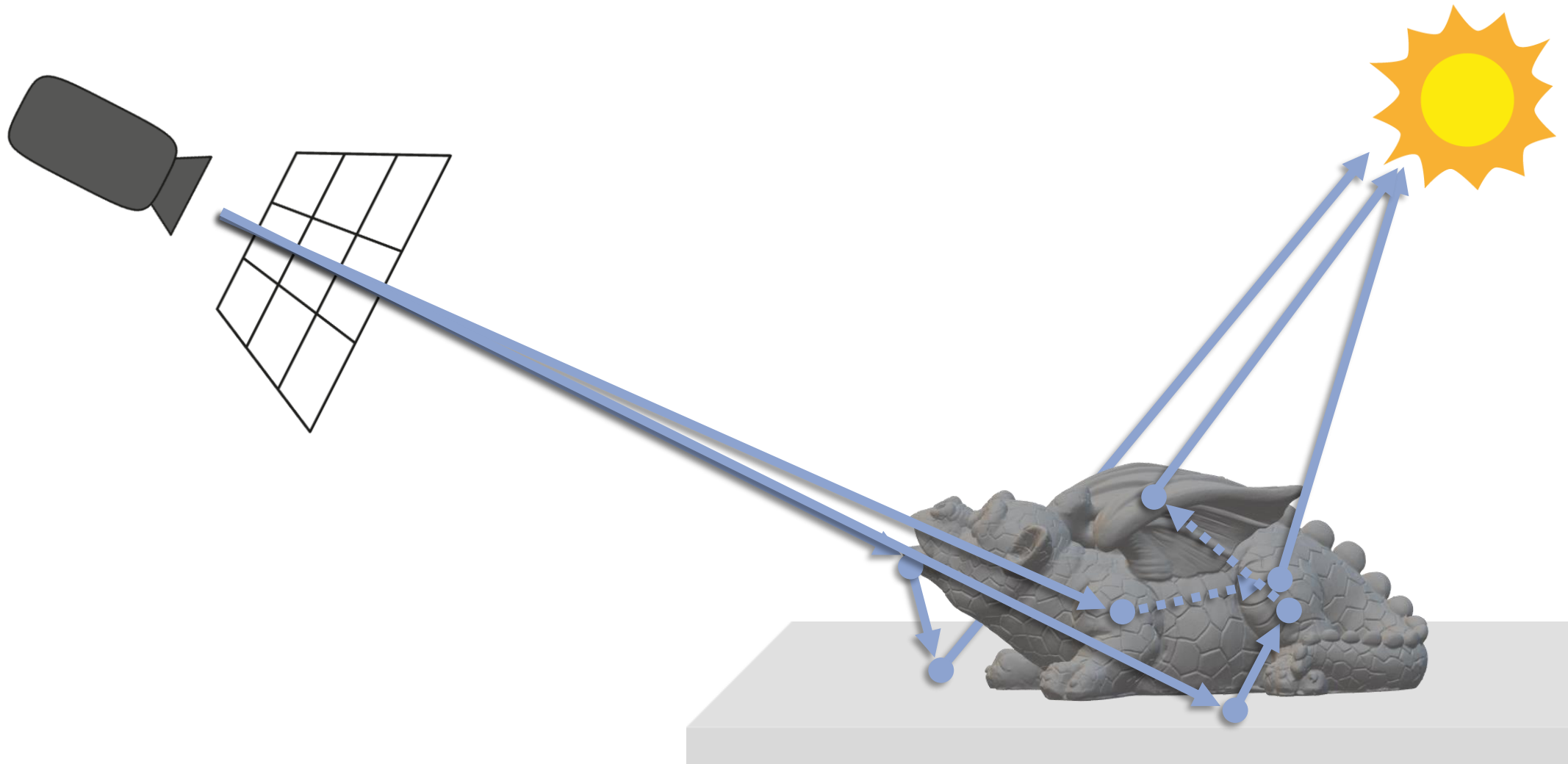


# SELF-SUPERVISED POST-CORRECTION FOR MONTE CARLO DENOISING

JONGHEE BACK<sup>1</sup> BINH-SON HUA<sup>2</sup> TOSHIYA HACHISUKA<sup>3</sup> BOCHANG MOON<sup>1</sup>  
GIST<sup>1</sup> VINAI RESEARCH<sup>2</sup> UNIVERSITY OF WATERLOO<sup>3</sup>

# MONTE CARLO PATH TRACING



# MONTE CARLO PATH TRACING



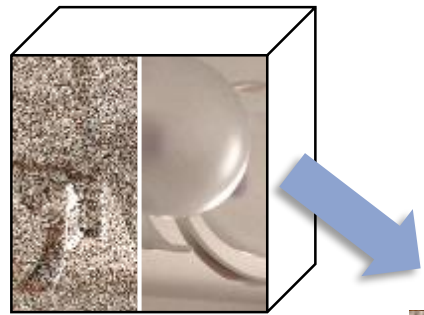
# MONTE CARLO PATH TRACING

Path tracing, 128 spp



Path tracing, 64K spp

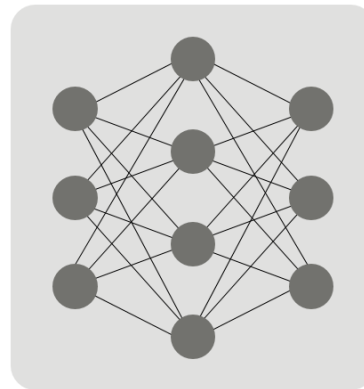
- Supervised learning based Monte Carlo denoising



Training dataset  
(noisy-reference pairs)

