Denoising Techniques

Sung-Eui Yoon
(윤성의)

Course URL:
http://sglab.kaist.ac.kr/~sungeui

KAIST

SIGGRAPH2015
Xroads of Discovery

The 42nd International Conference and Exhibition on Computer Graphics and Interactive Techniques

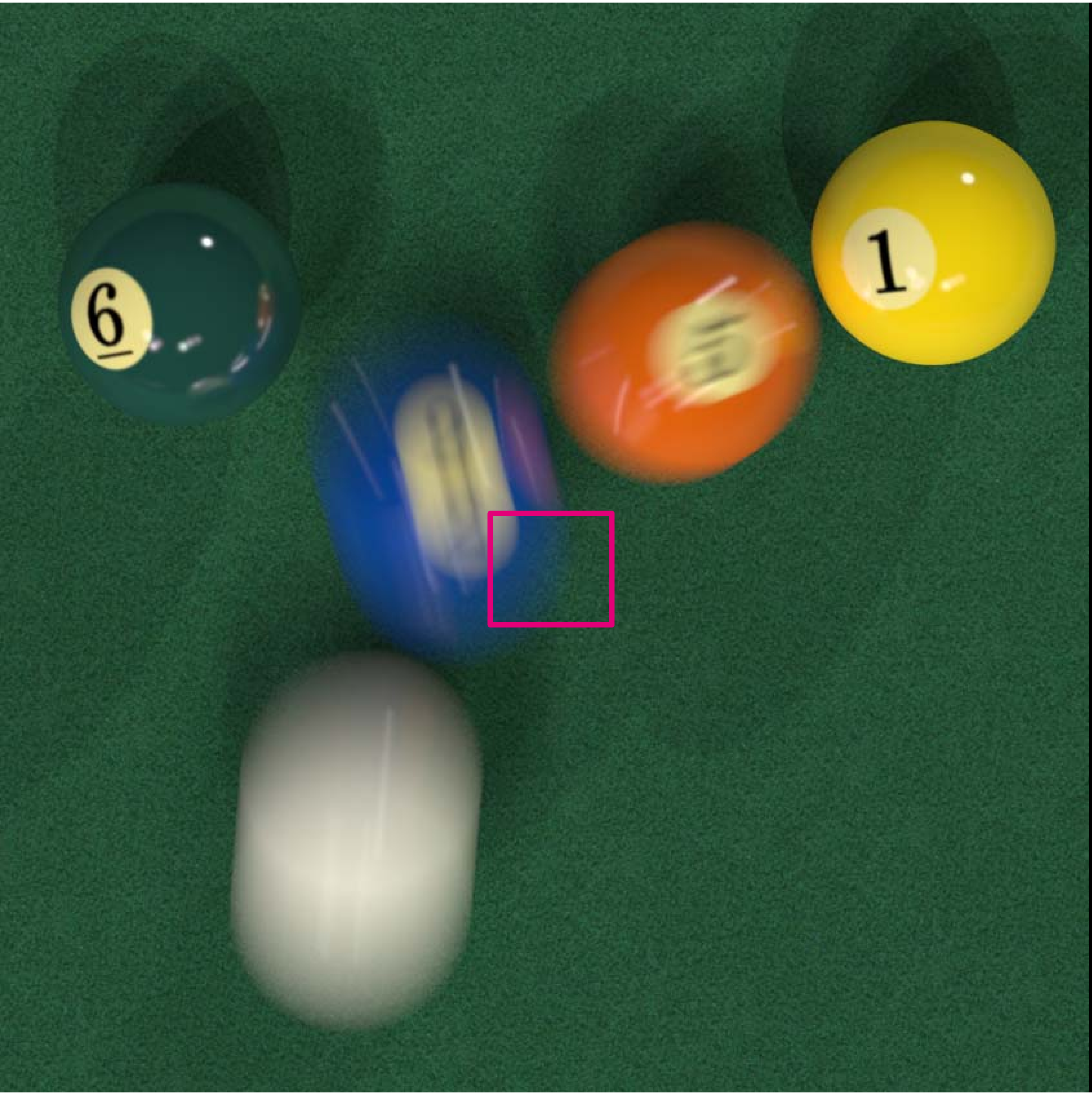
Adaptive Rendering based on Weighted Local Regression

Bochang Moon[1] Nathan Carr[2] Sung-Eui Yoon[1]

KAIST[1] Adobe[2]

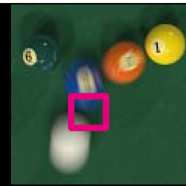
MC Ray Tracing

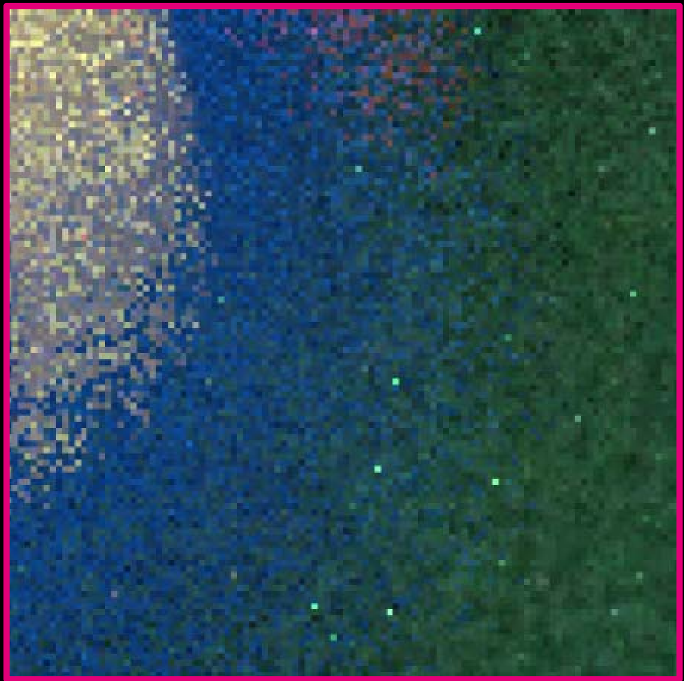
MC Ray Tracing



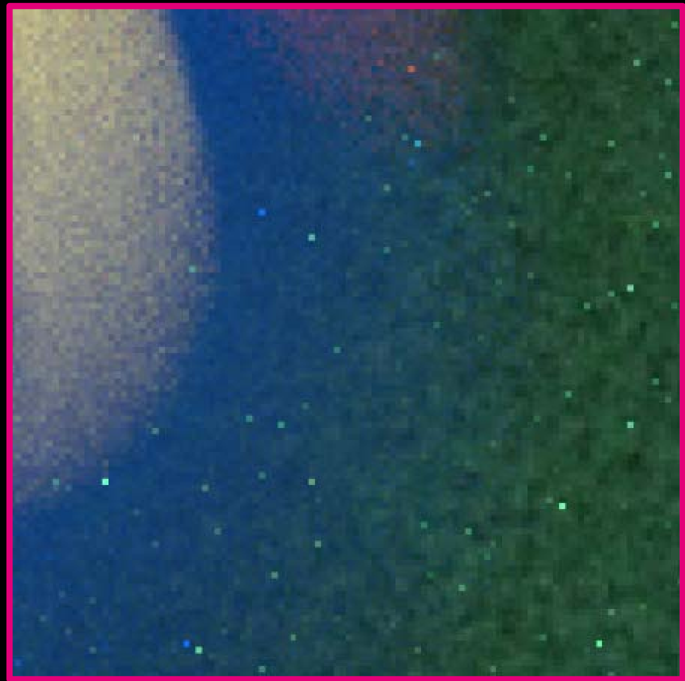
MC Ray Tracing

Slow Convergence





MC 4 spp
(7 secs)

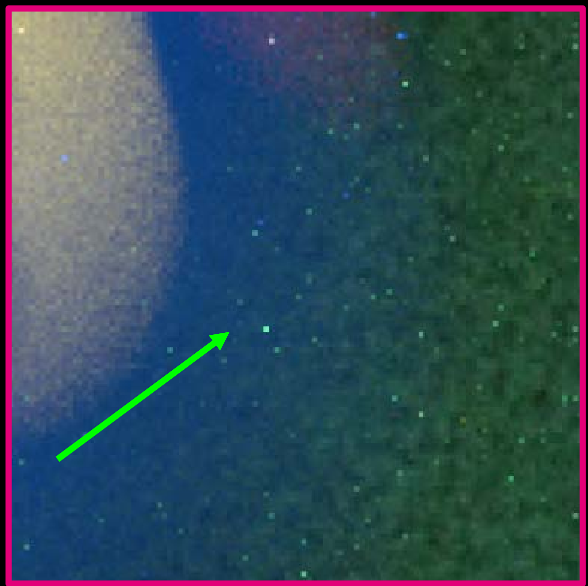


MC 32 spp
(56 secs)

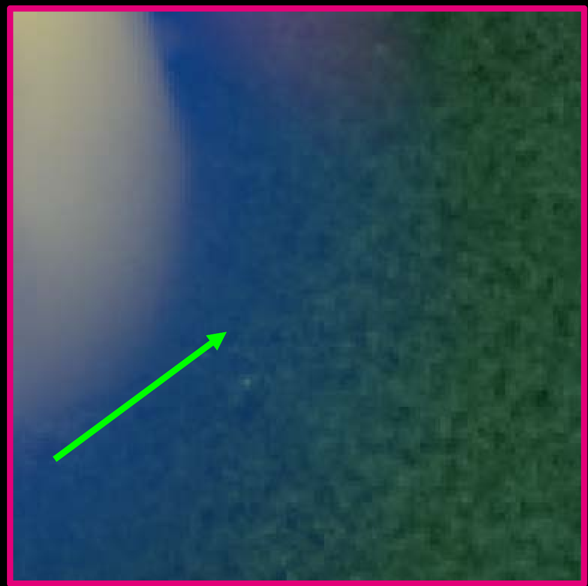
MC 64K spp
(32 hours)

Our Goal

- Reduce the required number of samples by an adaptive sampling and reconstruction



MC 53 spp
(93 secs)



Ours 32 spp
(91 secs)

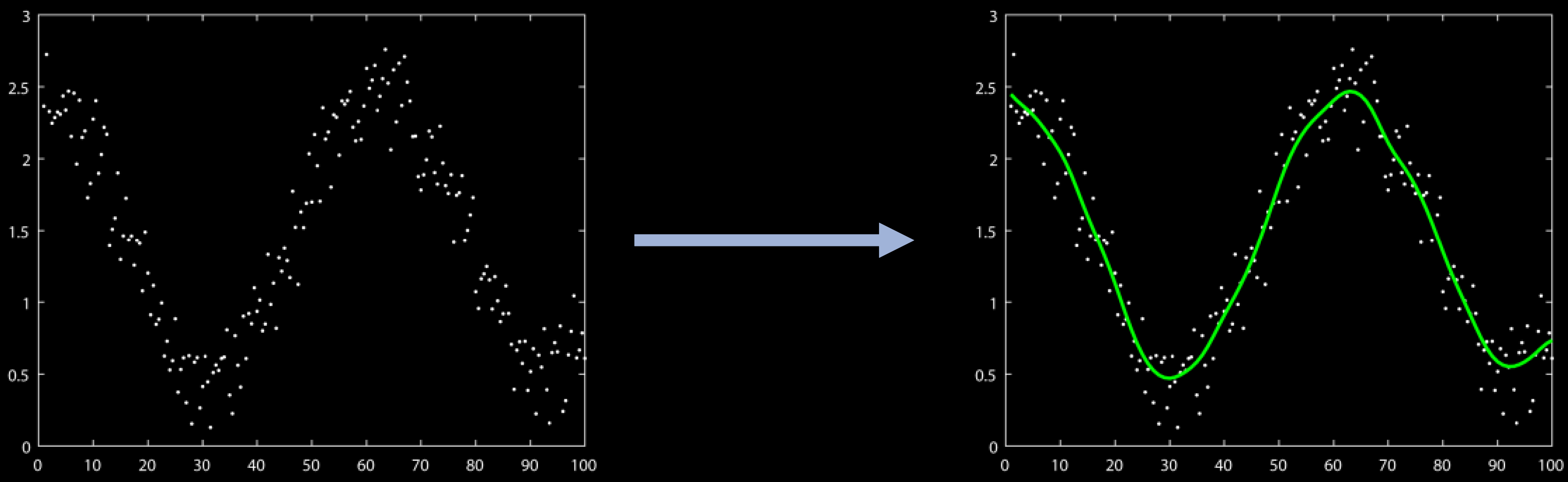
MC 64K spp
(32 hours)

Technical Contributions

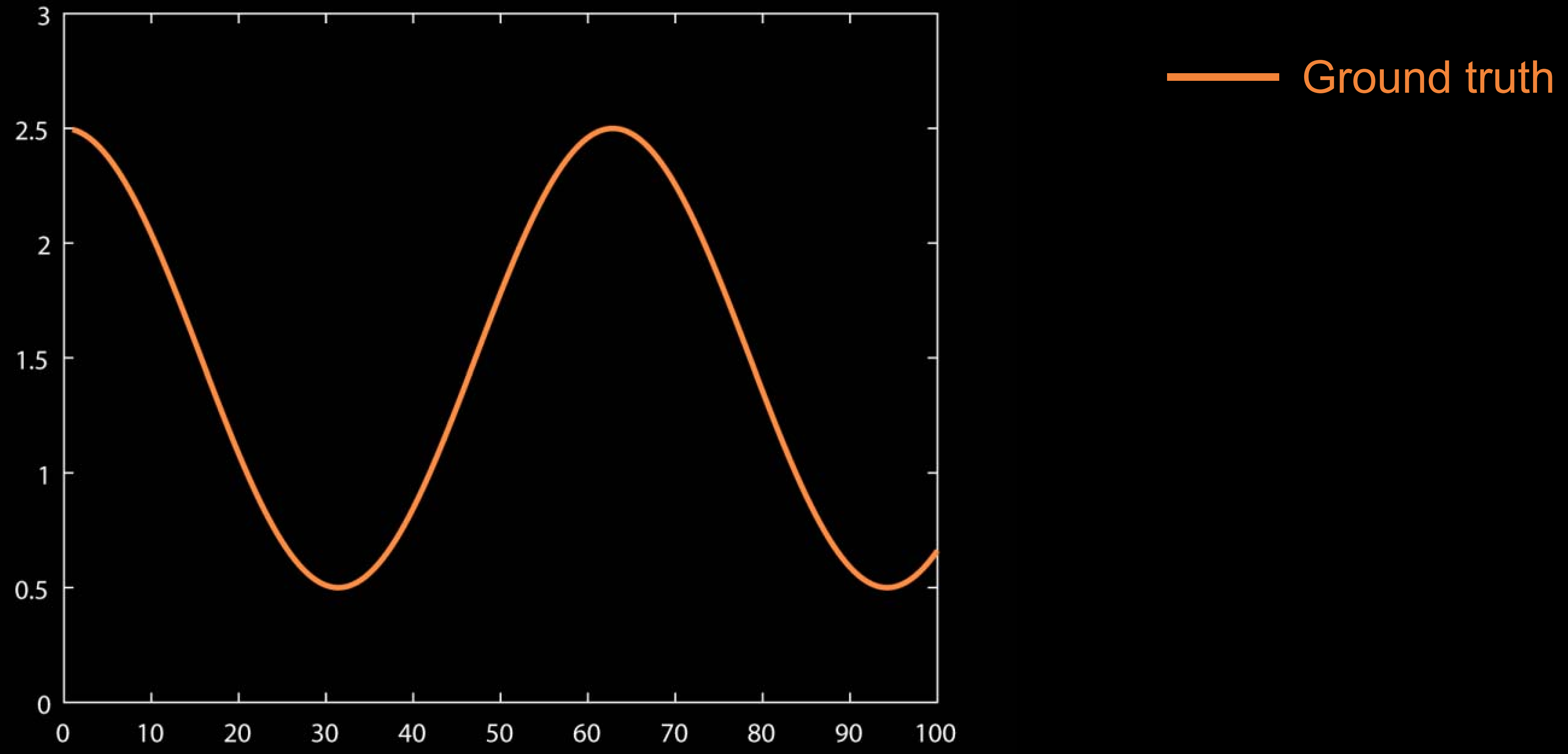
- Propose a weighted local regression based adaptive rendering method
- Adaptive Reconstruction: Estimate optimal filtering bandwidths for an arbitrary set of features
- Adaptive Sampling: Adaptively allocate ray samples using estimated errors

