SIGGRAPH ASIA 2019 BRISBANE

GradNet: Unsupervised Deep Screened Poisson Reconstruction for Gradient-Domain Rendering

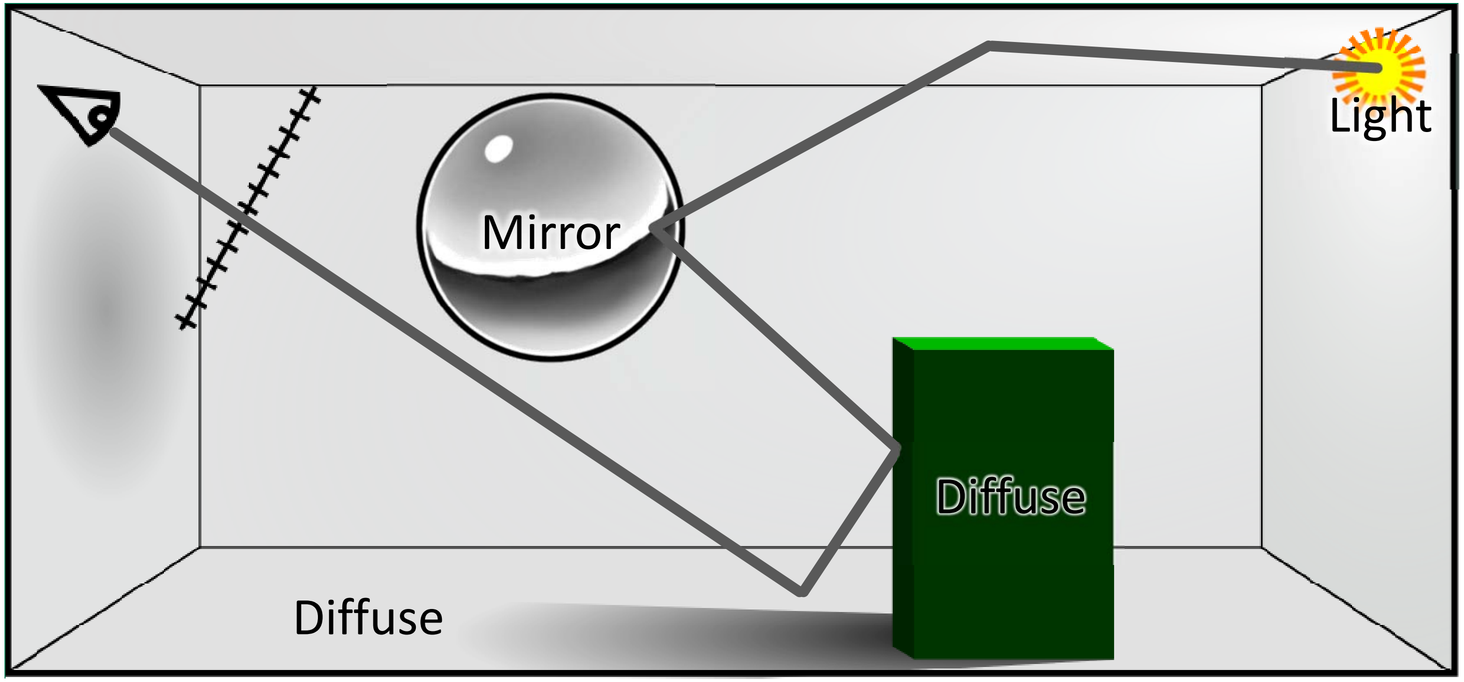
Jie Guo[1] Mengtian Li[1] Quewei Li[1] Yuting Qiang[1] Bingyang Hu[1] Yanwen Guo[1] Ling-Qi Yan[2]