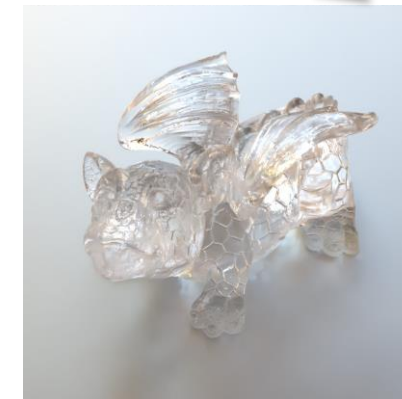


Pretraining step

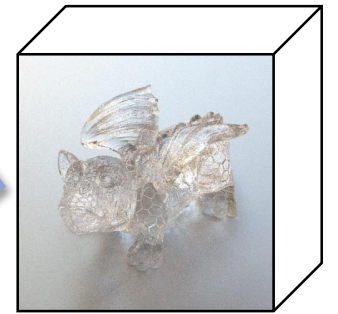


Neural network

Inference step



Denoised image



Noisy test image

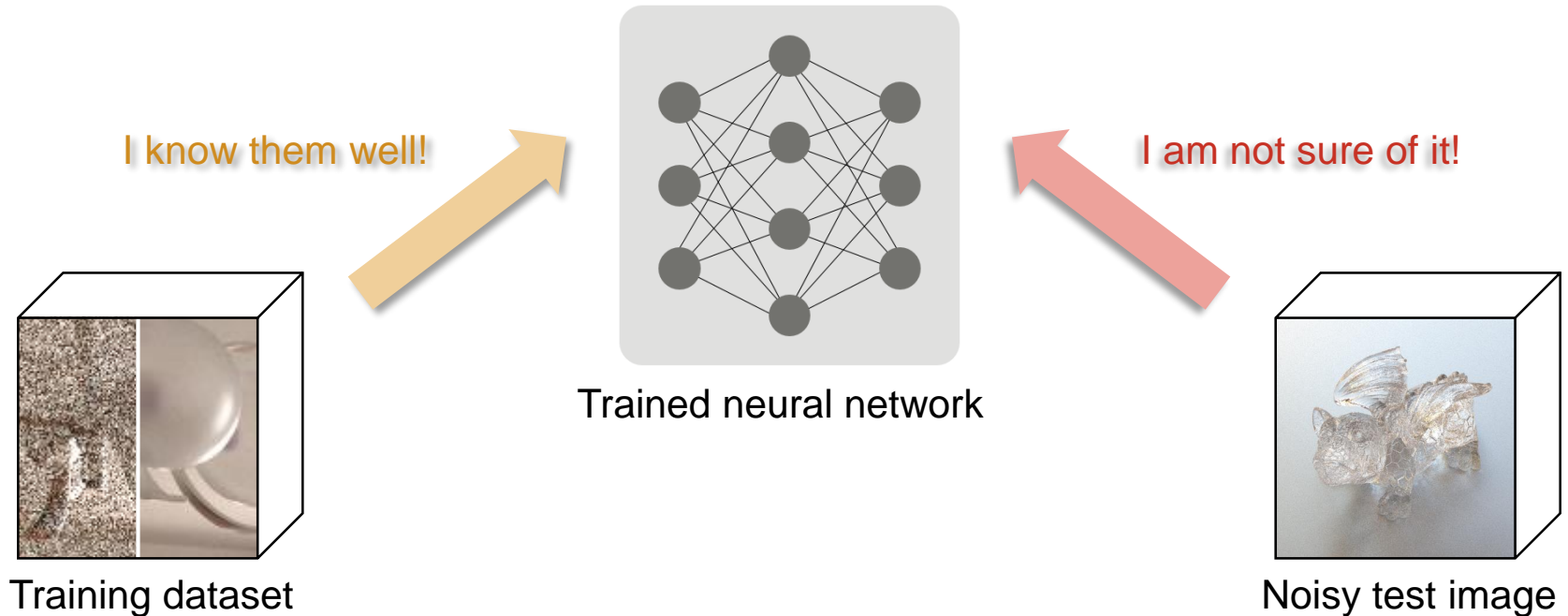
# MONTE CARLO DENOISING

Path tracing, 128 spp



Supervised MC denoising  
(AFGSA [Yu et al. 21])

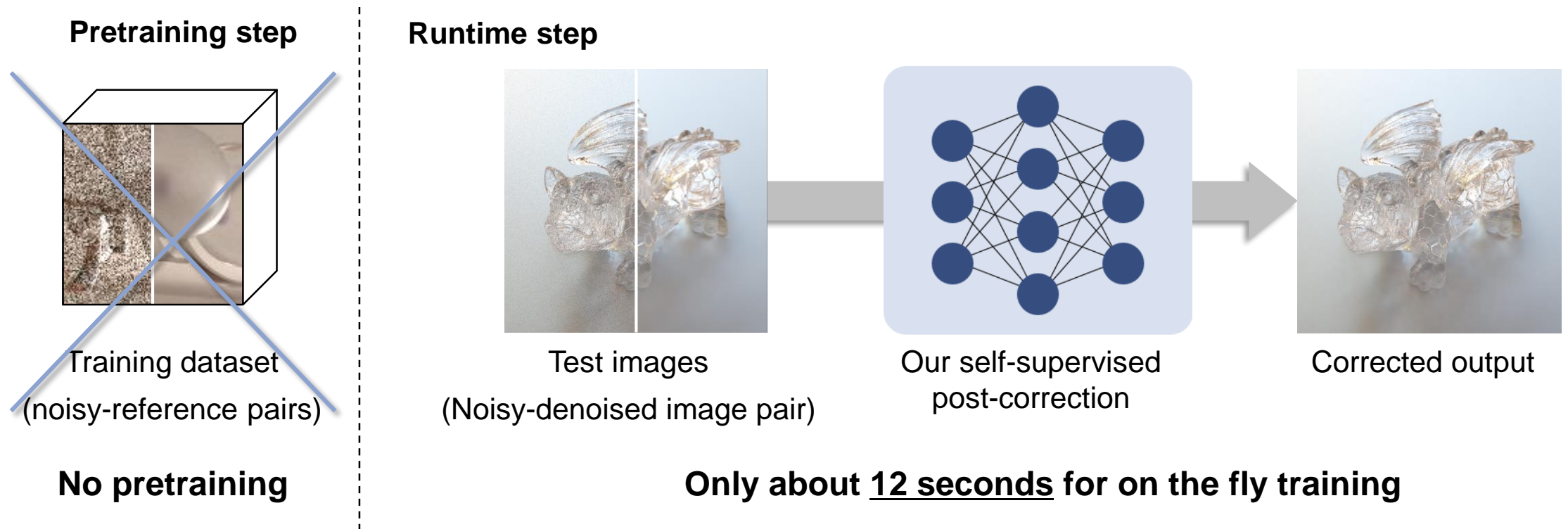
- Not practical to assume that training dataset covers all possible scenarios
  - The test image can be different from the training dataset.



- Not practical to assume that training dataset covers all possible scenarios
  - The test image can be different from the training dataset.



- We propose a novel post-correction framework for learning-based denoising.
  - Our work is the first to apply self-supervised learning to Monte Carlo denoising.



- We propose a novel post-correction framework for learning-based denoising.
  - Our work is the first to apply self-supervised learning to Monte Carlo denoising.

AFGSA, 128 spp



Reference, 64K spp

- We propose a novel post-correction framework for learning-based denoising.
  - Our work is the first to apply self-supervised learning to Monte Carlo denoising.



