Deep Graph Convolutional Networks

Computer Vision Final Project

Nikola Janjušević

NYU

December 17th 2020

Today's Papers

[1] Francesca Pistilli et al. "Learning Graph-Convolutional Representationsfor Point Cloud Denoising". In: *The European Conference on Computer Vision (ECCV)*. 2020.

[2] Martin Simonovsky and Nikos Komodakis. "Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs". In: *CVPR*. 2017. URL: https://arxiv.org/abs/1704.02901.

[3] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. "Image Denoising with Graph-Convolutional Neural Networks". In: *2019 26th IEEE International Conference on Image Processing (ICIP)*. 2019.

Table of Contents

Motivation: graph representation of data I

- Irregular data domains: point clouds, sensor networks

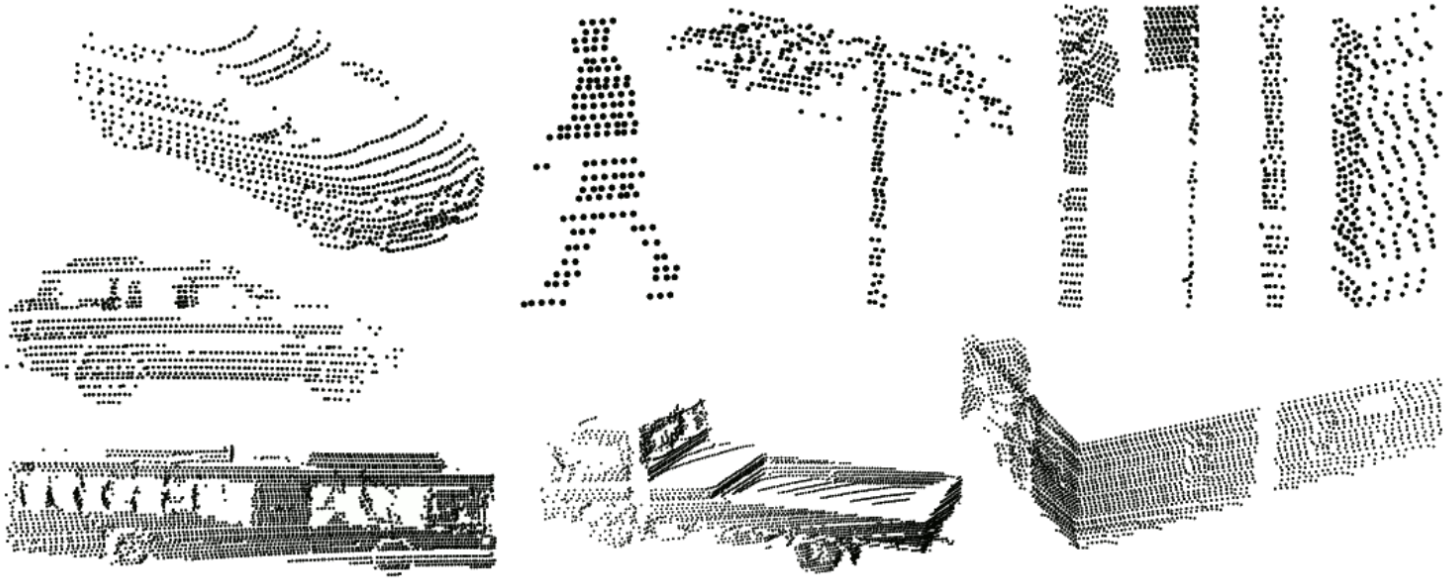


Figure: [2]