[1]State Key Lab for Novel Software Technology, Nanjing University

[2]University of California, Santa Barbara

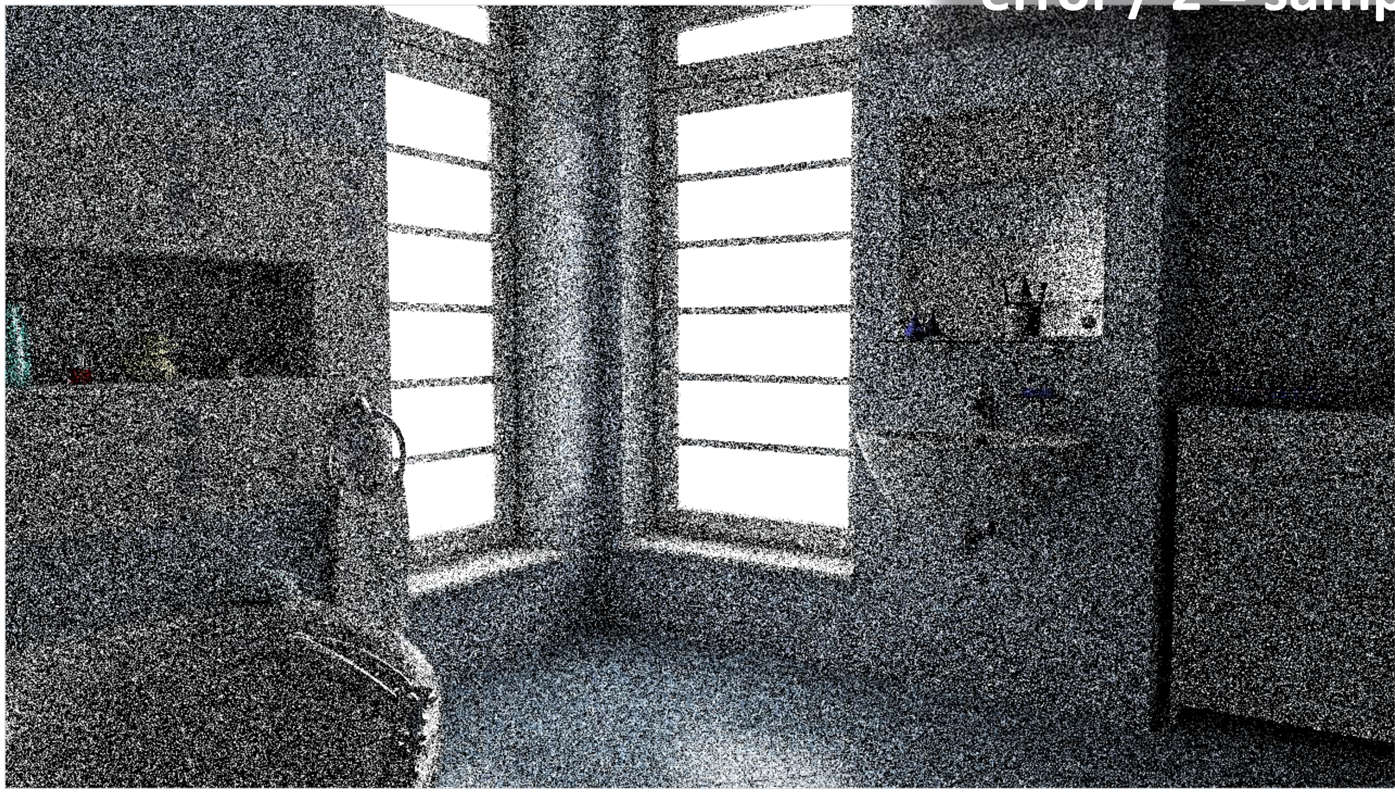
sa2019.siggraph.org

Path tracing

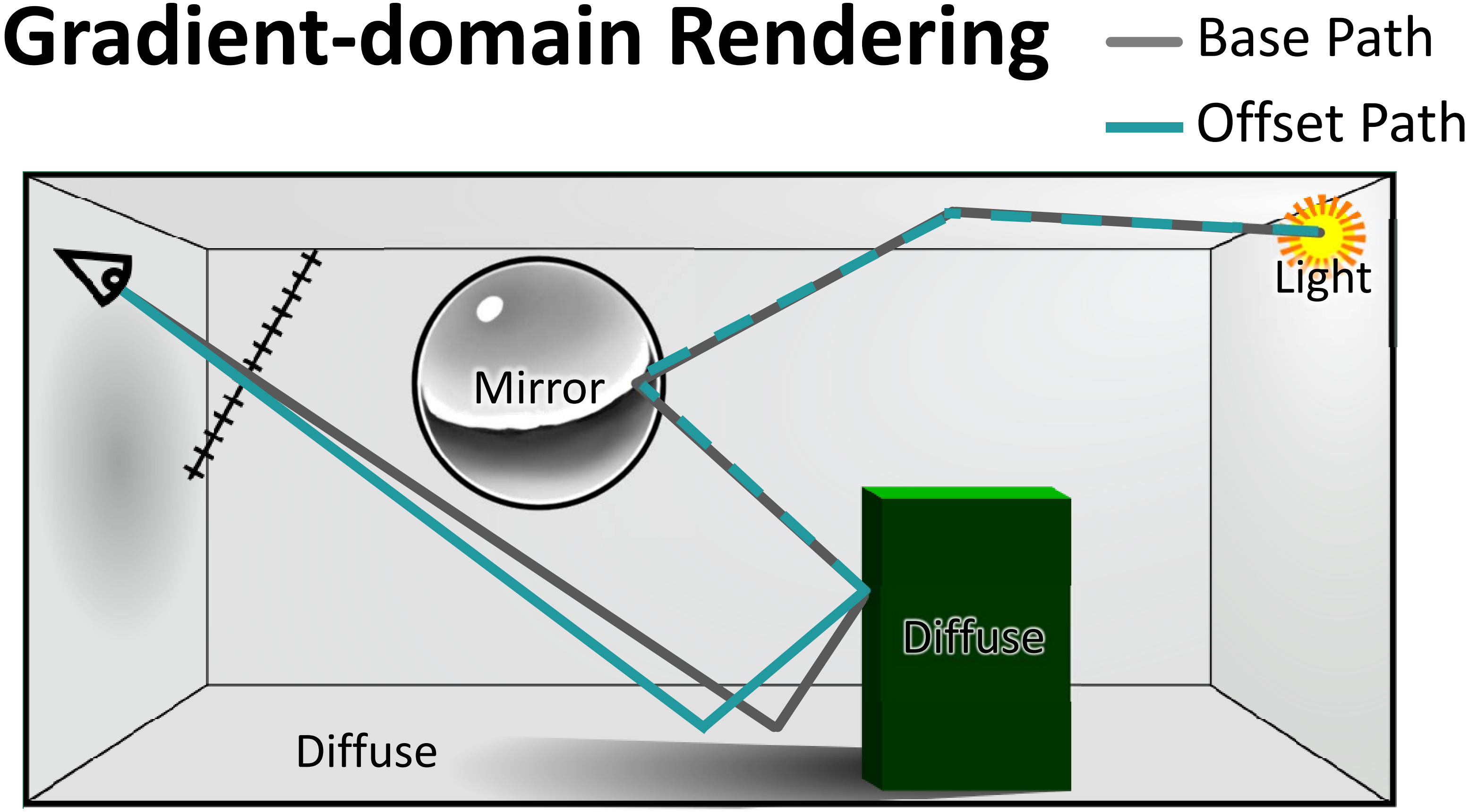


Path tracing

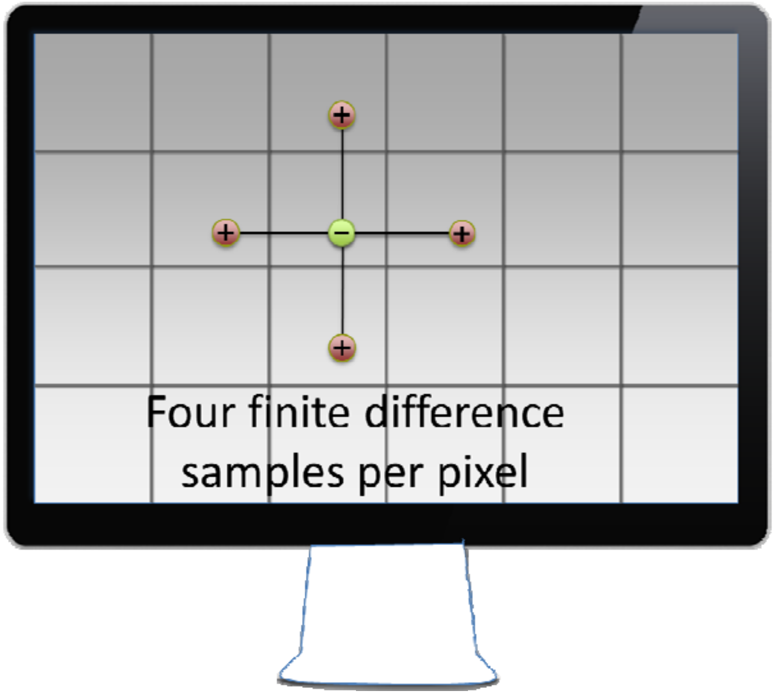
error / 2 = samples * 4



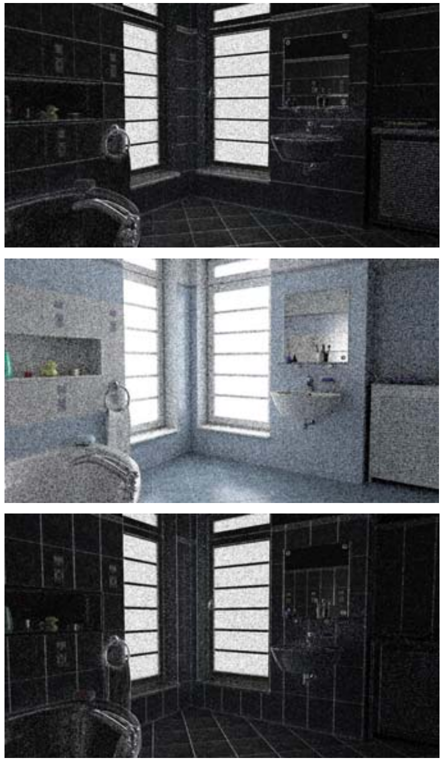
Gradient-domain Rendering



Gradient-domain Rendering



Gradient-domain Rendering



Screened
Poisson
Reconstruction

Screened Poisson Reconstruction

- For a base image and gradients rendered by any gradient-domain algorithms, we can reconstruct the final image by solving the following optimization problem

$$\hat{\mathbf{I}} = \arg\min_{\mathbf{I}} \left\{ \left\| \begin{pmatrix} \nabla_x \mathbf{I} \\ \nabla_y \mathbf{I} \end{pmatrix} - \begin{pmatrix} \mathbf{I}_{\mathrm{d}x} \\ \mathbf{I}_{\mathrm{d}y} \end{pmatrix} \right\|_p + \alpha \| (\mathbf{I} - \mathbf{I}_{\mathrm{b}}) \|_p \right\}$$

Screened Poisson Reconstruction

- For a base image and gradients rendered by any gradient-domain algorithms, we can reconstruct the final image by solving the following optimization problem

$$\hat{\mathbf{I}} = \arg\min_{\mathbf{I}} \left\{ \left\| \begin{pmatrix} \nabla_x \mathbf{I} \\ \nabla_y \mathbf{I} \end{pmatrix} - \begin{pmatrix} \mathbf{I}_{\mathrm{d}x} \\ \mathbf{I}_{\mathrm{d}y} \end{pmatrix} \right\|_p + \alpha \|(\mathbf{I} - \mathbf{I}_{\mathrm{b}})\|_p \right\}$$