- (Offline) Monte Carlo denoising & post-denoising
  - Pretraining-based optimization
    - Kalantari [2015], Bako [2017], Gharbi [2019], Kettunen [2019], Xu [2019], Guo [2019], Back [2020], Yu [2021]
  - Mean squared error (MSE) based optimization
    - Li [2012], Rousselle [2012, 2013], Moon [2014, 2016], Bitterli [2016], Zheng [2021]

	<b>Pretraining-based</b>	<b>MSE-based</b>	<b>Ours</b>
Neural network?	O	X	O
Optimization using test dataset?	X	O	O

- Optimizing a neural network that corrects a supervised denoising method using a test image pair

Minimizing the actual error  $\|\hat{\mu}_c - \mu_c\|^2$

$\hat{\mu}_c$  : post-corrected estimate at pixel  $c$

$\mu_c$  : ground truth at pixel  $c$

- Optimizing a neural network that corrects a supervised denoising method using a test image pair

Minimizing the actual error  $\|\hat{\mu}_c - \mu_c\|^2$



This is unknown in the context of self-supervised learning.

$\hat{\mu}_c$  : post-corrected estimate at pixel  $c$

$\mu_c$  : ground truth at pixel  $c$

- Optimizing a neural network that corrects a supervised denoising method using a test image pair

Minimizing the actual error  $\|\hat{\mu}_c - \mu_c\|^2$



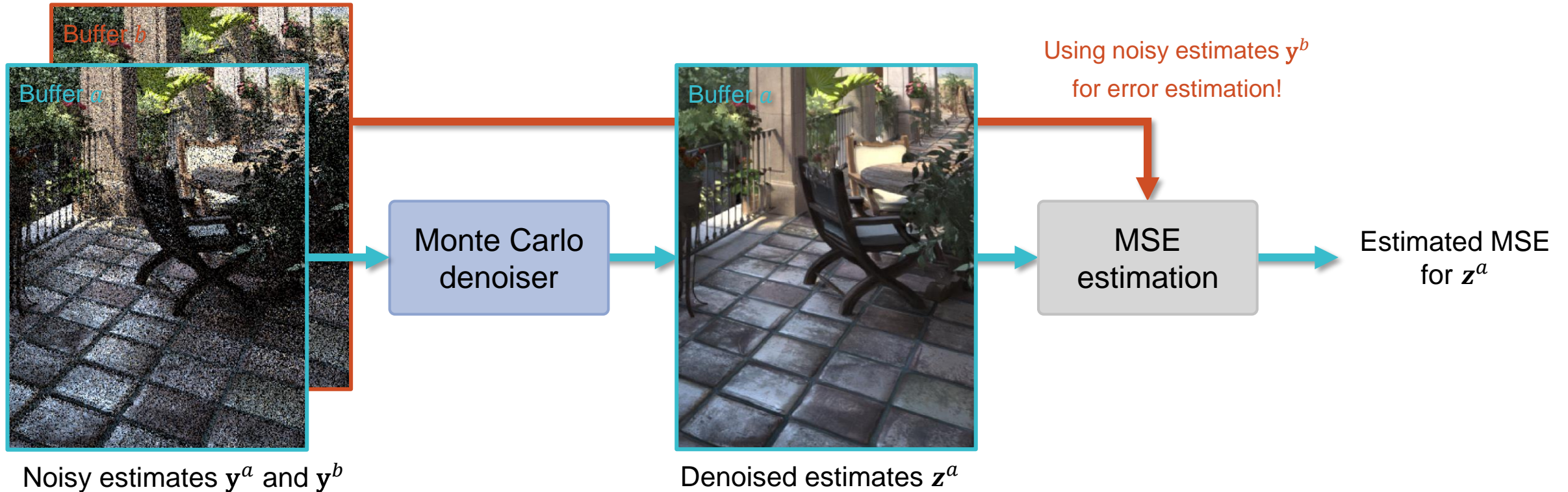
Estimating  $E\|\hat{\mu}_c - \mu_c\|^2$  instead and minimizing it!  
mean squared error (MSE)

$\hat{\mu}_c$  : post-corrected estimate at pixel  $c$

$\mu_c$  : ground truth at pixel  $c$

# PROPOSED METHOD: SELF-SUPERVISED LOSS

- Dual-buffer scheme for estimating denoising errors [Bitterli et al. 16]



- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\mathcal{L}(\hat{\mu}_c^a) = \frac{E \|\hat{\mu}_c^a - \mu_c\|^2}{\bar{\mu}_c^2 + 0.01}$$

Relative  $L_2$  error [Rousselle et al. 11]

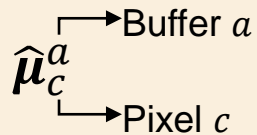
$\hat{\mu}$  : post-corrected estimates

$\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates

$\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$

$\mu$  : ground truth

$\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\mu, \mathbf{y}$  and  $\mathbf{z}$



- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\mathcal{L}(\hat{\mu}_c^a) = \frac{E \|\hat{\mu}_c^a - \mu_c\|^2}{\bar{\mu}_c^2 + 0.01}$$

Relative  $L_2$  error [Rousselle et al. 11]

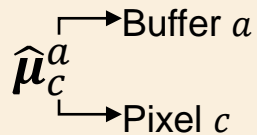
$\hat{\mu}$  : post-corrected estimates

$\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates

$\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$

$\boldsymbol{\mu}$  : ground truth

$\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\boldsymbol{\mu}, \mathbf{y}$  and  $\mathbf{z}$



- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\mathcal{L}(\hat{\mu}_c^a) \approx \frac{\|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2 - \mathbf{1}^T \hat{\sigma}^2(\mathbf{y}_c^b)}{\bar{\mu}_c^2 + 0.01}$$

Relative  $L_2$  error [Rousselle et al. 11]

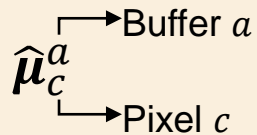
$\hat{\mu}$  : post-corrected estimates

$\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates

$\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$

$\boldsymbol{\mu}$  : ground truth

$\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\boldsymbol{\mu}, \mathbf{y}$  and  $\mathbf{z}$



Unbiased estimate of the MSE

$$E\|\hat{\mu}_c^a - \boldsymbol{\mu}_c\|^2 \approx \|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2 - \mathbf{1}^T \hat{\sigma}^2(\mathbf{y}_c^b)$$

(Please see our supplementary report!)