Motivation: graph representation of data II

- Capture non-local relations in regular-domain data

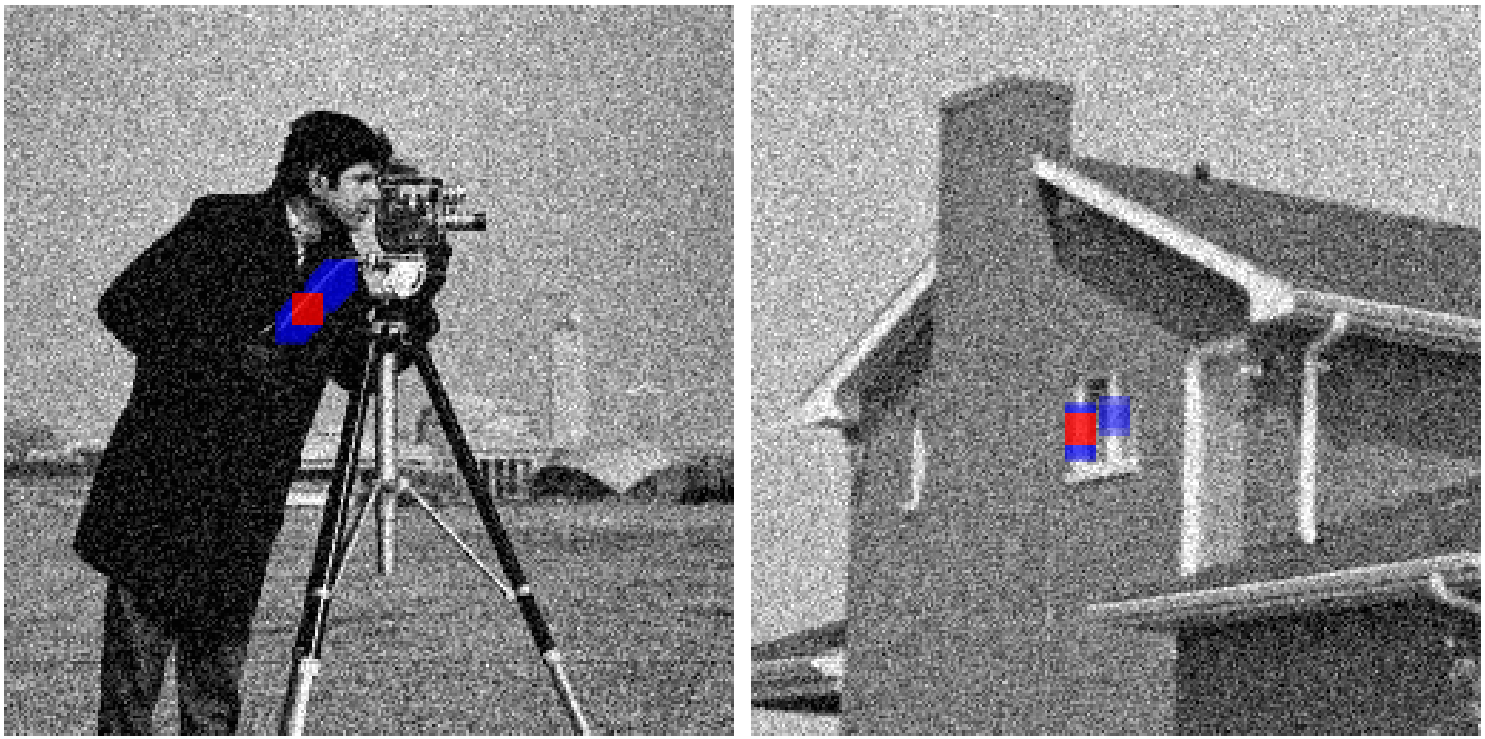


Figure: [**BM3D**]

Table of Contents

Linear Convolution as a special case:

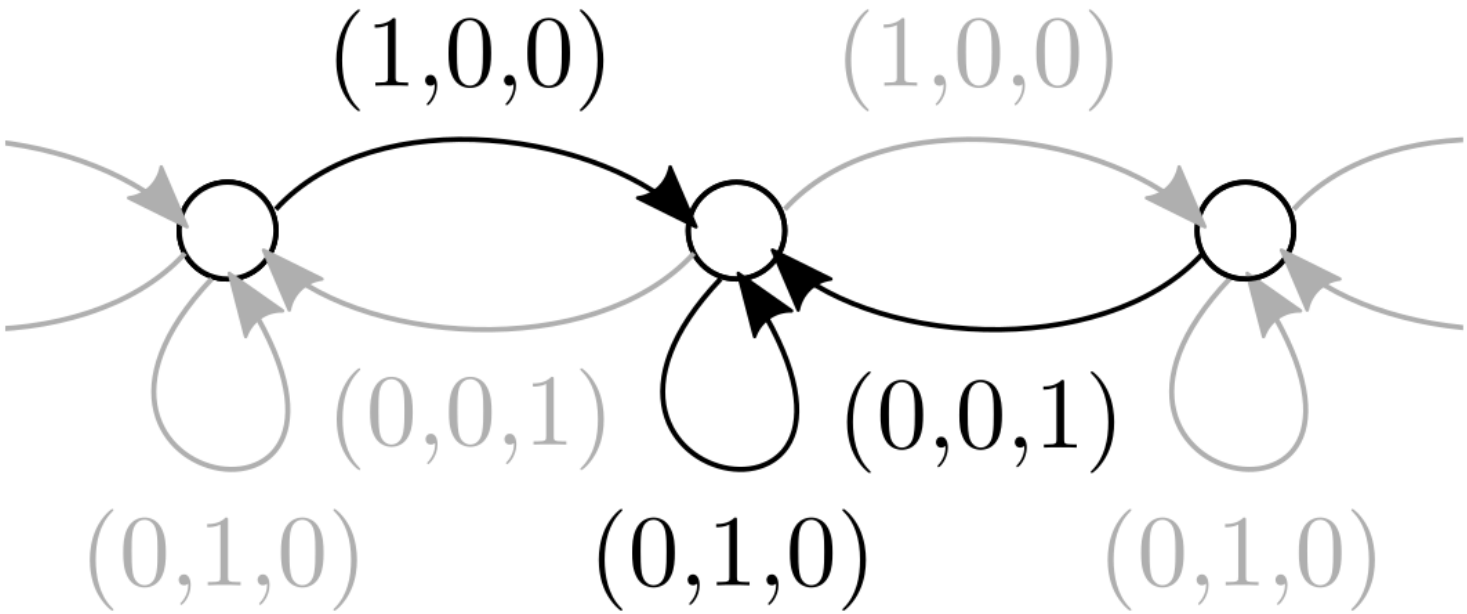


Figure: [2]

Frequency and spatial domain formulations

- Notion of spectrum can be defined via Graph-Laplacian. Filtering then naturally arises.
- Building/applying Graph-Laplacian can be expensive, not nice for dynamically changing graphs.
- Spatial domain: define filtering as node/vertex signal agregation. Less expensive but no clear formulation.

Notation

Consider $x \in \mathbb{R}^{CN}$, a multi-channel signal with N pixels.

- $x[i] \in \mathbb{R}^C$, feature vector of pixel i.
- $x^k \in \mathbb{R}^N$, channel k of x.
- $\mathcal{G}(\mathcal{V}, \mathcal{E})$ directed graph with vertices $\mathcal{V}$, edges $\mathcal{E}$.
- $\mathcal{V}$ set of verticies, $|\mathcal{V}| = N$
- $\mathcal{E}$ set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$.
- $\mathcal{N}(i) = \{j | (i, j) \in \mathcal{E}\}$ set of neighbors of vertex i.
- $\mathcal{L} : \mathcal{E} \to \mathbb{R}^D$ label map.
- $\mathcal{L}[i, j] \in \mathbb{R}^D$ label for edge connecting vertex j to i.

Edge Conditioned Convolution I

Definition (ECC [2])

Consider multi-channel signal $x \in \mathbb{R}^{C_{in}N}$, and *Label-Operator Mapping* $\mathcal{F} : \mathbb{R}^M \to \mathbb{R}^{C_{out} \times C_{in}}$. Define the *Edge-Conditioned Convolution* of x by $\mathcal{F}$ as

$$y[i] = \sum_{j \in \mathcal{N}(i)} \mathcal{F}(\mathcal{L}[i,j])x[j] \tag{1}$$

$$= \sum_{j \in \mathcal{N}(i)} \mathbf{H}_{ij} x[j] \in \mathbb{R}^{C_{out}} \tag{2}$$

where we say $\mathbf{H}_{ij}$ is the *edge-conditioned feature operator* from j to i.

- Consistent with multi-channel grid-convolution if $\mathbf{H}_{ij}$ is stationary (only function of $\tau = j - i$), though non-intuitive.

Edge Conditioned Convolution II

- For 1D grid-convolution, $\mathcal{F}(\mathcal{L}[i,j]) = \langle h, e_{j-i} \rangle$ (e_k dim k unit-vector), i.e. selecting coefficient $j-i$ from filter h.
- When ordering of neighbors no-longer makes sense, stationary kernel can be abandoned.
- [2] learn label-operator mapping $\mathcal{F}$ as 2-layer MLP
- Other spatial graph-convolution formulations don't have grid-convolution as special case, don't handle multi-channel signals properly, and/or use uninformative edge-labels
- [2] set SOA results on point-cloud classification

Table of Contents

GCDN

- Dynamically build K-Nearest Neighbors graph in feature-space to exploit non-local similarities for denoising images (like BM3D)
- Average local convolutions with non-local agregations over graph (ECC)

Motivating Results [3]

