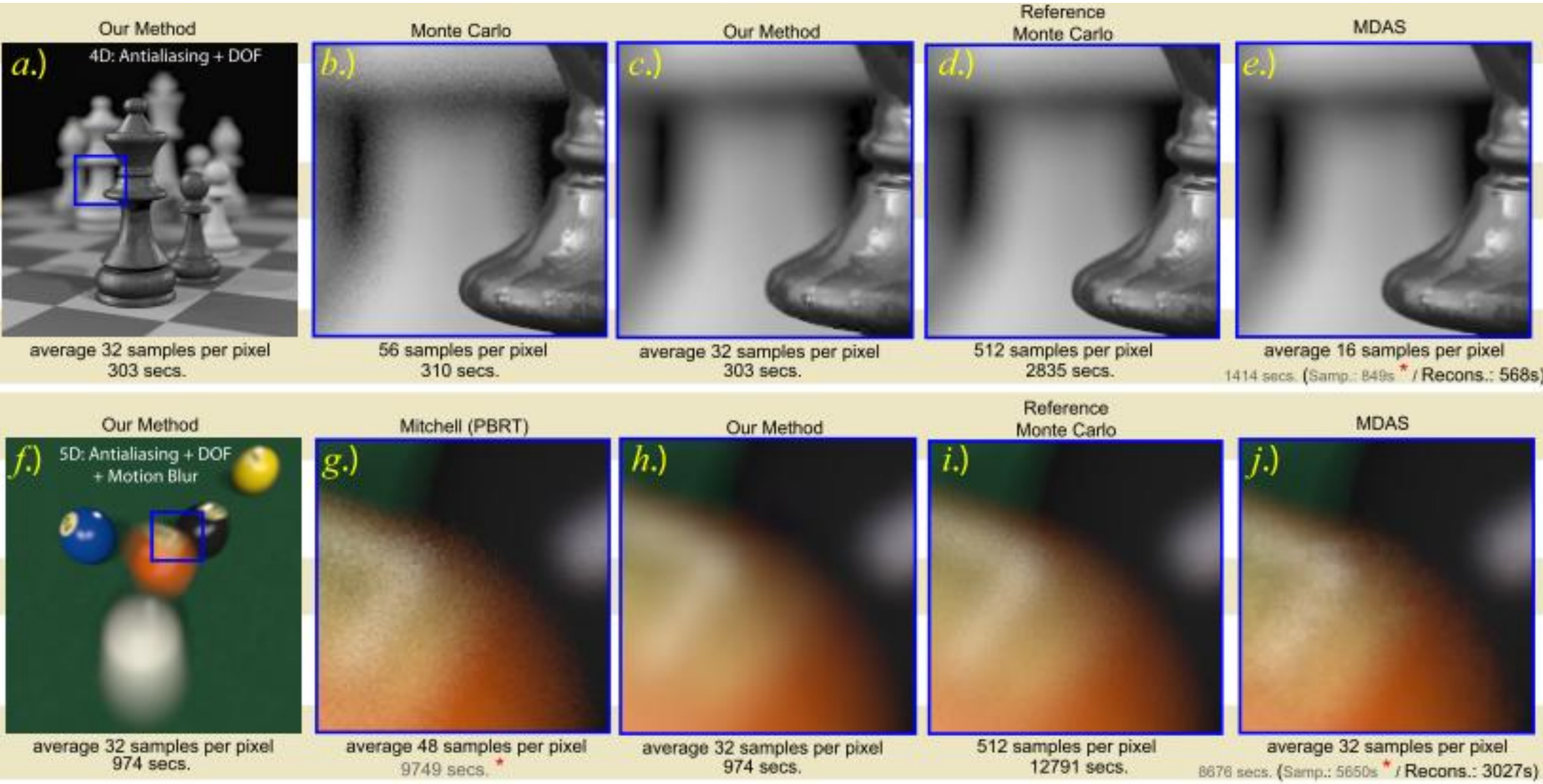
KAIST

After Sampling

After Reconstruction