Local Regression

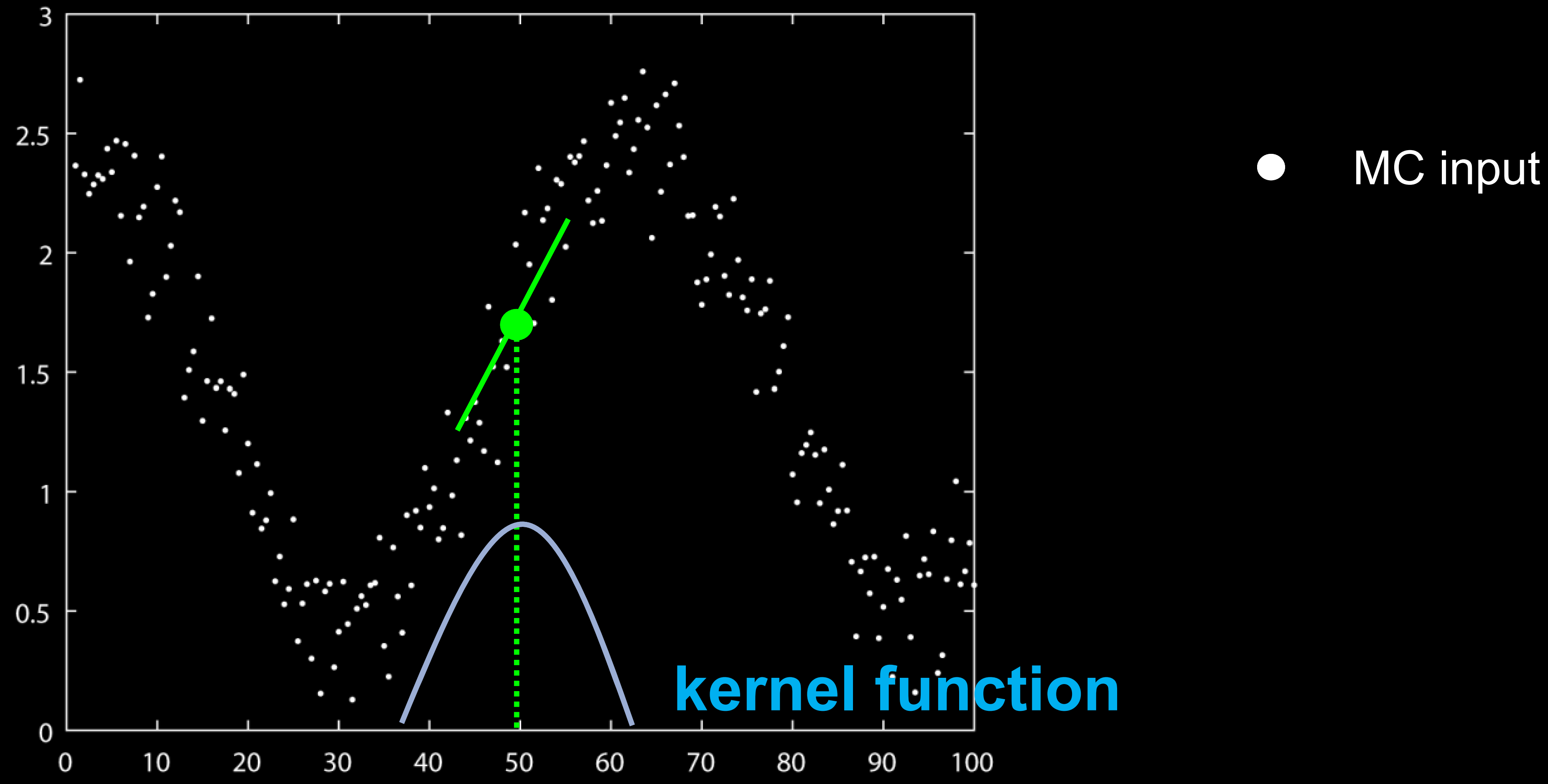
- Well-established theory in the statistics literature [Ruppert and Wand 94, Cleveland and Loader 96]
- Locally approximate an unknown function with low order polynomials (e.g., linear)



