Gene Cheung
National Institute of Informatics
8th June, 2018

Recent Advances in Graph Spectral Image Processing

.

Acknowledgement

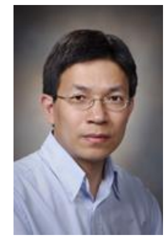
Collaborators:

- Y. Nakatsukasa (NII, Japan)
- S. Muramatsu (Niigata, Japan)
- **A. Ortega** (USC, USA)
- **D. Florencio** (MSR, USA)
- **P. Frossard** (EPFL, Switzerland)
- J. Liang, I. Bajic (SFU, Canada)
- X. Wu (McMaster U, Canada)
- V. Stankovic (U of Strathclyde, UK)
- **P. Le Callet** (U of Nantes, France)
- **X. Liu** (HIT, China)
- W. Hu, J. Liu, Z. Guo, W. Gao (Peking U., China)
- X. Ji, L. Fang (Tsinghua, China)
- Y. Zhao (BJTU, China)
- **C.-W. Lin** (National Tsing Hua University, Taiwan)
- E. Peixoto, B. Macchiavello, E. M. Hung (U. Brasilia, Brazil)

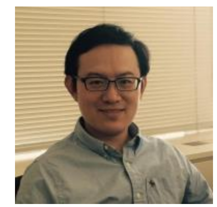


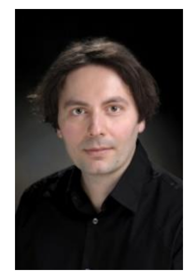






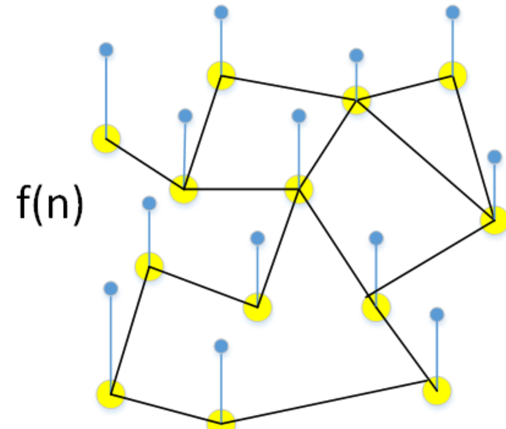






Why GSP for Image Processing?

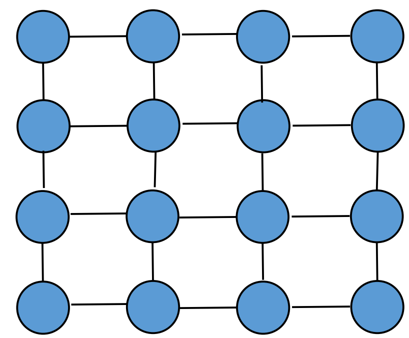
- **GSP**: signals on ***irregular*** data kernels described by graphs.
  - Graph: nodes and edges.
  - Edges reveals *node-to-node relationships*.



example graph-signal

1. Graph captures underlying image statistics.
   - Ex: GFT decorrelates signal for compression.

2. Graph captures (dis)similarities of pixels.
   - Ex: Bilateral filter weights based on Euclidean / photometric distance.



GSP provides design space of spectrum for image filtering.

[1] G. Cheung et al. "**Graph Spectral Image Processing**," *Proceedings of IEEE*, vol.106, no.4, pp. 907-930, May 2018.

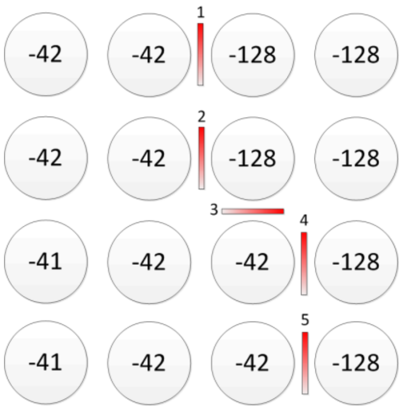
Outline

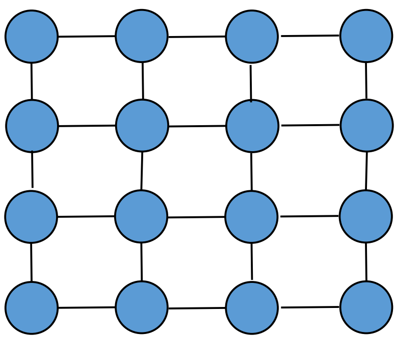
- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV
- Deep GLR
- Ongoing & Future Work

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV
- Deep GLR
- Ongoing & Future Work

GFT for Image Compression





- DCT are ***fixed*** basis. Can we do better?
- **Idea**: use ***adaptive*** GFT to improve sparsity [1].

1. Assign edge weight 1 to adjacent pixel pairs.
2. Assign edge weight 0 to sharp signal discontinuity.
3. Compute GFT for transform coding, transmit coeff.

GFT

$$\tilde{x} = V^T x$$

4. Transmit bits (*contour*) to identify chosen GFT to decoder (overhead of GFT).

[1] G. Shen et al., "**Edge-adaptive Transforms for Efficient Depth Map Coding**," *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[2] W. Hu, G. Cheung, X. Li, O. Au, "**Depth Map Compression using Multi-resolution Graph-based Transform for Depth-image-based Rendering**," *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

GFT: Derivation of Optimal Edge Weights

- Assume a 1D 1st-order ***autoregressive (AR) process*** $\mathbf{x} = [x_1, ..., x_N]^T$ where,

0-mean r.v. with large var. σ^2

$$x_k = \begin{cases} \eta & k = 1 \\ x_{k-1} + e_k & 1 < k \le N \end{cases}$$

0-mean r.v. with var. σ_k^2

$$\mathbf{F}\mathbf{x} = \mathbf{b}, \qquad \mathbf{x} = \mathbf{F}^{-1}\mathbf{b}$$

$$\begin{aligned} x_1 &= \eta \\ x_2 - x_1 &= e_2 \\ &\vdots \\ x_N - x_{N-1} &= e_N \end{aligned} \Longrightarrow \mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & \ddots & \ddots & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \quad \mathrm{b} = \begin{bmatrix} \eta \\ e_2 \\ \vdots \\ \\ \\ e_N \end{bmatrix}$$

GFT: Derivation of Optimal Edge Weights

- Covariance matrix

$$\mathbf{C} = E\left[(\mathrm{x}-\mu)(\mathrm{x}-\mu)^T\right]$$
$$= E\left[\mathrm{x}\,\mathrm{x}^T\right] = E\left[\mathbf{F}^{-1}\mathrm{b}\,\mathrm{b}^T\left(\mathbf{F}^{-1}\right)^T\right]$$
$$= \mathbf{F}^{-1}E\left[\mathrm{b}\,\mathrm{b}^T\right]\left(\mathbf{F}^{-1}\right)^T$$

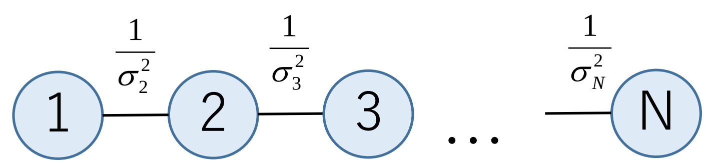
$$E\left[\mathbf{bb}^T\right] = \begin{bmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_N^2 \end{bmatrix}$$

- Precision matrix (tri-diagonal)

large

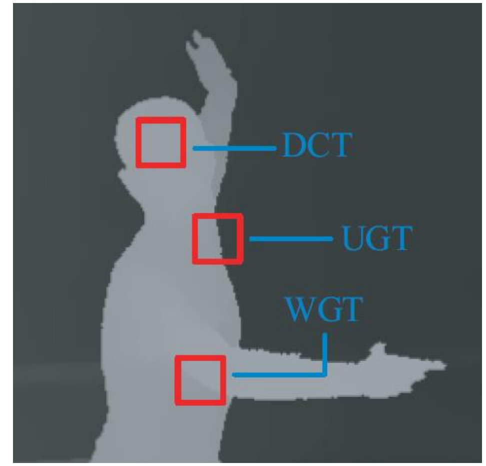
$$\mathrm{Q} = \mathrm{C}^{-1} = \begin{bmatrix} \frac{1}{\sigma^2}+\frac{1}{\sigma_2^2} & -\frac{1}{\sigma_2^2} & 0 & \cdots & 0 \\ -\frac{1}{\sigma_2^2} & \frac{1}{\sigma_2^2}+\frac{1}{\sigma_3^2} & -\frac{1}{\sigma_2^2} & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -\frac{1}{\sigma_{N-1}^2} & \frac{1}{\sigma_{N-1}^2}+\frac{1}{\sigma_N^2} & -\frac{1}{\sigma_N^2} \\ 0 & \cdots & 0 & -\frac{1}{\sigma_N^2} & \frac{1}{\sigma_N^2} \end{bmatrix} \approx \mathbf{L}$$

Graph Laplacian matrix!



GFT for PWS Image Coding

- Graph Laplacian ≈ Precision Matrix → GFT approx. *Karhunen-Loeve Transform* (KLT).



- Encode blocks with **signal-decorrelation** GFT.

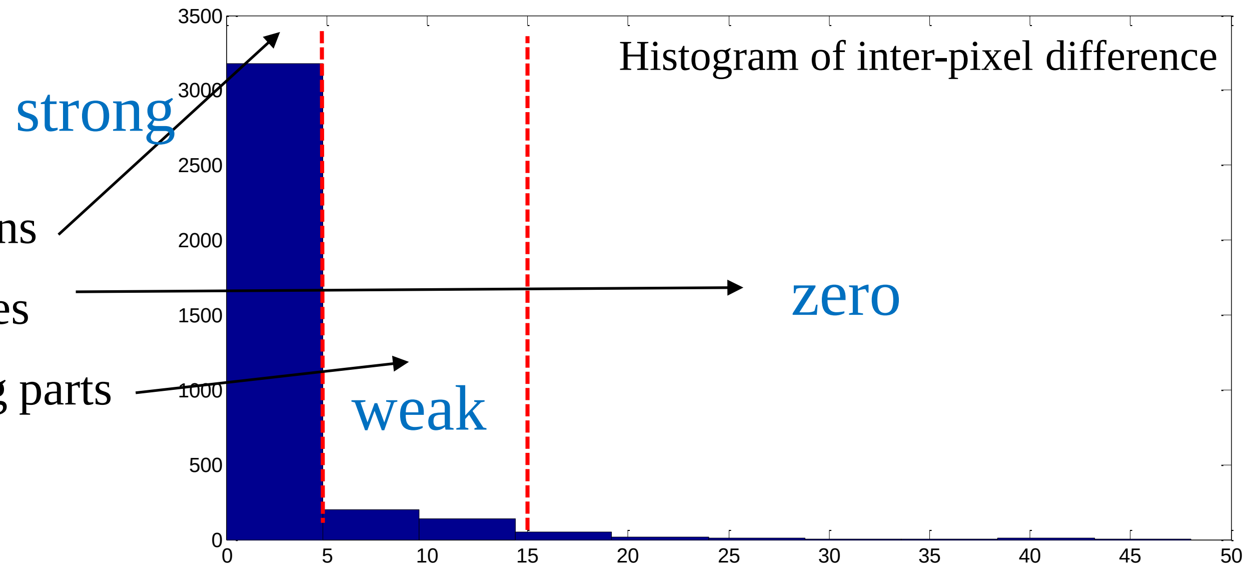
Rate of transform coefficient vector α

Rate of transform description T

$$\min_{\mathbf{W}} \; R_\alpha(\mathbf{x}, \mathbf{W}) + R_T(\mathbf{W})$$

- To limit the description cost R_T, restrict weights to a small discrete set $\mathcal{C} = \{1, 0, c\}$

- "1": *strong correlation* in smooth regions
- "0": *zero correlation* in sharp boundaries
- "c": *weak correlation* in slowly-varying parts



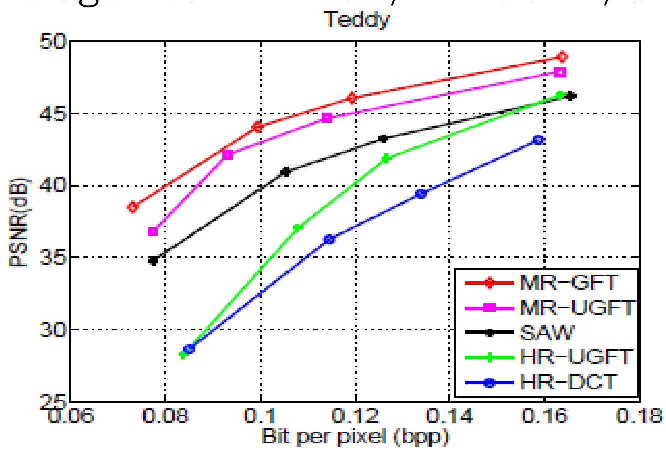
Transform Comparison

	Transform Representation	**Transform Description**
Karhunen-Loeve Transform (KLT)	"Sparsest" signal representation given available statistical model	Can be expensive (if poorly structured)
Discrete Cosine Transform (DCT)	*non-sparse signal representation* across sharp boundaries	little (fixed transform)
Graph Fourier Transform (GFT)	minimizes the total rate of signal's transform representation & transform description	

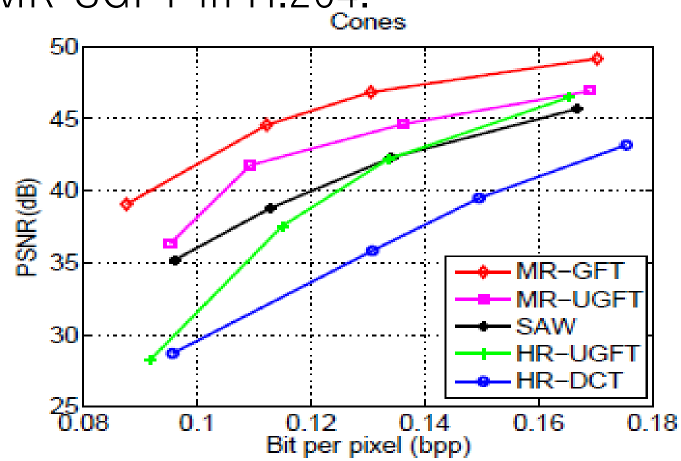
[1] Wei Hu, Gene Cheung, Antonio Ortega, Oscar Au, "**Multiresolution Graph Fourier Transform for Compression of Piecewise Smooth Images**," *IEEE Transactions on Image Processing*, vol.24, no.1, pp.419-433, January 2015.