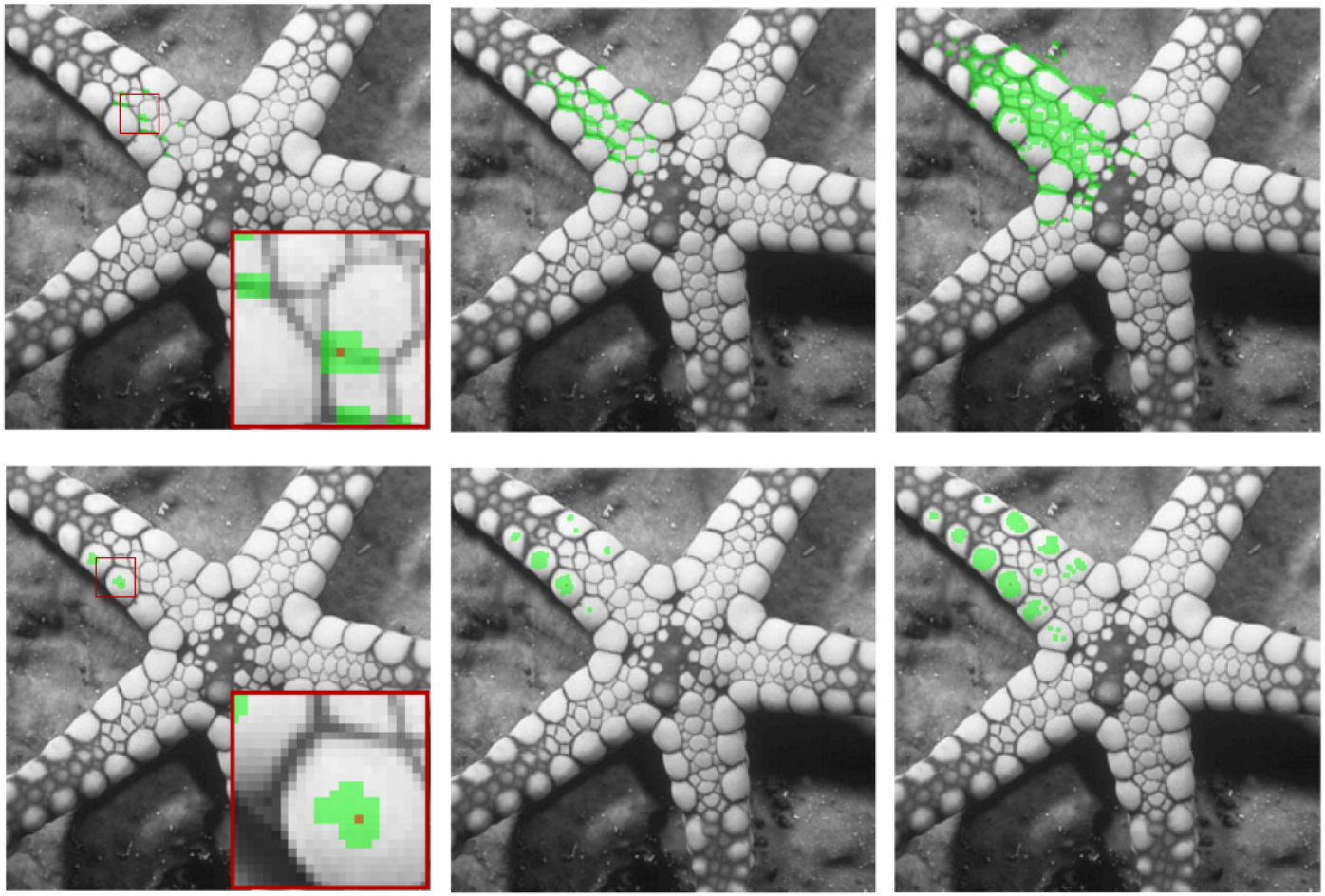
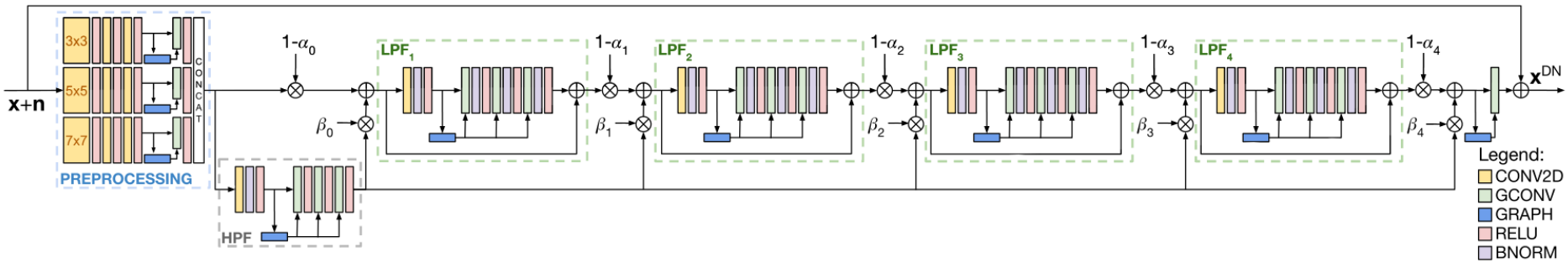