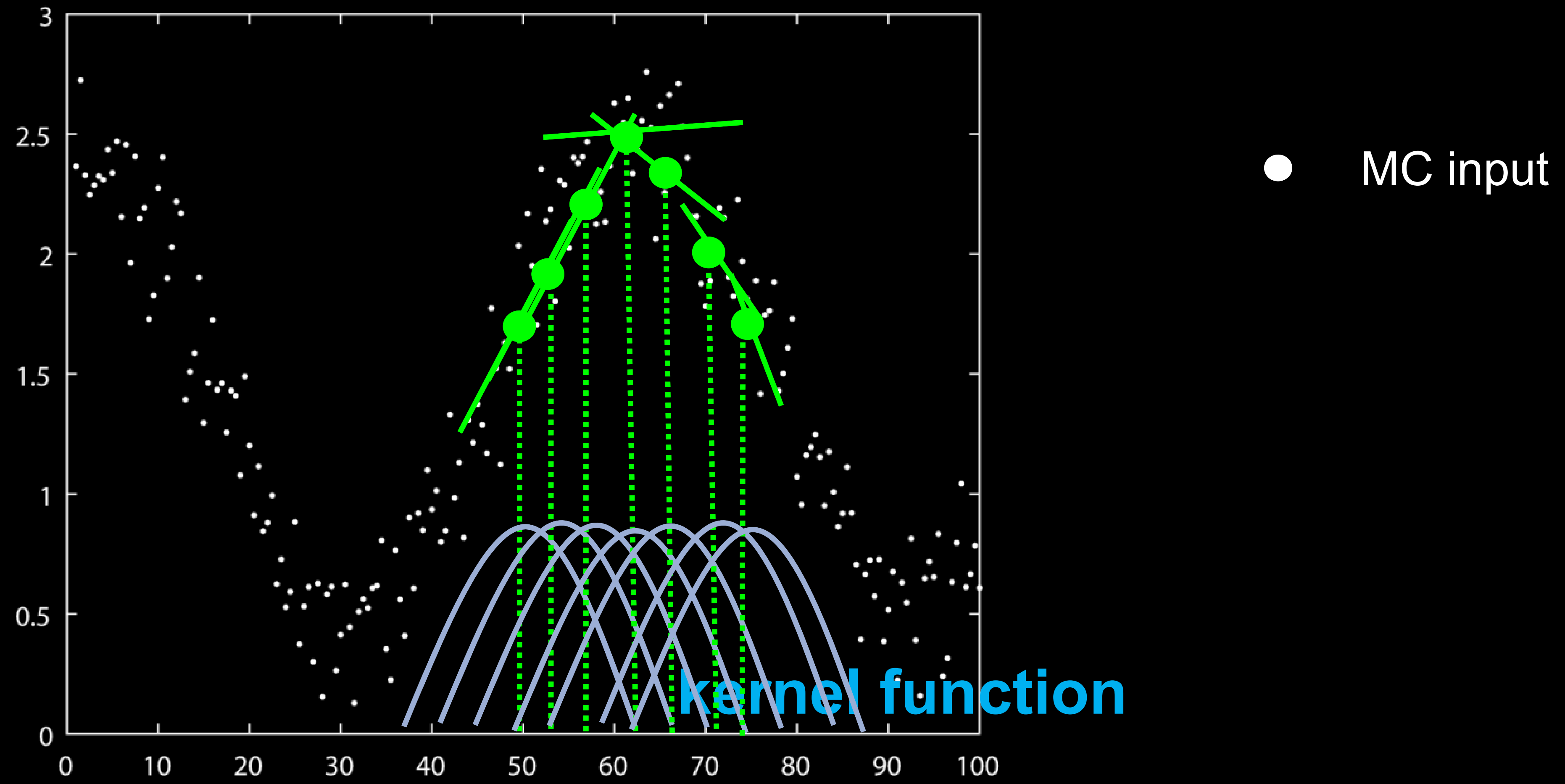
Local Regression



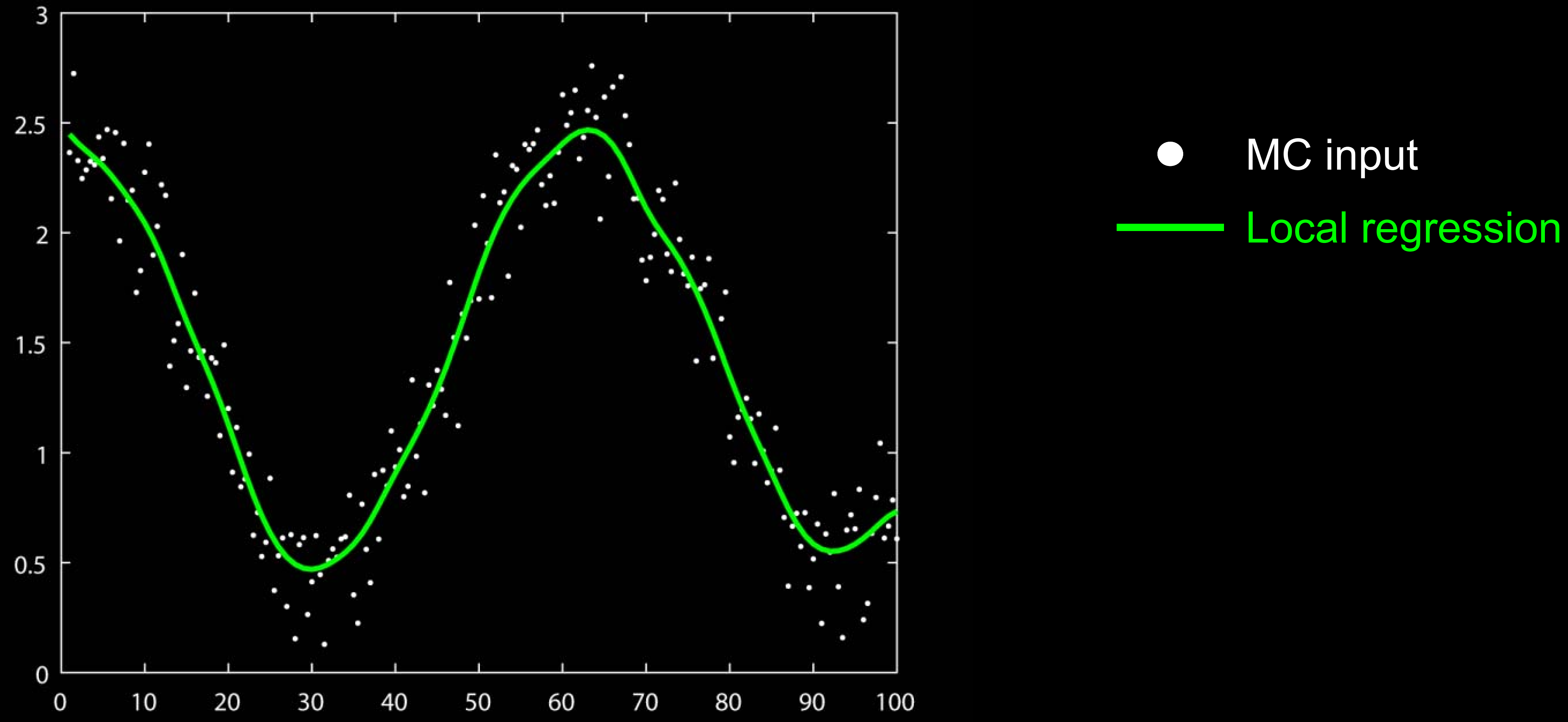
Local Regression



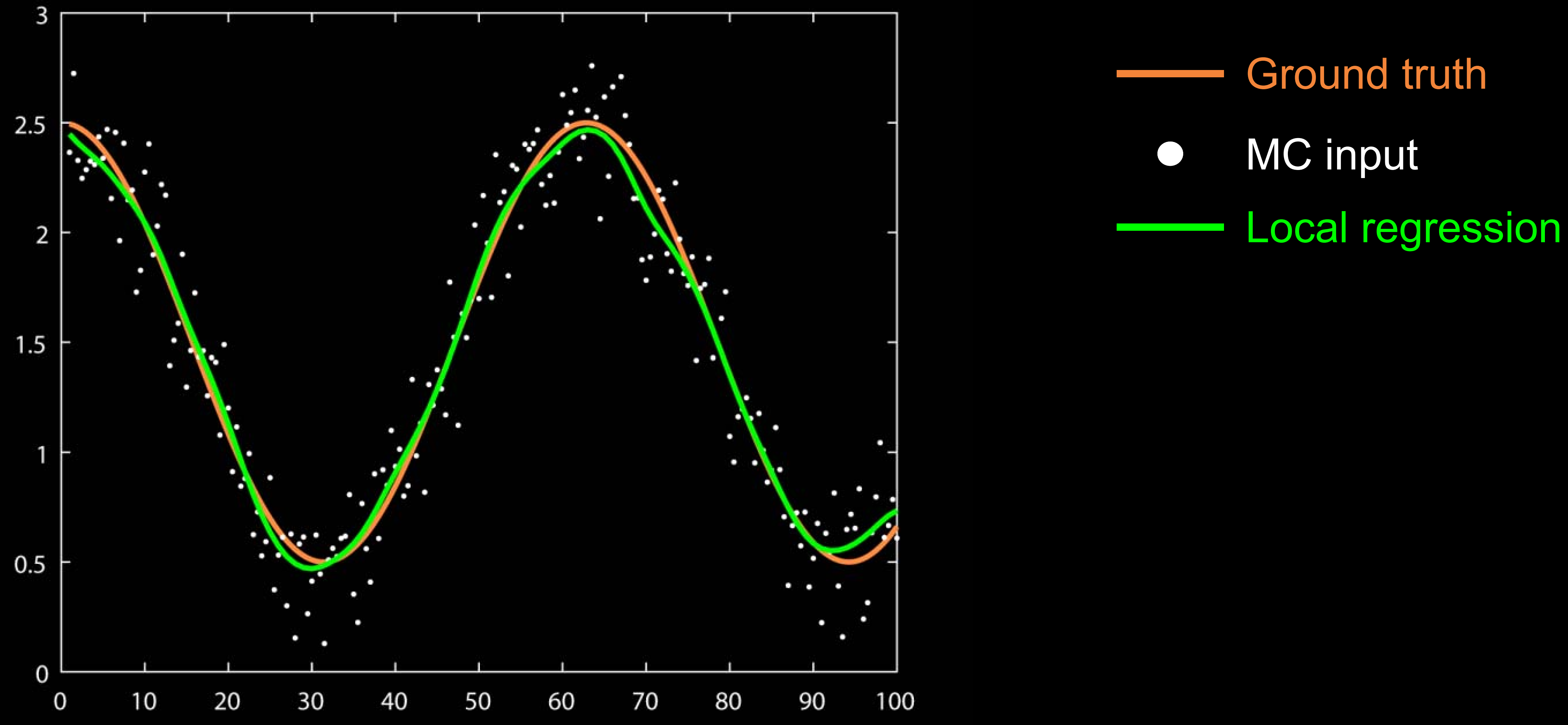
Local Regression



Local Regression



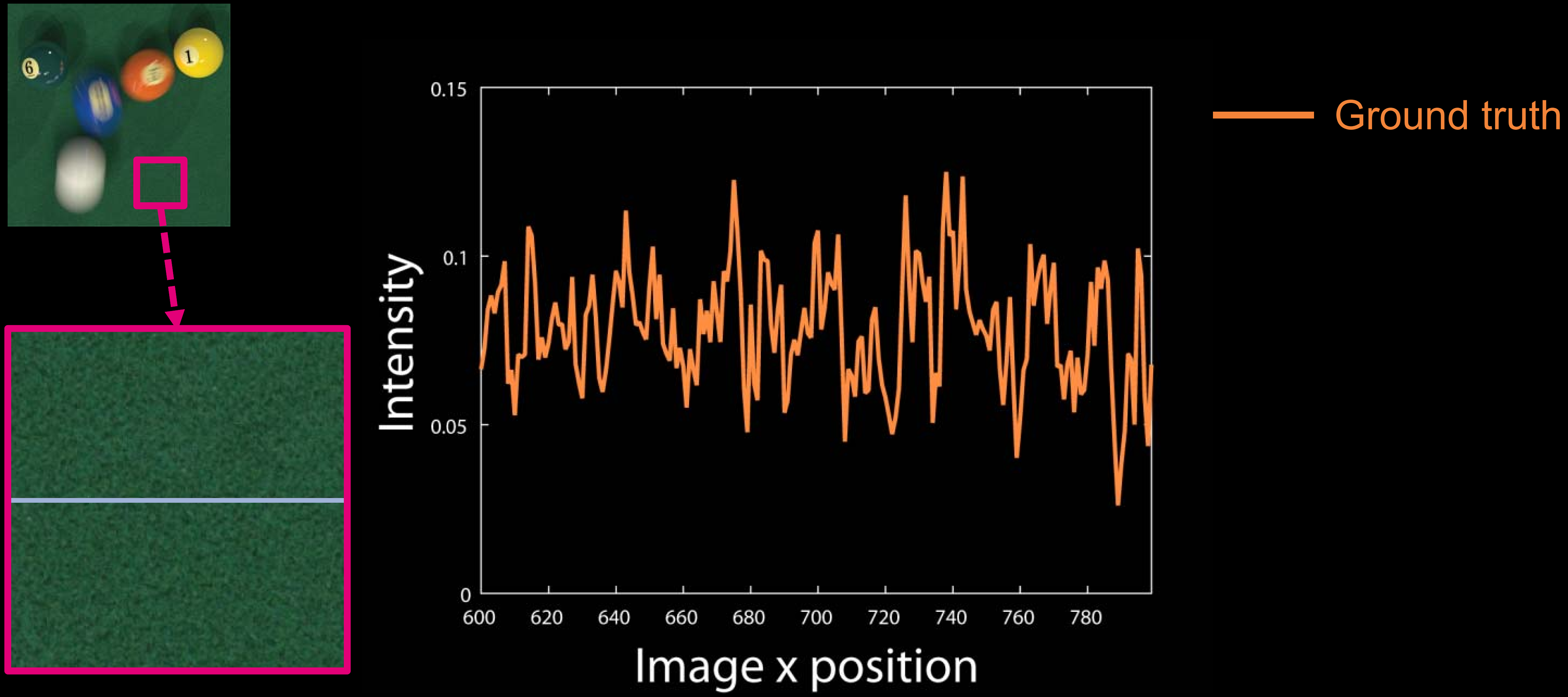
Local Regression



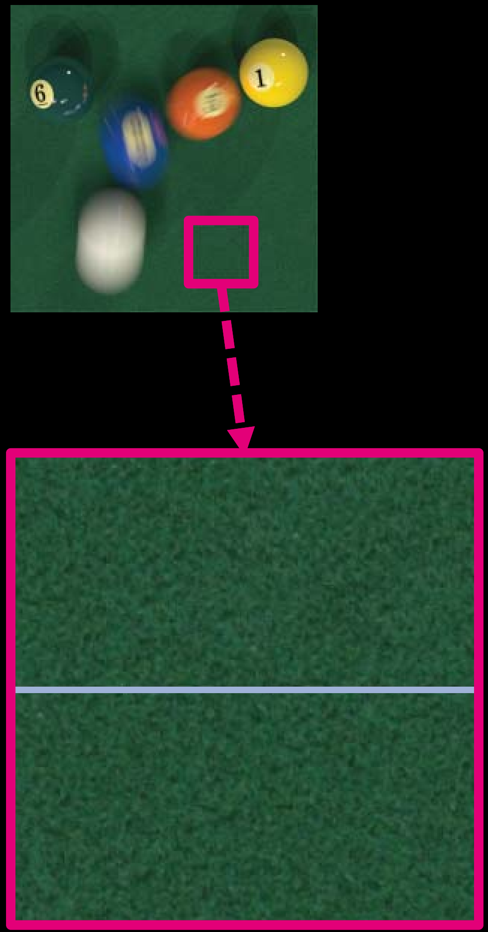
Local Linear Approximation

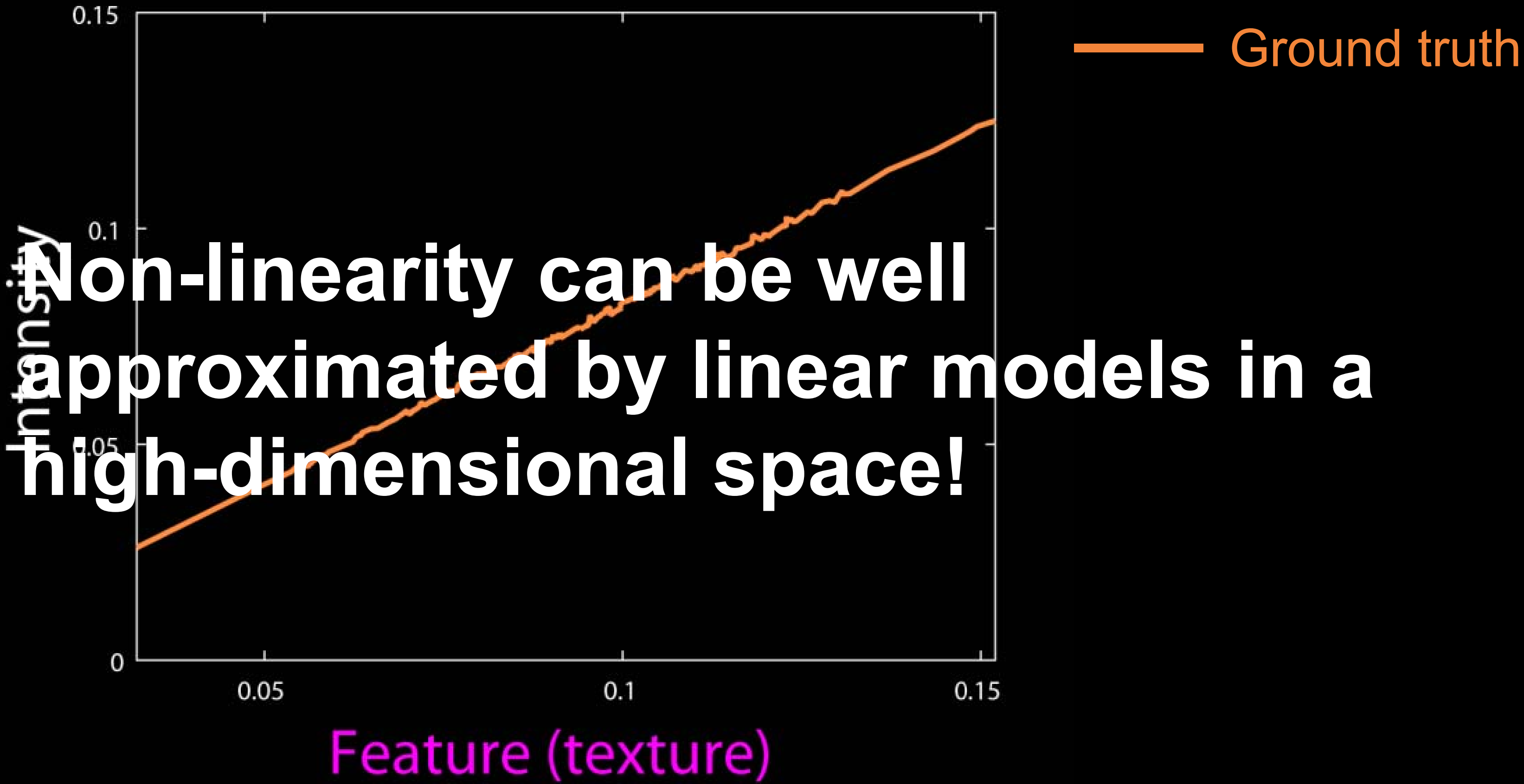
- Locally approximate an unknown image function with linear functions
- Simple and efficient compared to its alternatives such as logistic regression and higher order functions
- Q. Non-linear functions (edges)?

Local Linear Approximation



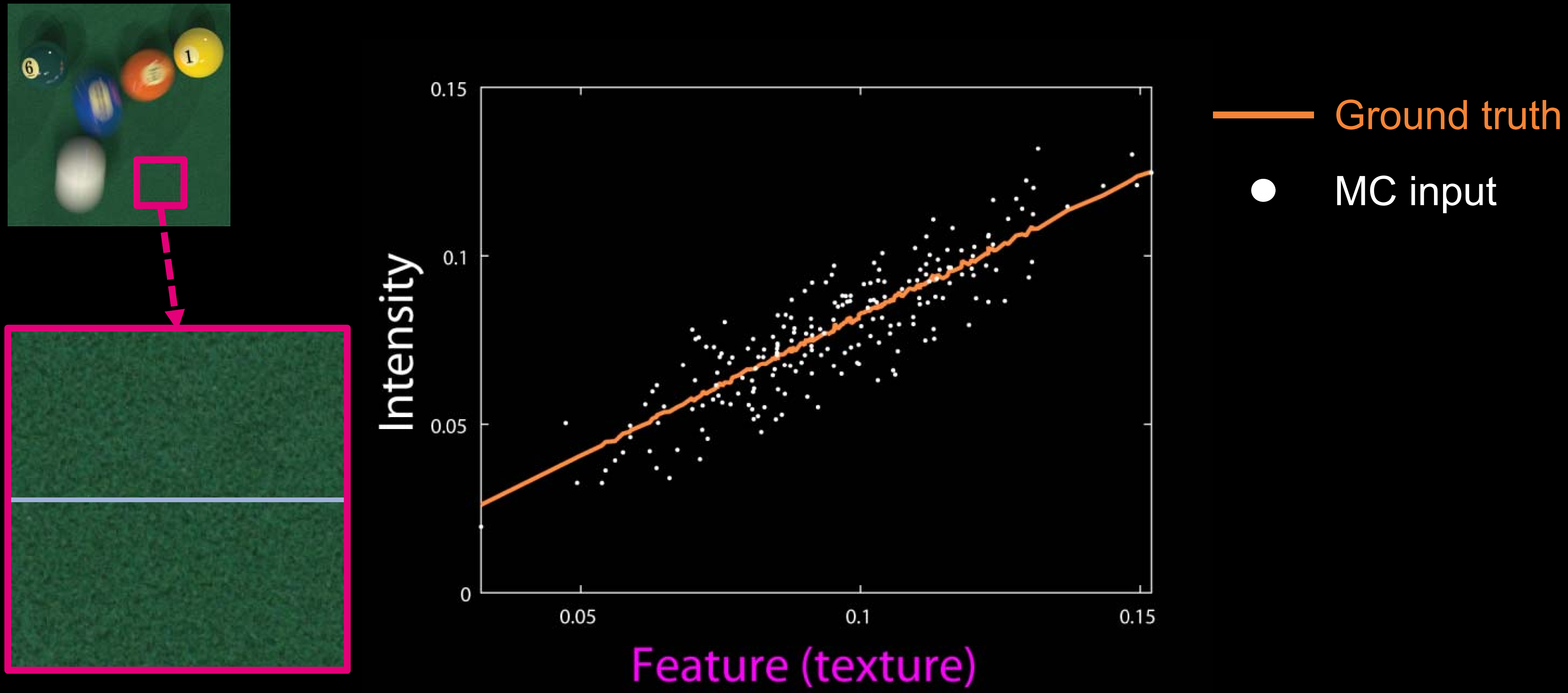
Local Linear Approximation



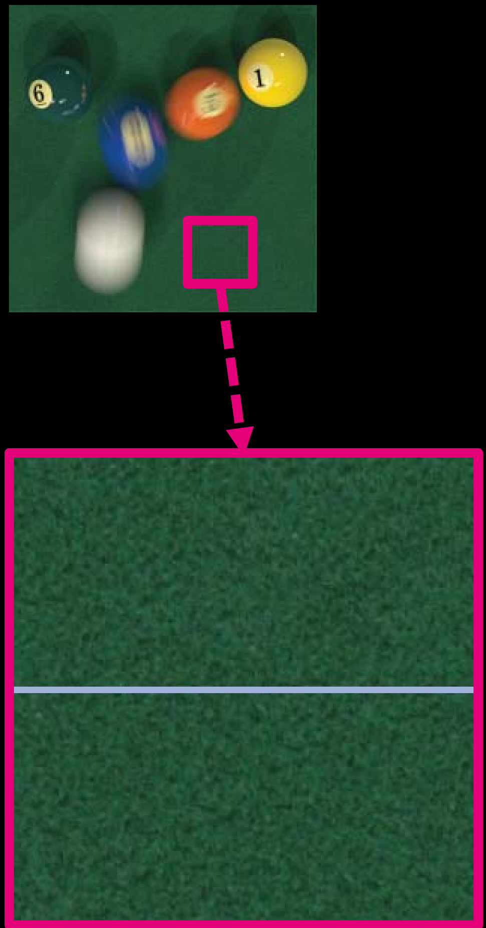


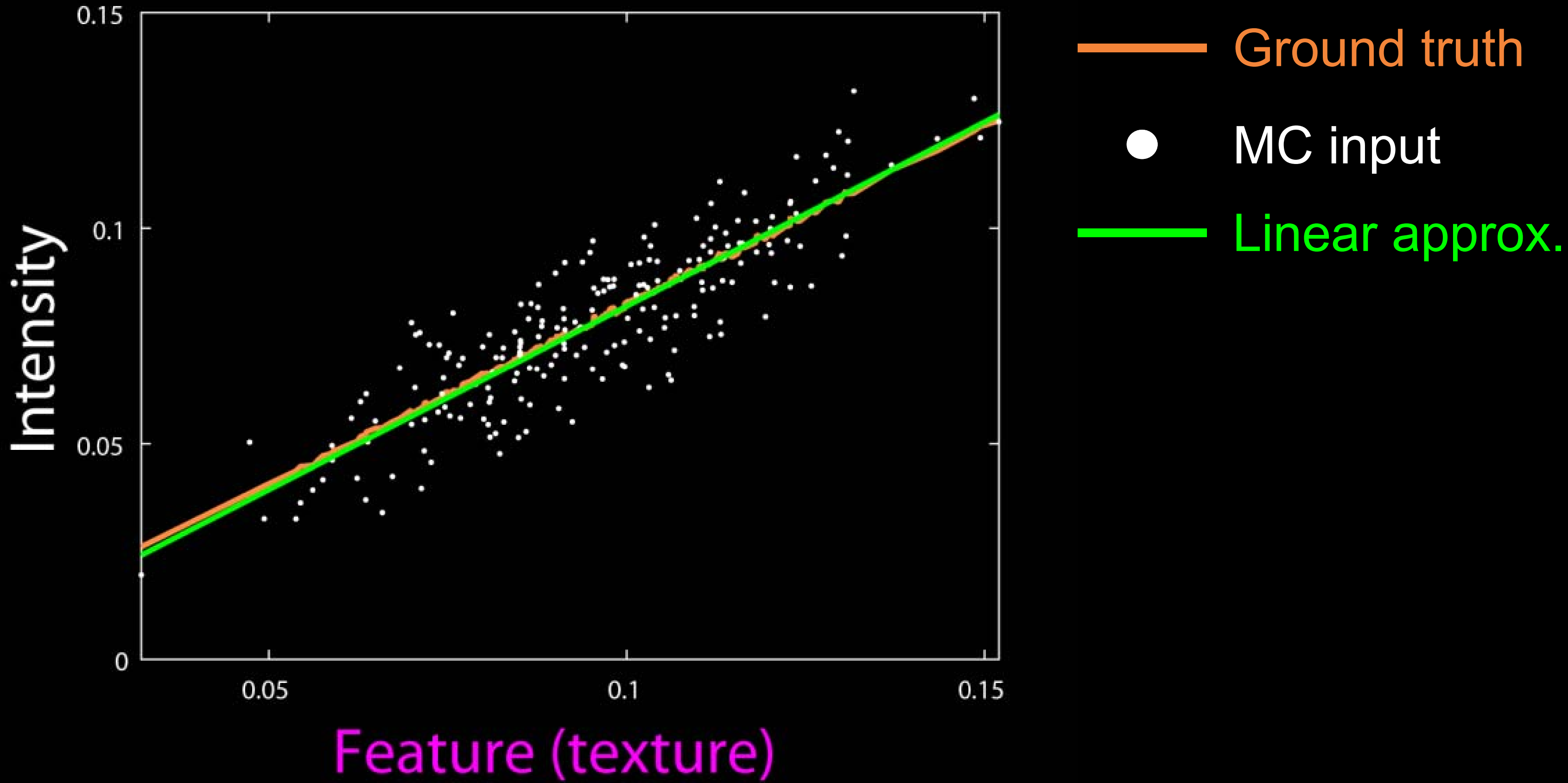
Non-linearity can be well approximated by linear models in a high-dimensional space!