# PROPOSED METHOD: SELF-SUPERVISED LOSS

- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\mathcal{L}(\hat{\mu}_c^a) \approx \frac{\|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2 - \mathbf{1}^T \hat{\sigma}^2(\mathbf{y}_c^b)}{\boxed{\bar{\mu}_c^2 + 0.01}}$$

Relative  $L_2$  error [Rousselle et al. 11]

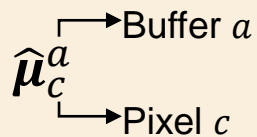
$\hat{\mu}$  : post-corrected estimates

$\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates

$\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$

$\boldsymbol{\mu}$  : ground truth

$\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\boldsymbol{\mu}, \mathbf{y}$  and  $\mathbf{z}$

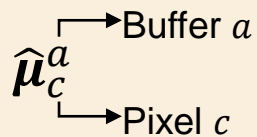


- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\mathcal{L}(\hat{\mu}_c^a) \approx \frac{\|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2 - \mathbf{1}^T \hat{\sigma}^2(\mathbf{y}_c^b)}{(\bar{z}_c^b)^2 + 0.01}$$

Relative  $L_2$  error [Rousselle et al. 11]

$\hat{\mu}$  : post-corrected estimates  
 $\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates  
 $\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$   
 $\boldsymbol{\mu}$  : ground truth  
 $\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\boldsymbol{\mu}, \mathbf{y}$  and  $\mathbf{z}$



Practical choice of the denominator

1. Making it statistically independent of  $\hat{\mu}_c^a$
2. Selecting the accurate one of candidates  $\bar{y}_c^b$  and  $\bar{z}_c^b$

# PROPOSED METHOD: SELF-SUPERVISED LOSS

- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\mathcal{L}(\hat{\mu}_c^a) \approx \frac{\|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2 - \mathbf{1}^T \hat{\sigma}^2(\mathbf{y}_c^b)}{(\bar{z}_c^b)^2 + 0.01}$$

Relative  $L_2$  error [Rousselle et al. 11]

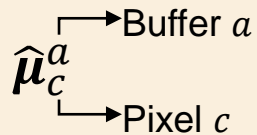
$\hat{\mu}$  : post-corrected estimates

$\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates

$\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$

$\boldsymbol{\mu}$  : ground truth

$\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\boldsymbol{\mu}, \mathbf{y}$  and  $\mathbf{z}$



- Let's derive a self-supervised loss for post-corrected estimate  $\hat{\mu}_c^a$  first.

$$\hat{\mathcal{L}}(\hat{\mu}_c^a) = \frac{\|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2}{(\bar{z}_c^b)^2 + 0.01}$$

Derived loss for the output estimate  $\hat{\mu}_c^a$

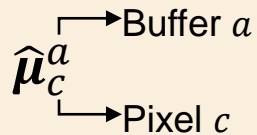
$\hat{\mu}$  : post-corrected estimates

$\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates

$\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$

$\mu$  : ground truth

$\bar{\mu}, \bar{y}, \bar{z}$ : intensity of  $\mu, \mathbf{y}$  and  $\mathbf{z}$



# PROPOSED METHOD: SELF-SUPERVISED LOSS

- Neural network optimization using our self-supervised loss

$$\theta^* = \operatorname{argmin}_{\theta} \frac{1}{3N} \sum_{c=1}^N 0.5 \left( \hat{\mathcal{L}}(\hat{\mu}_c^a) + \hat{\mathcal{L}}(\hat{\mu}_c^b) \right)$$

$$\hat{\mathcal{L}}(\hat{\mu}_c^a) = \frac{\|\hat{\mu}_c^a - \mathbf{y}_c^b\|^2}{(\bar{z}_c^b)^2 + 0.01}$$

Loss for the output estimate  $\hat{\mu}_c^a$

$$\hat{\mathcal{L}}(\hat{\mu}_c^b) = \frac{\|\hat{\mu}_c^b - \mathbf{y}_c^a\|^2}{(\bar{z}_c^a)^2 + 0.01}$$

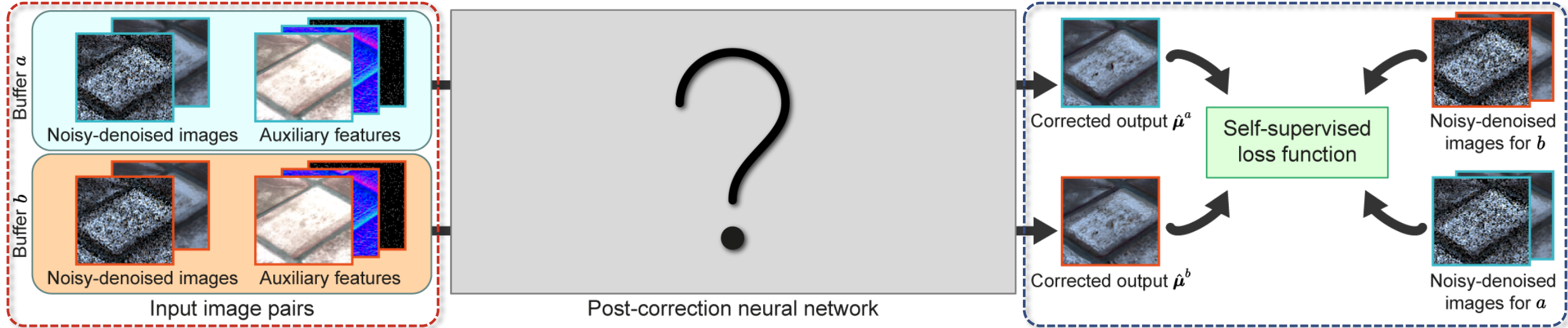
Loss for the output estimate  $\hat{\mu}_c^b$

$\hat{\mu}$  : post-corrected estimates  
 $\mathbf{y}, \mathbf{z}$  : noisy/denoised estimates  
 $\hat{\sigma}^2(\mathbf{y})$  : estimated variance of  $\mathbf{y}$   
 $\boldsymbol{\mu}$  : ground truth  
 $\bar{\mu}, \bar{y}, \bar{z}$  : intensity of  $\boldsymbol{\mu}, \mathbf{y}$  and  $\mathbf{z}$

$\hat{\mu}_c^a$  → Buffer  $a$   
Pixel  $c$

# PROPOSED METHOD: PRACTICAL NETWORK

- Overall framework



Which network is a good fit for this problem?

# PROPOSED METHOD: PRACTICAL NETWORK

- Result of self-supervised learning with the existing model from DC [Back et al. 20]
  - The model results in overfitting to noise!