Fig. 6. Receptive field (green) of a single pixel (red) at the output of the three graph-convolutional layers in the LPF_1 block with respect to the input of the first graph-convolutional layer in the block. Top row: gray pixel on an edge. Bottom row: white pixel in a uniform area.

Architecture [3]

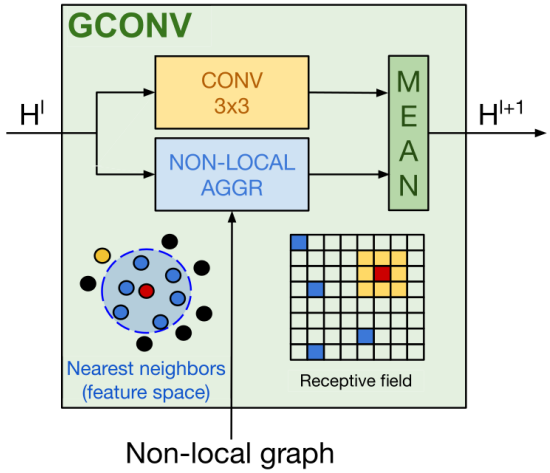


- Residual learning with batch-norm
- Build KNN graph for each "block" (not every convolution)
- All intermediate signals stay at same spatial resolution (no subsampling)
- Can be (loosely) viewed as unrolled proximal gradient descent with graph-laplacian operator,

$$\hat{x} = \arg\min_x \frac{1}{2}\|\mathbf{A}x - y\|_2^2 + \frac{\beta}{2} x^\top \mathbf{L} x$$

(for denoising $\mathbf{A} = \mathbf{I}$)

GConv Module [3] I



- $$x^{(\ell+1)} = \frac{1}{2}\left(x^{(\ell+1),L} + x^{(\ell+1),NL}\right) + b^{\ell+1}$$
- $x^{(\ell+1),L}$ local 3x3 convolution

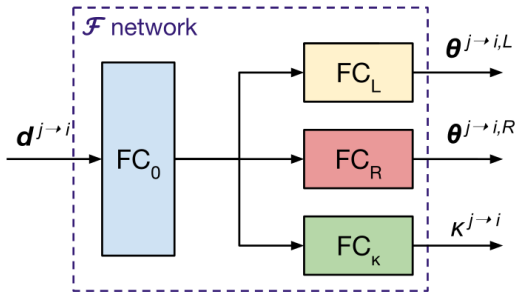
GConv Module [3] II

- $x^{(\ell+1),NL}$ non-local ECC,

$$x^{(\ell+1),NL}[i] = \sum_{j \in \mathcal{N}^{(\ell)}(i)} \mathcal{F}^{(\ell)}(\mathcal{L}^{(\ell)}[i,j]) x^{(\ell)}[j] \tag{3}$$

$$= \sum_{j \in \mathcal{N}^{(\ell)}(i)} \mathbf{H}_{ij}^{(\ell)} x^{(\ell)}[j] \tag{4}$$

- Implement label operator map $\mathcal{F}$ as low-rank MLP, via sum of r outer-products



SOA Synthetic Denoising Results [3]