Local Linear Approximation



Local Linear Approximation





Locally Linear Regression

- $[\hat{\alpha}, \hat{\beta}] = \min_{\alpha,\beta} \sum_{i=1}^{n} \left(y^{i} - \alpha - \beta^{T}\left(x^{i} - x^{c}\right)\right)^{2} \prod_{j=1}^{k} \boxed{K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right)}$

- Kernel function (e.g., Gaussian) to allocate higher weights to closer pixels

- Filtering bandwidths (parameters) for features to control smoothing locally

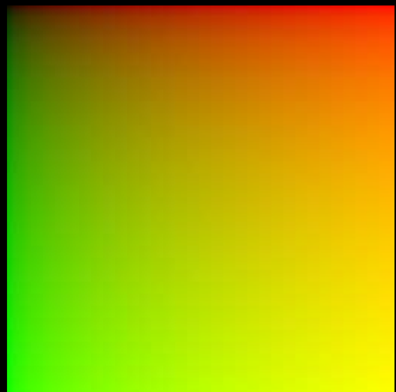
Technical Contributions

- Propose a weighted local regression based adaptive rendering method
- Adaptive Reconstruction: Estimate optimal filtering bandwidths for an arbitrary set of features
- Adaptive Sampling: Adaptively allocate ray samples using estimated errors

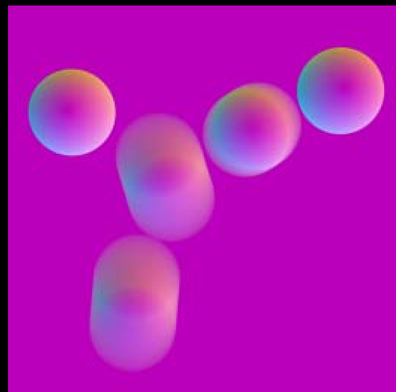
Locally Linear Regression

- $[\hat{\alpha},\hat{\beta}] = \min_{\alpha,\beta} \sum_{i=1}^{n} \left(y^i - \alpha - \beta^T(x^i - x^c)\right)^2 \prod_{j=1}^{k} \boxed{K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right)}$

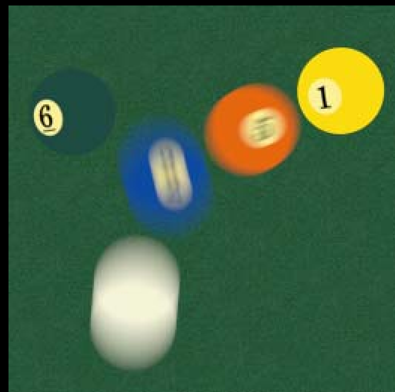
MC input



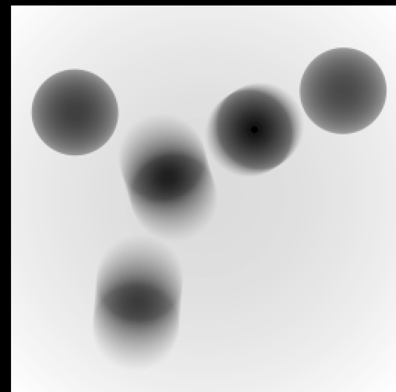
Pixel position



Normal



Texture



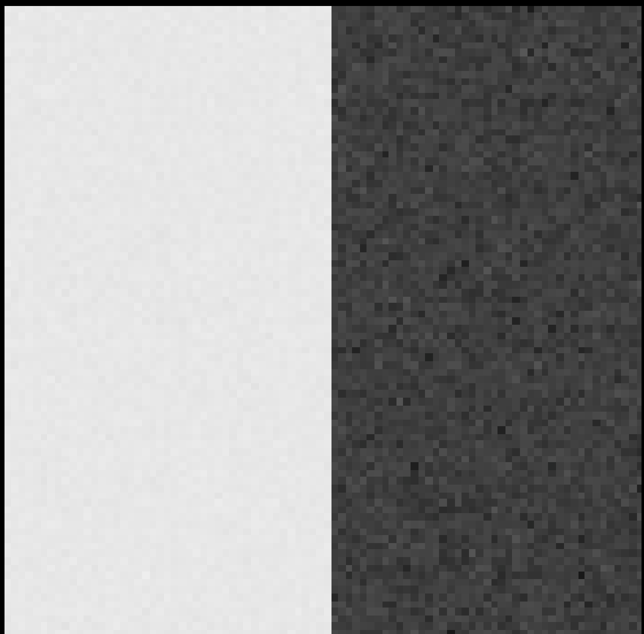
Depth

Bias-Variance Tradeoff

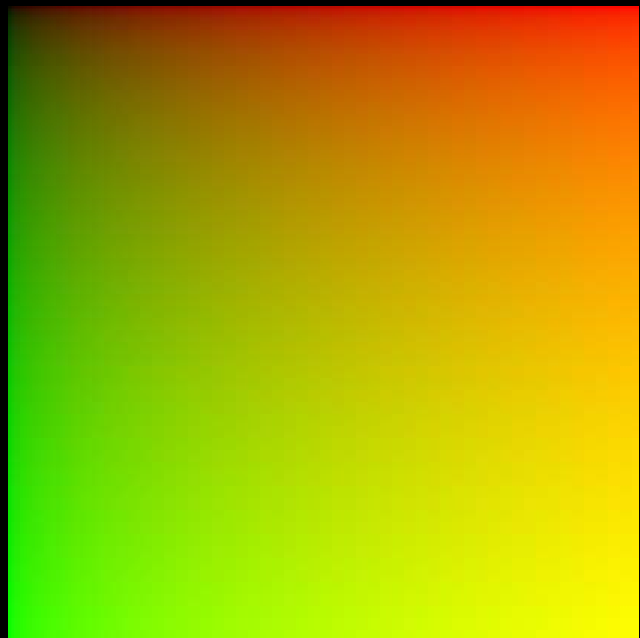
- $\prod_{j=1}^{k} K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$

x position

y position



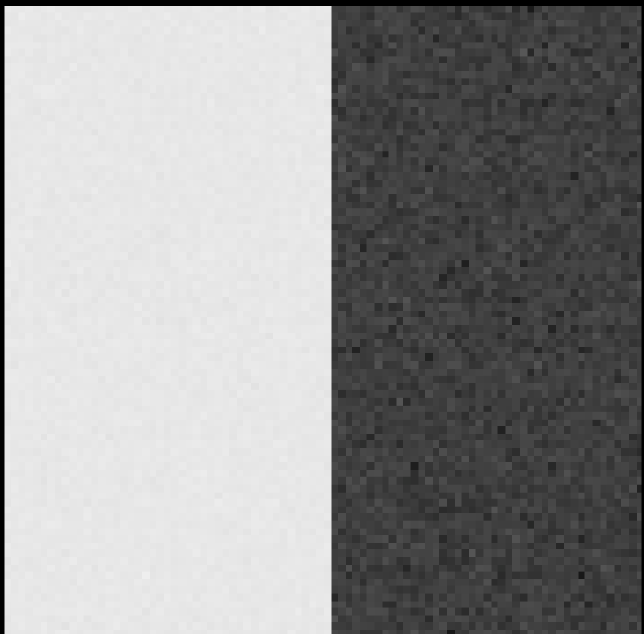
MC input



Pixel position

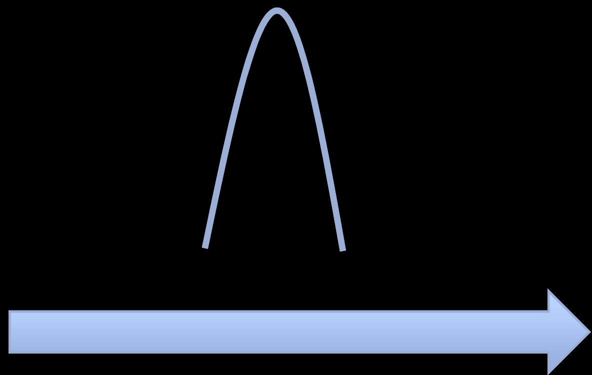
Bias-Variance Tradeoff

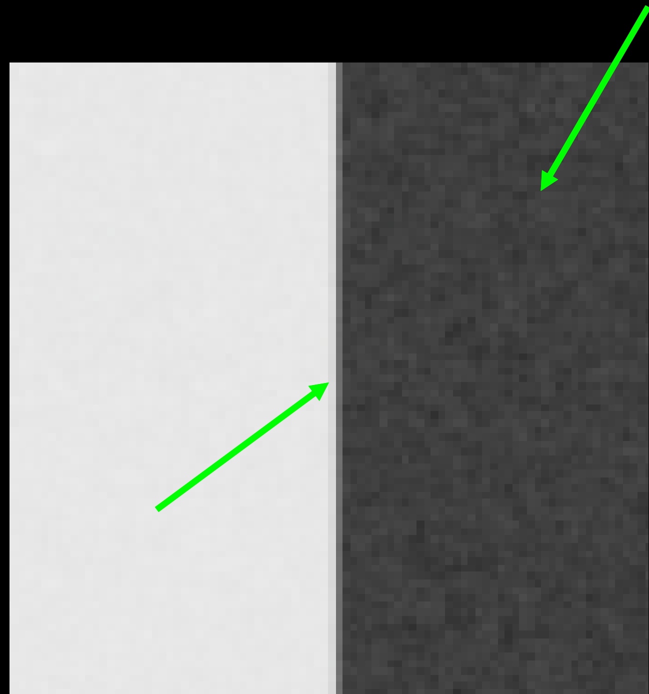
- $\prod_{j=1}^{k} K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$



MC input

small
bandwidths

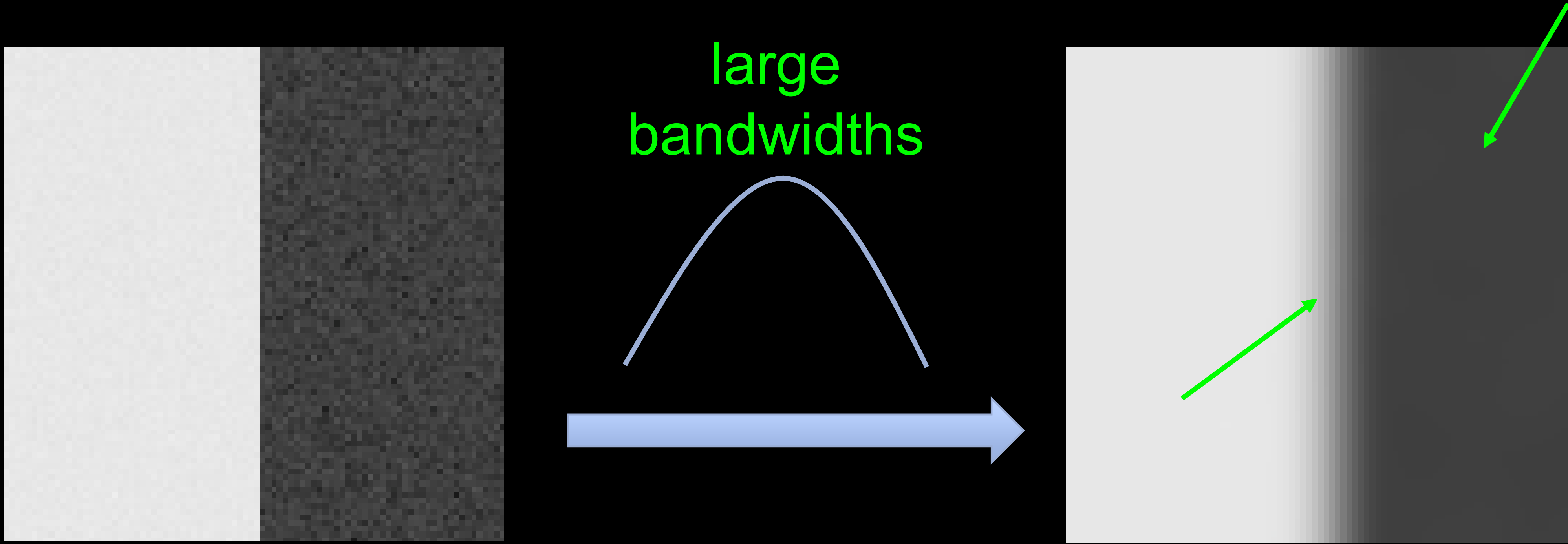




Result

Bias-Variance Tradeoff

- $\prod_{j=1}^{k} K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$



MC input

Result

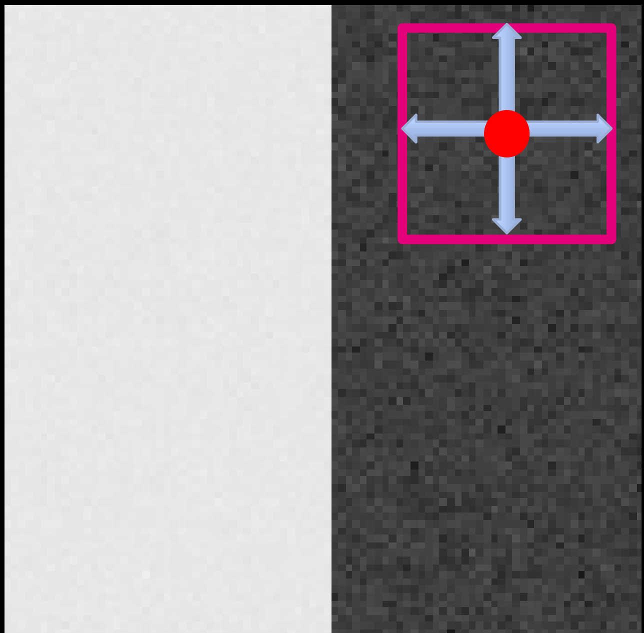
Adaptive Bandwidth Selection

- $\prod_{j=1}^{k} K(\frac{x_j^i - x_j^c}{bandwidth_j}) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$

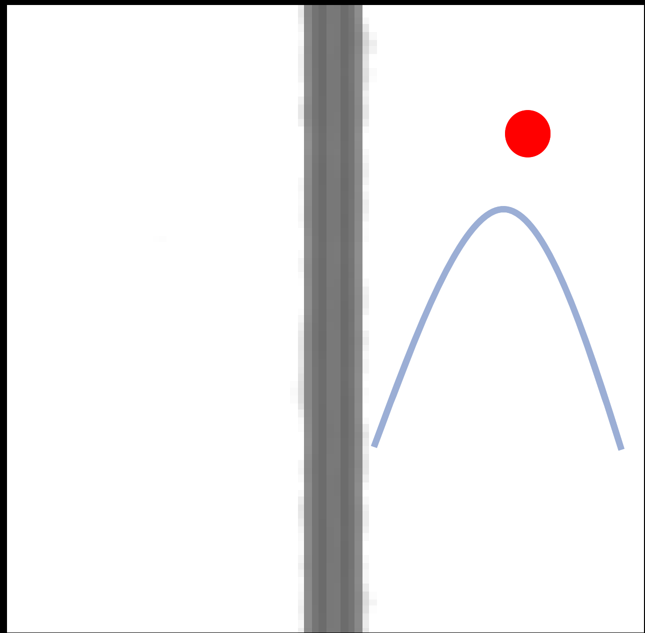
- At each center pixel, our method uses different bandwidths for each feature in a data-driven way

Adaptive Bandwidth Selection

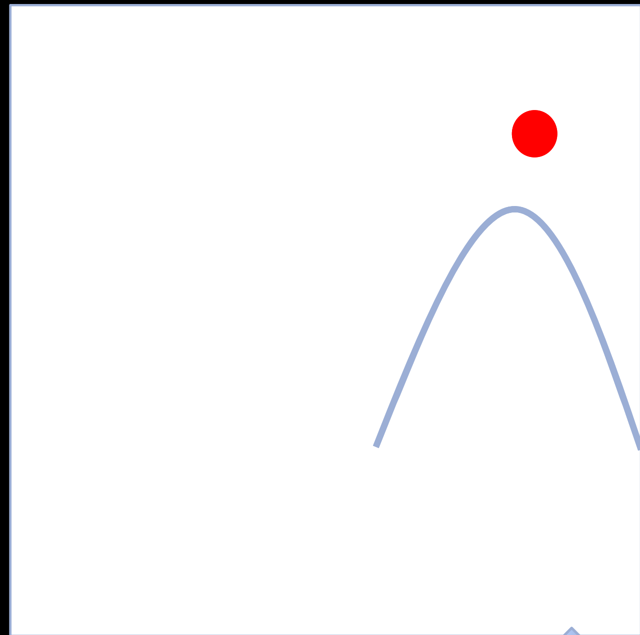
- $\prod_{j=1}^{k} K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$



MC Input



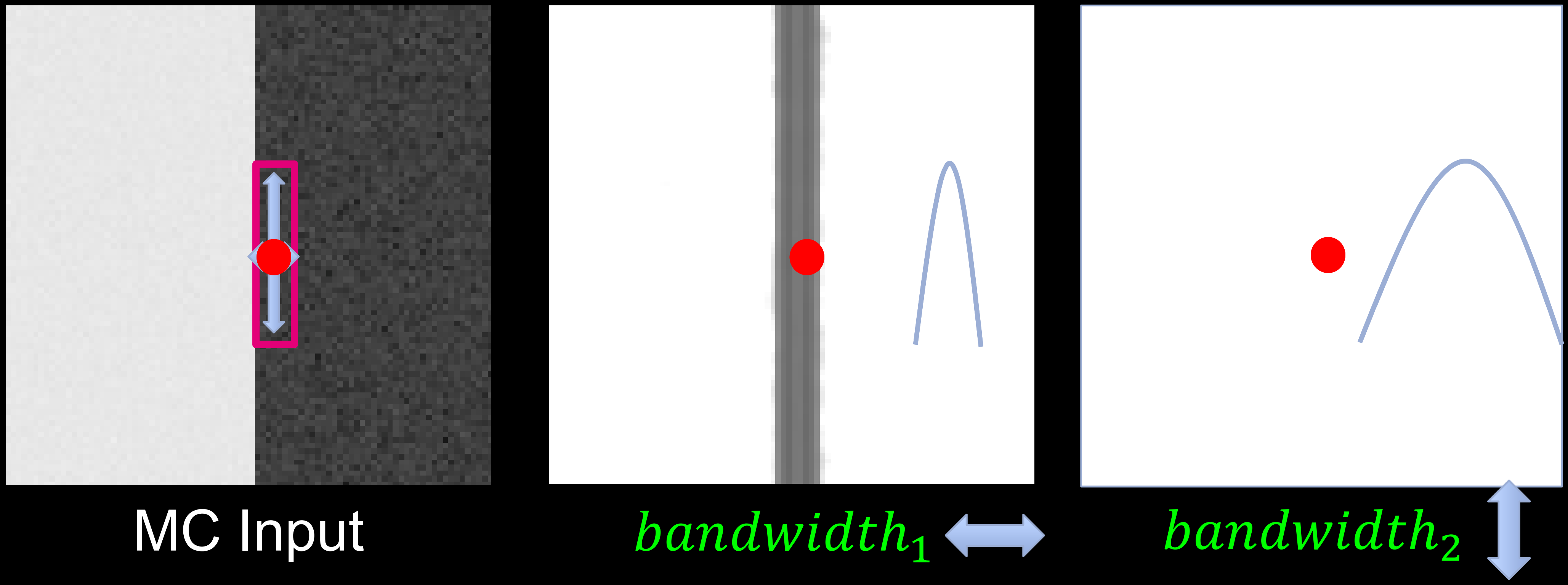
$bandwidth_1$



$bandwidth_2$

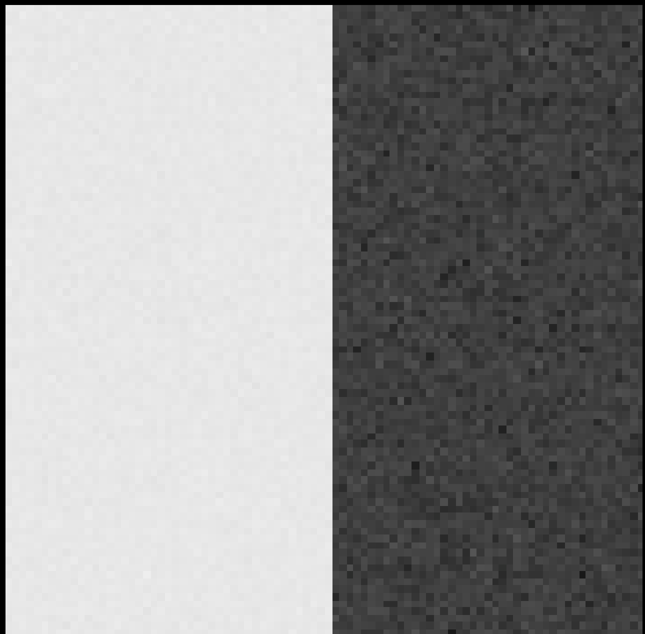
Adaptive Bandwidth Selection

- $\prod_{j=1}^{k} K\left(\frac{x_j^i - x_j^c}{bandwidth_j}\right) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$