Gradient term

Data term

Screened Poisson Reconstruction

- Lp norm

$p = 2$ means L_2 reconstruction

$p = 1$ means L_1 reconstruction

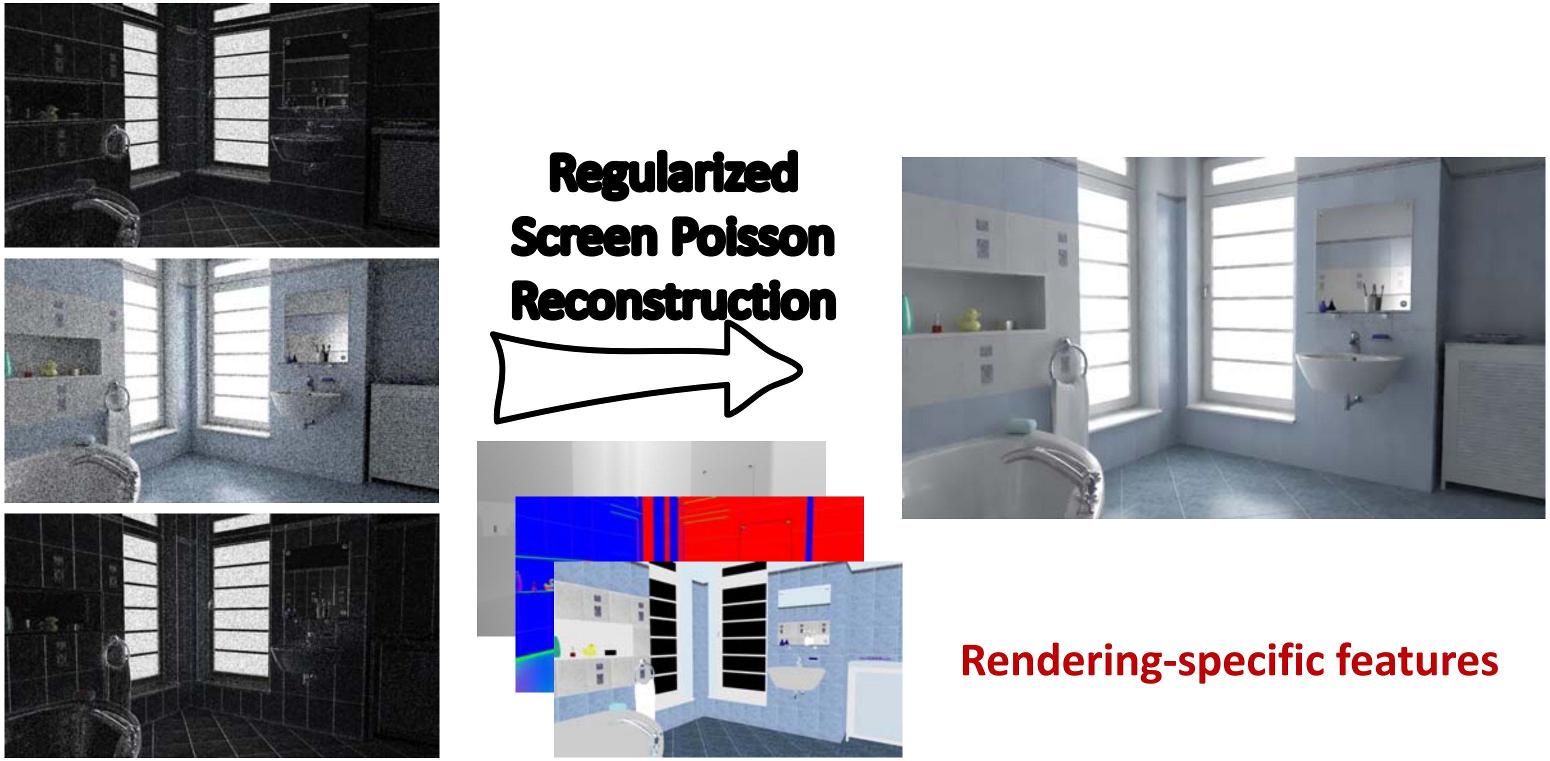
Regularization

- A regularized version of the screened Poisson solver can be written as

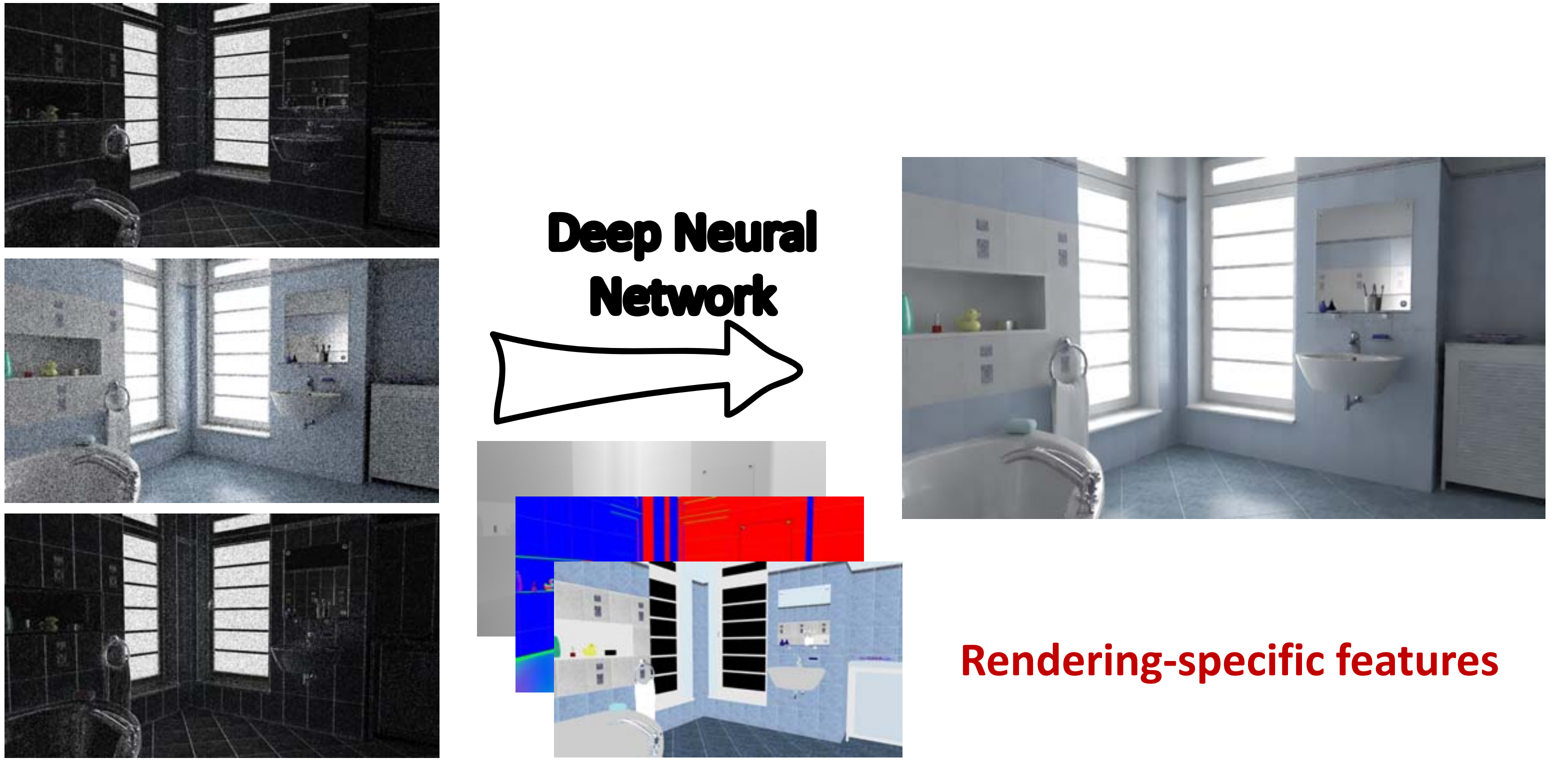
$$\hat{\mathbf{I}} = \arg\min_{\mathbf{I}} \left\{ \left\| \begin{pmatrix} \nabla_x \mathbf{I} \\ \nabla_y \mathbf{I} \end{pmatrix} - \begin{pmatrix} \mathbf{I}_{\mathrm{d}x} \\ \mathbf{I}_{\mathrm{d}y} \end{pmatrix} \right\|_p + \alpha \|(\mathbf{I} - \mathbf{I}_{\mathrm{b}})\|_p + \lambda \Omega(\mathbf{I}, \mathbf{F}) \right\}$$

Regularizer

Regularized Screen Poisson Reconstruction



Our Idea



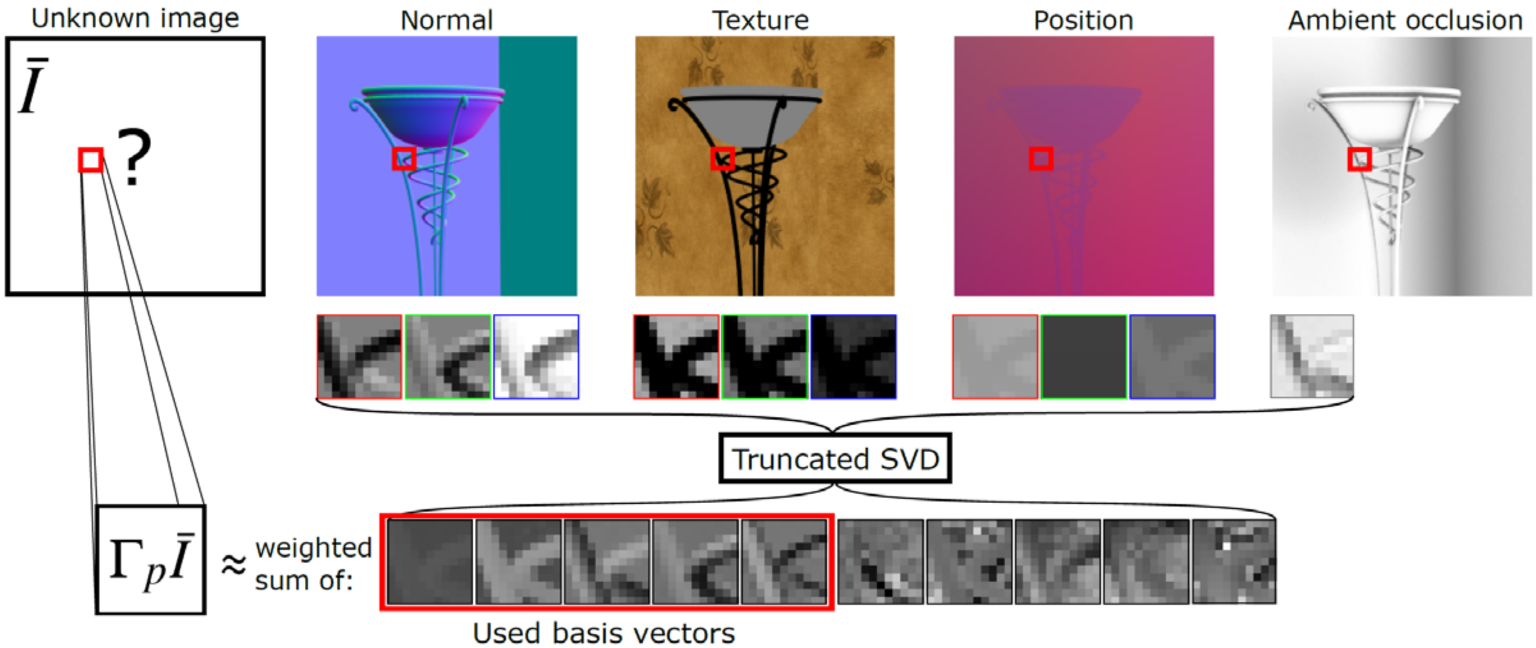
Related Work

METHOD	DEEP LEARNING BASED	AUXILIARY BUFFERS	PERFORMANCE
L1	×	×	≈0.45s GPU
CV [Rousselle et al. 2016]	×	×	≈2s CPU
LTS [Ha et al. 2019]	×	×	≈1.7s CPU
NFOR [Bitterli et al. 2016]	×	√	≈200s CPU
REG [Manzi et al. 2016]	×	√	≈60s GPU
KPCN [Bako et al. 2017]	√ Supervised	√	≈1.7s GPU
[Kettunen et al. 2019]	√ Supervised	√	≈0.3s GPU

Related Work

METHOD	DEEP LEARNING BASED	AUXILIARY BUFFERS	PERFORMANCE
L1	×	×	≈0.45s GPU
CV [Rousselle et al. 2016]	×	×	≈2s CPU
LTS [Ha et al. 2019]	×	×	≈1.7s CPU
NFOR [Bitterli et al. 2016]	×	√	≈200s CPU
REG [Manzi et al. 2016]	×	√	≈60s GPU
KPCN [Bako et al. 2017]	√ **Supervised**	√	≈1.7s GPU
[Kettunen et al. 2019]	√ **Supervised**	√	≈0.3s GPU