TABLE III

NATURAL IMAGE DENOISING RESULTS. METRICS ARE PNSR (DB) AND SSIM

Dataset	Noise σ	BM3D [5]	WNNM [31]	TNRD [34]	DnCNN [7]	N^3Net [13]	NLRN [14]	**GCDN**
Set12	15	32.37 / 0.8952	32.70 / 0.8982	32.50 / 0.8958	32.86 / 0.9031	- / -	**33.16** / 0.9070 ± 1.06/0.0222	33.14 / **0.9072** ± 1.05/0.0214
	25	29.97 / 0.8504	30.28 / 0.8557	30.06 / 0.8512	30.44 / 0.8622	30.55 / -	**30.80** / **0.8689** ± 1.19/0.0305	30.78 / 0.8687 ± 1.20/0.0295
	50	26.72 / 0.7676	27.05 / 0.7775	26.81 / 0.7680	27.18 / 0.7829	27.43 / -	**27.64** / **0.7980** ± 1.31/0.0430	27.60 / 0.7957 ± 1.33/0.0443
BSD68	15	31.07 / 0.8717	31.37 / 0.8766	31.42 / 0.8769	31.73 / 0.8907	- / -	**31.88** / 0.8932 ± 2.48/0.0380	31.83 / **0.8933** ± 2.47/0.0383
	25	28.57 / 0.8013	28.83 / 0.8087	28.92 / 0.8093	29.23 / 0.8278	29.30 / -	**29.41** / 0.8331 ± 2.63/0.0564	29.35 / **0.8332** ± 2.66/0.0570
	50	25.62 / 0.6864	25.87 / 0.6982	25.97 / 0.6994	26.23 / 0.7189	26.39 / -	**26.47** / 0.7298 ± 2.78/0.0882	26.38 / **0.7389** ± 2.70/0.0877
Urban100	15	32.35 / 0.9220	32.97 / 0.9271	31.86 / 0.9031	32.68 / 0.9255	- / -	33.42 / 0.9348 ± 2.56/0.0324	**33.47** / **0.9358** ± 2.58/0.0326
	25	29.70 / 0.8777	30.39 / 0.8885	29.25 / 0.8473	29.97 / 0.8797	30.19 / -	30.88 / 0.9003 ± 2.63/0.0460	**30.95** / **0.9020** ± 2.65/0.0476
	50	25.95 / 0.7791	26.83 / 0.8047	25.88 / 0.7563	26.28 / 0.7874	26.82 / -	27.40 / **0.8244** ± 2.52/0.0711	**27.41** / 0.8160 ± 2.53/0.0758

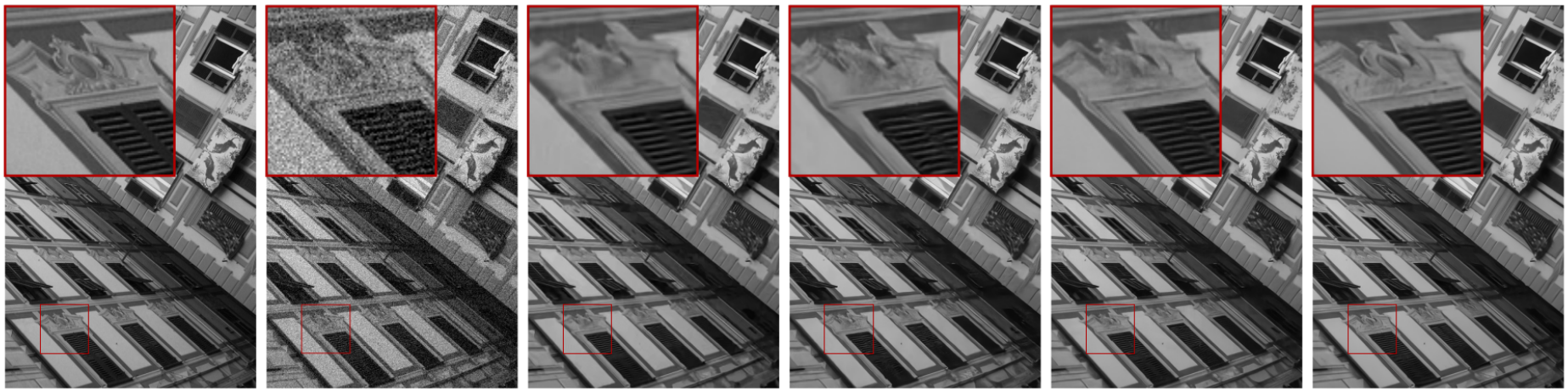


Fig. 11. Extract from Urban100 scene 13, $\sigma = 25$. Left to right: ground truth, noisy (20.16 dB), BM3D (30.40 dB), DnCNN (30.71 dB), NLRN (31.41 dB), GCDN (**31.53 dB**).