Adaptive Bandwidth Selection

- $\prod_{j=1}^{k} K(\frac{x_j^i - x_j^c}{bandwidth_j}) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$

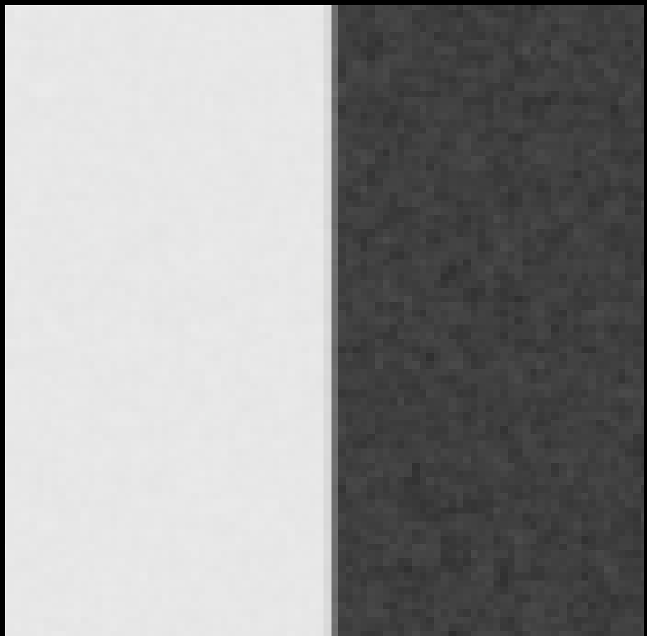


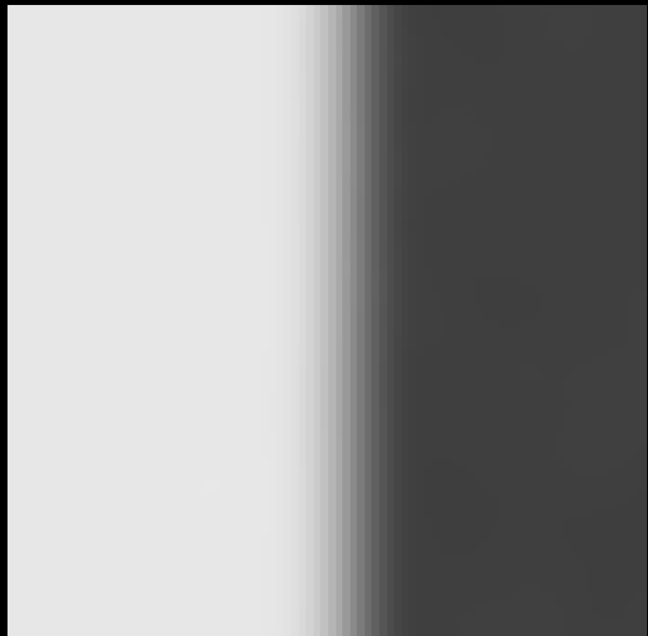
MC input

Our result

Adaptive Bandwidth Selection

- $\prod_{j=1}^{k} K(\frac{x_j^i - x_j^c}{bandwidth_j}) = K\left(\frac{x_1^i - x_1^c}{bandwidth_1}\right) \times K\left(\frac{x_2^i - x_2^c}{bandwidth_2}\right)$

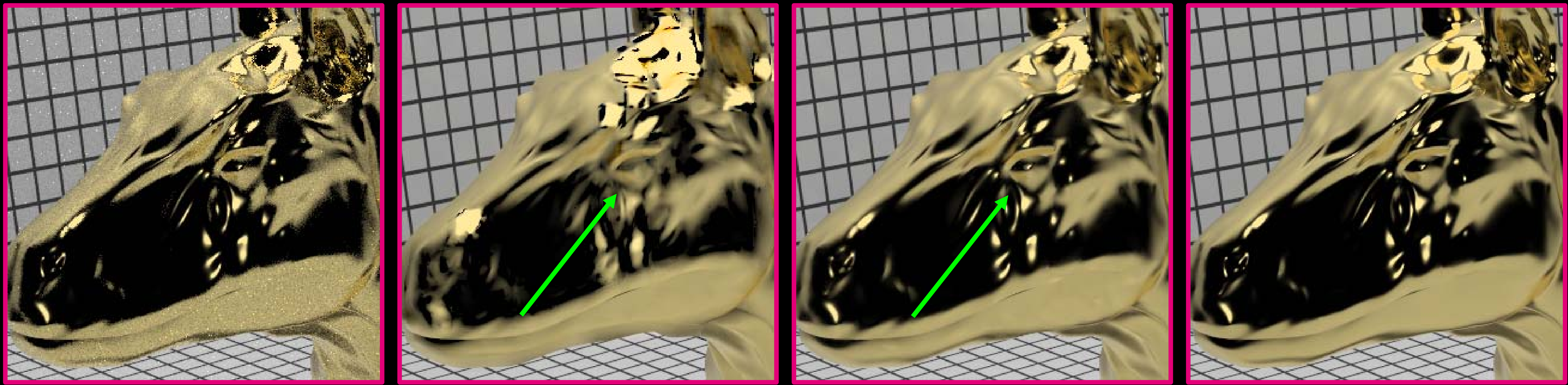




Results with global bandwidths

Our result

Linear Approximation vs. Ours



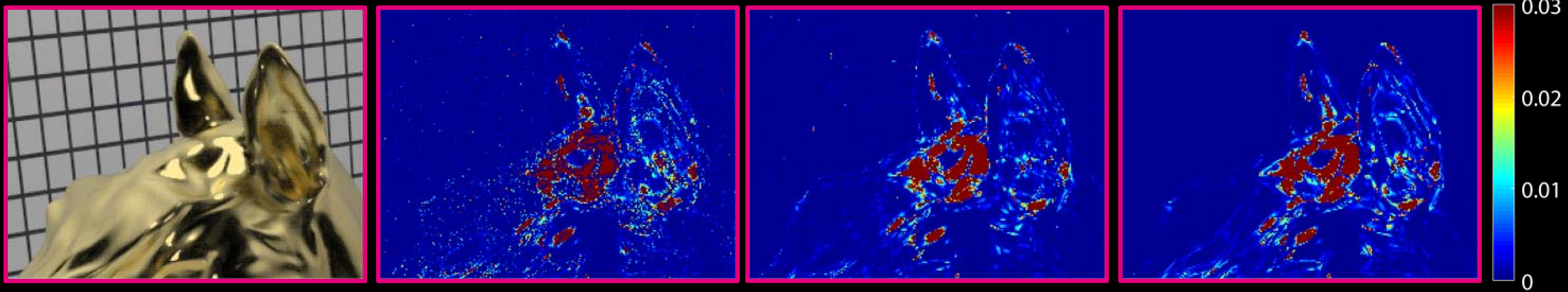
MC input
32 spp

Linear approx.
rMSE 0.50889
[Bauszat et al. 11]

Ours
rMSE 0.00176

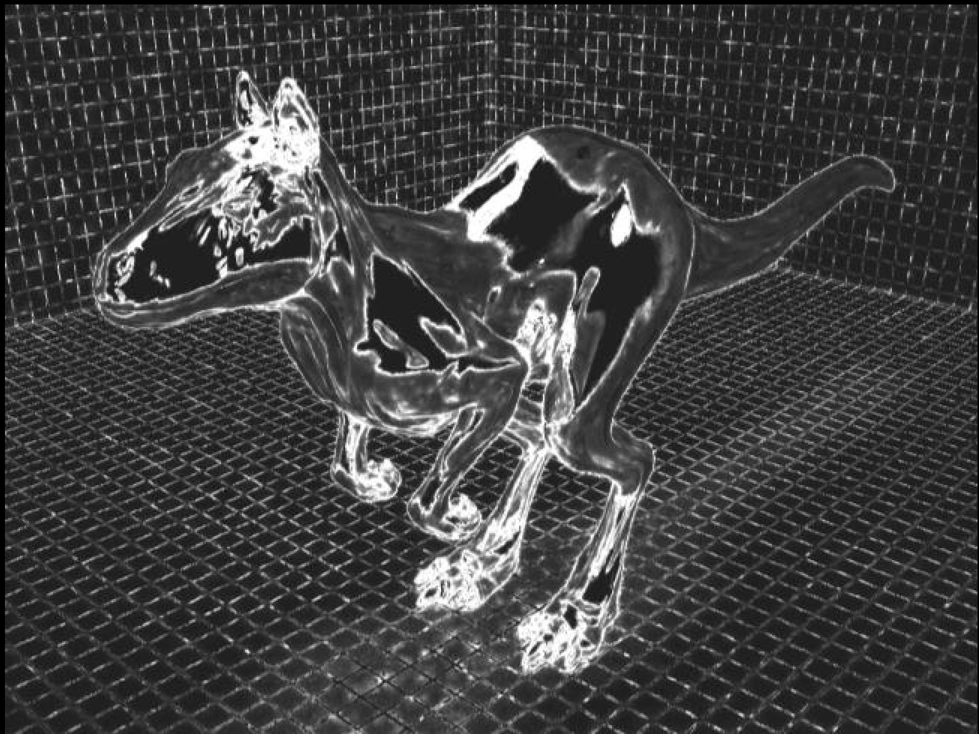
Reference
16K spp

Estimation for Reconstruction Error

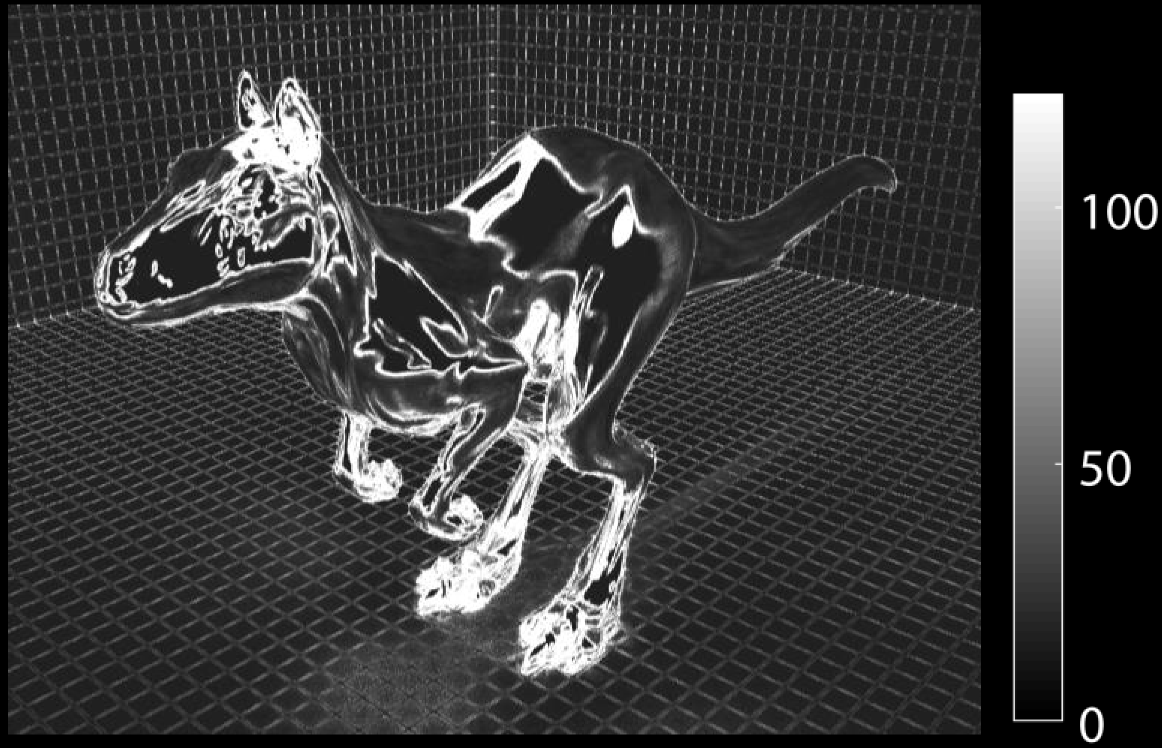


Reconstruction result | SURE [Li et al. 12] | Our estimation | Reference error

Adaptive Sampling



Our map using our error
(Correlation 0.78)