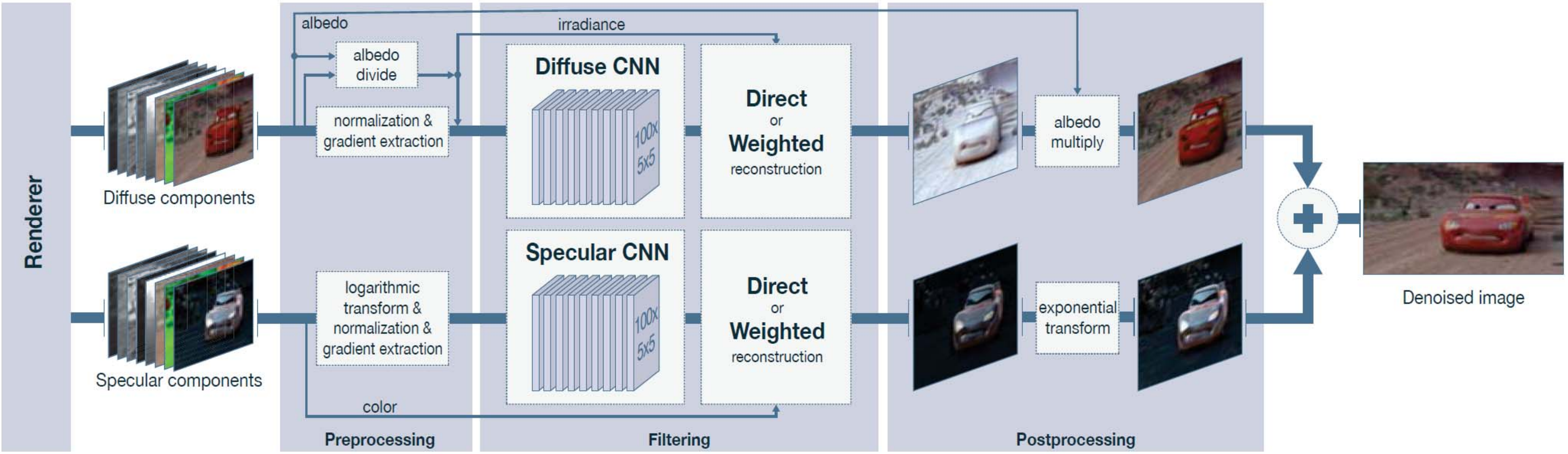
Related Work (REG [Manzi et al. 2016])



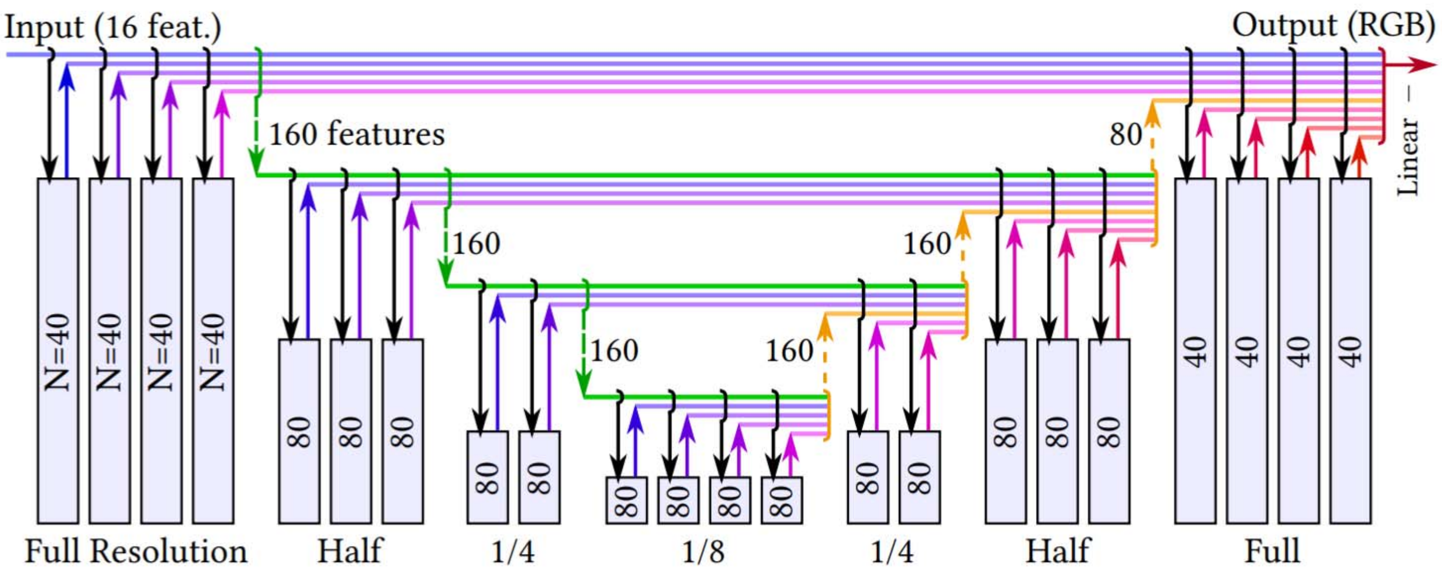
Feature Bases via Truncated SVD.

Solved by an iteratively reweighted least squares (IRLS) approach

Related Work (KPCN)



Related Work ([Kettunen et al. 2019])



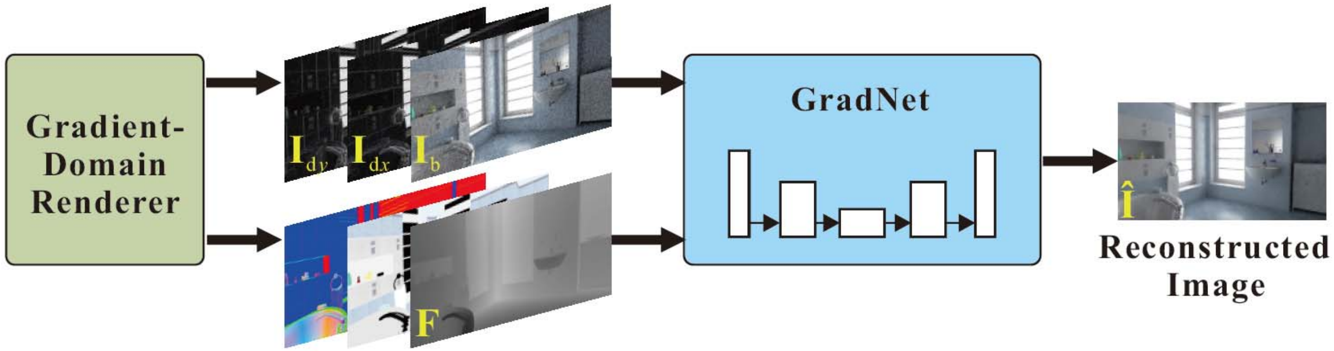
1. Using gradients as an additional feature
2. Adopting a new perceptual loss

Our Method

- Deep learning based
- Unsupervised
- Fast to reconstruct high-quality image

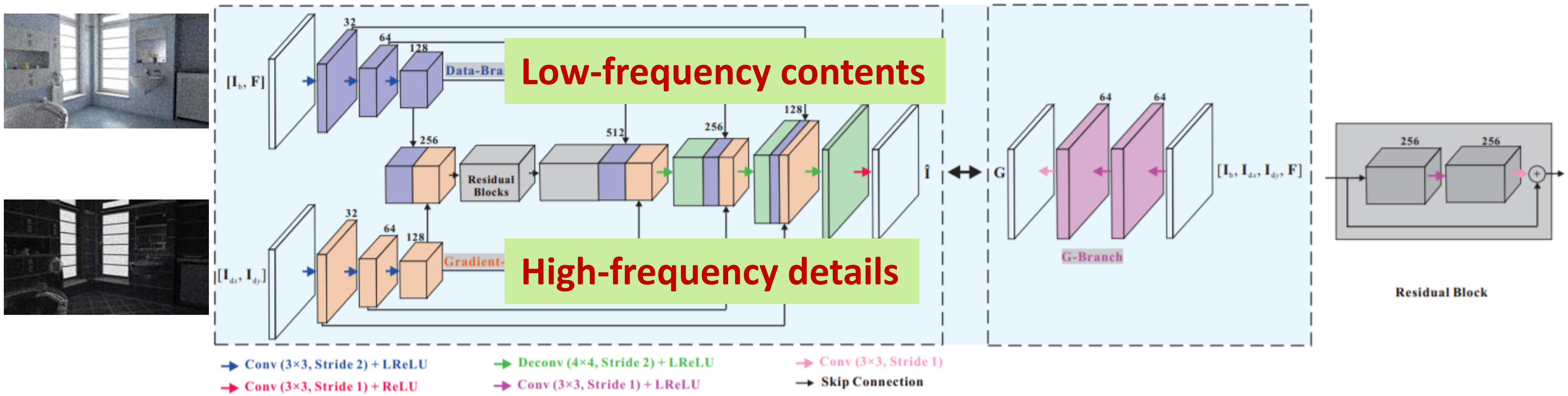
Our Solution

- Replace the traditional optimization in screened Poisson reconstruction with GradNet



Network Architecture

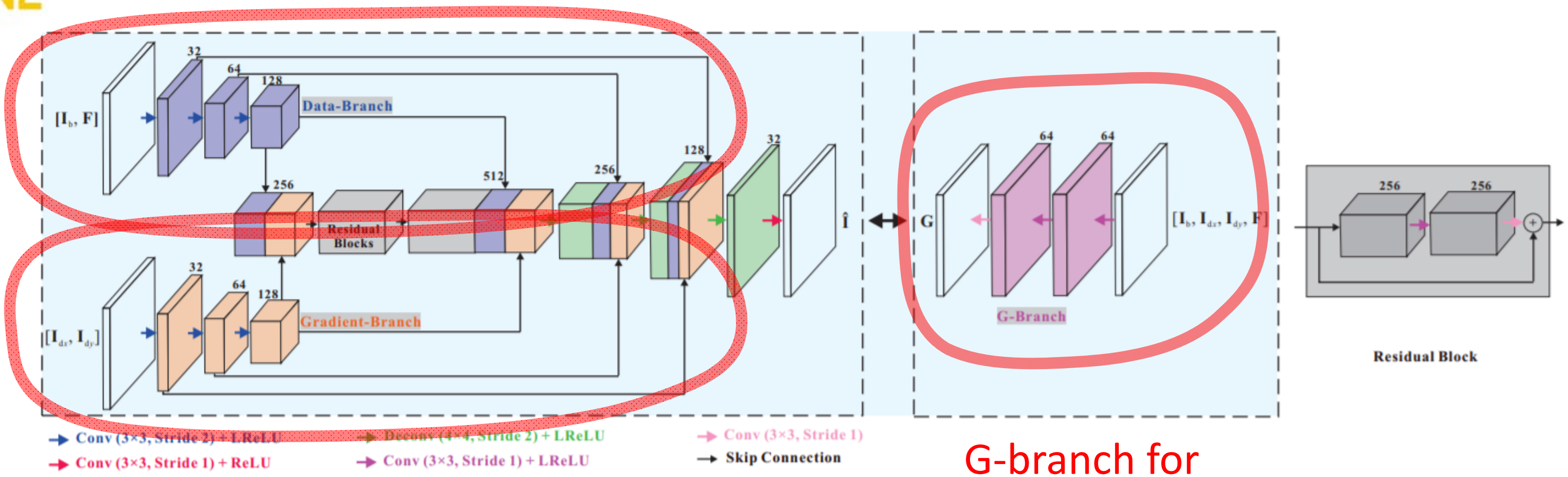
- Multi-branch auto-encoder with dual skip connection