Experimentation

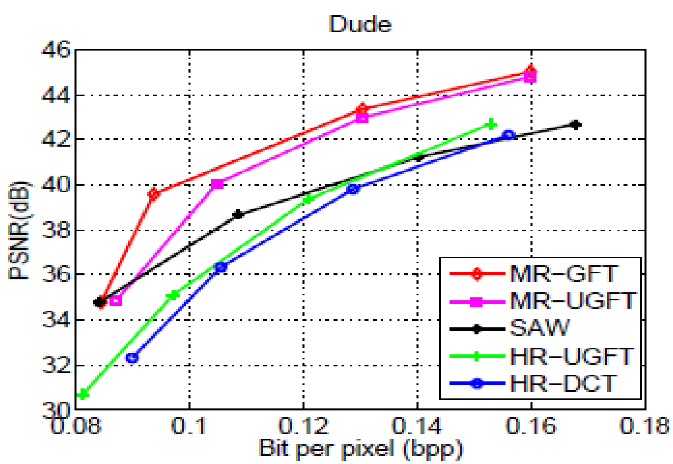
- Setup
  - Test images: depth maps of *Teddy* and *Cones*, and graphics images of *Dude* and *Tsukuba*.
  - Compare against: HR-DCT, HR-SGFT, SAW, MR-SGFT in H.264.
- Results



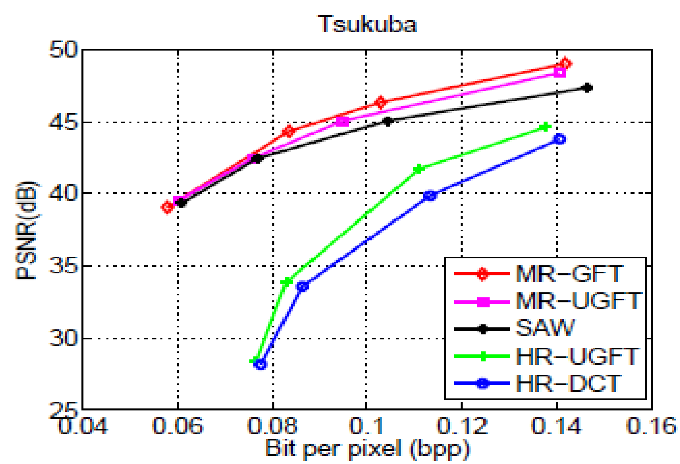
(a)



(b)



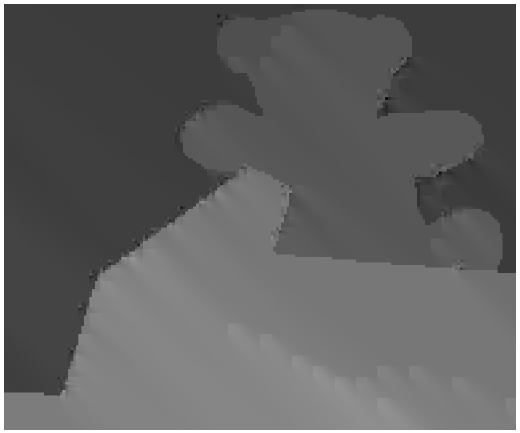
(c)

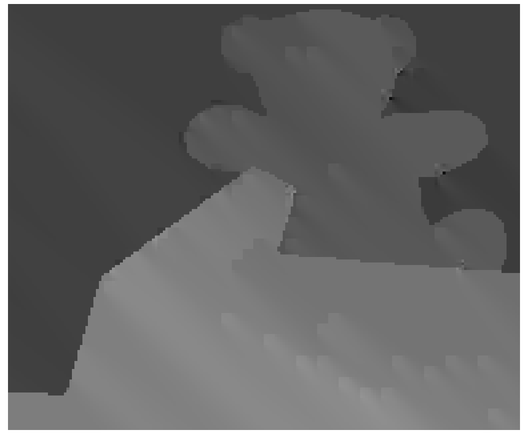


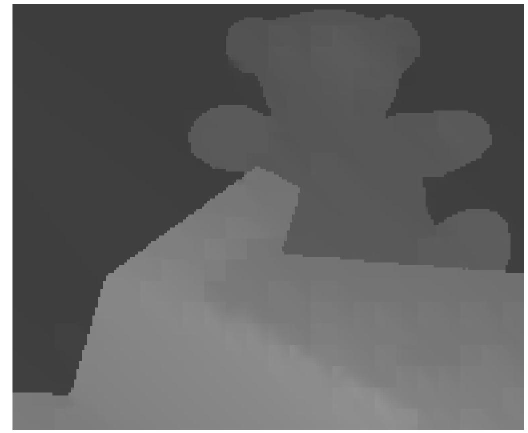
(d)

HR-DCT: 6.8dB
HR-SGFT: 5.9dB
SAW: 2.5dB
MR-SGFT: 1.2dB

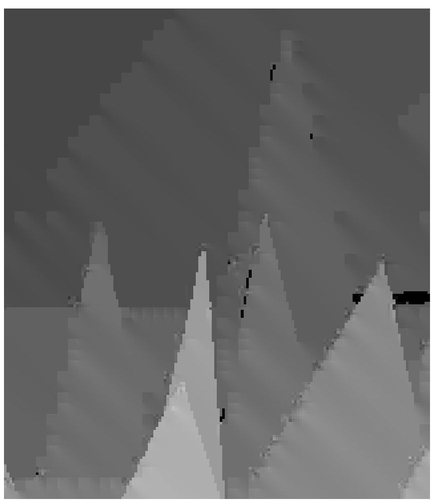
Subjective Results





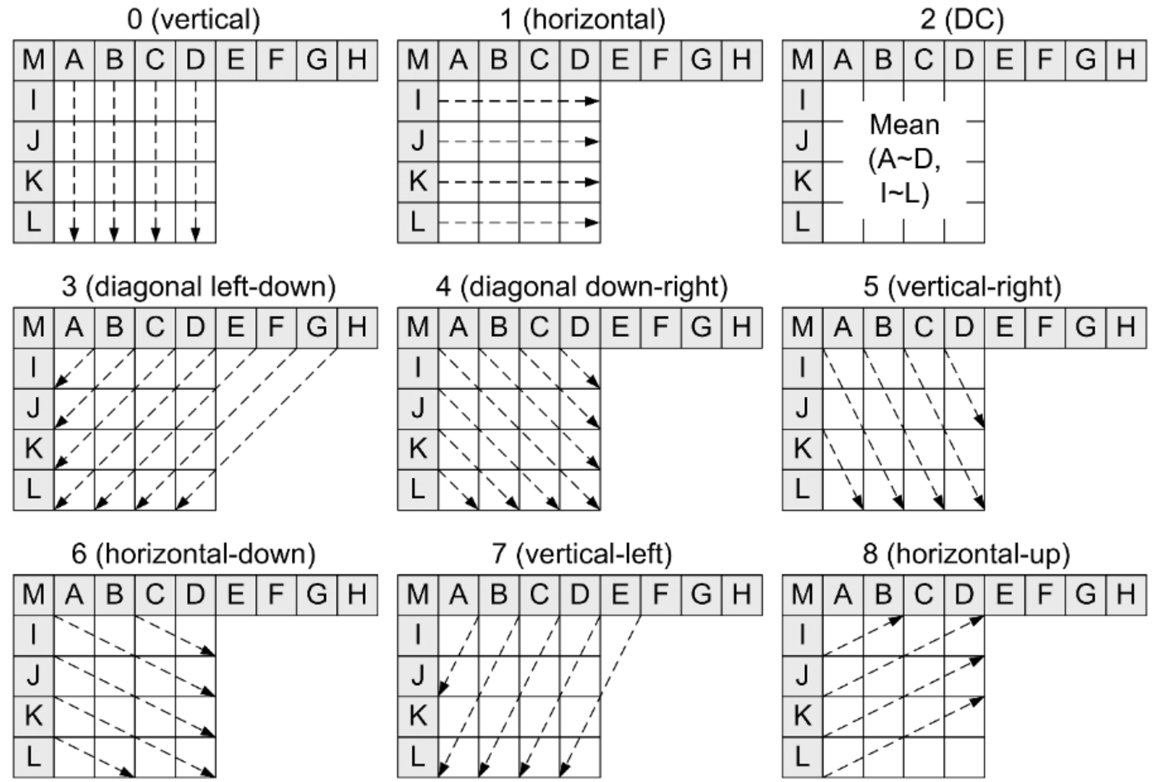


HR-DCT HR-SGFT MR-GFT

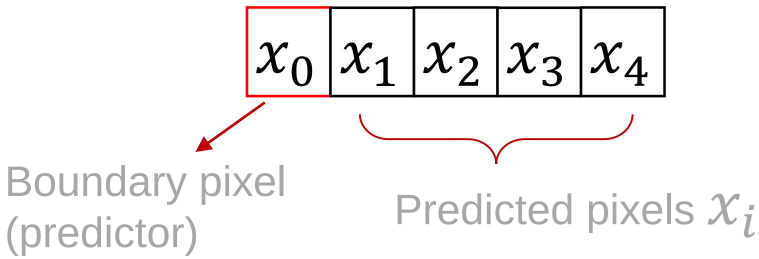


Generalized GFT

Intra-prediction in H.264



- **Intra-prediction**



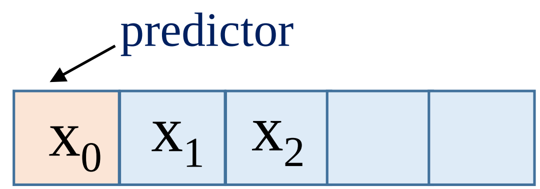
$x_i - x_0$: prediction residuals

- Statistical model for prediction residuals?

[1] W. Hu et al., "**Intra-Prediction and Generalized Graph Fourier Transform for Image Coding**," *IEEE Signal Processing Letters*, vol.22, no.11, pp. 1913-1917, November, 2015.