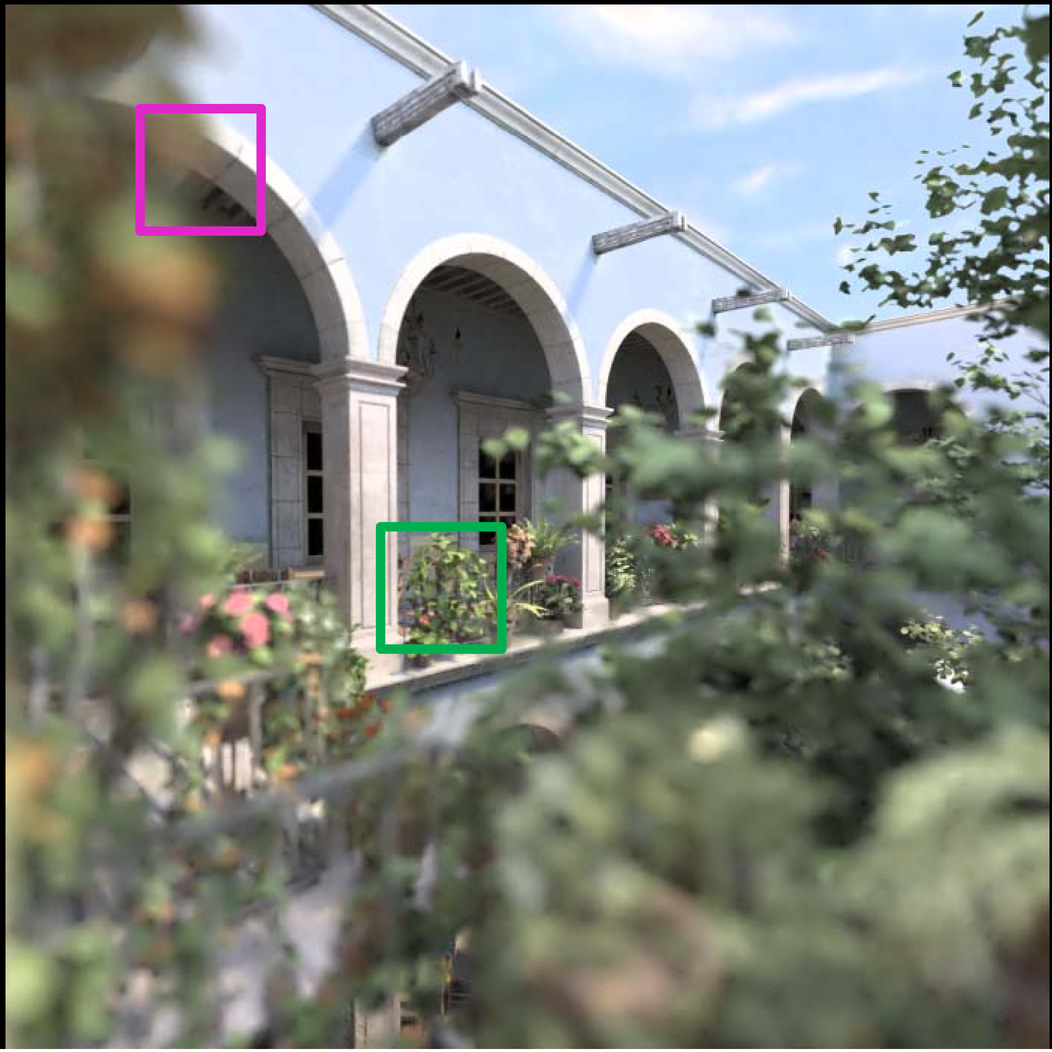


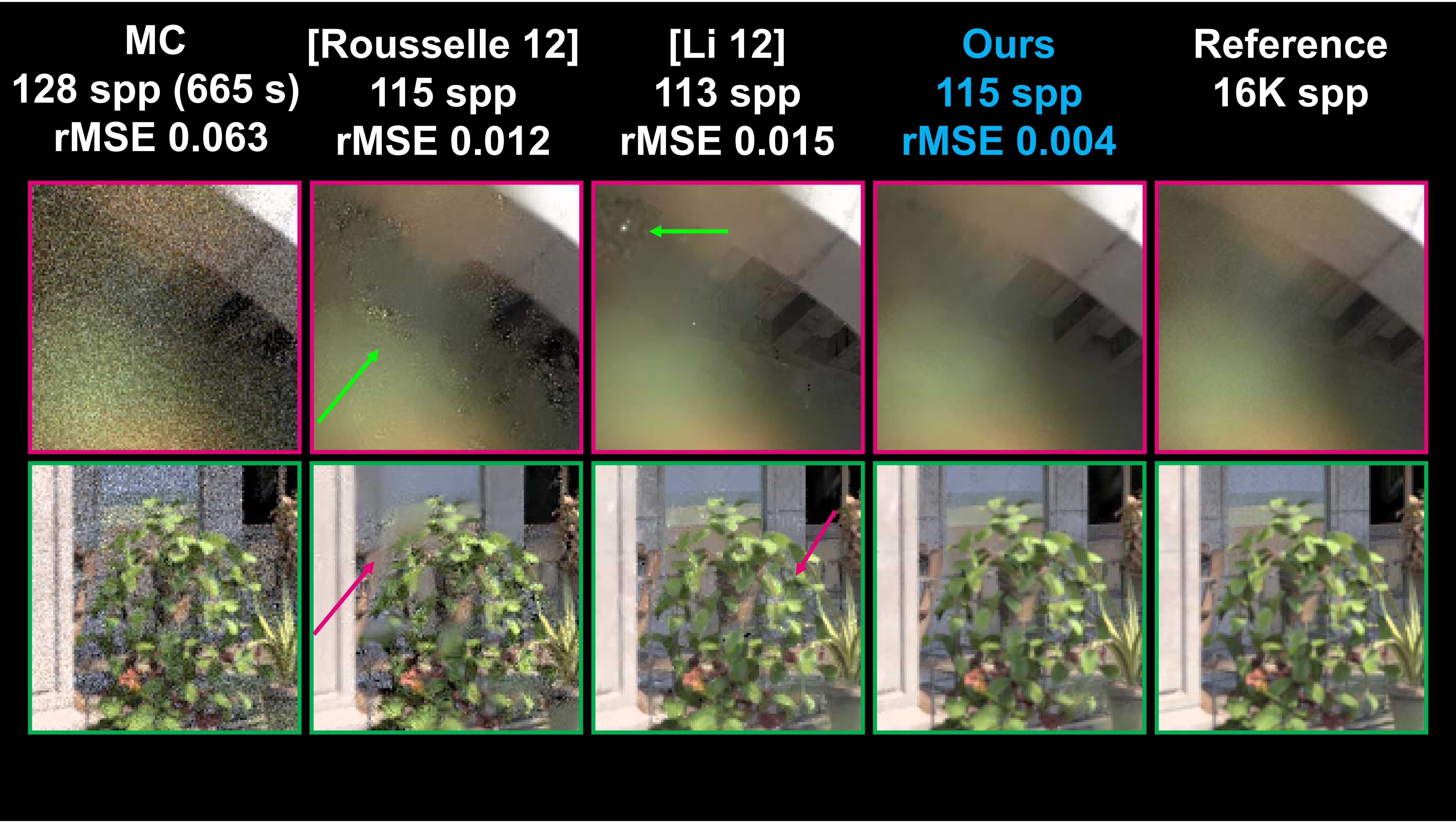
Reference map using
reference error

RESULTS (IMAGES)

Equal-Time Comparison

- San Miguel
  - Path traced
  - Complex geometries
  - Depth-of-field

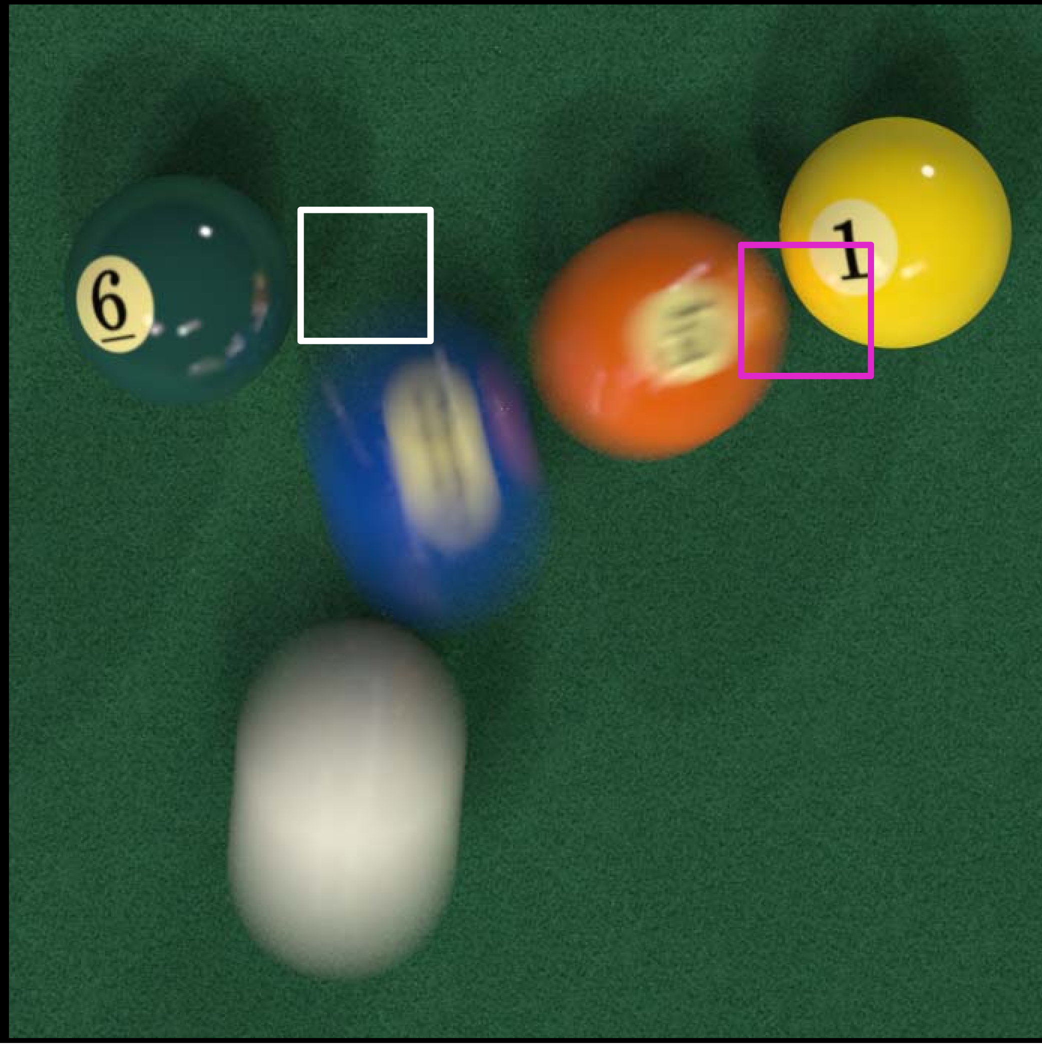


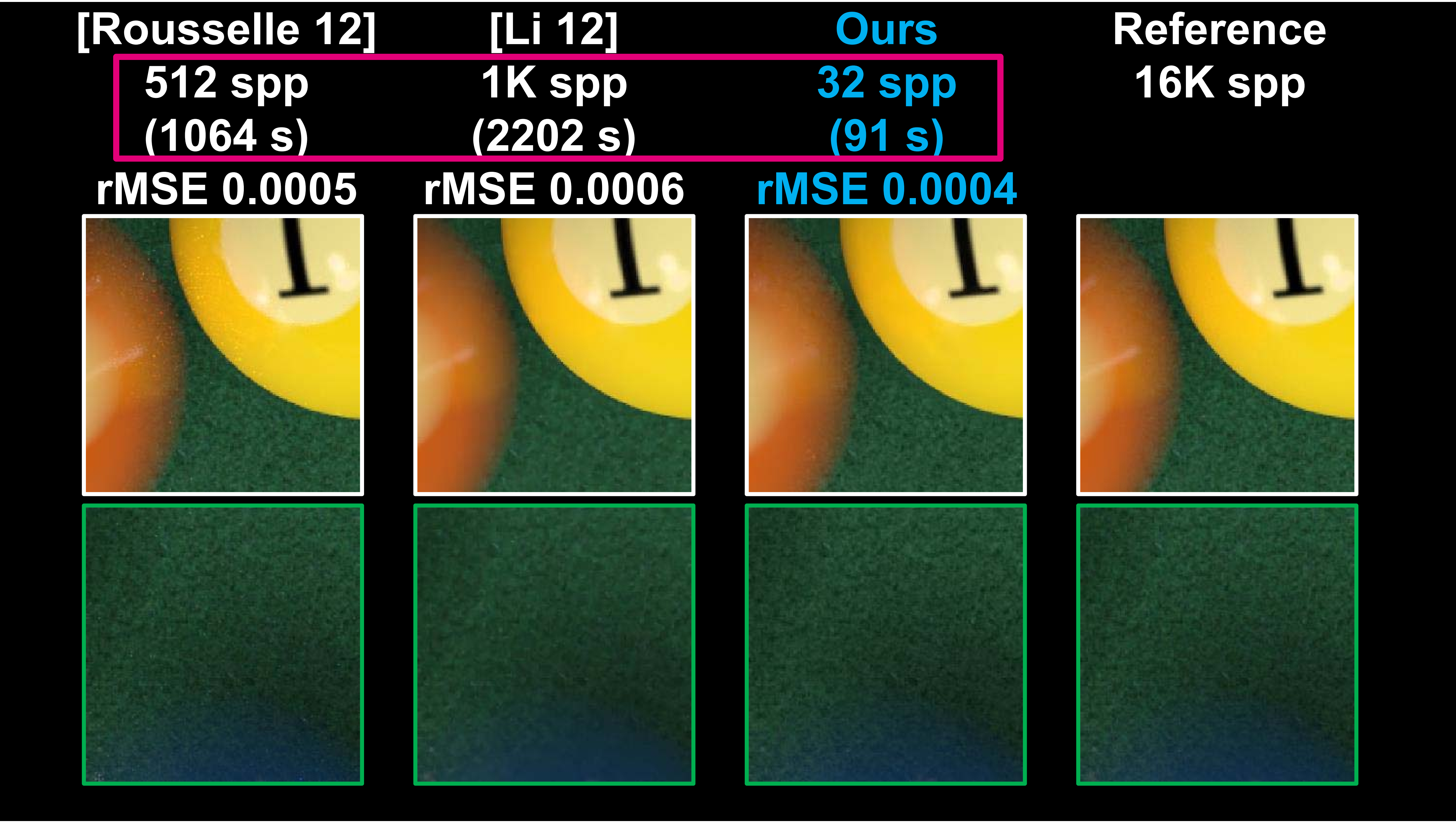


MC
128 spp (665 s)
rMSE 0.063
[Rousselle 12]
115 spp
rMSE 0.012
[Li 12]
113 spp
rMSE 0.015
Ours
115 spp
rMSE 0.004
Reference
16K spp

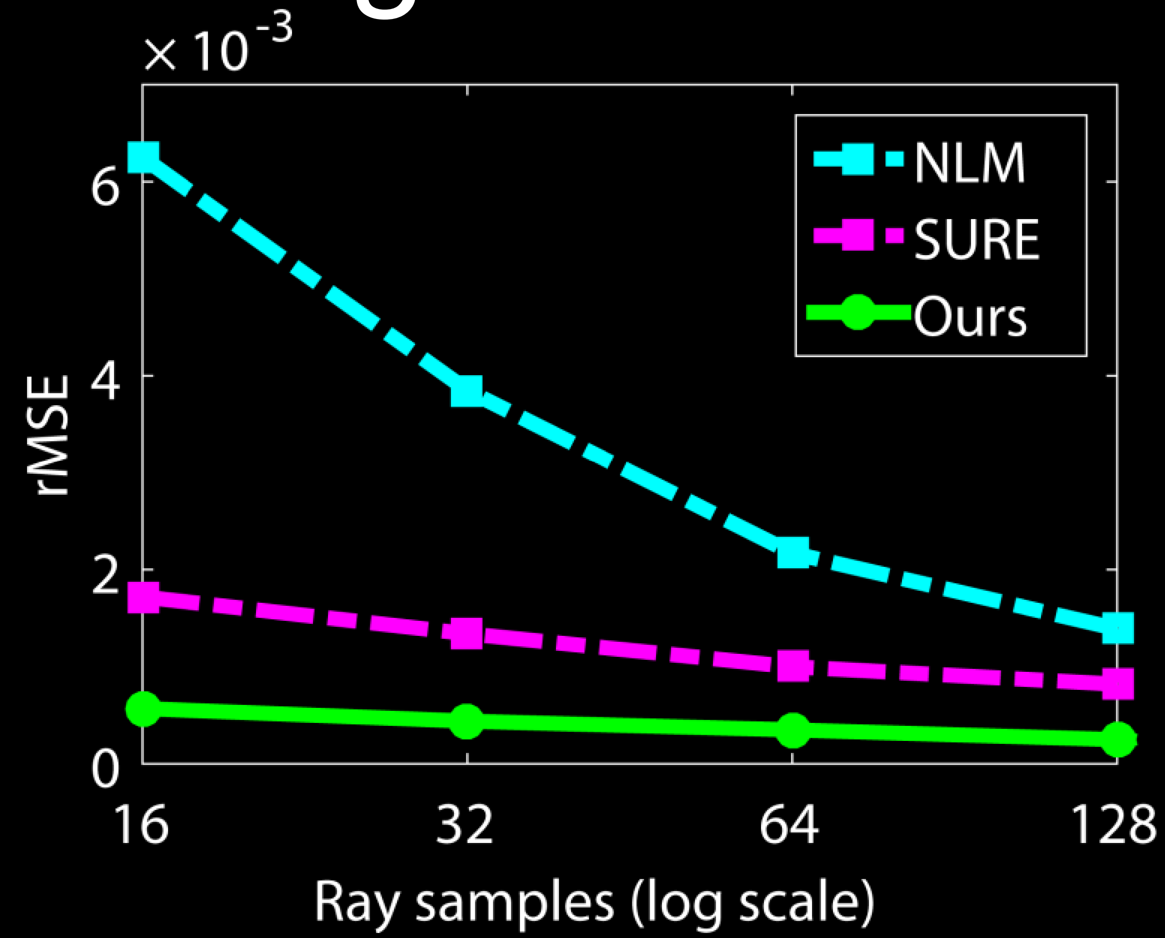
Equal-Quality Comparison

- Pool
  - Path traced
  - Motion blur
  - Noisy textured floor

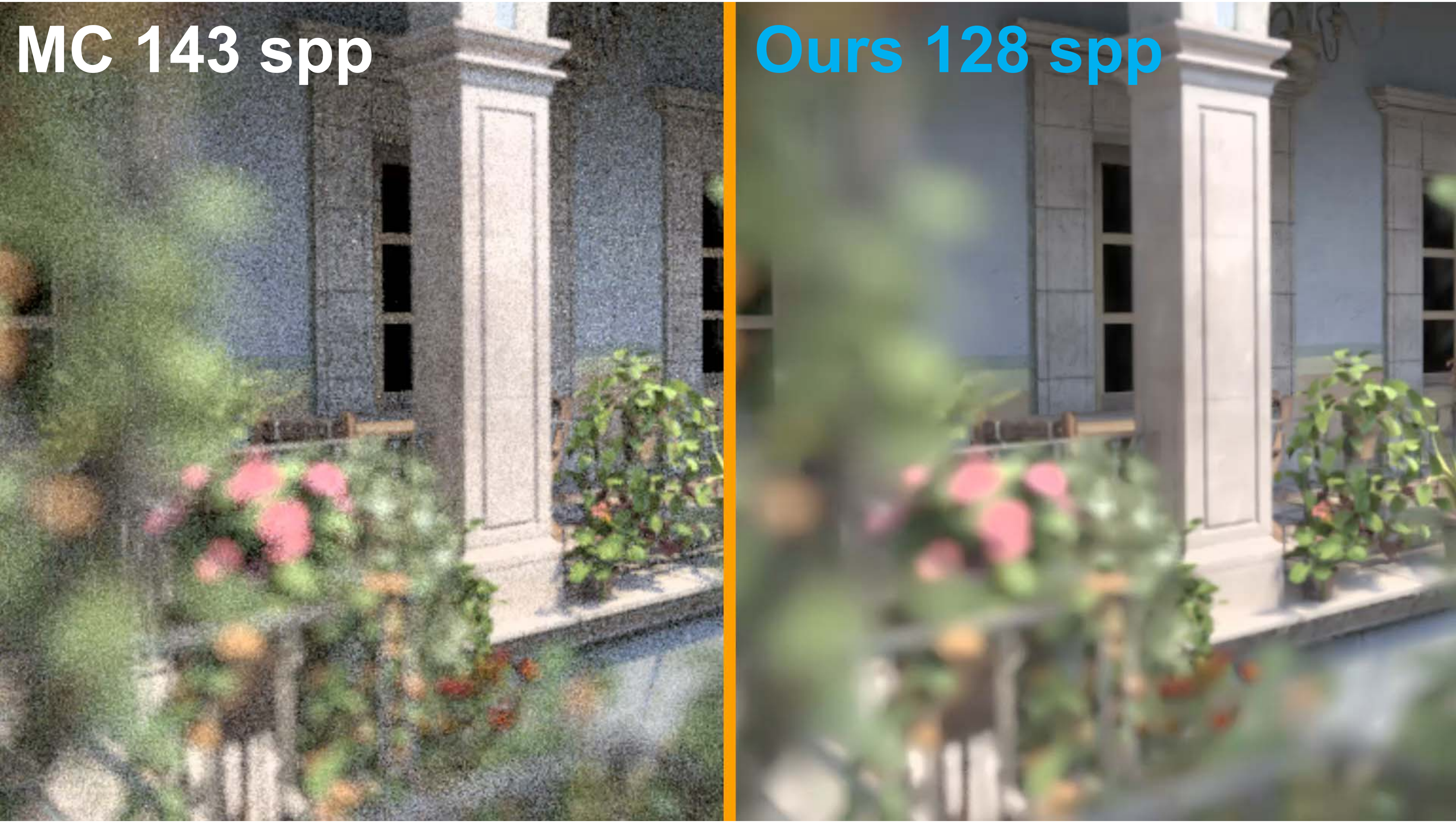



[Rousselle 12]
512 spp
(1064 s)
rMSE 0.0005
[Li 12]
1K spp
(2202 s)
rMSE 0.0006
Ours
32 spp
(91 s)
rMSE 0.0004
Reference
16K spp