SOA Real Image Denoising Results [3]

REAL IMAGE DENOISING (SIDD DATASET)

	CBM3D	DnCNN	NLRN	GCDN
PSNR	38.73 ± 2.95 dB	39.98 ± 3.17 dB	41.24 ± 2.64 dB	**41.48** ± 2.15 dB
SSIM	0.9587 ± 0.0138	0.9605 ± 0.0158	0.9652 ± 0.0144	**0.9697** ± 0.0132

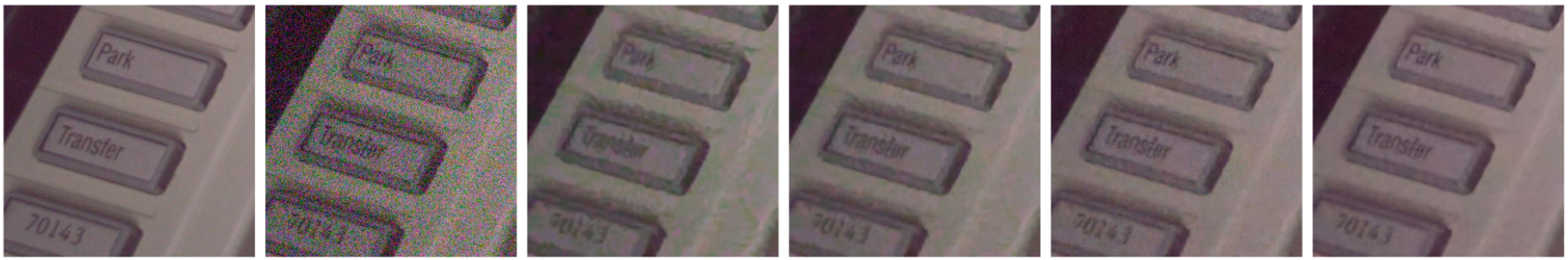
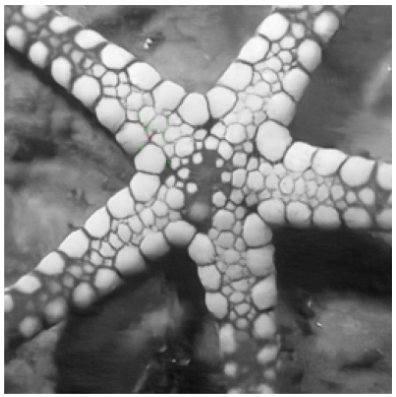
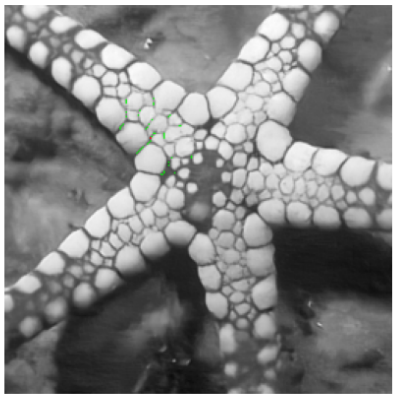
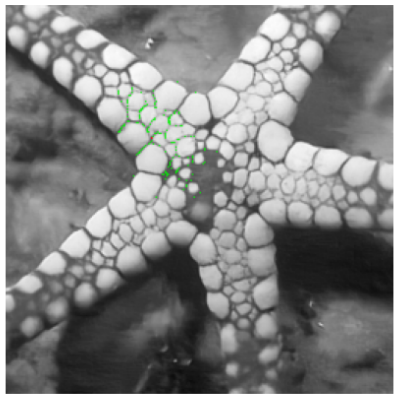
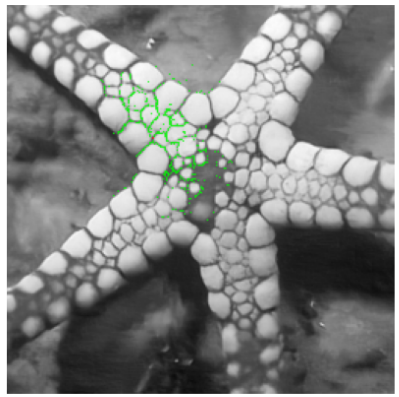
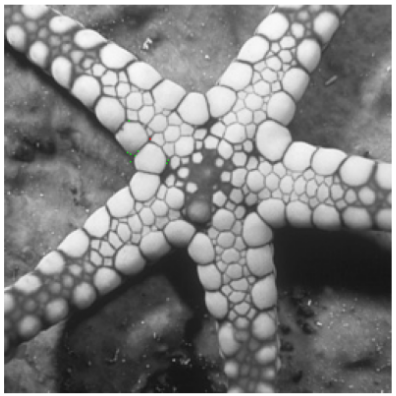
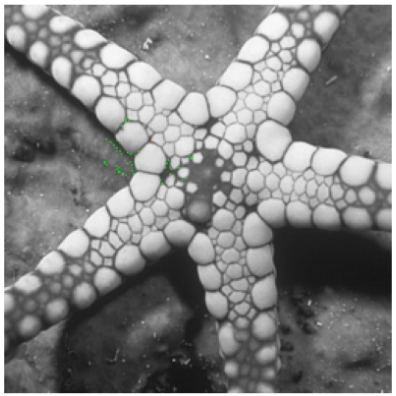
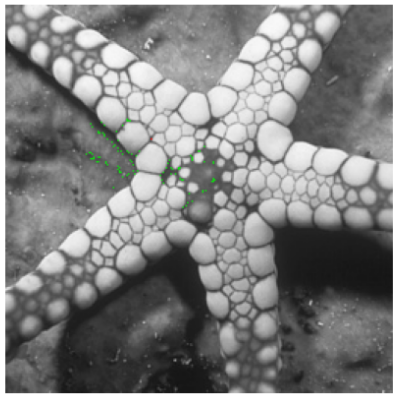
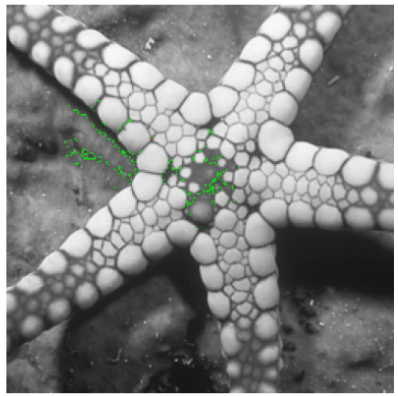