GGFT: Derivation of Optimal Edge Weights

- Assume a 1D 1st-order ***autoregressive (AR) process*** $\mathbf{x} = [x_1, \ldots, x_N]^T$ where,



predictor from decoded block

$$x_k = \begin{cases} \alpha\, x_0 + e_1 & k = 1 \\ x_{k-1} + e_k & 1 < k \le N \end{cases}$$

0-mean r.v. with var. σ_k^2

$$x_1 = e_1 + \alpha\, x_0$$
$$x_2 - x_1 = e_2$$
$$\vdots$$
$$x_N - x_{N-1} = e_N$$

Prediction residual

Optimal predictor

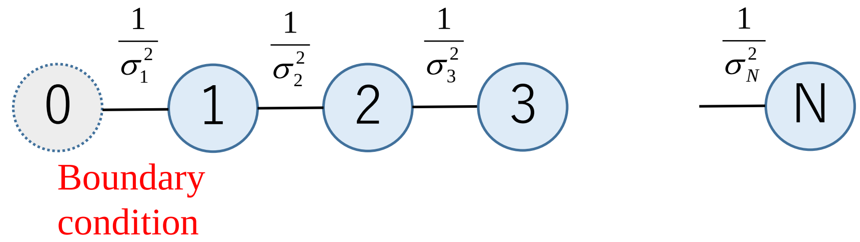
$$\mathbf{F}\,\mathbf{x} = \mathbf{b} + \mathbf{c} \qquad \mathbf{x} = \mathbf{F}^{-1}\mathbf{b} + \mathbf{F}^{-1}\mathbf{c}$$

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & \ddots & \ddots & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \quad \mathrm{b} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ \\ \\ e_N \end{bmatrix}, \quad \mathrm{c} = \begin{bmatrix} \alpha\, x_0 \\ 0 \\ \vdots \\ \\ \\ 0 \end{bmatrix} \qquad \mathbf{F}^{-1}\mathrm{c} = \begin{bmatrix} \alpha\, x_0 \\ \vdots \\ \alpha\, x_0 \end{bmatrix}$$

GGFT: Derivation of Optimal Edge Weights

- Covariance matrix

$$\mathbf{C} = E\left[(\mathrm{r}-\mu)(\mathrm{r}-\mu)^T\right]$$
$$= E\left[\mathrm{r}\,\mathrm{r}^T\right] = E\left[\mathbf{F}^{-1}\mathrm{b}\mathrm{b}^T\left(\mathbf{F}^T\right)^{-1}\right]$$
$$= \mathbf{F}^{-1}E\left[\mathrm{b}\,\mathrm{b}^T\right]\left(\mathbf{F}^T\right)^{-1}$$



- Precision matrix (tri-diagonal)

Graph Laplacian matrix L

diagonal matrix D

Generalized graph Laplacian

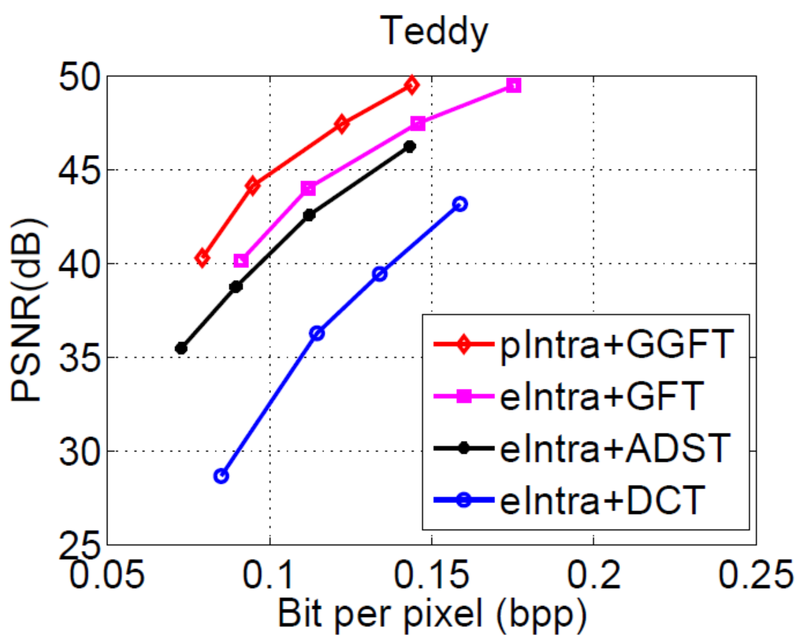
$$\mathbf{Q} = \mathbf{C}^{-1} = \begin{bmatrix} \frac{1}{\sigma_2^2} & -\frac{1}{\sigma_2^2} & 0 & \cdots & 0 \\ -\frac{1}{\sigma_2^2} & \frac{1}{\sigma_2^2}+\frac{1}{\sigma_3^2} & -\frac{1}{\sigma_2^2} & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -\frac{1}{\sigma_{N-1}^2} & \frac{1}{\sigma_{N-1}^2}+\frac{1}{\sigma_N^2} & -\frac{1}{\sigma_N^2} \\ 0 & \cdots & 0 & -\frac{1}{\sigma_N^2} & \frac{1}{\sigma_N^2} \end{bmatrix} + \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & \cdots & 0 \\ 0 & 0 & & \\ \vdots & & \ddots & \\ 0 & \cdots & & 0 \end{bmatrix} \approx \mathcal{L}$$

Defaults to ADST if $\sigma_i^2=1$

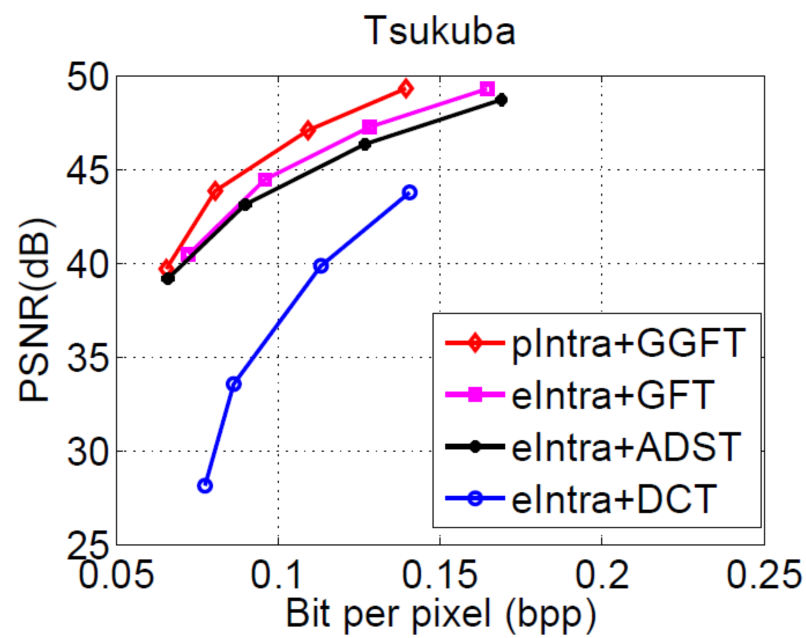
[1] J. Han et al., "**Jointly Optimized Spatial Prediction and Block Transform for video and Image Coding**," IEEE TIP, April 2012.
[2] G. Strang, "**The discrete cosine transform**," **SIAM Review**, vol. 41, pp.135–147, 1999.

Experimental Results

- Test images: PWS images and natural images
- Compare *proposed intra-prediction (pIntra) + GGFT* against:
  - edge-aware intra-prediction (eIntra) + DCT
  - eIntra + ADST
  - eIntra + GFT



(a) Teddy



(b) Tsukuba

SGFT: Derivation of Optimal Edge Weights

- Assume a 1D 1st-order ***autoregressive (AR) process*** $\mathbf{x} = [x_1, ..., x_N]^T$ where,

negative correlation

$$x_k = \begin{cases} \eta & k = 1 \\ -x_{l-1} + e_l & k = l \\ x_{k-1} + e_k & 1 < k \le N \end{cases}$$

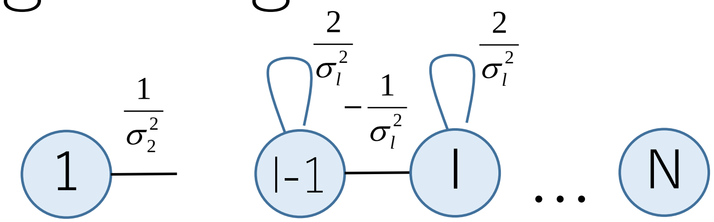
$$\mathbf{F}\mathbf{x} = \mathbf{b}, \qquad \mathbf{x} = \mathbf{F}^{-1}\mathbf{b}$$

$$x_1 = \eta$$
$$x_2 - x_1 = e_2$$
$$x_l + x_{l-1} = e_l$$
$$x_N - x_{N-1} = e_N$$

$$\Rightarrow \quad \mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & \ddots & \ddots & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \quad \mathrm{b} = \begin{bmatrix} \eta \\ e_2 \\ \vdots \\ \\ \\ e_N \end{bmatrix}$$

l-th row

SGFT: Derivation of Optimal Edge Weights



- Precision matrix (tri-diagonal)

large

$$
Q = C^{-1} = \begin{bmatrix}
\frac{1}{\sigma^2} + \frac{1}{\sigma_2^2} & -\frac{1}{\sigma_2^2} & 0 & \cdots & & 0 \\
-\frac{1}{\sigma_2^2} & \frac{1}{\sigma_2^2} + \frac{1}{\sigma_3^2} & -\frac{1}{\sigma_2^2} & & & \vdots \\
0 & \ddots & & & & \\
\vdots & -\frac{1}{\sigma_{l-1}^2} & \frac{1}{\sigma_{l-1}^2} + \frac{1}{\sigma_l^2} & \frac{1}{\sigma_l^2} & & \\
 & & \frac{1}{\sigma_l^2} & \frac{1}{\sigma_l^2} + \frac{1}{\sigma_{l+1}^2} & -\frac{1}{\sigma_l^2} & \\
 & & & & \ddots & \\
0 & \cdots & & 0 & -\frac{1}{\sigma_N^2} & \frac{1}{\sigma_N^2}
\end{bmatrix} \approx \mathcal{L}
$$

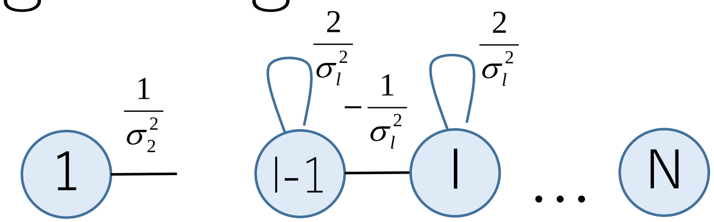
l-th row

(l+1)-th row

Graph Laplacian with negative edges & self-loops

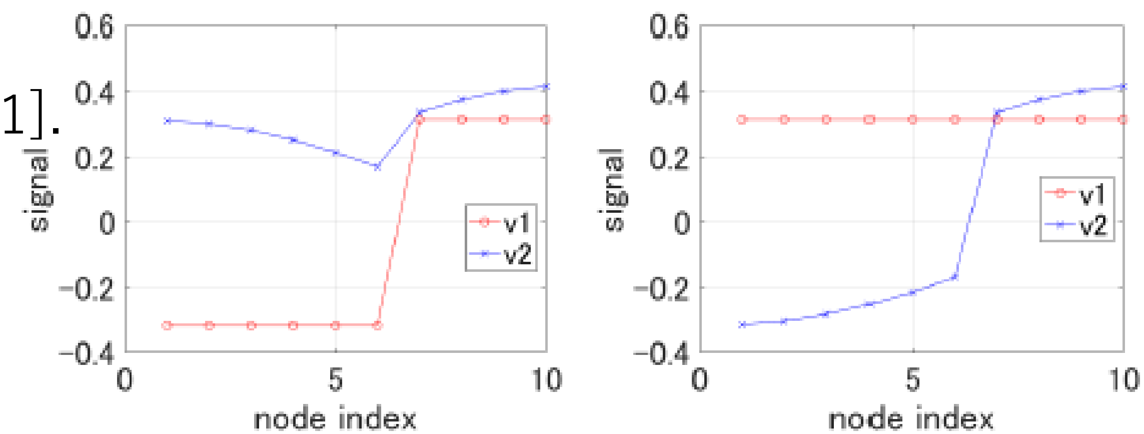
[1] Weng-Tai Su, Gene Cheung, Chia-Wen Lin, "**Graph Fourier Transform with Negative Edges for Depth Image Coding**," *IEEE International Conference on Image Processing*, Beijing, China, September, 2017.

SGFT: Derivation of Optimal Edge Weights



- SGFT approx. KLT for anti-correlation.
- Self-loops ensure PSD (Gershgorin).
- 1st eigenvector is PWC.
- 2nd eigenvector is *not* constant.

- **Negative Edges**:
  - Improve classifier robustness [1].
  - Enhance image contrast.



(a) SGFT basis vectors (b) GFT basis vectors

Fig. 2. First two eigenvectors of: a) loopy Laplacian **Q** for a 10-node graph with negative edge weight −0.1 between nodes 6 and 7; b) graph Laplacian **L** for the same graph with small edge weight 0.1 between nodes 6 and 7. Other edge weights are 1.

[1] Gene Cheung, Weng-Tai Su, Yu Mao, Chia-Wen Lin, "Robust Semi-Supervised Graph Classifier Learning with Negative Edge Weights," accepted to *IEEE Transactions on Signal and Information Processing over Networks*, March 2018.

Summary of GFT for Image Coding

- Optimality of GFT, Generalized GFT, Signed GFT for variants of AR models.

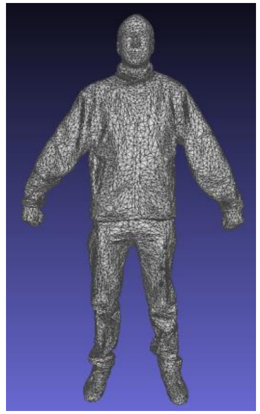
- Selection of models trades off encoding cost of SI.

- Fast implementation (w/o eigen-decomposition) via **Graph Lifting Transform** (GLT) [1] or **Fast Graph Fourier Transform** (FGFT) [2].

[1] Y.-H. Chao et al., "Edge-Adaptive Depth Map Coding with Lifting Transform on Graphs," 31st Picture Coding Symposium, Cairns, Australia, May, 2015.

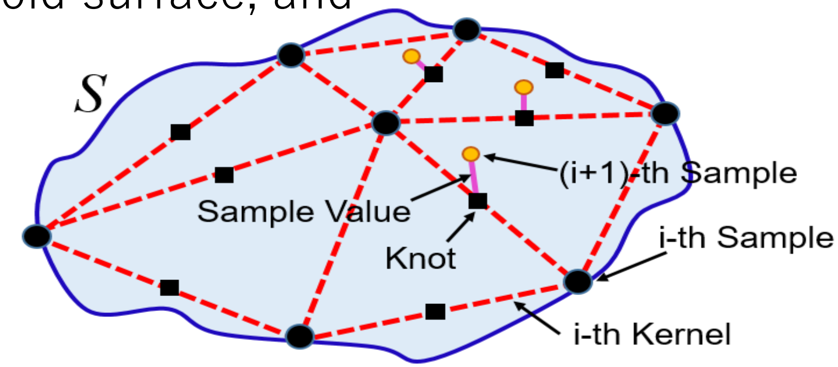
[2] L. Le Magoarou et al., "Approximate Fast Graph Fourier Transforms via Multilayer Sparse Approximations," *IEEE TSIPN*, May, 2018.