MSE Convergence



RESULTS (ANIMATIONS)


MC 143 spp
Ours 128 spp


MC 153 spp
Ours 128 spp

SIGGRAPH2015
Xroads of Discovery
The 42nd International Conference and Exhibition on Computer Graphics and Interactive Techniques

Adaptive Rendering with Linear Predictions

Bochang Moon[1] Jose A. Iglesias-Guitian[1] Sung-Eui Yoon[2] Kenny Mitchell[1]

Disney Research Zürich (based on Edinburgh)[1] KAIST[2]

Disney Research

KAIST

Main Idea

- Approximate images with a sparse number of linear models
  - Apply expensive reconstruction only at a sparse number of pixels
  - Predict most of pixels from linear models

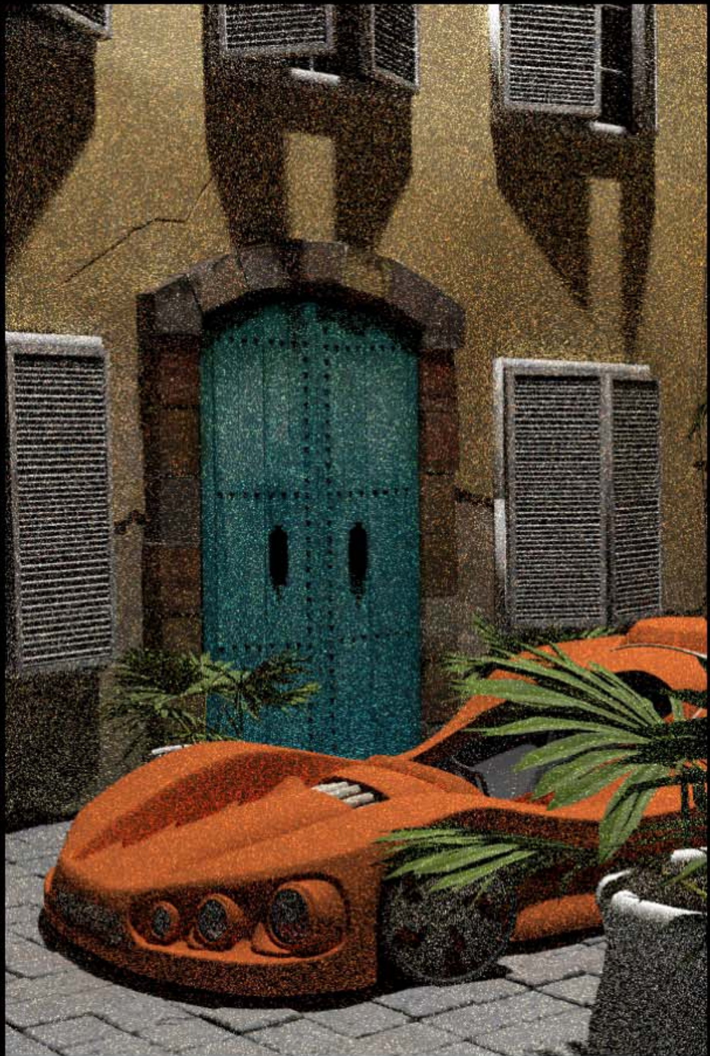
[Moon et al. 14] vs. Ours

- The previous work performs reconstruction at each pixel
  – i.e., Number of models = pixel count
- We conduct reconstruction at a sparse number of pixels
  – Reduce its computational overhead (e.g., 28X)

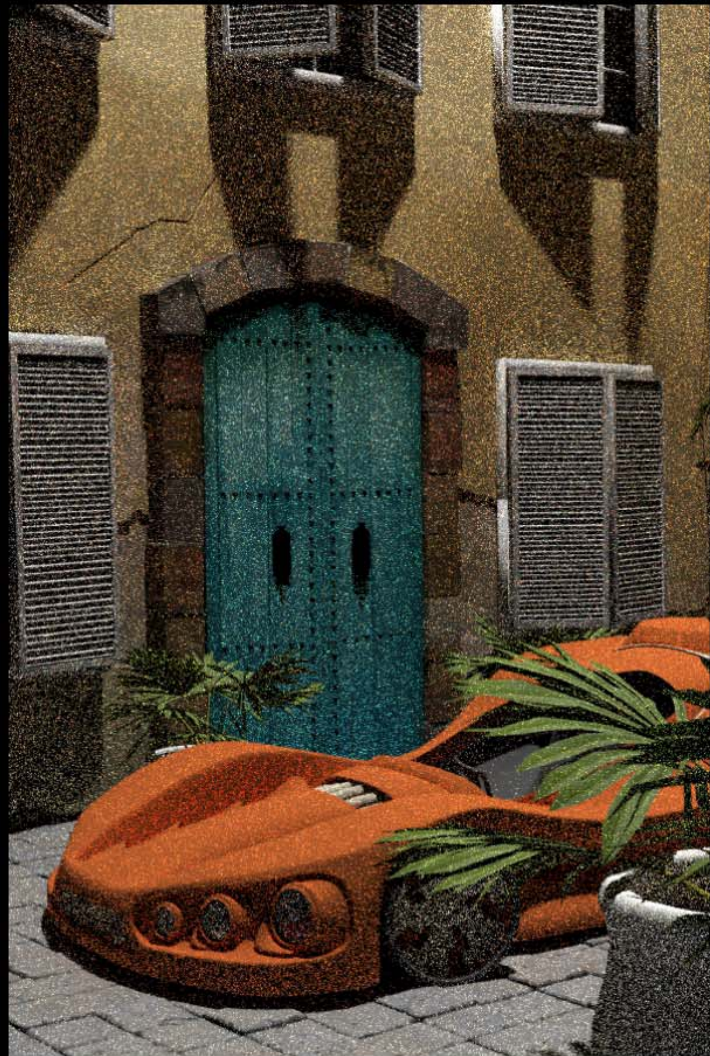
Previous Methods

Our Method

Previous Methods



Our Method



Equal-Time Comparisons

[Rousselle 12]
62 spp (57.2 s)
rMSE 0.0012

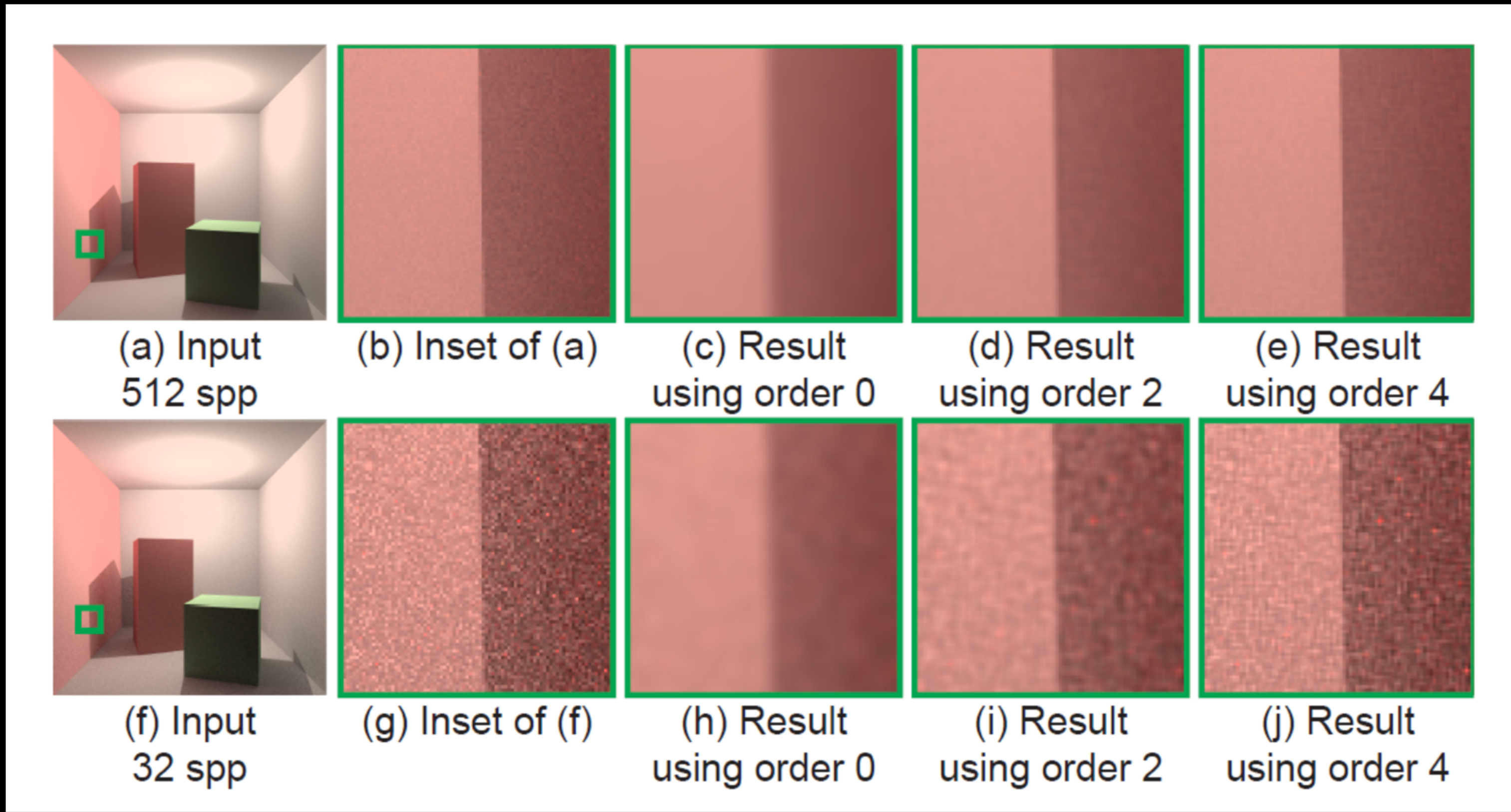
[Moon 14]
45 spp
rMSE 0.0019

Ours
76 spp
rMSE 0.0009
Sparsity (13.3%)

Reference
32K spp

Adaptive Polynomial Rendering [SIG 16]

- Find the optimal order of the polynomial functions



Denoising with Kernel Prediction and Asymmetric Loss Functions

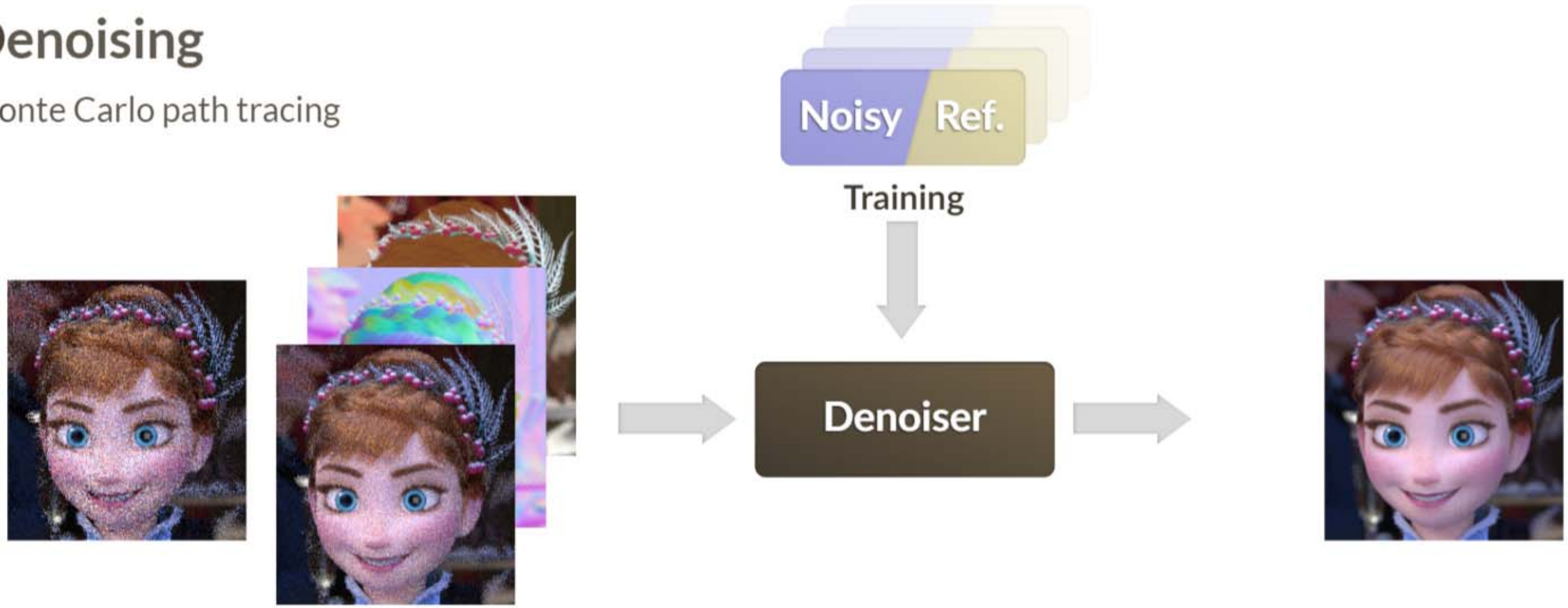
Thijs Vogels, Fabrice Rousselle, Brian McWilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, Jan Novák

SIGGRAPH 2018

Adopted from authors' slide

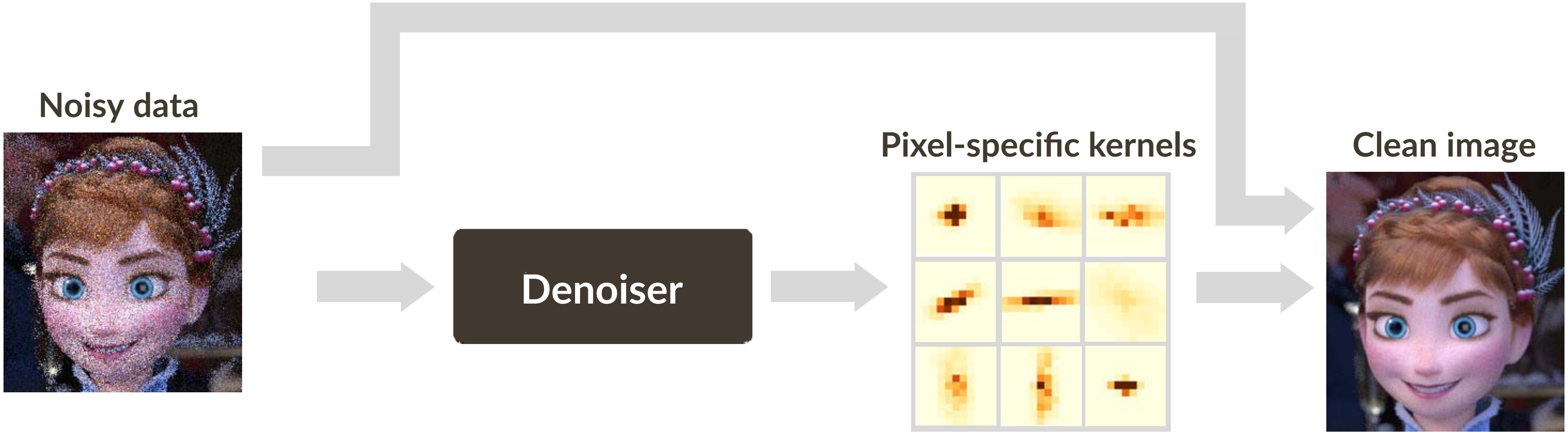
Denoising

Monte Carlo path tracing

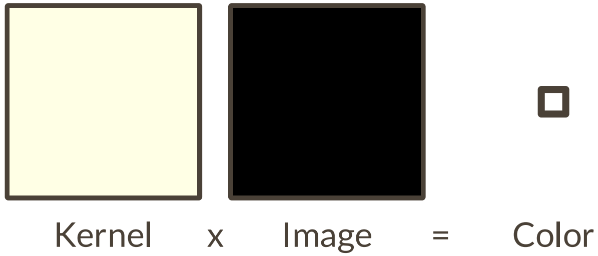


Kernel Prediction

Basic single-frame denoiser (Bako et al. 2017)