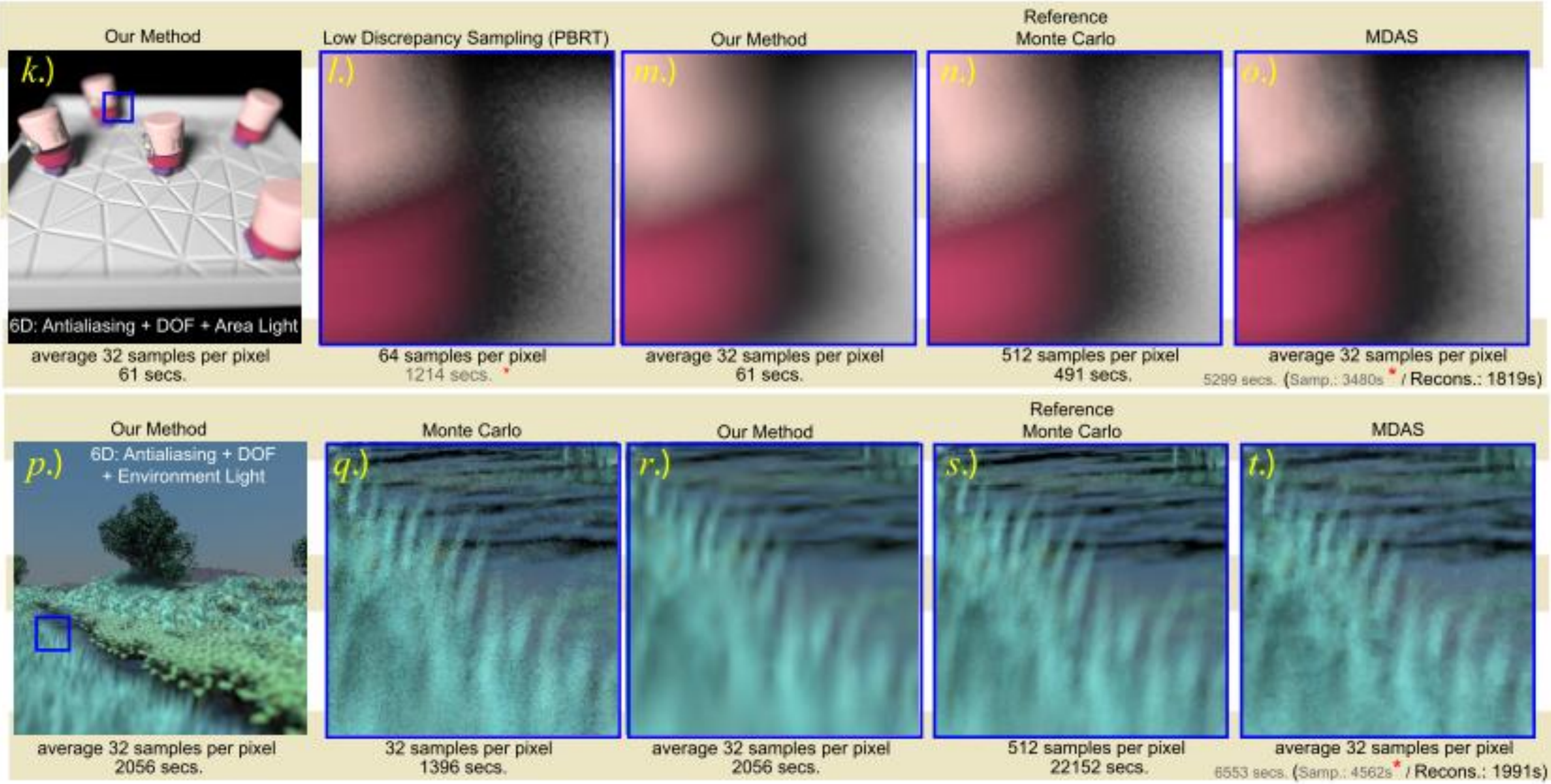


Results (1/2)



KAIST

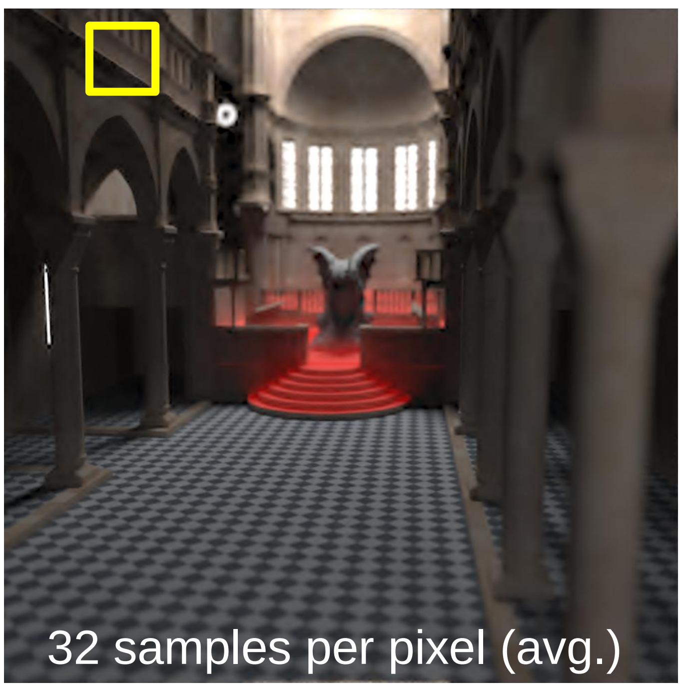
Results (2/2)