$relL_2$  0.034826

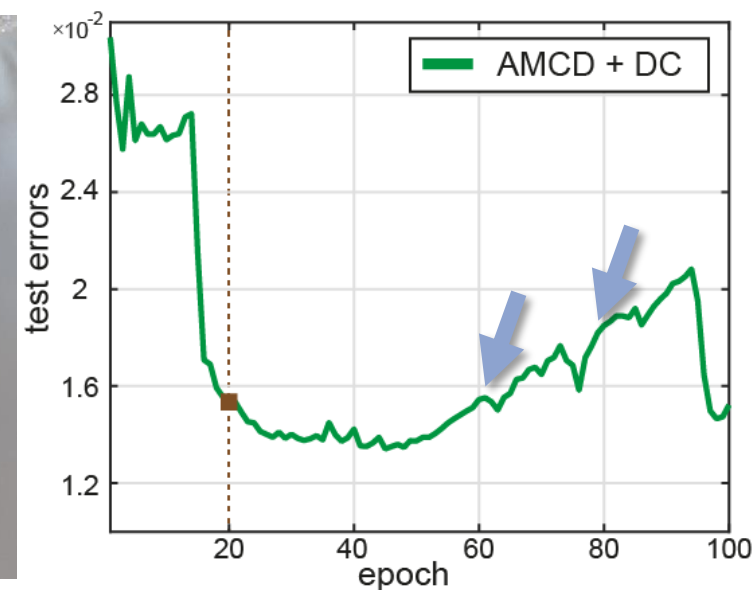
AMCD [Xu et al. 19]  
(64 spp)

$relL_2$  0.013846

AMCD + DC  
(using our loss)

64K spp

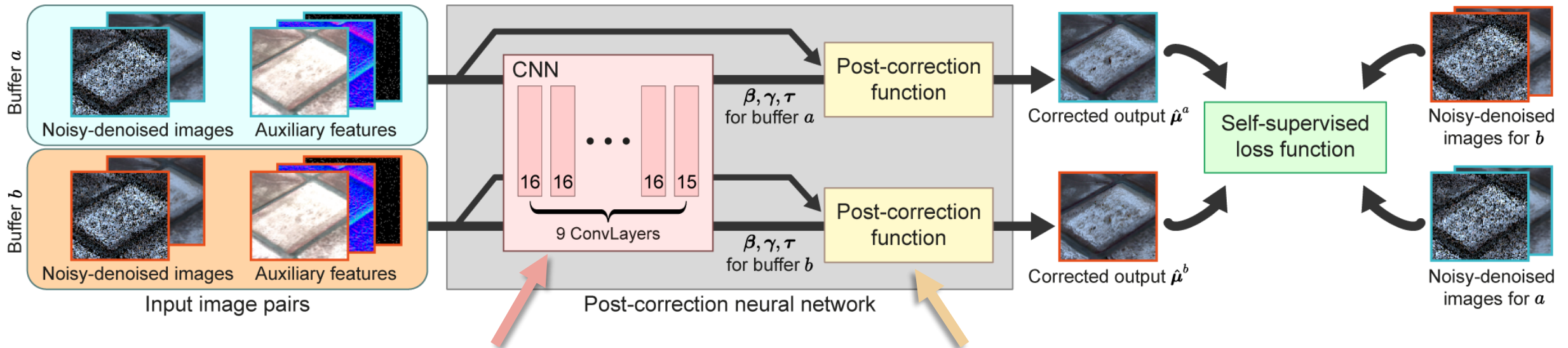
Reference



Convergence graph

# PROPOSED METHOD: PRACTICAL NETWORK

- Practical post-correction neural network for self-supervised learning



Lightweight network  
(only 0.02M parameters)

Our post-correction function  
refined from [Back et al. 20]

$\Omega_c$  : local window centered at pixel  $c$   
 $w_i$  : combination weight at pixel  $i$

- Revisit: combination function [Back et al. 20]

$$f_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \mathbf{y}_i + \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i (\mathbf{z}_c - \mathbf{z}_i)$$

# PROPOSED METHOD: PRACTICAL NETWORK

$\Omega_c$  : local window centered at pixel  $c$   
 $w_i$  : combination weight at pixel  $i$

- Revisit: combination function [Back et al. 20]

$$f_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \mathbf{y}_i + \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i (\mathbf{z}_c - \mathbf{z}_i)$$

Locally weighted average of  $\mathbf{y}_i$  and  $\mathbf{z}_c - \mathbf{z}_i$

# PROPOSED METHOD: PRACTICAL NETWORK

$\Omega_c$  : local window centered at pixel  $c$   
 $w_i$  : combination weight at pixel  $i$

- Revisit: combination function [Back et al. 20]

$$f_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \mathbf{y}_i + \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i (\mathbf{z}_c - \mathbf{z}_i)$$

Per-pixel weights  $w_i$  are produced by a neural network.

$\rho, \mathbf{n}, v$  : auxiliary features (albedo/normal/visibility buffers)  
 $\beta_c^z, \beta_c^\rho, \beta_c^n$  : scale parameters at pixel  $c$  for  $\mathbf{z}$ ,  $\rho$  and  $\mathbf{n}$

- Our post-correction function

$$g_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \{ \mathbf{y}_i + \beta_c^z \circ (\mathbf{z}_c - \mathbf{z}_i) + \beta_c^\rho \circ (\rho_c - \rho_i) + \beta_c^n \circ (\mathbf{n}_c - \mathbf{n}_i) \}$$

$\rho, \mathbf{n}, v$  : auxiliary features (albedo/normal/visibility buffers)  
 $\beta_c^z, \beta_c^\rho, \beta_c^n$  : scale parameters at pixel  $c$  for  $\mathbf{z}$ ,  $\rho$  and  $\mathbf{n}$

- Our post-correction function

$$g_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \{ \mathbf{y}_i + \beta_c^z \circ (\mathbf{z}_c - \mathbf{z}_i) + \beta_c^\rho \circ (\rho_c - \rho_i) + \beta_c^n \circ (\mathbf{n}_c - \mathbf{n}_i) \}$$

Introducing auxiliary features such as albedo and normal buffers

$\rho, \mathbf{n}, v$  : auxiliary features (albedo/normal/visibility buffers)  
 $\beta_c^z, \beta_c^\rho, \beta_c^n$  : scale parameters at pixel  $c$  for  $\mathbf{z}$ ,  $\rho$  and  $\mathbf{n}$

- Our post-correction function

$$g_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \{ \mathbf{y}_i + \beta_c^z \circ (\mathbf{z}_c - \mathbf{z}_i) + \beta_c^\rho \circ (\rho_c - \rho_i) + \beta_c^n \circ (\mathbf{n}_c - \mathbf{n}_i) \}$$