Graph-Signal Sampling / Encoding for 3D Point Cloud



MIT $dataset^*$

- **Problem**: Point clouds require encoding specific 3D coordinates.
- **Assumption**: smooth 2D manifold in 3D space.
- **Proposal**: progressive 3D geometry rep. as series of graph-signals.
  1. adaptively identifies new samples on the manifold surface, and
  2. encodes them efficiently as graph-signals.

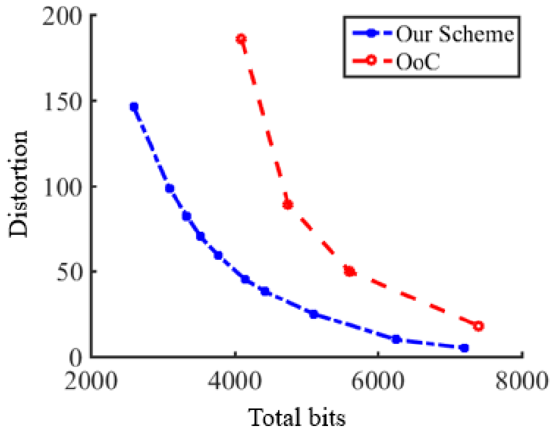


- **Example**:

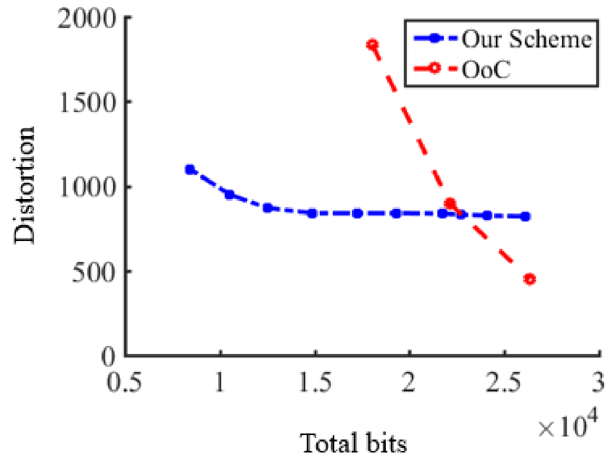
1. Interpolate i^{th} iteration samples (black circles) to a **continuous kernel** (mesh), an approximation of the target surface $\boldsymbol{S}$.
2. New sample locations, **knots** (squares), are located on the kernel surface.
3. **Signed distances** between knots and $\boldsymbol{S}$ are recorded as sample values.
4. **Sample values** (green circles) are encoded as a **graph-signal via GFT**.

Graph-Signal Sampling / Encoding for 3D Point Cloud

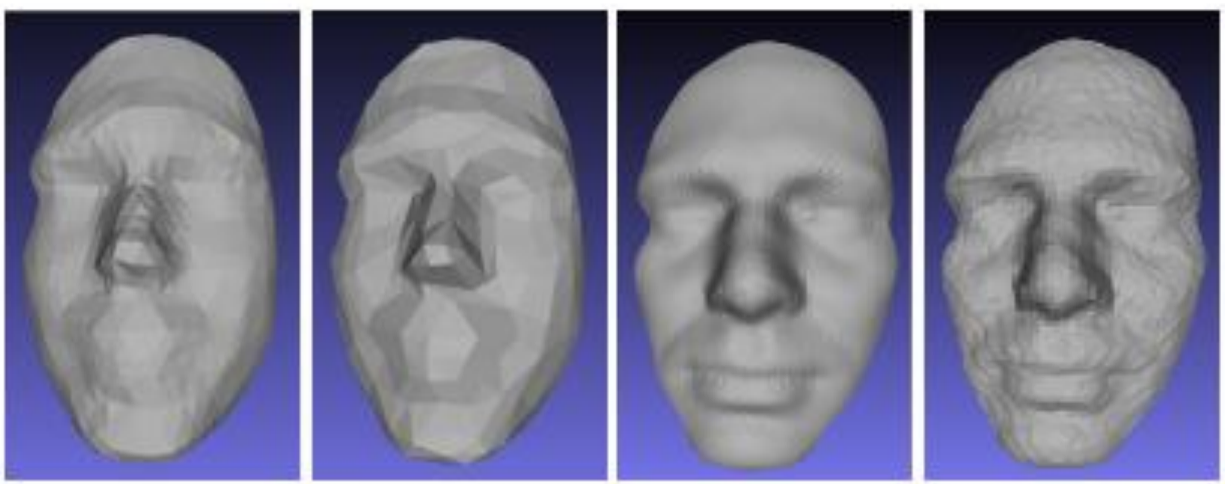
- **Experimental Results**:



(a) Dataset1



(b) Dataset2

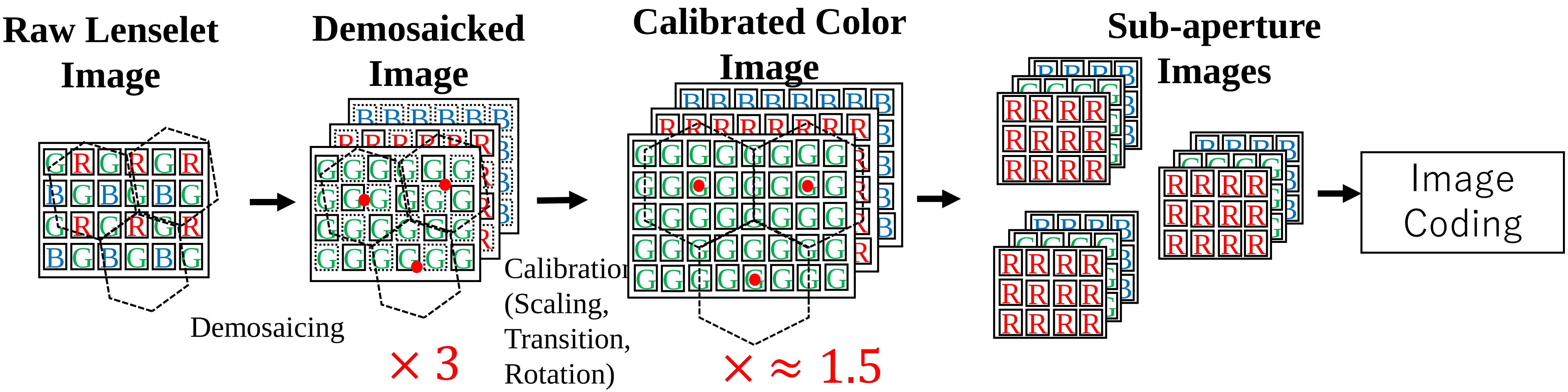


(a) (b)

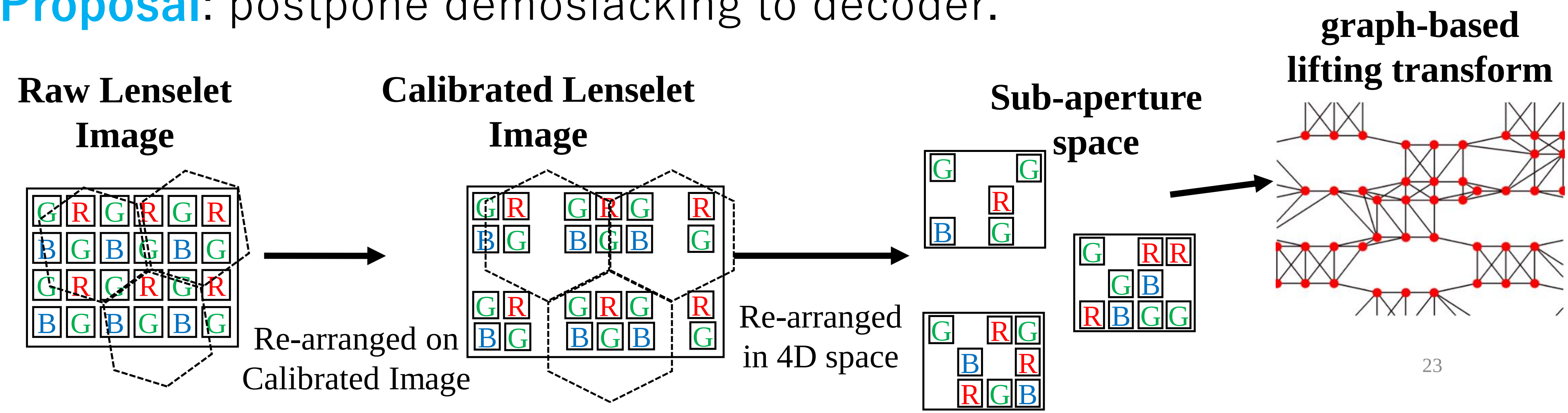
[1] M. Zhao, G. Cheung, D. Florencio, X. Ji, "**Progressive Graph-Signal Sampling and Encoding for Static 3D Geometry Representation**," *IEEE International Conference on Image Processing*, Beijing, China, September, 2017.

Pre-Demosiac Light Field Image Compression Using Graph Lifting Transform

- **Problem**: Sub-aperture images in Light field data are huge.



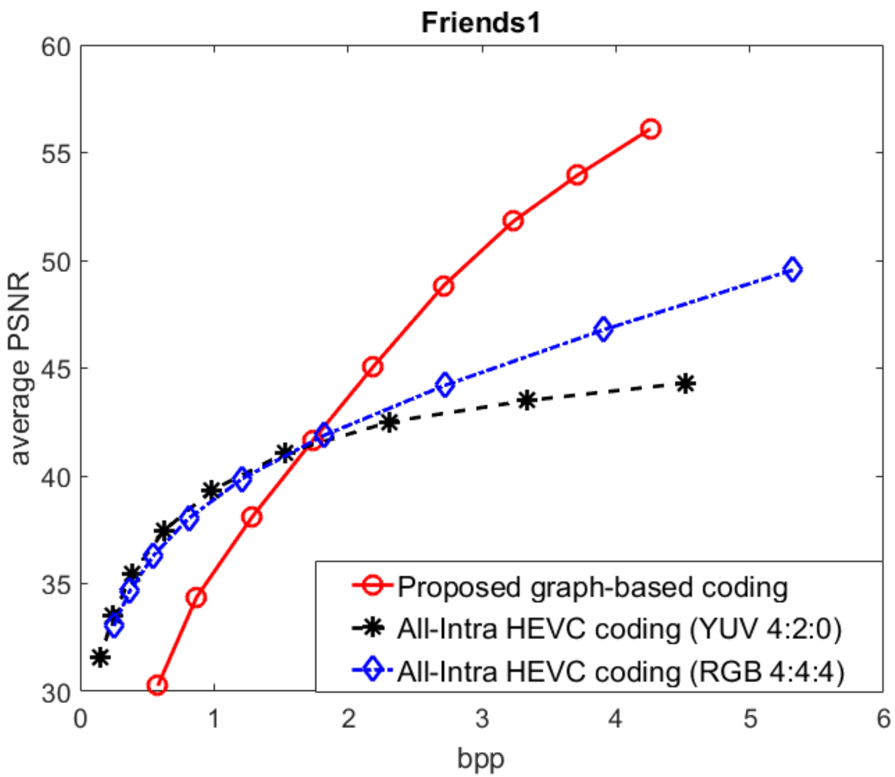
- **Proposal**: postpone demosiacking to decoder.