Fig. 13. Real image denoising. Left to right: ground truth, noisy (23.69 dB), CBM3D (36.90 dB), DnCNN (38.76 dB), NLRN (40.78 dB), GCDN (**41.21 dB**). Image intensity rescaled for visualization.

Receptive Field: Learned feature-domain vs. image-domain

Extensions

- SOA Point-Cloud denoising [1]

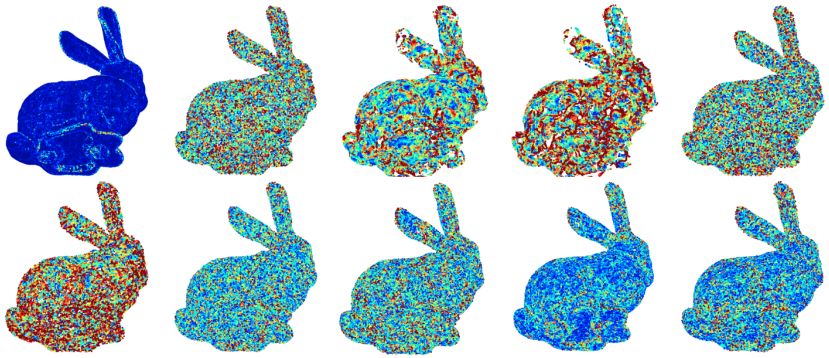
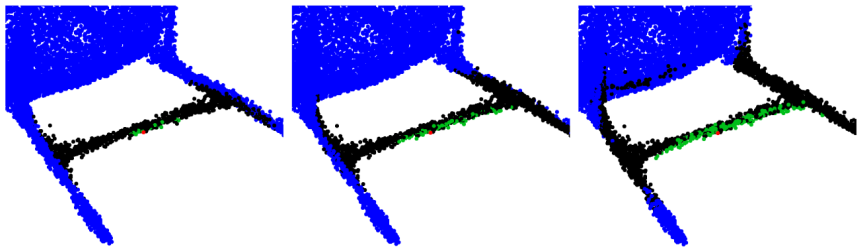


Fig. 3. Denosing results for $\sigma = 0.015$. Color represents unoriented normal angle error (red is high, blue is low). Top left to bottom right: clean point cloud (UNAE = 3.75°), DGCNN (29.73°), APSS (26.29°), RIMLS (33.63°), AWLOP (29.18°), MRPCA (37.50°), GLR (22.08°), PointCleanNet (23.63°), GPDNet MSE (16.62°), GPDNet MSE-SP (23.11°).



References I

[1] Francesca Pistilli et al. "Learning Graph-Convolutional Representationsfor Point Cloud Denoising". In: *The European Conference on Computer Vision (ECCV)*. 2020.

[2] Martin Simonovsky and Nikos Komodakis. "Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs". In: *CVPR*. 2017. URL: https://arxiv.org/abs/1704.02901.

[3] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. "Image Denoising with Graph-Convolutional Neural Networks". In: *2019 26th IEEE International Conference on Image Processing (ICIP)*. 2019.