Relative importance between difference terms

$\rho, \mathbf{n}, v$  : auxiliary features (albedo/normal/visibility buffers)  
 $\beta_c^z, \beta_c^\rho, \beta_c^n$  : scale parameters at pixel  $c$  for  $\mathbf{z}$ ,  $\rho$  and  $\mathbf{n}$

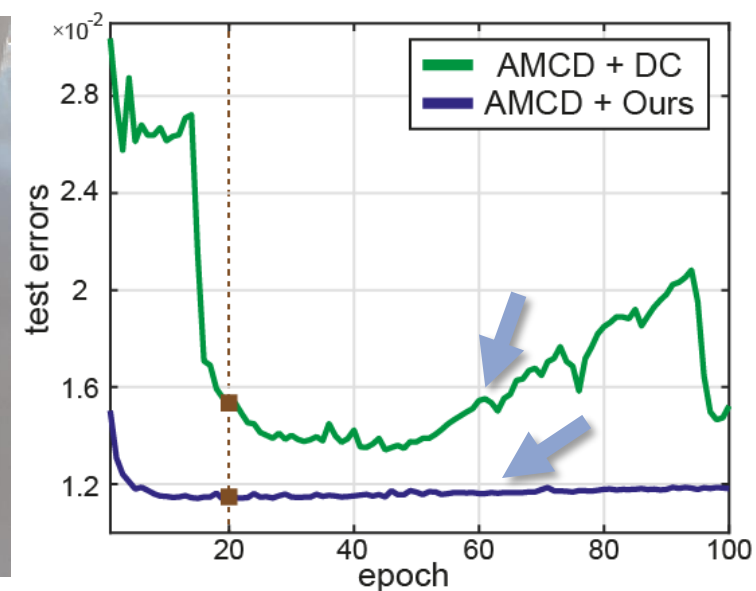
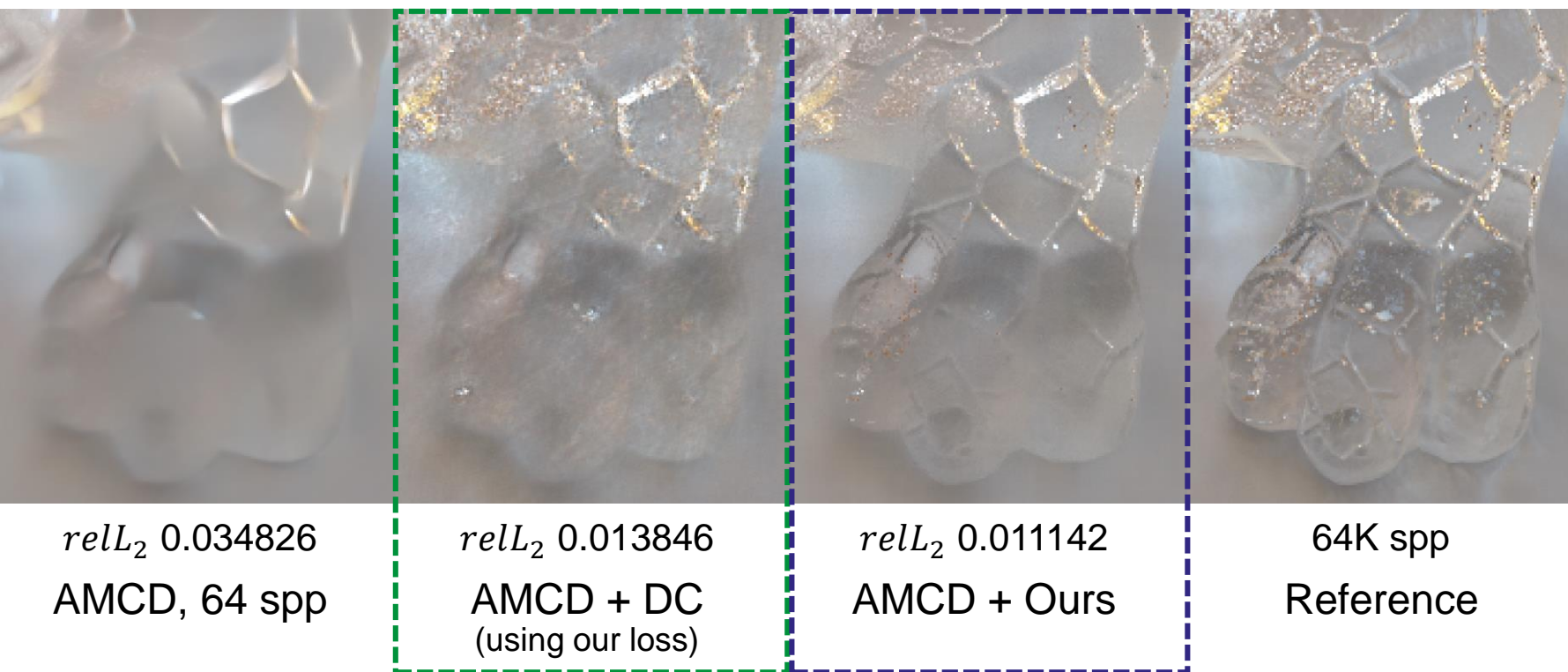
- Our post-correction function

$$g_c(\mathbf{y}, \mathbf{z}) = \frac{1}{\sum_{i \in \Omega_c} w_i} \sum_{i \in \Omega_c} w_i \{ \mathbf{y}_i + \beta_c^z \circ (\mathbf{z}_c - \mathbf{z}_i) + \beta_c^\rho \circ (\rho_c - \rho_i) + \beta_c^n \circ (\mathbf{n}_c - \mathbf{n}_i) \}$$

Cross-bilateral weighting [Kalantari et al. 15, Işık et al. 21]

# PROPOSED METHOD

- Comparisons between neural networks of DC and ours in a self-supervised manner
  - Both networks are optimized using our self-supervised loss.