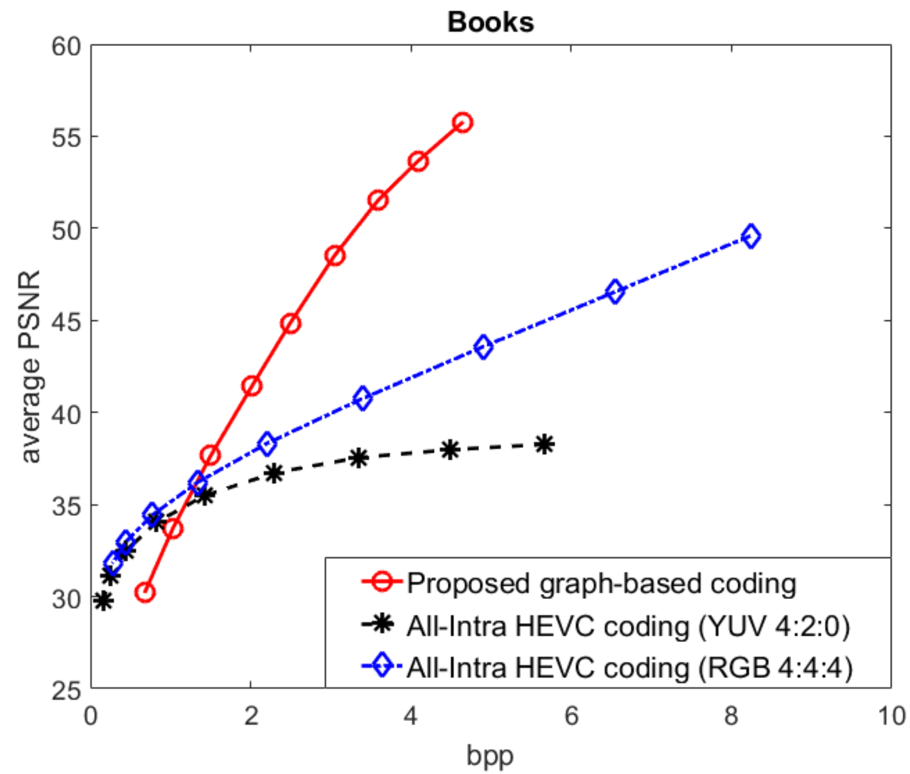


Pre-Demosiac Light Field Image Compression Using Graph Lifting Transform

- **Experimental Results**:

Dataset: EPFL light field image dataset
Baseline: All-intra HEVC coding in YUV4:2:0 and RGB 4:4:4





[1] Y.-H. Chao, G. Cheung, A. Ortega, "**Pre-Demosiac Light Field Image Compression Using Graph Lifting Transform**," *IEEE Int'l Conf. on Image Processing*, Beijing, China, September, 2017. (**Best student paper award**)

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV
- Deep GLR
- Ongoing & Future Work

Graph Laplacian Regularizer

- $\mathbf{x}^T \mathbf{L} \mathbf{x}$ (graph Laplacian regularizer) [1]) is one smoothness measure.

$$\mathbf{x}^T \mathbf{L} \mathbf{x} = \frac{1}{2} \sum_{i,j} w_{i,j} \left(x_i - x_j \right)^2 = \sum_k \lambda_k \tilde{x}_k^2$$

signal smooth in nodal domain

signal contains mostly low graph freq.

- **Signal Denoising**:

observation

desired signal

$$\mathbf{y} = \mathbf{x} + \mathbf{v}$$

noise

- **MAP Formulation**:

pixel intensity diff.

pixel location diff.

$$w_{i,j} = \exp\left(\frac{-\left\| x_i - x_j \right\|_2^2}{\sigma_1^2} \right) \exp\left(\frac{-\left\| l_i - l_j \right\|_2^2}{\sigma_2^2} \right)$$

Bilateral filter weights

fidelity term

$$\min_x \left\| \mathbf{y} - \mathbf{x} \right\|_2^2 + \mu\, \mathbf{x}^T \mathbf{L} \mathbf{x}$$

smoothness prior

update edge weights

$$\left(\mathbf{I} + \mu \mathbf{L} \right) \mathbf{x}^* = \mathbf{y}$$

linear system of eqn's w/ sparse, symmetric PD matrix

[1] P. Milanfar, "**A Tour of Modern Image Filtering: New Insights and Methods, Both Practical and Theoretical**," *IEEE Signal Processing Magazine*, vol.30, no.1, pp.106-128, January 2013.

Graph Laplacian Regularizer

$$w_{i,j} = \exp\left(\frac{-\left\|x_i - x_j\right\|_2^2}{\sigma_1^2}\right)\exp\left(\frac{-\left\|l_i - l_j\right\|_2^2}{\sigma_2^2}\right)$$

- $\mathrm{x}^T \mathrm{L}(\mathrm{x})\mathrm{x}$ promotes ***piecewise smooth*** (PWS) signal behavior [1].

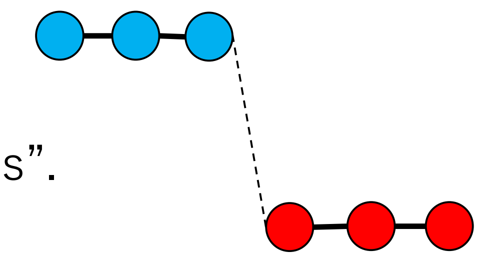
$$\mathrm{x}^T \mathrm{L}(\mathrm{x})\mathrm{x} = \frac{1}{2}\sum_{i,j} w_{i,j}\left(x_i - x_j\right)^2 = \frac{1}{2}\sum_{i,j} u_{i,j}\exp\left\{-\frac{\left(x_i - x_j\right)^2}{\sigma^2}\right\}\left(x_i - x_j\right)^2$$

- **Spectral Clustering** [2]:

Rayleigh quotient

$$\mathrm{v}^* = \arg\min_{\mathrm{v}} \frac{\mathrm{v}^T \mathrm{L_n} \mathrm{v}}{\mathrm{v}^T \mathrm{v}} \quad s.t.\ \mathrm{v}^T \mathrm{v}_1 = 0$$

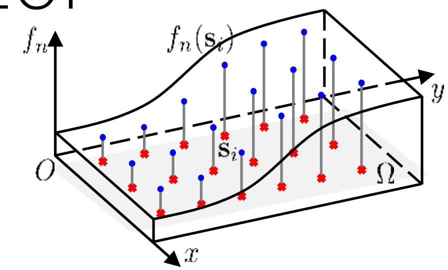
- v_1 minimizes obj → Sol'n is 2nd eigenvector of $\mathrm{L_n}$.
- 2nd eigenvalue—**Fiedler number**—measures "connectedness".
- PWS signal = 2 clusters of similar nodes → $\mathrm{x}^T \mathrm{L} \mathrm{x} \approx 0$



[1] X. Liu, G. Cheung, X. Wu, D. Zhao, "**Random Walk Graph Laplacian based Smoothness Prior for Soft Decoding of JPEG Images**," *IEEE Transactions on Image Processing*, vol.26, no.2, pp.509-524, February 2017.

[2] J. Shi and J. Malik, "**Normalized cuts and image segmentation**," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

Analysis of Graph Laplacian Regularizer



- [1] shows $S_G(\mathbf{u}) = \mathbf{u}^T \mathbf{L}\mathbf{u}$ converges to continuous functional S_Ω and objective becomes:

$$u^\star = \arg\min_u \|u - z_0\|_\Omega^2 + \tau \cdot \int_\Omega \nabla u^T \mathbf{D} \nabla u \, ds,$$

$$\mathbf{D} = \mathbf{G}^{-1} \left(\sqrt{\det \mathbf{G}} \right)^{2\gamma - 1}$$

- Solution can be implemented as anisotropic diffusion:

feature function vector

$$\mathbf{v}_i = [\mathbf{f}_1(i)\ \mathbf{f}_2(i) \ldots \mathbf{f}_N(i)]^T$$

$$\partial_t u = \operatorname{div}(\mathbf{D}\nabla u),$$

$$u(\mathbf{s}, t = 0) = z_0(\mathbf{s}).$$

diffusivity

distance $d_{ij}^2 = \|\mathbf{v}_i - \mathbf{v}_j\|_2^2$

edge weight

$$w_{ij} = (\rho_i \rho_j)^{-\gamma} \psi(d_{ij})$$

  - it not only smooths but may also sharpens the image,
  - promote piecewise smooth images, like **total variation** (TV).

metric space $\mathbf{G} = \sum_{n=1}^{N} \nabla f_n \cdot \nabla f_n^T$

[1] J. Pang, G. Cheung, "**Graph Laplacian Regularization for Inverse Imaging: Analysis in the Continuous Domain**," *IEEE Transactions on Image Processing*, vol. 26, no.4, pp.1770-1785, April 2017.

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer: Image Denoising
  - Reweighted Graph TV:
- Deep GLR
- Ongoing & Future Work

Optimal Graph Laplacian Regularization for Denoising

- Adopt a patch-based recovery framework, for a noisy patch $\mathbf{p}_0$
  1. Find $K-1$ patches similar to $\mathbf{p}_0$ in terms of Euclidean distance.
  2. Compute **feature functions**, leading to edge weights and Laplacian.
  3. Solve the unconstrained quadratic optimization:

$$\boxed{\mathrm{q}^* = \arg\min_{\mathrm{q}} \left\|\mathrm{p}_0 - \mathrm{q}\right\|_2^2 + \lambda \mathrm{q}^T \mathrm{L} \mathrm{q} \quad \Rightarrow \quad \mathrm{q} = \left(\mathrm{I} + \lambda \mathrm{L}\right)^{-1} \mathrm{p}_0}$$

  to obtain the denoised patch.

Spatial

$$\mathbf{f}_1^D(i) = \sqrt{\sigma^2 + \alpha} \cdot x_i$$

$$\mathbf{f}_2^D(i) = \sqrt{\sigma^2 + \alpha} \cdot y_i$$

Intensity

$$\mathbf{f}_3^D = \frac{1}{K + \sigma_e^2 / \sigma_g^2} \sum_{k=0}^{K-1} \mathbf{p}_k$$

- Aggregate denoised patches to form an updated image.
- Denoise the image iteratively to gradually enhance its quality.
- Optimal Graph Laplacian Regularization for Denoising (OGLRD).

[1] J. Pang, G. Cheung, "**Graph Laplacian Regularization for Inverse Imaging: Analysis in the Continuous Domain**," *IEEE Transactions on Image Processing*, vol. 26, no.4, pp.1770-1785, April 2017.

Denoising Experiments (natural images)

- Subjective comparisons ($\sigma_I = 40$)

Original

Noisy, 16.48 dB

K-SVD, 26.84 dB

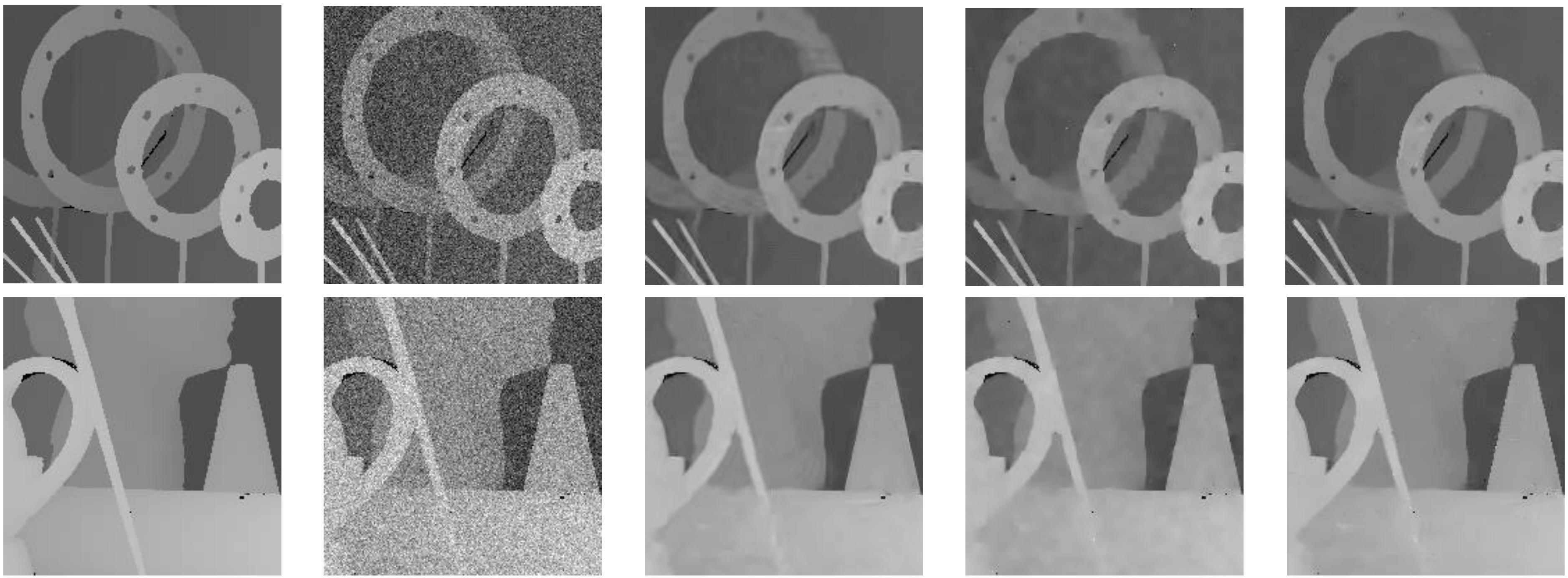
BM3D, 27.99 dB

PLOW, 28.11 dB

OGLR, 28.35 dB

Denoising Experiments (depth images)

- Subjective comparisons ($\sigma_I = 30$)



Original　Noisy, 18.66 dB　BM3D, 33.26 dB　NLGBT, 33.41dB　OGLR, 34.32 dB

[1] W. Hu et al., "**Depth Map Denoising using Graph-based Transform and Group Sparsity**," *IEEE International Workshop on Multimedia Signal Processing*, Pula (Sardinia), Italy, October, 2013.