Network Architecture

- Multi-branch auto-encoder with dual skip connection

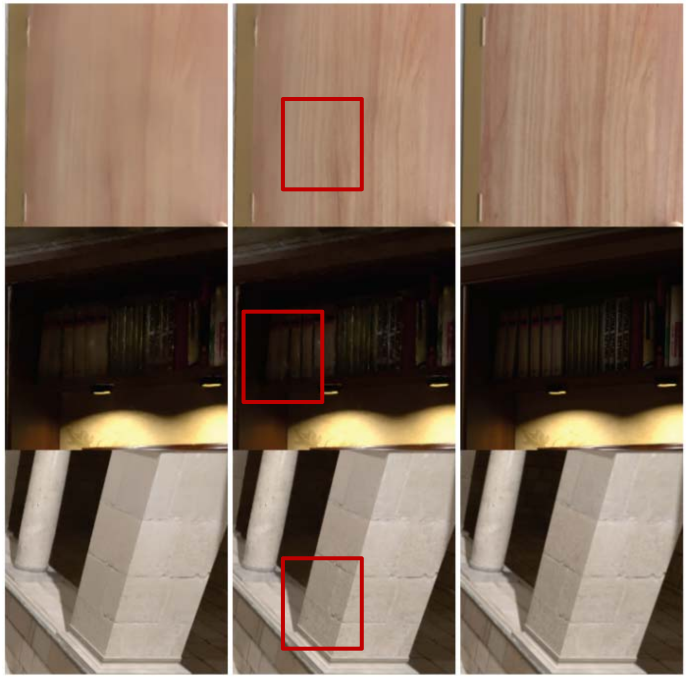
Data branch



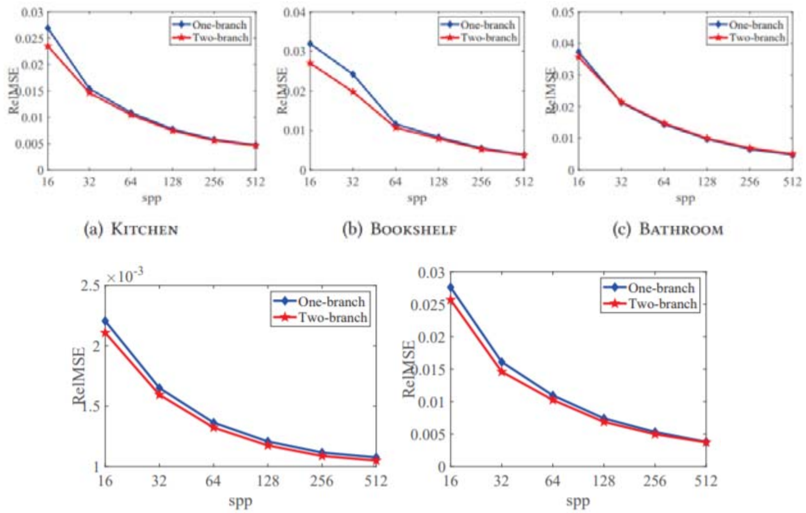
Gradient branch

G-branch for generating derivative

Impact of the Branches


(a) One-branch (b) Two-branch (c) Reference

One-branch encoder weaken the effects of sparse image gradients

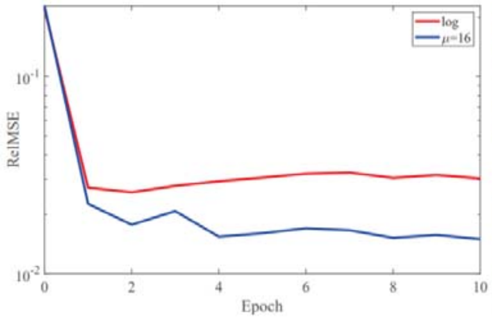


Dynamic Range Compression

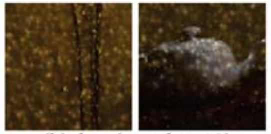
- We employ the μ-law transformation to compress HDR data

$$\mathcal{T}(\mathbf{I}) = \mathrm{sign}(\mathbf{I})\frac{\log(1+\mathrm{abs}(\mathbf{I})\mu)}{\log(1+\mu)}$$

- The μ-law transformation makes the training process easier than naïve log transformation



(a) Evolution of RelMSE



(b) log (epoch= 10)

(c) μ = 16 (epoch= 10)

Loss Function

- The loss function contains 3 items
  - data item, gradient item and first-order item

$$\mathcal{L}_{\text{all}} = \mathcal{L}_{\text{grad}}(\hat{\mathbf{I}}, \mathbf{I}_{\text{d}x}, \mathbf{I}_{\text{d}y}) + \alpha\mathcal{L}_{\text{data}}(\hat{\mathbf{I}}, \mathbf{I}_{\text{b}}) + \lambda\mathcal{L}_{\text{1st}}(\hat{\mathbf{I}}, \mathbf{F})$$

- Data item:

$$\mathcal{L}_{\text{data}}(\hat{\mathbf{I}}, \mathbf{I}_{\text{b}}) = \frac{1}{N}\sum_{i=1}^{N}||\mathcal{T}(\hat{\mathbf{I}}_i) - \mathcal{T}(\mathbf{I}_{\text{b},i})||$$

Loss Function

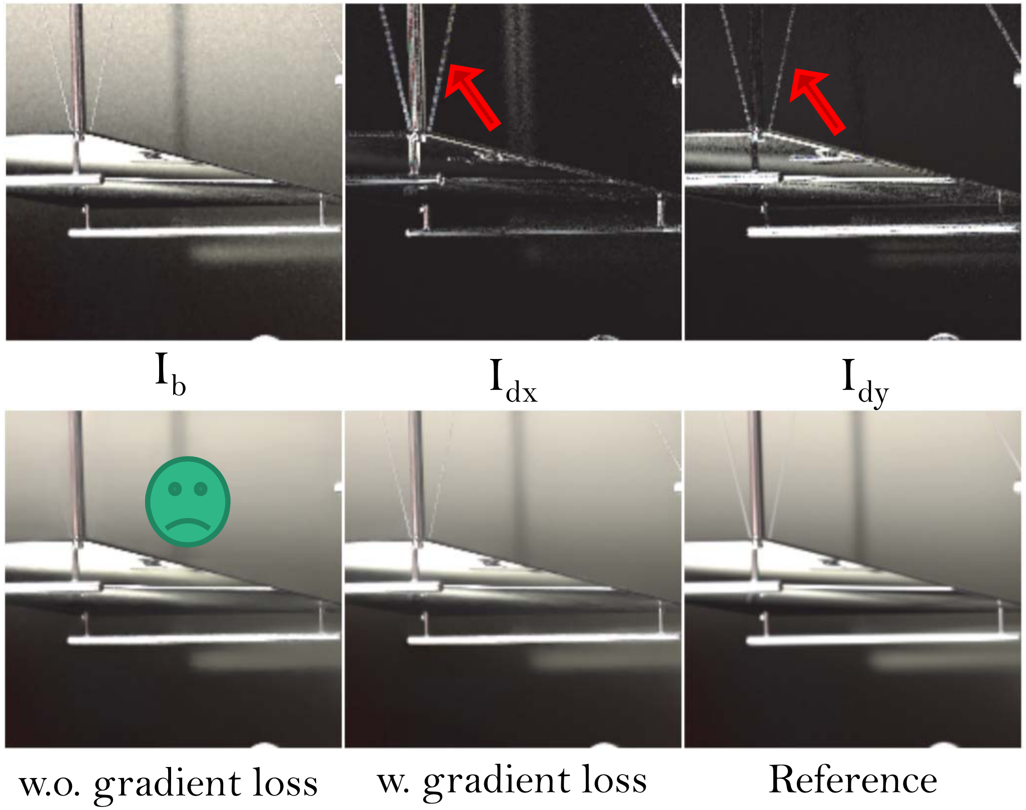
- The loss function contains 3 items
  - data item, gradient item and first-order item

$$\mathcal{L}_{\text{all}} = \mathcal{L}_{\text{grad}}(\hat{\mathbf{I}}, \mathbf{I}_{\text{d}x}, \mathbf{I}_{\text{d}y}) + \alpha\mathcal{L}_{\text{data}}(\hat{\mathbf{I}}, \mathbf{I}_{\text{b}}) + \lambda\mathcal{L}_{1\text{st}}(\hat{\mathbf{I}}, \mathbf{F})$$