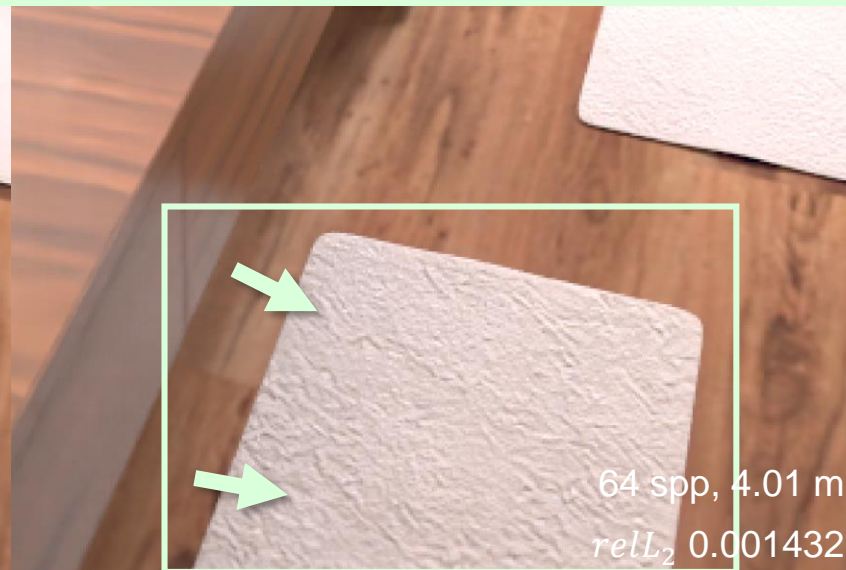
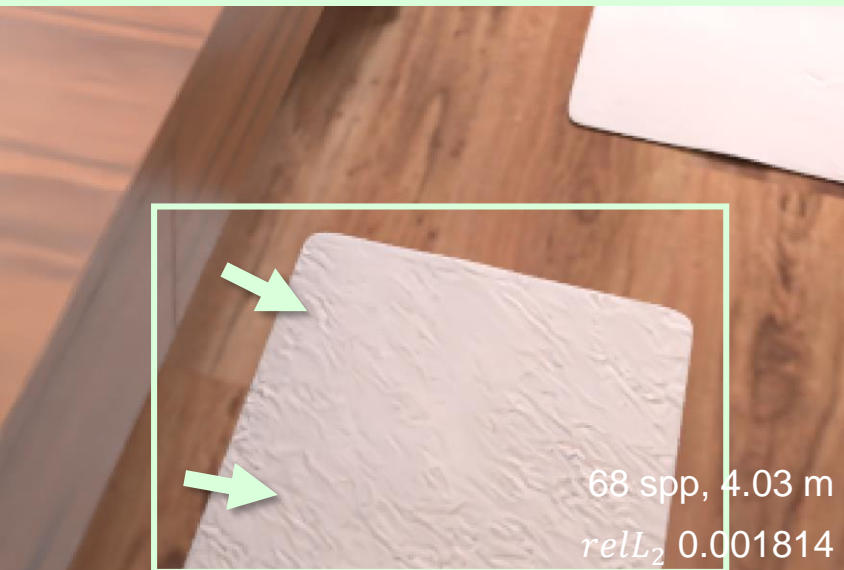
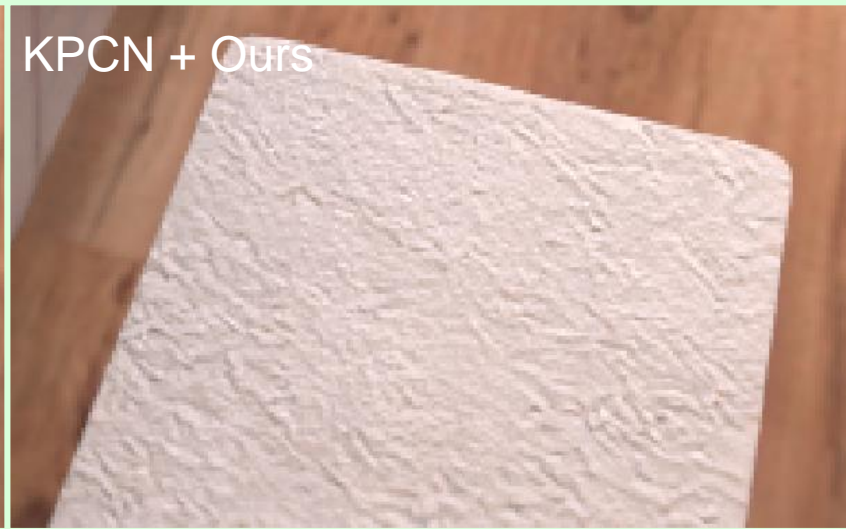
Convergence graph

- Tested Monte Carlo denoising techniques
  - KPCN [Bako et al. 17], AMCD [Xu et al. 19] and AFGSA [Yu et al. 21]
- Error metric
  - Relative L2 error ( $relL_2$ ) [Rousselle et al. 11]