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer: Soft Decoding of JPEG Images
  - Reweighted Graph TV
- Deep GLR
- Ongoing & Future Work

Soft Decoding of JPEG Images

- **Setting**: **JPEG** compresses natural images:
  1. Divide image into 8x8 blocks, DCT.
  2. Perform DCT transform per block and quantize:

DCT

$$q_i = round(Y_i / Q_i), \quad \mathbf{Y} = \mathbf{T}\mathbf{y}$$

8x8 pixel block

quantization parameter

DCT Coefficients

  3. Quantized DCT coeff entropy coded.
- **Decoder**: uncertainty in signal reconstruction:

$$q_i Q_i \le Y_i \le (q_i + 1) Q_i, i = 1, 2, \cdots, 64.$$

[1] A. Zakhor, "**Iterative procedures for reduction of blocking effects in transform image coding**," *IEEE Transactions on Circuits and Systems for Video Technology*,, vol. 2, no. 1, pp. 91–95, Mar 1992.

[2] K. Bredies and M. Holler, "**A total variation-based JPEG decompression model**," *SIAM J. Img. Sci.*, vol. 5, no. 1, pp. 366–393, Mar. 2012.

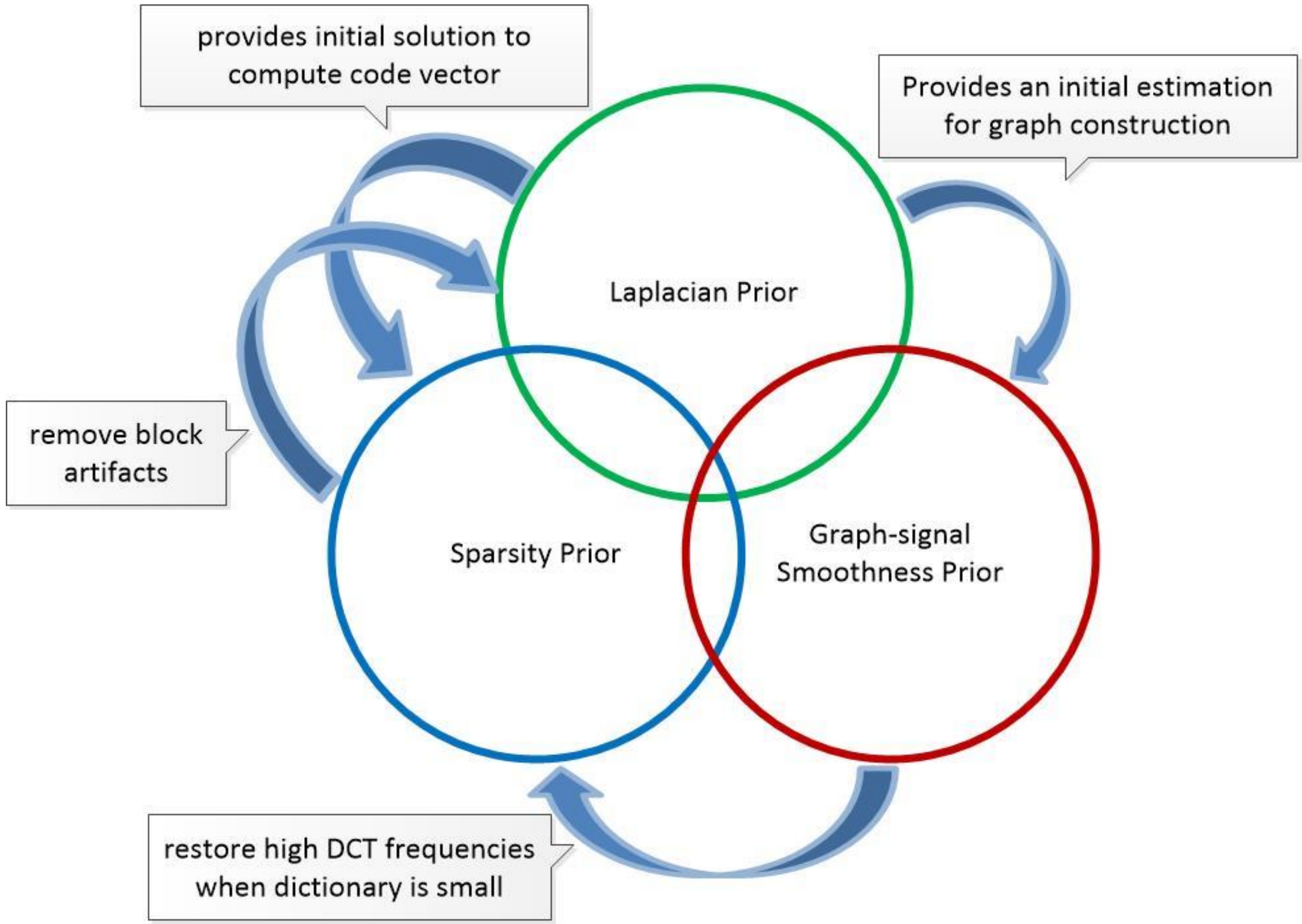
[3] H. Chang, M. Ng, and T. Zeng, "**Reducing artifacts in jpeg decompression via a learned dictionary**," *IEEE Transactions on Signal Processing*,,vol. 62, no. 3, pp. 718–728, Feb 2014.

LERaG for Soft Decoding of JPEG Images

- **Problem**: reconstruct image given indexed quant. bin in 8x8 DCT.

- **Procedure**:

1. Initialize *per-block* MMSE sol'n via Laplacian prior.
2. Solve *per-patch* signal restoration problem w/ 2 priors:
    1. Sparsity prior
    2. Graph-signal smoothness prior



[1] X. Liu, G. Cheung, X. Wu, D. Zhao, "**Random Walk Graph Laplacian based Smoothness Prior for Soft Decoding of JPEG Images**," *IEEE Transactions on Image Processing*, vol.26, no.2, pp.509-524, February 2017.

Soft Decoding Algorithm w/ Prior Mixture

- Objective:

fidelity term

sparsity prior

graph-signal smoothness prior

$$\arg\min_{\{\mathbf{x},\boldsymbol{\alpha}\}} \|\mathbf{x} - \mathbf{\Phi}\boldsymbol{\alpha}\|_2^2 + \lambda_1 \|\boldsymbol{\alpha}\|_0 + \lambda_2 \mathbf{x}^T (d_{\min}^{-1}) \mathbf{L}\mathbf{D}^{-1}\mathbf{L}\mathbf{x},$$

graph-signal, code vector

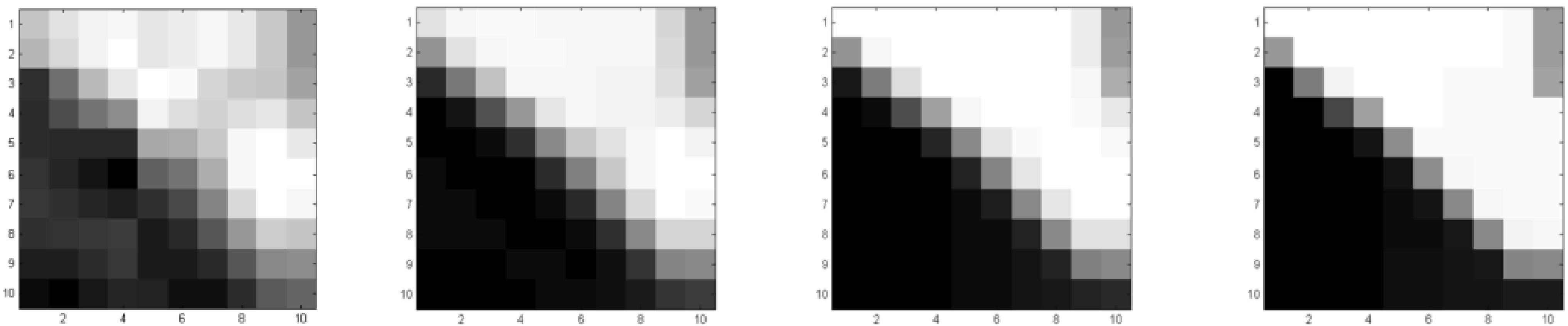
$$\text{s.t. } \mathbf{q}\mathbf{Q} \preceq \mathbf{T}\mathbf{M}\mathbf{x} \prec (\mathbf{q}+1)\mathbf{Q}$$

quantization bin constraint

- Optimization:
  1. Laplacian prior provides an initial estimation;
  2. Fix $\mathbf{x}$ and solve for $\boldsymbol{\alpha}$;
  3. Fix $\boldsymbol{\alpha}$ and solve for $\mathbf{x}$.

Evolution of 2^{nd} Eigenvector

- 2^{nd} eigenvector becomes more PWS:



(a) initialization, LERaG = 76453.02, 2^{nd} Eigenvalue = 0.001079

(b) iter = 1, LERaG = 64827.99, 2^{nd} Eigenvalue = 315e-6

(c) iter = 2, LERaG = 14057.09, 2^{nd} Eigenvalue = 35e-6

(d) iter = 3, LERaG = 6440.29, 2^{nd} Eigenvalue = 3e-6

1. better pixel clusters,
2. smaller Fiedler number (2^{nd} eigenvalue),
3. Smaller smoothness penalty term.

Experimental Setup

- Compared methods
  - **BM3D**: well-known denoising algorithm
  - **KSVD**: with a large enough over-complete dictionary (100x4000); our method uses a much smaller one (100x400).
  - **ANCE**: non-local self similarity [**Zhang et al. TIP14**]
  - **DicTV**: Sparsity + TV [**Chang et al, TSP15**]
  - **SSRQC**: Low rank + Quantization constraint [**Zhao et al. TCSVT16**]

PSNR / SSIM Comparison

QUALITY COMPARISON WITH RESPECT TO PSNR (IN dB) AND SSIM AT QF = 40

Images	JPEG		BM3D [38]		KSVD [8]		ANCE [18]		DicTV [3]		SSRQC [20]		Ours	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Butterfly	29.97	0.9244	31.35	0.9555	31.57	0.9519	31.38	0.9548	31.22	0.9503	32.02	0.9619	**32.87**	**0.9627**
Leaves	30.67	0.9438	32.55	0.9749	33.04	0.9735	32.74	0.9728	32.45	0.9710	32.13	0.9741	**34.42**	**0.9803**
Hat	32.78	0.9022	33.89	0.9221	33.62	0.9149	33.69	0.9169	33.20	0.8988	34.10	0.9237	**34.46**	**0.9268**
Boat	33.42	0.9195	34.77	0.9406	34.28	0.9301	34.64	0.9362	26.08	0.7550	33.88	0.9306	**34.98**	**0.9402**
Bike	28.98	0.9131	29.96	0.9356	30.19	0.9323	30.31	0.9357	29.75	0.9154	30.35	0.9411	**31.14**	**0.9439**
House	35.07	0.8981	36.09	0.9013	36.05	0.9055	36.12	0.9048	35.17	0.8922	36.49	0.9072	**36.55**	**0.9071**
Flower	31.62	0.9112	32.81	0.9357	32.63	0.9271	32.67	0.9314	31.86	0.9084	33.02	0.9362	**33.37**	**0.9371**
Parrot	34.03	0.9291	34.92	0.9397	34.91	0.9371	35.02	0.9397	33.92	0.9227	35.11	0.9401	**35.32**	**0.9401**
Pepper512	34.21	0.8711	34.94	0.8767	34.89	0.8784	34.99	0.8803	34.24	0.8639	35.05	0.8795	**35.19**	**0.8811**
Fishboat512	32.76	0.8763	33.61	0.8868	33.36	0.8809	33.60	0.8861	32.53	0.8496	33.68	0.8859	**33.73**	**0.8871**
Lena512	35.12	0.9089	36.03	0.9171	35.82	0.9146	36.04	0.9177	34.85	0.8986	36.09	0.9187	**36.11**	**0.9191**
Airplane512	33.36	0.9253	34.38	0.9361	34.36	0.9341	34.53	0.9358	33.75	0.9134	35.81	0.9355	**36.07**	**0.9439**
Bike512	29.43	0.9069	30.47	0.9299	30.66	0.9258	30.71	0.9298	30.05	0.9043	32.26	0.9372	**32.55**	**0.9387**
Statue512	32.78	0.9067	33.61	0.9188	33.55	0.9149	33.55	0.9193	32.53	0.8806	34.88	0.9249	**34.95**	**0.9273**
Average	32.44	0.9097	33.52	0.9264	33.50	0.9229	33.57	0.9258	32.25	0.8945	33.91	0.9283	**34.41**	**0.9311**

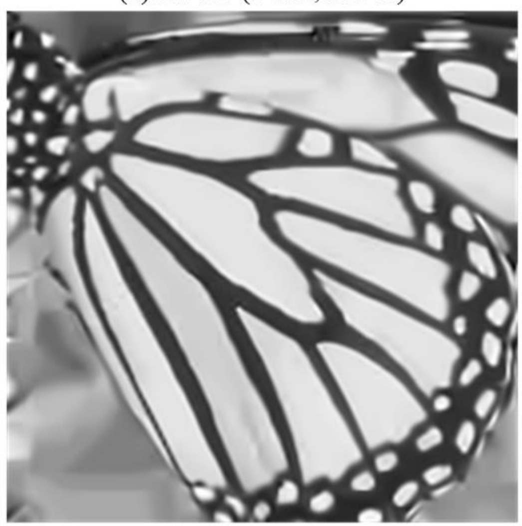
Subjective Quality Evaluation

（a) BM3D (23.91,0.8266)

(b) KSVD (24.55,0.8549)

(c) ANCE (24.34,0.8532)