- Gradient item:

$$\mathcal{L}_{\text{grad}}(\hat{\mathbf{I}}, \mathbf{I}_{\text{d}x}, \mathbf{I}_{\text{d}y}) = \frac{1}{N}\sum_{i=1}^{N}(\|\mathcal{T}(\nabla_x\hat{\mathbf{I}}_i) - \mathcal{T}(\mathbf{I}_{\text{d}x,i})\| + \|\mathcal{T}(\nabla_y\hat{\mathbf{I}}_i) - \mathcal{T}(\mathbf{I}_{\text{d}y,i})\|)$$

Impact of the Gradient Loss



Loss Function

- The loss function contains 3 items
  - data item, gradient item and first-order item

$$\mathcal{L}_{\mathrm{all}} = \mathcal{L}_{\mathrm{grad}}(\hat{\mathbf{I}}, \mathbf{I}_{\mathrm{d}x}, \mathbf{I}_{\mathrm{d}y}) + \alpha \mathcal{L}_{\mathrm{data}}(\hat{\mathbf{I}}, \mathbf{I}_{\mathrm{b}}) + \lambda \mathcal{L}_{\mathrm{1st}}(\hat{\mathbf{I}}, \mathbf{F})$$

- The first-order item:

$$\mathcal{L}_{\mathrm{1st}}(\hat{\mathbf{I}}, \mathbf{F}) = \frac{1}{N|\mathcal{N}_i|} \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} w_{i,j} \|\mathcal{T}(\hat{\mathbf{I}}_j) - \mathcal{T}(\hat{\mathbf{I}}_i) - \mathbf{G}_i^{\top}(\mathbf{F}_j - \mathbf{F}_i)\|$$

First-order Regularization

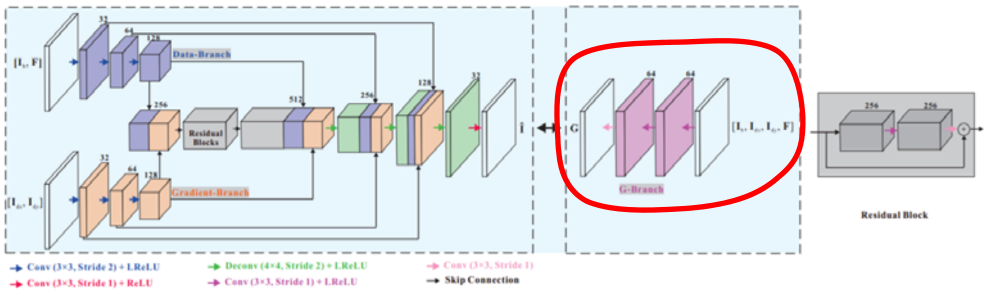
- The first-order regularization defines as follow

$$\frac{1}{N|N_i|}\sum_{i=1}^{N}\sum_{j\in N_i} w_{i,j}\left\|\hat{I}_j-\hat{I}_i-G_i^T(F_j-F_i)\right\|$$

Neighboring pixels around index *i*

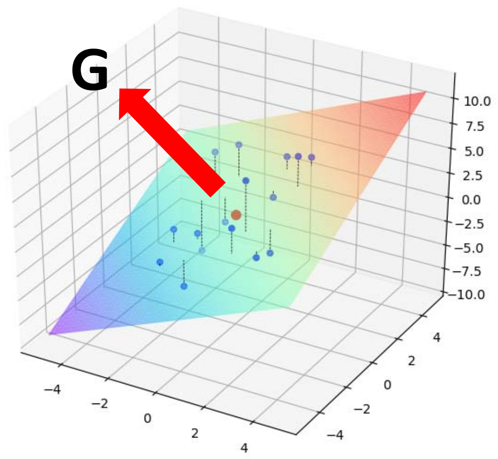
Derivative of *I* with respect to *F*

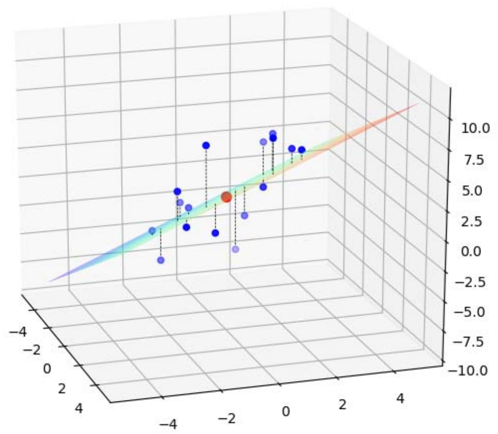
Difference of auxiliary features



First-order Regularization

- The first-order regularization encourages nearby pixels to lie on a hyper-plane parameterized by G





Impact of the First-order Loss

w.o. the first-order loss | w. the first-order loss | reference

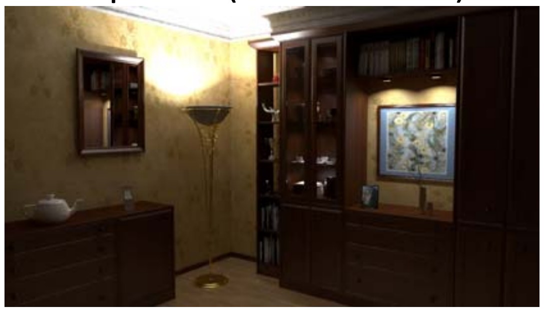
Bias of Network

μ will introduce bias to the reconstructed image

Reference (mean: 0.204)

μ = 16 (mean: 0.187)

μ = 128 (mean: 0.176)



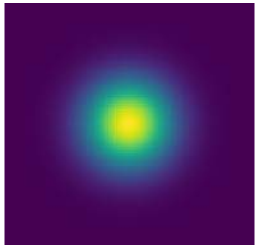
μ = 1024 (mean: 0.163)

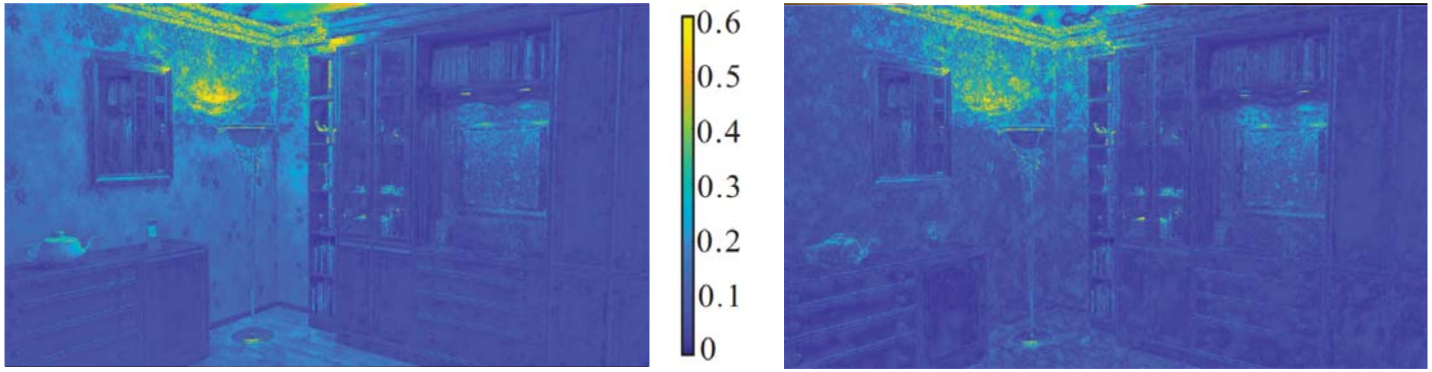
Post-Processing

- We find a simple post-processing step can reduce the bias

$$\hat{I} \leftarrow \frac{I_b * k}{\hat{I} * k} \odot \hat{I}$$

Gaussian filter
with r = 45 and σ = 15





Without P.P.

With P.P.

Training Details

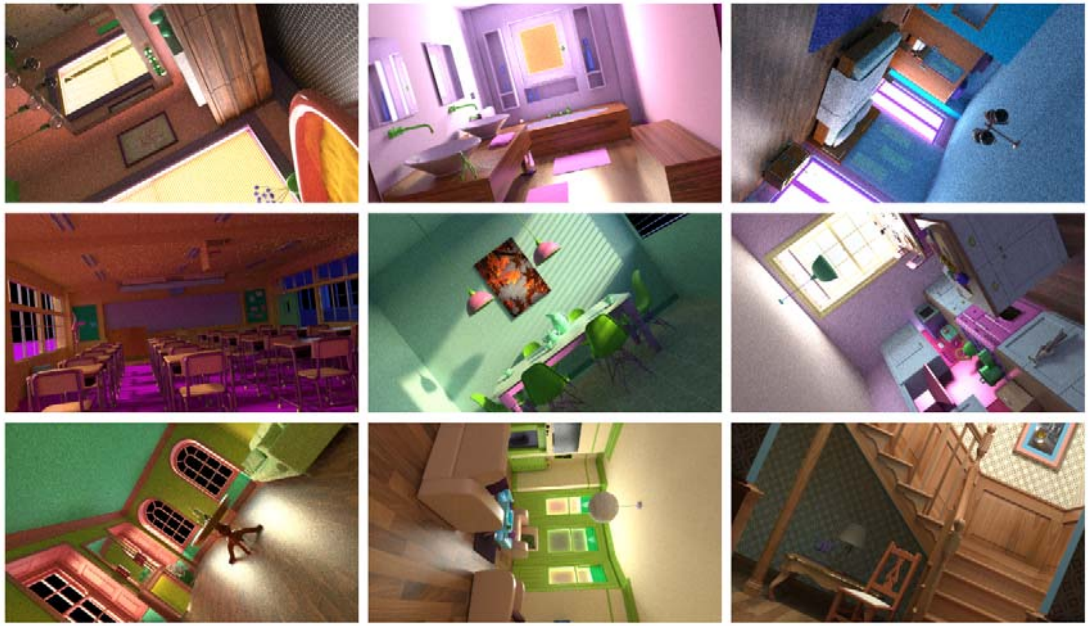
- Adam optimizer
  - $\beta1 = 0.5$ and $\beta2 = 0.999$
- Initial learning rate = 0.0001 and decays with the power of 0.95 for every other epoch
- λ follows the schedule

$$\lambda = \begin{cases} 0 & \text{if} \quad \text{epoch} \leq 5 \\ \min(0.1 \times 1.1^{(\text{epoch}-5)}, 2.0) & \text{if} \quad \text{epoch} > 5 \end{cases}$$