Our Method
Low Discrepancy Sampling (PBRT)
Our Method
Reference
Monte Carlo
MDAS
k.)
l.)
m.)
n.)
o.)
6D: Antialiasing + DOF + Area Light
average 32 samples per pixel
61 secs.
64 samples per pixel
1214 secs. *
average 32 samples per pixel
61 secs.
512 samples per pixel
491 secs.
average 32 samples per pixel
5299 secs. (Samp.: 3480s * / Recons.: 1819s)
Our Method
Monte Carlo
Our Method
Reference
Monte Carlo
MDAS
p.)
6D: Antialiasing + DOF
+ Environment Light
q.)
r.)
s.)
t.)
average 32 samples per pixel
2056 secs.
32 samples per pixel
1396 secs.
average 32 samples per pixel
2056 secs.
512 samples per pixel
22152 secs.
average 32 samples per pixel
6553 secs. (Samp.: 4562s * / Recons.: 1991s)

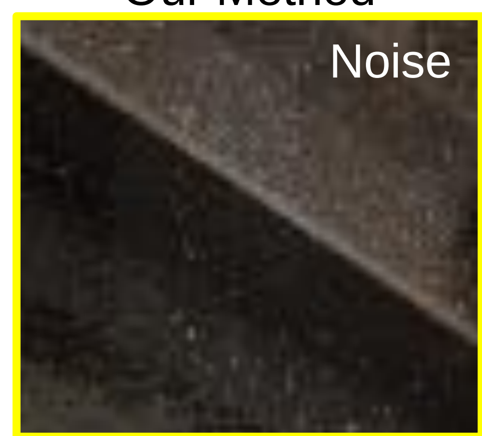
Limitations (1/3)

Wavelet artifacts when not enough samples

Ringing around edges



Our Method

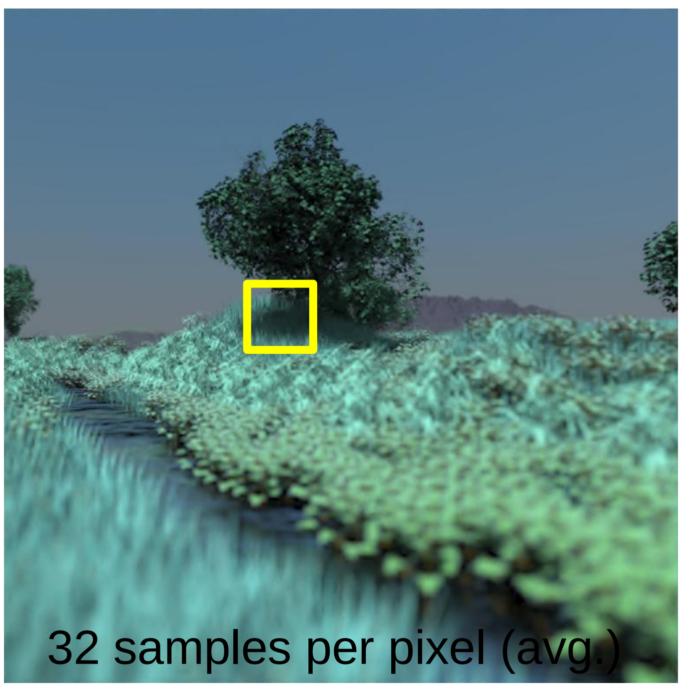


Monte Carlo

Limitations (2/3)

Wavelet artifacts when not enough samples

- **Ringing around edges**
- **Overly smoothing**



Our Method

Monte Carlo

Limitations (3/3)

- Wavelet artifacts when not enough samples
  - Ringing around edges
  - Overly smoothing

- Potential solutions
  - Variance reduction (path splitting, QMC, etc.)
  - Reduce smoothing during reconstruction
  - Use depth and normals to improve statistics
  - Use more samples

Conclusion/Summary

Sample and reconstruct in wavelet basis

Features

- Low Sample Counts
- Efficient
- General

Best for smooth image features