(d) DicTV (23.42,0.8176)



(e) SSRQC (25.31,0.8764)

(f) Proposed (25.82,0.8861)

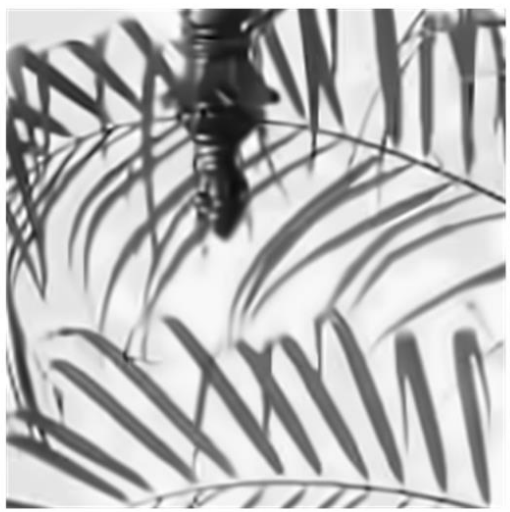
Subjective Quality Evaluation

(a) BM3D (23.78,0.8408)

(b) KSVD (24.39,0.8684)

(c) ANCE (24.18, 0.8551)

(d) DicTV (23.27,0.8245)



(e) SSRQC (25.01,0.8861)

(f) Proposed (25.57,0.8979)

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV
- Deep GLR
- Ongoing & Future Work

Reweighted Graph Total Variation

- TV on graphs.

pixel intensity difference

$$w_{i,j} = \exp\left(\frac{-\left\| x_i - x_j \right\|_2^2}{\sigma_1^2} \right)$$

Gradient of nodes on the graph:

$$(\nabla_i \mathbf{x})_j \triangleq x_j - x_i,$$

Conventional Graph TV:

$$\begin{aligned}\|\mathbf{x}\|_{GTV} &= \sum_{i \in \mathcal{V}} \| \operatorname{diag}(\mathbf{W}_{i,\cdot}) \nabla_i \mathbf{x} \|_1 \\ &= \sum_{i=1}^{N} \sum_{j=1}^{N} w_{i,j} |x_j - x_i|\end{aligned}$$

Reweighted Graph TV:

$$\begin{aligned}\|\mathbf{x}\|_{RGTV} &= \sum_{i \in \mathcal{V}} \| \operatorname{diag}(\mathbf{W}_{i,\cdot}(\mathbf{x})) \nabla_i \mathbf{x} \|_1 \\ &= \sum_{i=1}^{N} \sum_{j=1}^{N} w_{i,j}(x_i, x_j) |x_j - x_i|,\end{aligned}$$

[1] M. Hidane, O. Lezoray, and A. Elmoataz, “Nonlinear multilayered representation of graph-signals,” in *Journal of Mathematical Imaging and Vision*, February 2013, vol. 45, no.2, pp. 114–137.

[2] P. Berger, G. Hannak, and G. Matz, “Graph signal recovery via primal-dual algorithms for total variation minimization,” in *IEEE Journal on Selected Topics in Signal Processing*, September 2017, vol. 11, no.6, pp. 842–855.

Reweighted Graph Total Variation

- RGTV is separable. analyze each node pair.
  - Promotes *bi-modal inter-pixel differences*.

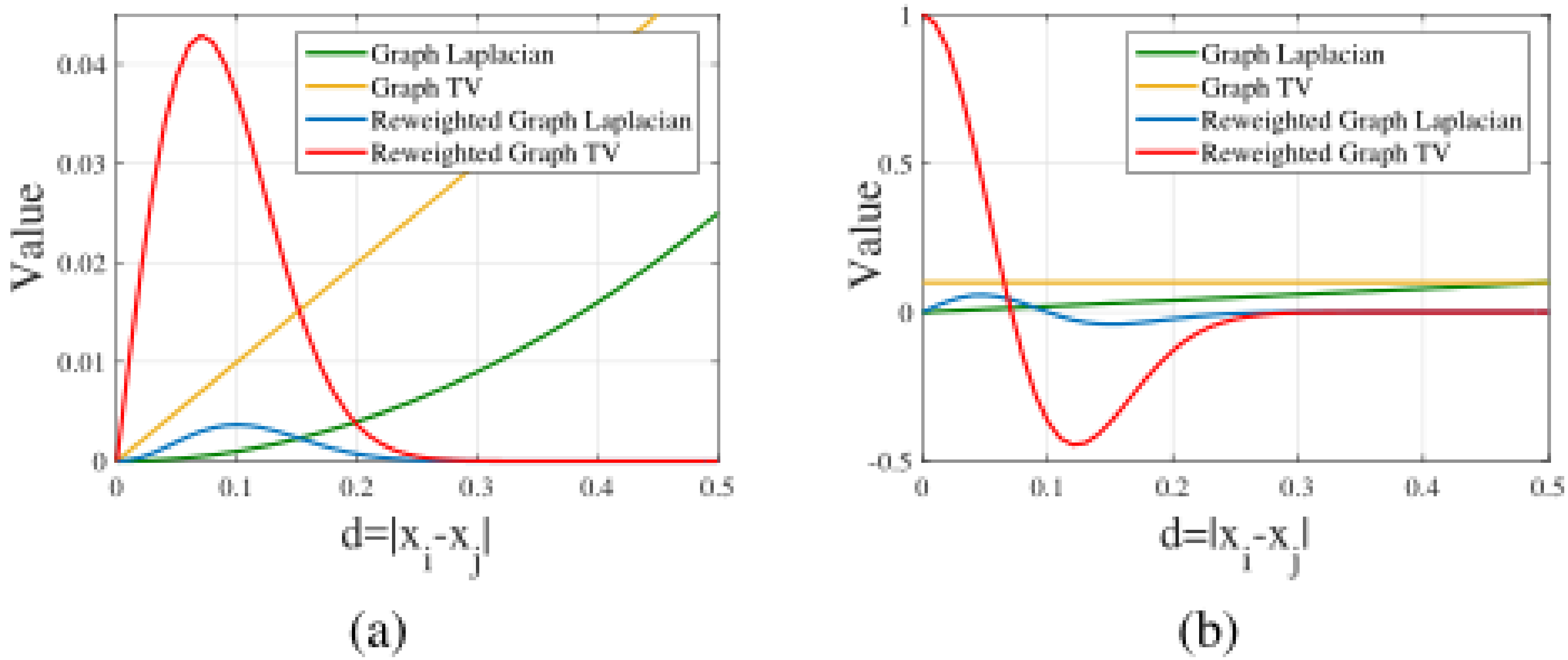


Fig. 3. Curves of regularizers and their corresponding first-derivatives for each (i, j) pair. d is normalized to $[0, 1]$. $w_{i,j} = 0.1$ for graph Laplacian and graph TV. $\sigma = 0.1$ for reweighted graph Laplacian and reweighted graph TV.

[1] Y. Bai, G. Cheung, X. Liu, W. Gao, "Blind Image Deblurring via Reweighted Graph Total Variation," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Alberta, April, 2018.

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT

- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV: Image Deblurring

- Deep GLR

- Ongoing & Future Work

Background for Image Deblurring

- Image blur is a common image degradation.
- Typically, blur process is modeled:

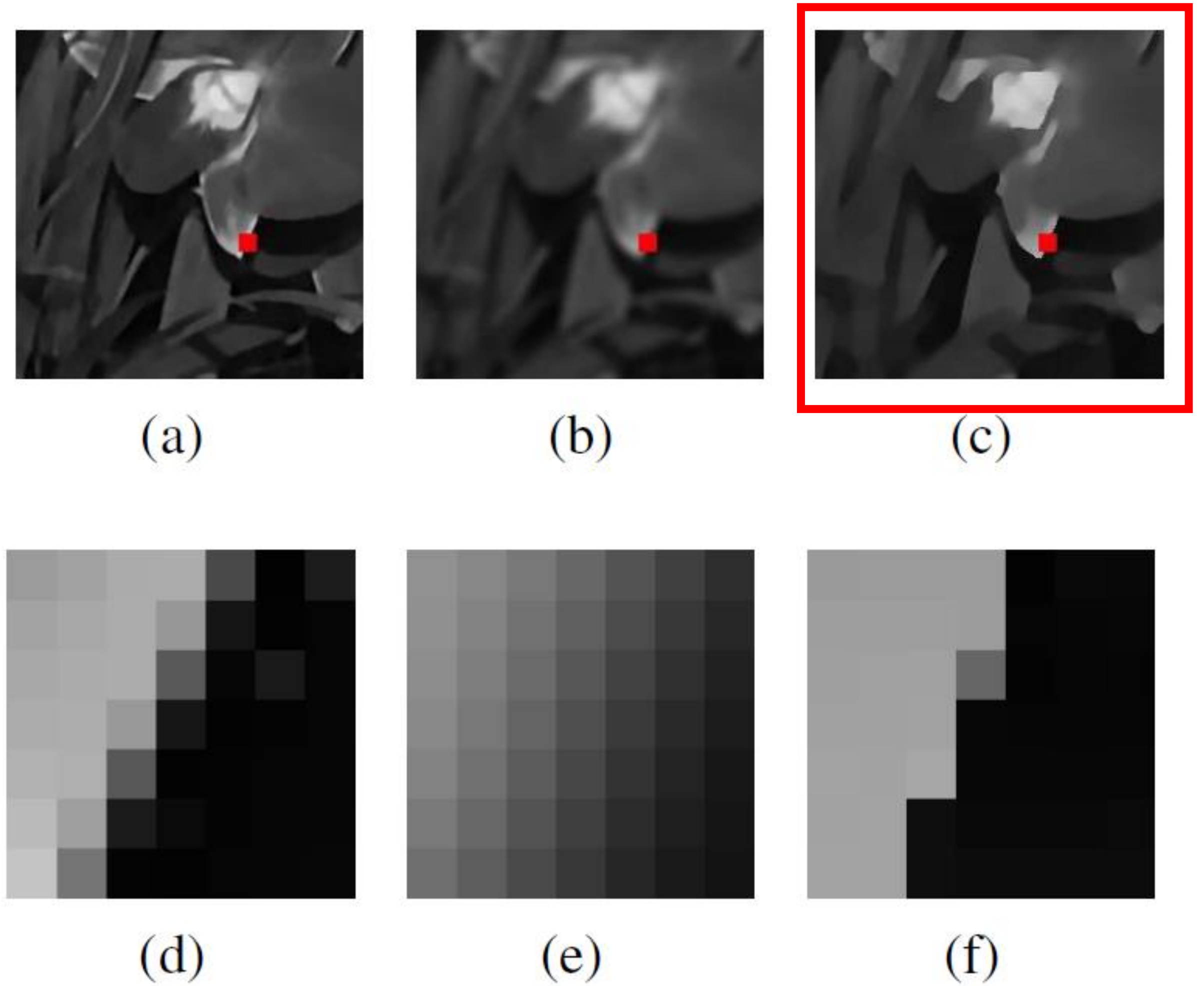
$$y = k \otimes x$$

where y is the blurry image, k is the blur kernel, x is the original sharp image.

- **Blind-image deblurring** focuses on estimating blur kernel k.
- Given k, problem becomes ***de-convolution***.

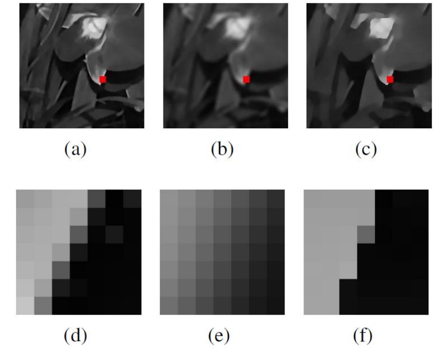
Observation

- ***Skeleton image***:
  - PWS image keeping only structural edges.
  - Proxy to estimate blur kernel k.

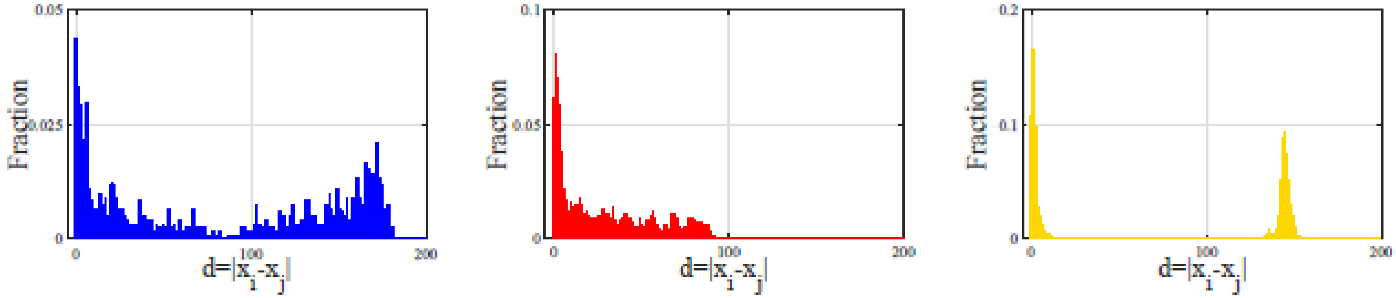


Observation

- Examine statistical properties of local patches:
  - Edge weight distribution of a fully connected graph.



$$[\mathbf{W}]_{i,j} = w_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2}\right),$$



(a) (b) (c)

Fig. 2: Graph weight distribution properties around edges. (a) a true natural patch. (b) a blurry patch. (c) a skeleton patch.