- Train main branches and G-branch alternatively for 50 epochs with 32 mini-batches

Dataset

- Randomly perturb 9 base scenes and render to 900 high-resolution images with 64 spp



Dataset

- 12654 patches with 256x256 resolution are extracted from these 900 high-resolution images

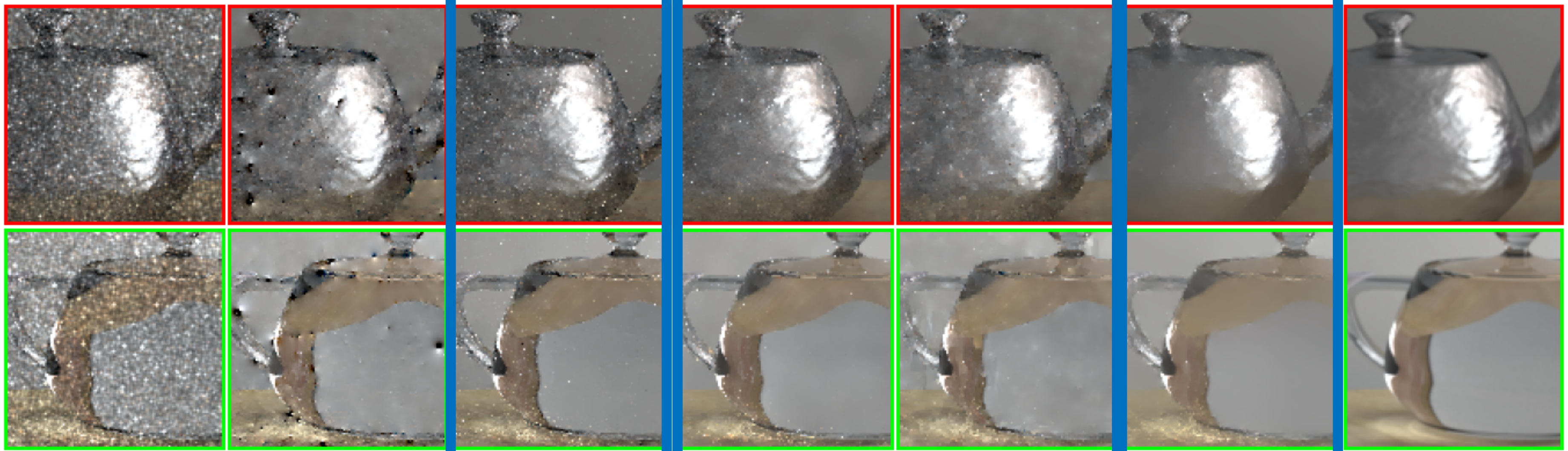
Results

Comparison with traditional methods

Comparison with traditional methods

Comparison with traditional methods

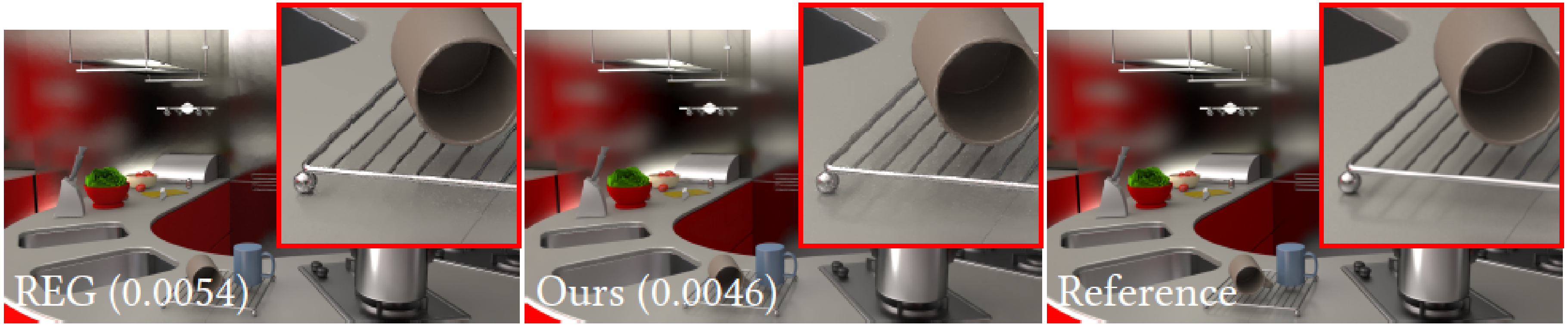
Comparison with traditional methods



0.2577	0.0237	0.0257	0.0086	0.0110	0.0050	
(b) Base ($\mathbf{I}_b$)	(c) L_2	(d) L_1	(e) CV	(f) LTS	(g) Ours	(h) Reference

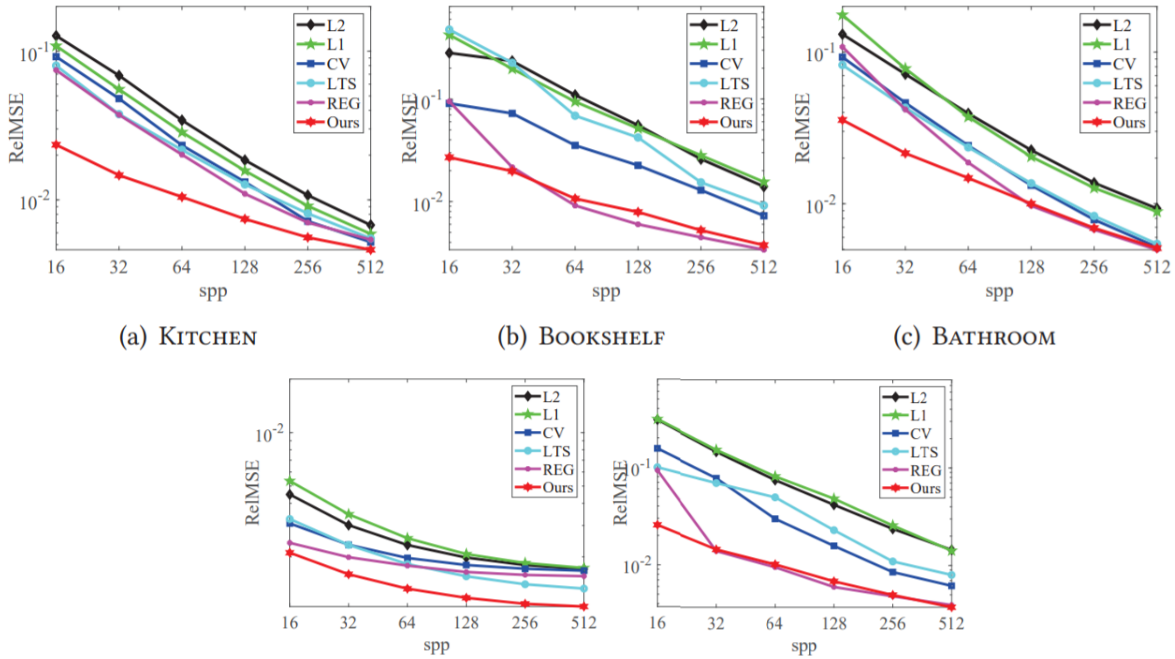
Comparison with traditional methods

- Comparison with the regularized reconstruction method of Manzi et al. [2016]



- Time: >1min (REG) vs. 0.16s (ours)