# EQUAL-TIME COMPARISONS



# EQUAL-TIME COMPARISONS



# EQUAL-TIME COMPARISONS



# EQUAL-TIME COMPARISONS



# EQUAL-TIME COMPARISONS

AFGSA



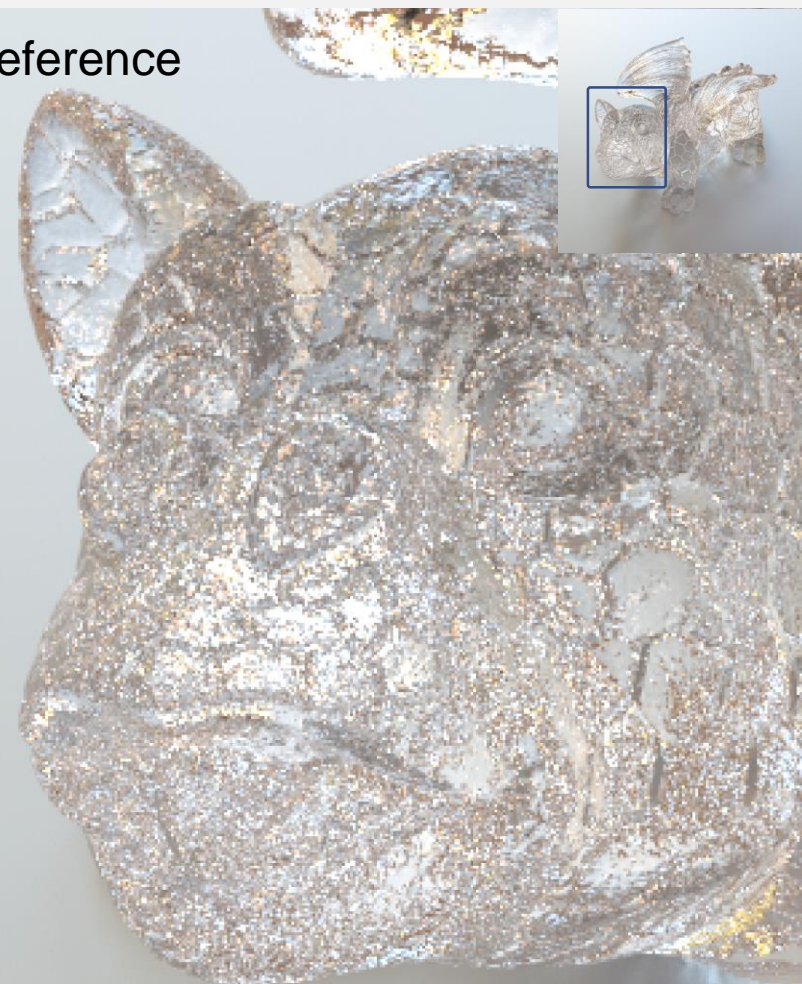
74 spp, 1.43 m  
 $relL_2$  0.030630

AFGSA + Ours



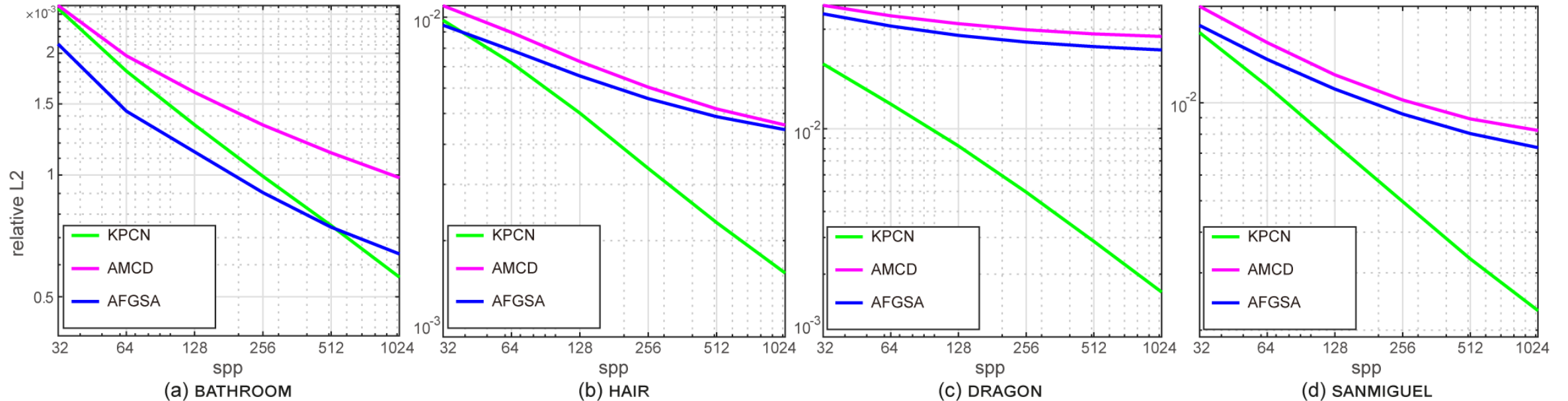
62 spp, 1.41 m  
 $relL_2$  0.011699

Reference

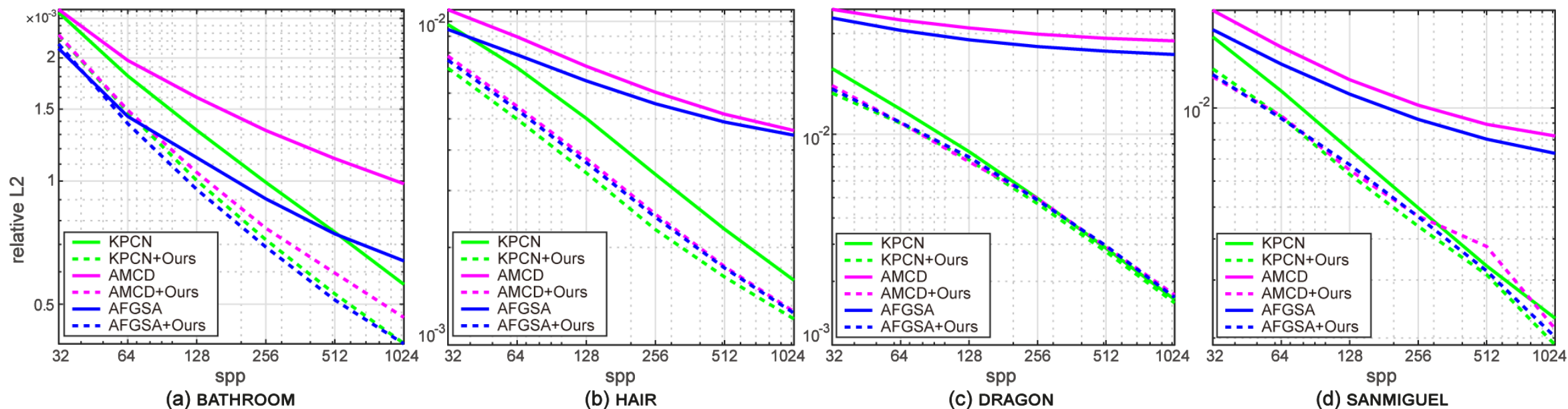


64K spp

- Convergence graphs for denoising methods with and without our post-correction

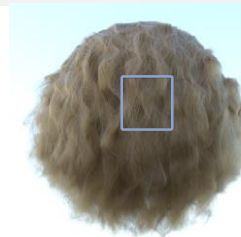


- Convergence graphs for denoising methods with and without our post-correction



# LIMITATIONS & FUTURE WORK

- What if noisy estimates  $y$  are severely noisy compared to denoised estimates  $z$ ?



$relL_2$  0.062694  
Path tracing, 32 spp



$relL_2$  0.009440  
AFGSA



$relL_2$  0.007756  
AFGSA + Ours



16K spp  
Reference

- What if noisy estimates  $y$  are severely noisy compared to denoised estimates  $z$ ?
- Applying to animated sequences
- Extending to self-supervised loss for gradient-domain rendering
  - [Lehtinen et al. 13, Kettunen et al. 15, Hua et al. 19]

- Self-supervised post-correction framework for learning based denoiser
  - Self-supervised loss → no pretraining with training dataset
  - Practical implementation for our post-correction framework
- Practical advantage of ours in scenarios where input data can be different from training dataset

Code, interactive viewer and presentation slides are available at our project page:



- Constructive comments and suggestions
  - Anonymous reviewers
- Funding agencies
  - NRF funded by the Korea government (MSIT) (No. 2020R1A2C4002425)
  - Ministry of Culture, Sports and Tourism and Korea Creative Content Agency (No. R2021080001)
- Scenes
  - nacimus (bathroom), Cem Yuksel (hair), Christian Schüller (dragon), Guillermo M. Leal Llaguno (sanmiguel) and SlykDrako (bedroom)