© Disney



Multi-renderer support

Baseline generalization ability

Renderer B Input

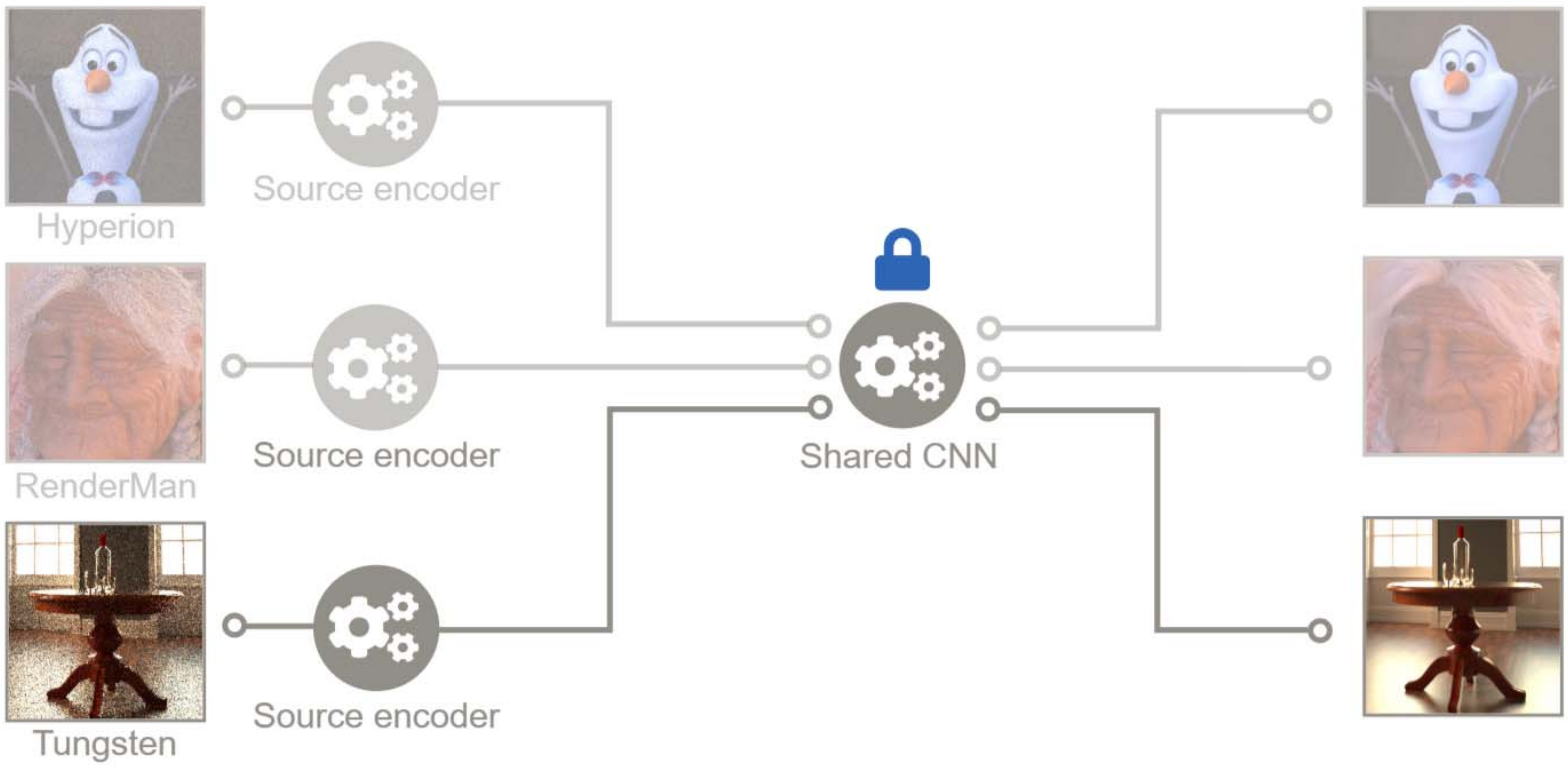
Trained on Renderer A data

Retrained on Renderer B data

Scene by nacimus (Blendswap)

Multi-renderer support

Source-aware encoders



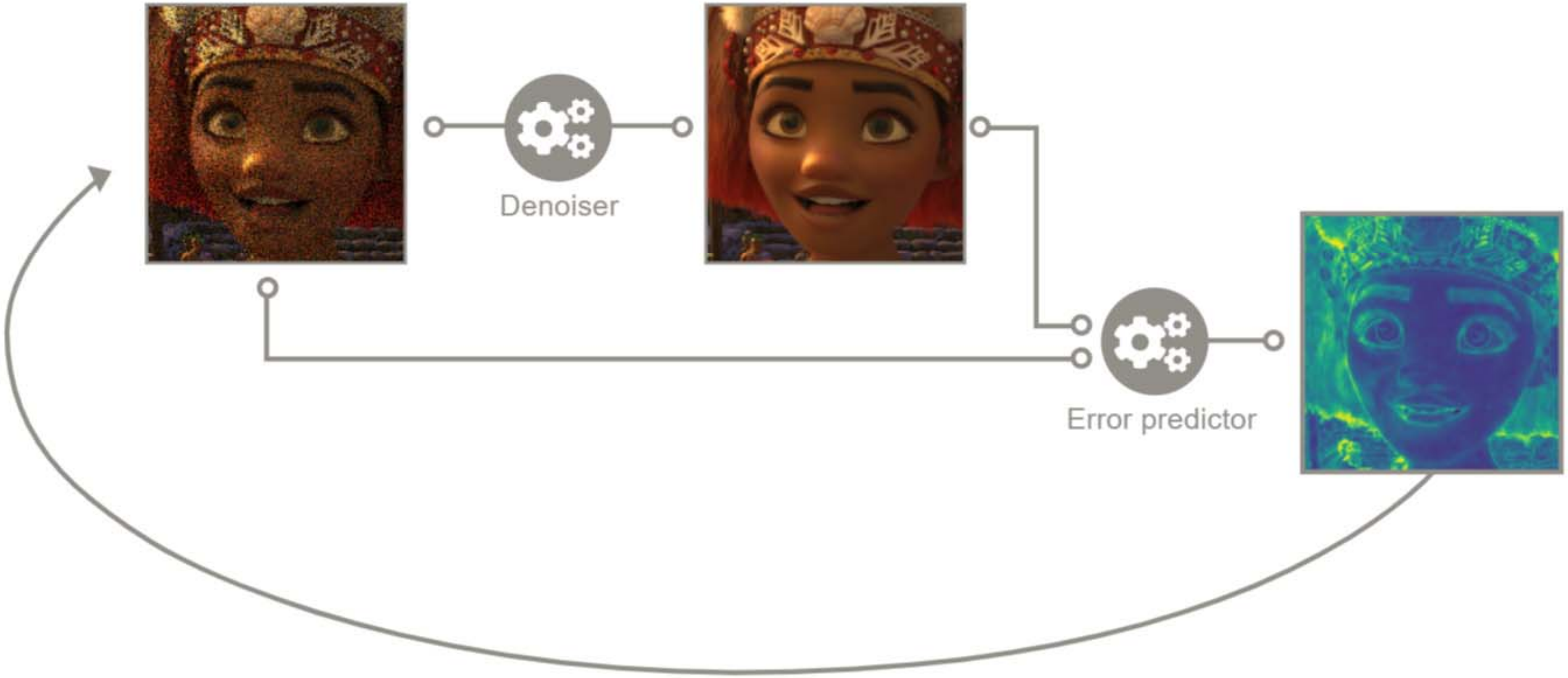
Temporal denoising

Reusing a pre-trained single-frame denoiser

Video Comparisons:
https://tvogels.nl/kpal/slides/#/5/5

Adaptive sampling

with an error predictor




Predicted sampling map

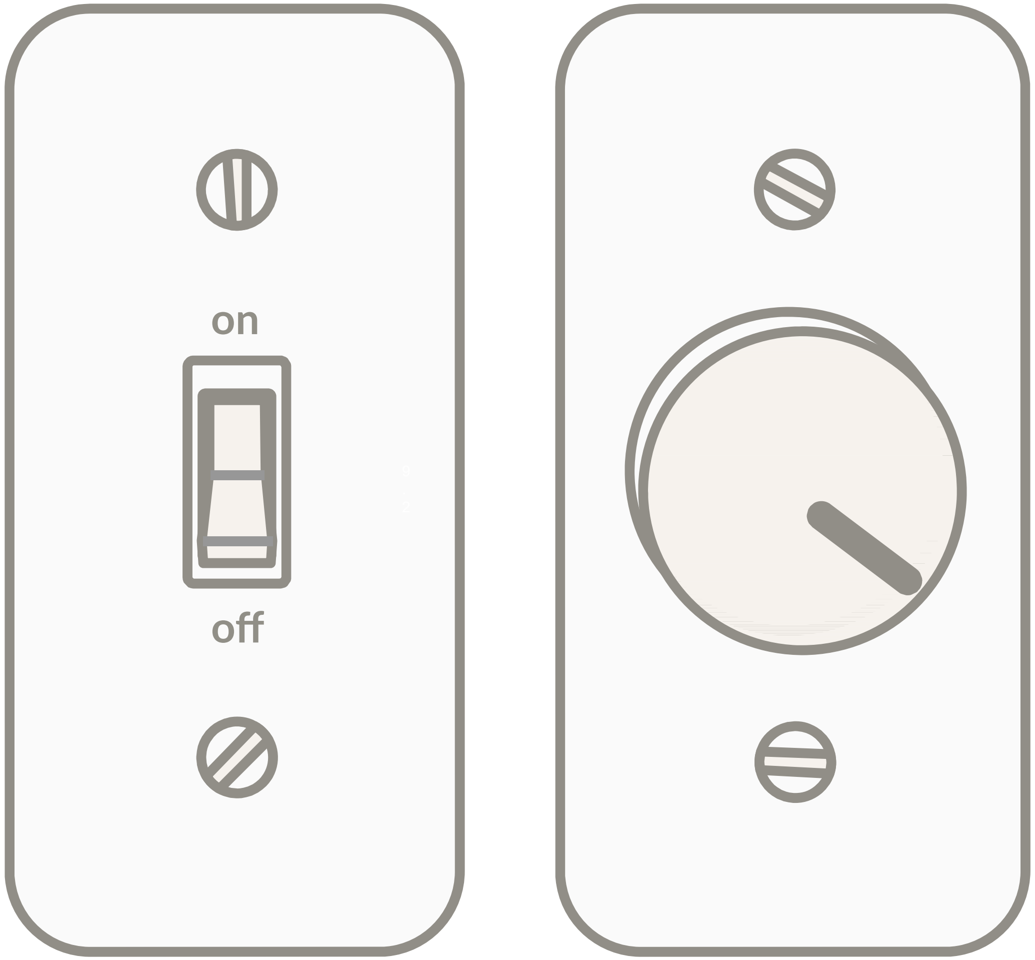
MULTI-RENDERER SUPPORT
source-aware encoders

TEMPORAL STABILITY
cross-frame denoising

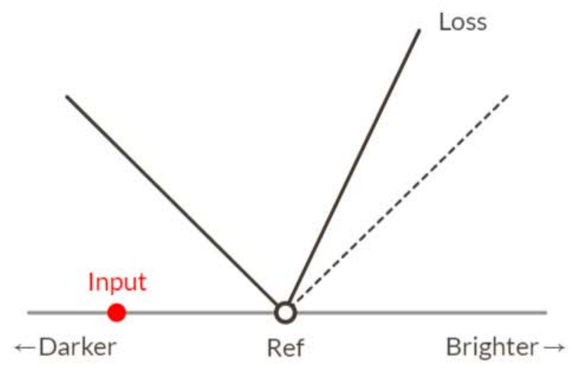
DENOISING OF LOW FREQUENCIES
a multi-scale approach

ADAPTIVE SAMPLING
the error predictor

USER CONTROL
asymmetric loss functions

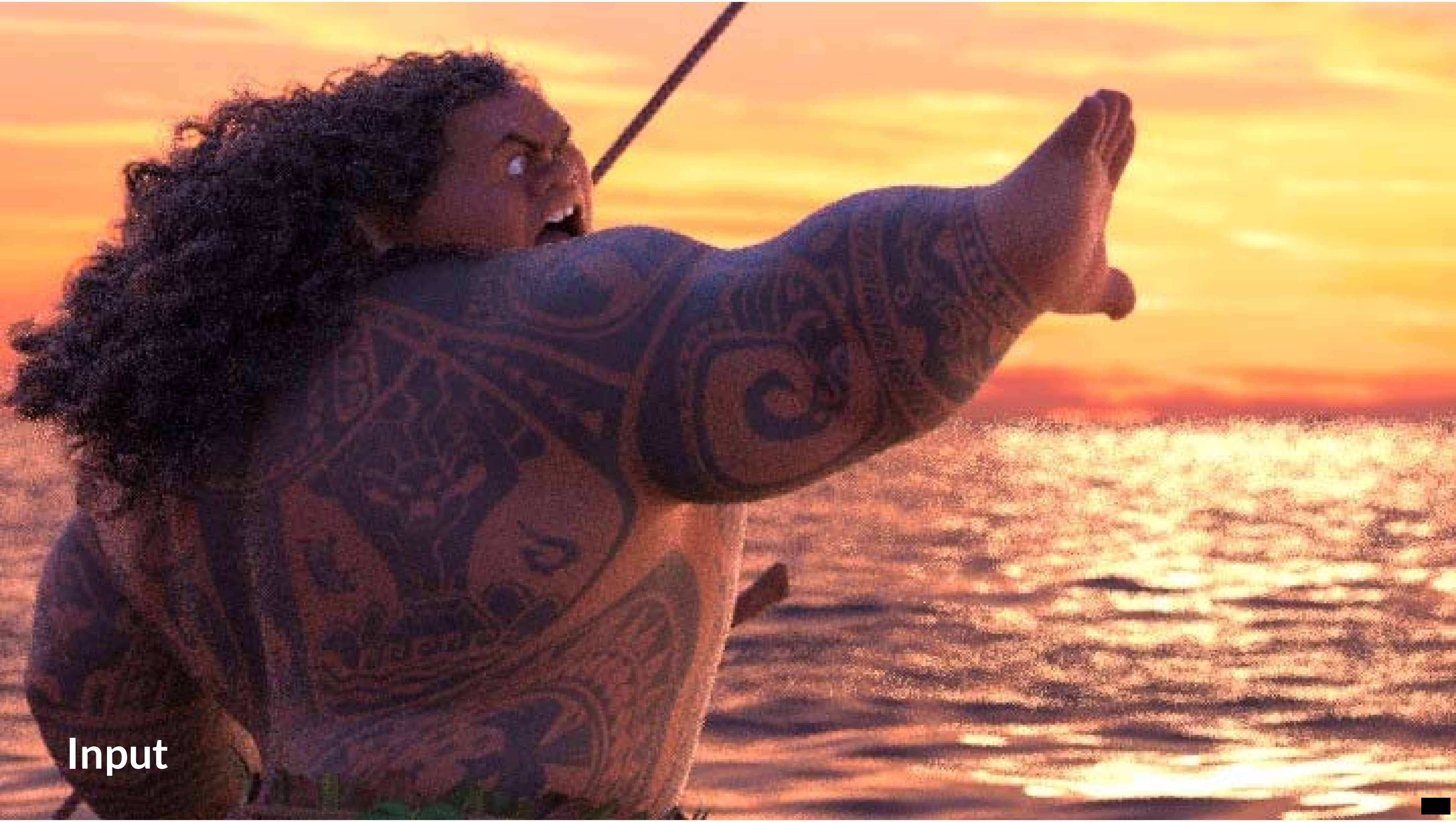

on
off

© Disney / Pixar


Loss
Input
←Darker
Ref
Brighter→


Input


Denoised


Input


Denoised