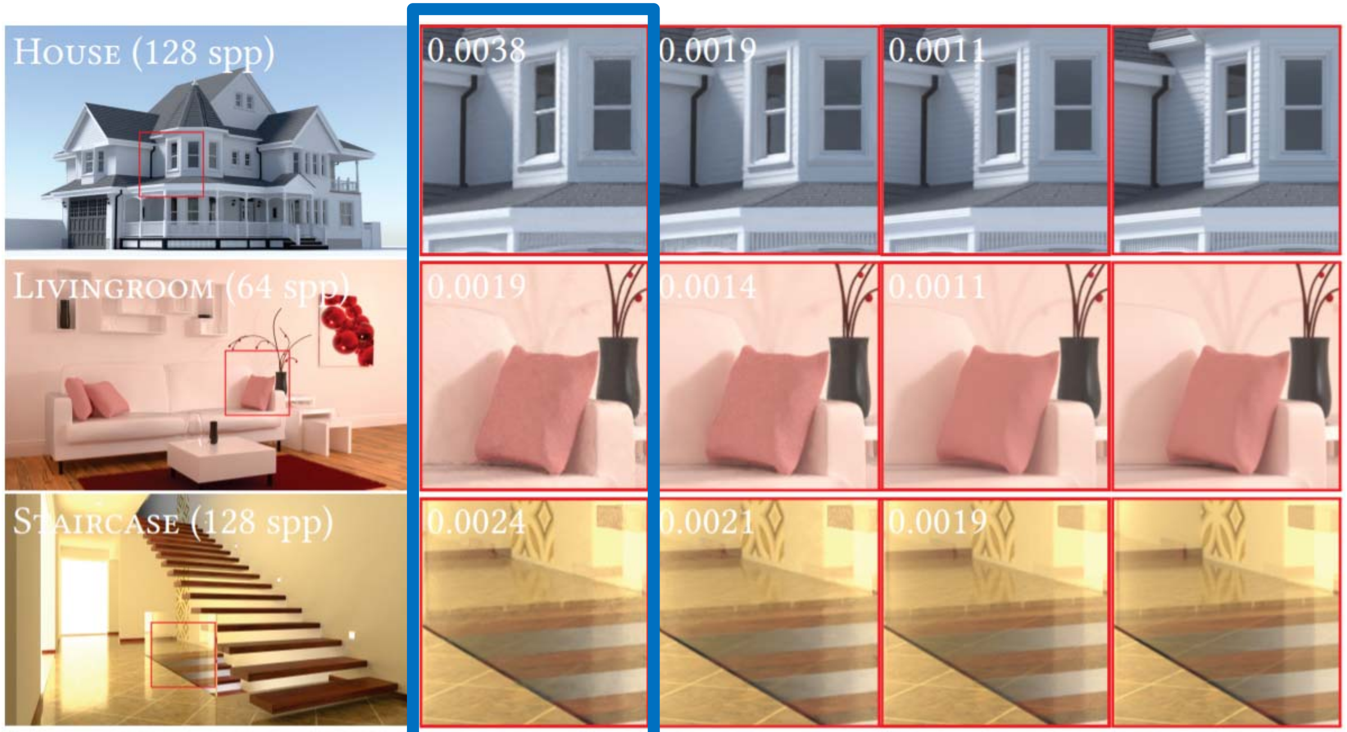
Comparison with traditional methods



(a) KITCHEN (b) BOOKSHELF (c) BATHROOM

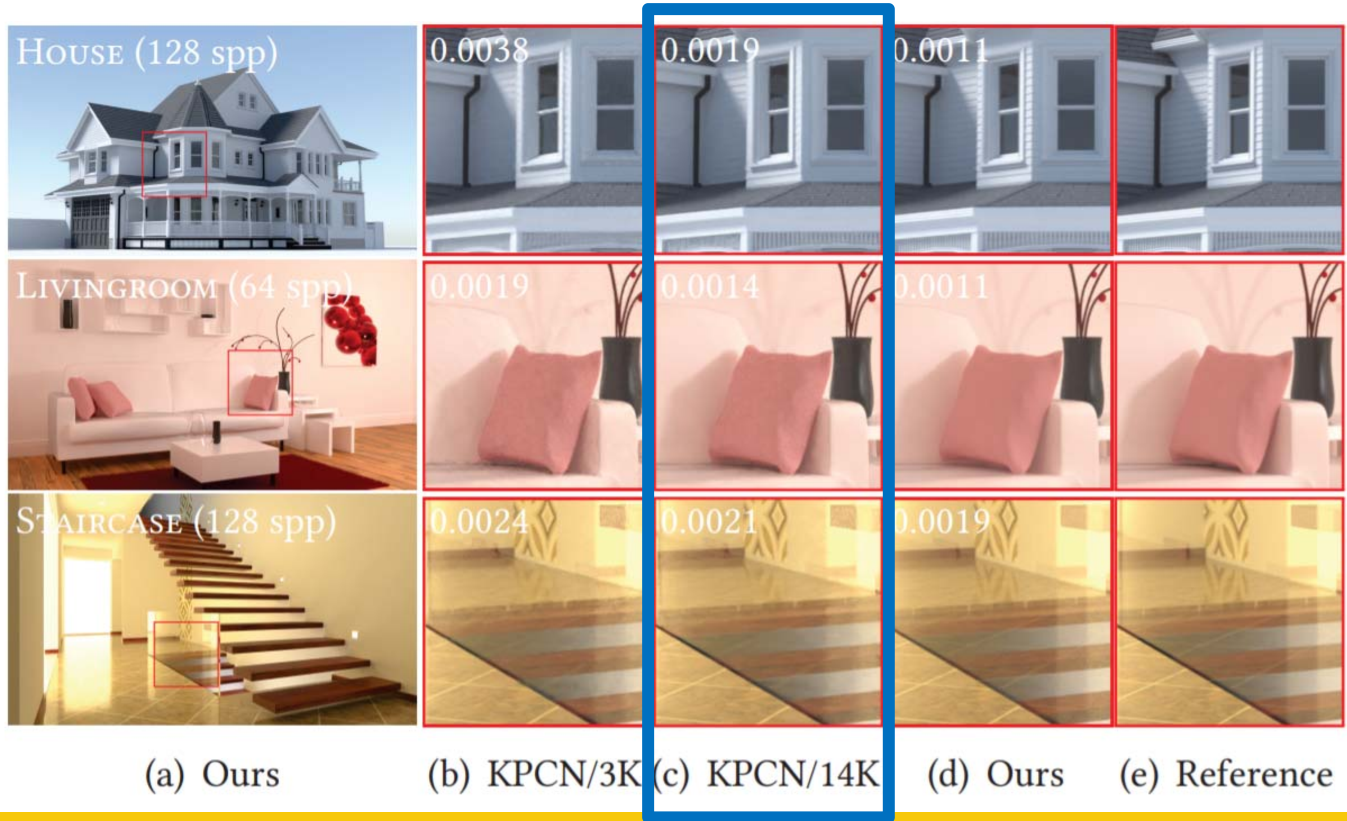
(d) SPONZA (e) DOOR

Comparison with KPCN

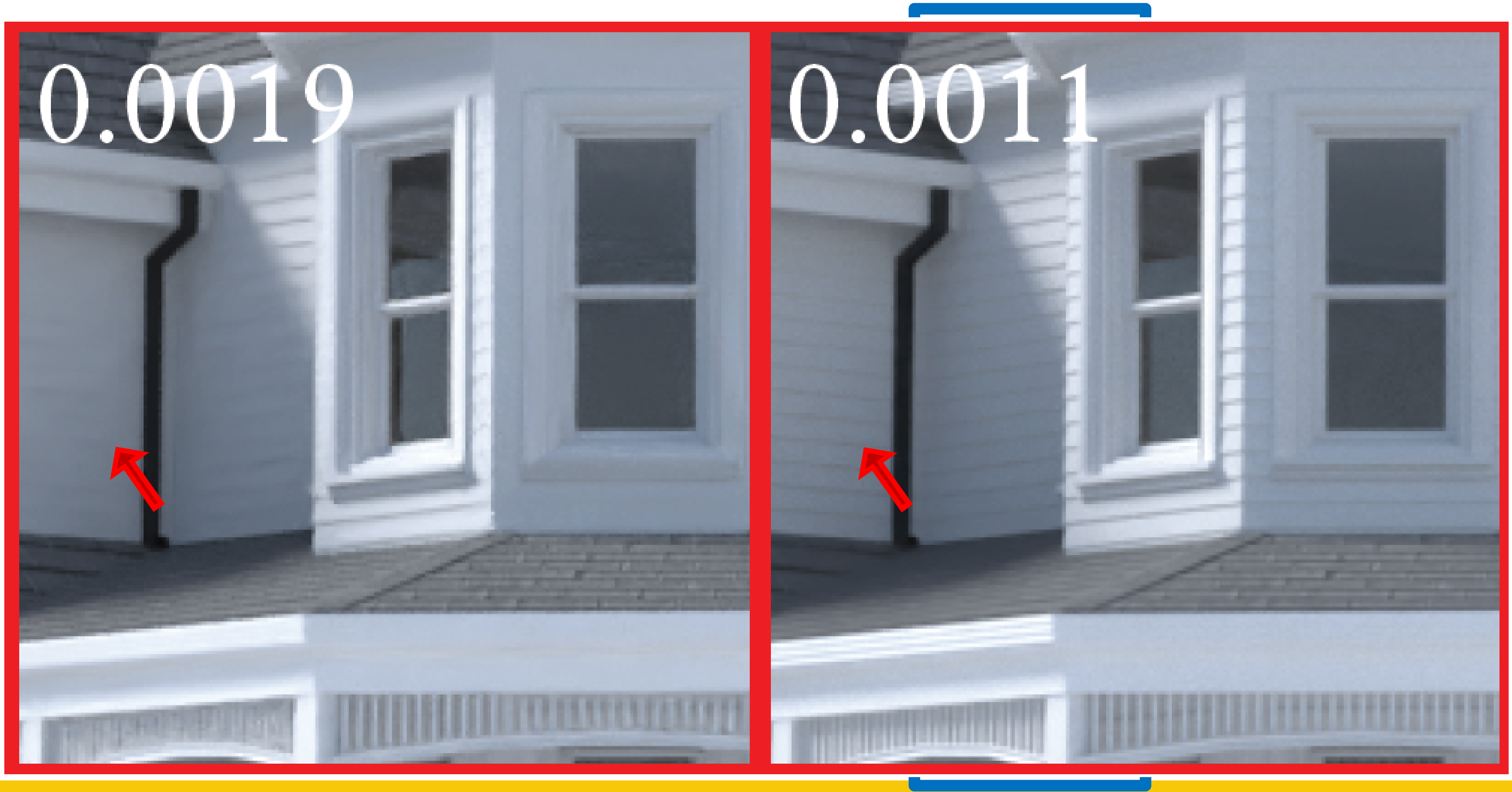


(a) Ours (b) KPCN/3K (c) KPCN/14K (d) Ours (e) Reference

Comparison with KPCN



Comparison with KPCN



Impact of the Training Datasets

1/4 dataset 1/2 dataset Full dataset

Conclusion

- The first unsupervised deep learning solution to screened Poisson reconstruction.
- A multi-branch auto-encoder allowing extracting both low-frequency contents and high-frequency details.
- A novel reconstruction loss function incorporating auxiliary feature buffers.

Future Work

- Adding an adversarial loss to further enhance important local structures of reconstructed images.
- Extend to the temporal domain by introducing temporal finite differences.
- Combine our technology with adaptive sampling.

SIGGRAPH ASIA 2019 BRISBANE

Thank you

Jie Guo

guojie@nju.edu.cn