- Skeleton Image enjoys both *Sharpness* and *bi-modal Weight distribution*, thus useful to estimate blur kernel.

Key Idea

- Propose a *Reweighted Graph Total Variation* (RGTV) to promote a skeleton image patch.

Conventional Graph TV:

$$\|\mathbf{x}\|_{GTV} = \sum_{i \in \mathcal{V}} \| \operatorname{diag}(\mathbf{W}_{i,\cdot}) \nabla_i \mathbf{x} \|_1$$
$$= \sum_{i=1}^{N} \sum_{j=1}^{N} w_{i,j} |x_j - x_i|$$

Reweighted Graph TV:

$$\|\mathbf{x}\|_{RGTV} = \sum_{i \in \mathcal{V}} \| \operatorname{diag}(\mathbf{W}_{i,\cdot}(\mathbf{x})) \nabla_i \mathbf{x} \|_1$$
$$= \sum_{i=1}^{N} \sum_{j=1}^{N} w_{i,j}(x_i, x_j) |x_j - x_i|,$$

Our algorithm

- The optimization function can be written as follows,

$$\hat{\mathbf{x}}, \hat{\mathbf{k}} = \underset{\mathbf{x},\mathbf{k}}{\operatorname{argmin}}\, \varphi(\mathbf{x} \otimes \mathbf{k} - \mathbf{b}) + \mu_1 \cdot \theta_x(\mathbf{x}) + \mu_2 \cdot \theta_k\,(\mathbf{k})$$

- Assume L_2 norm for fidelity term $\varphi(\cdot)$.
- $\theta_x(\cdot) = RGTV(\cdot)$.
- $\theta_k(\cdot) = ||\cdot||_2$, assuming zero mean Gaussian distribution of k.

- RGTV is non-differentiable and non-convex.

Solution:

- Solve x and k alternatingly.
- For x, spectral interpretation of GTV, fast spectral filter.

Spectral domain

- Deduction for spectrum of GTV

$$
\begin{aligned}
\left(\partial \mathbf{x}^T \mathbf{L}\mathbf{x}\right)_i &= 2 \cdot (\mathbf{L}\mathbf{x})_i = c \cdot \sum_{j=1}^{N} w_{i,j} \cdot (x_i - x_j), \\
&= c \cdot \left(\sum_{j=1}^{N} w_{i,j} x_i - \sum_{j=1}^{N} w_{i,j} x_j\right)
\end{aligned}
$$

$$
\begin{aligned}
\left(\partial \|\mathbf{x}\|_{GTV}\right)_i &= c' \cdot \sum_{j=1}^{N} \gamma_{i,j} \cdot (x_i - x_j), \\
&= c' \cdot \left(\sum_{j=1}^{N} \gamma_{i,j} x_i - \sum_{j=1}^{N} \gamma_{i,j} x_j\right)
\end{aligned}
$$

$$
\boxed{\gamma_{i,j} = \frac{w_{i,j}}{\max\{|x_j - x_i|, \epsilon\}}}
$$

New weight function

Spectral domain

- Explanation: $\gamma_{i,j} = \dfrac{w_{i,j}}{\max\{|x_j - x_i|, \epsilon\}}$ ➡ New Adjacency matrix $\mathbf{\Gamma}$

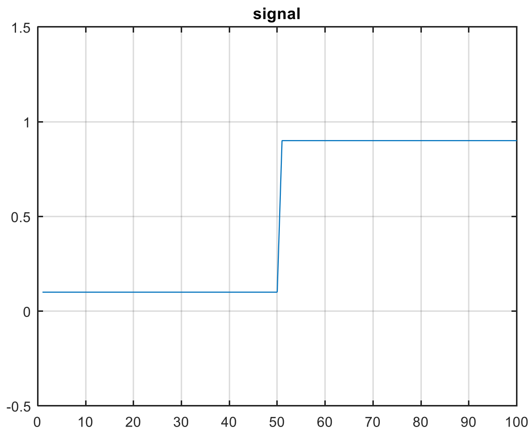
$$\|\mathbf{x}\|_{GTV} = \sum_{(i,j)\in E} \gamma_{i,j} \left(x_i - x_j\right)^2$$

$$= \mathbf{x}^T \mathbf{L}_\Gamma \, \mathbf{x}$$

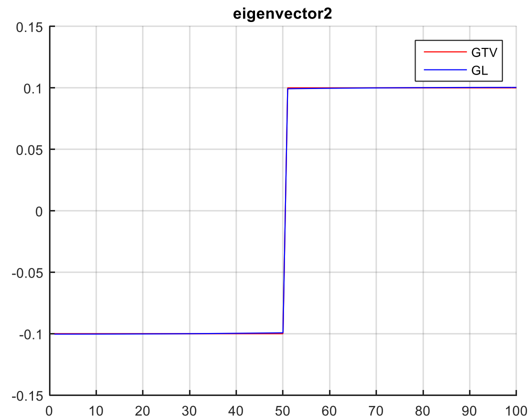
$$\mathbf{L}_\Gamma = \mathbf{U}_\Gamma \mathbf{\Lambda}_\Gamma \mathbf{U}_\Gamma^T$$ ← Graph L_1 spectrum

$$\mathbf{x}^* = \mathbf{U}_{\{W,\Gamma\}} \,\mathrm{diag}\left(\frac{1}{1+\mu\cdot\lambda_k^{\{W,\Gamma\}}}\right)\mathbf{U}_{\{W,\Gamma\}}^T\mathbf{y},$$

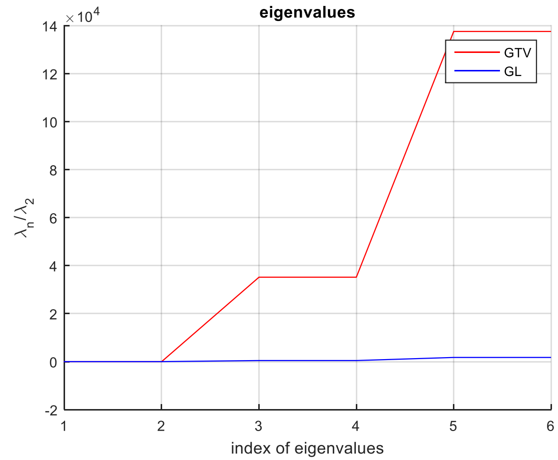
Spectral domain



Second eigenvector

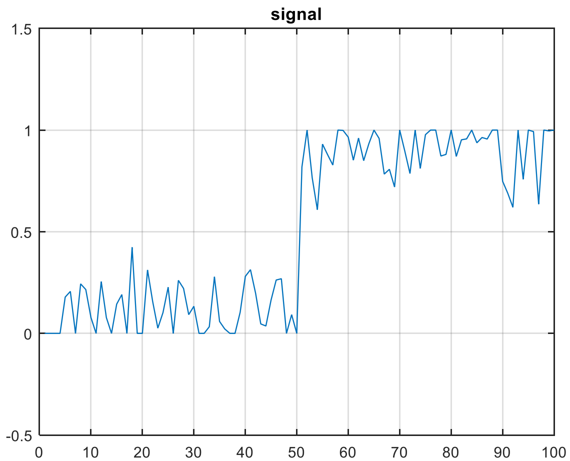


Relative eigenvalues $\frac{\lambda_k}{\lambda_2}$



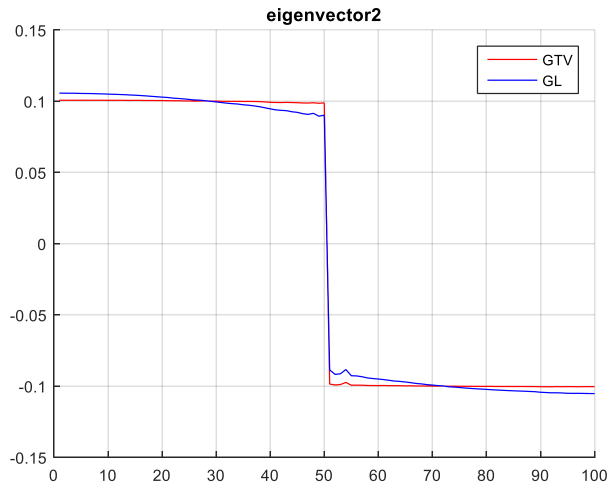
- We do 1D illustrative experiments to compute Graph Spectrum with conventional Graph Laplacian and GTV Laplacian.
- The parameter are the same for both. 4 neighbors and ¥sigma=0.3

Spectral domain

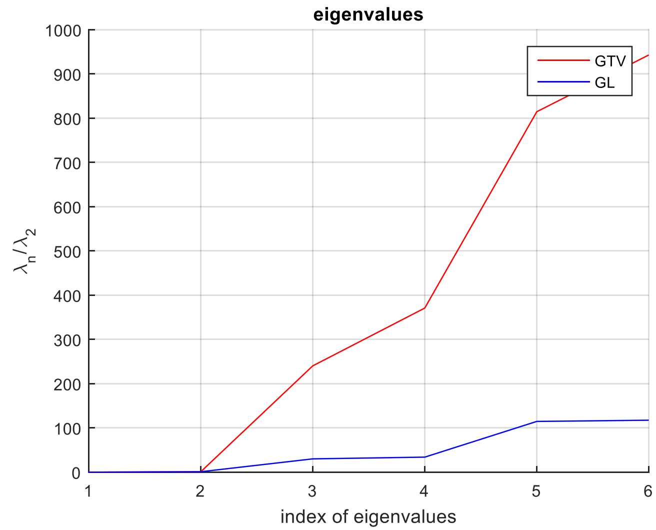


After adding noise

Second eigenvector



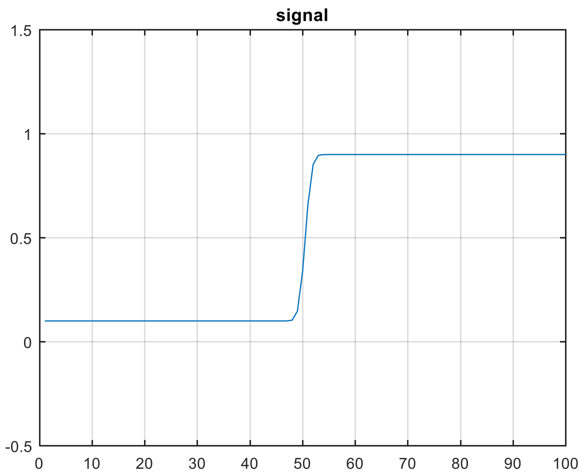
Relative eigenvalues $\frac{\lambda_k}{\lambda_2}$

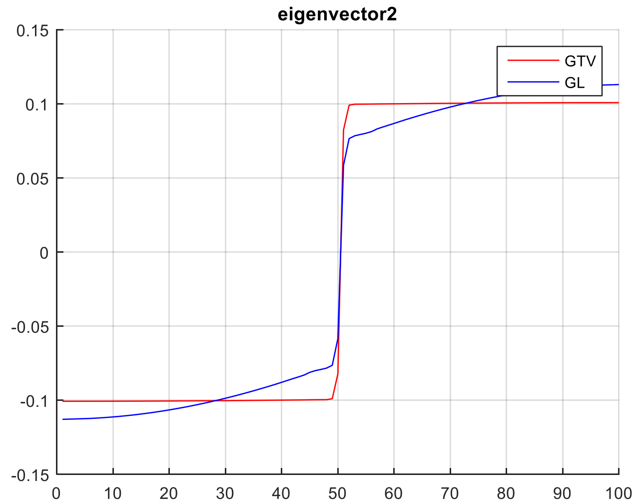


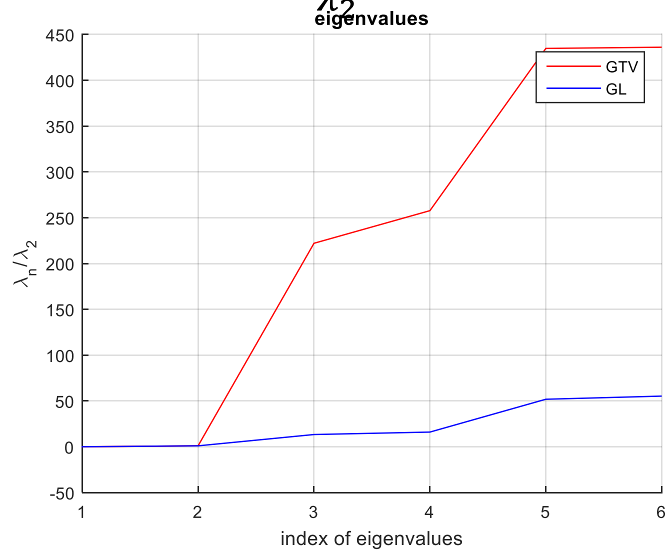
Spectral domain

Second eigenvector

Relative eigenvalues $\frac{\lambda_k}{\lambda_2}$







After Blurring

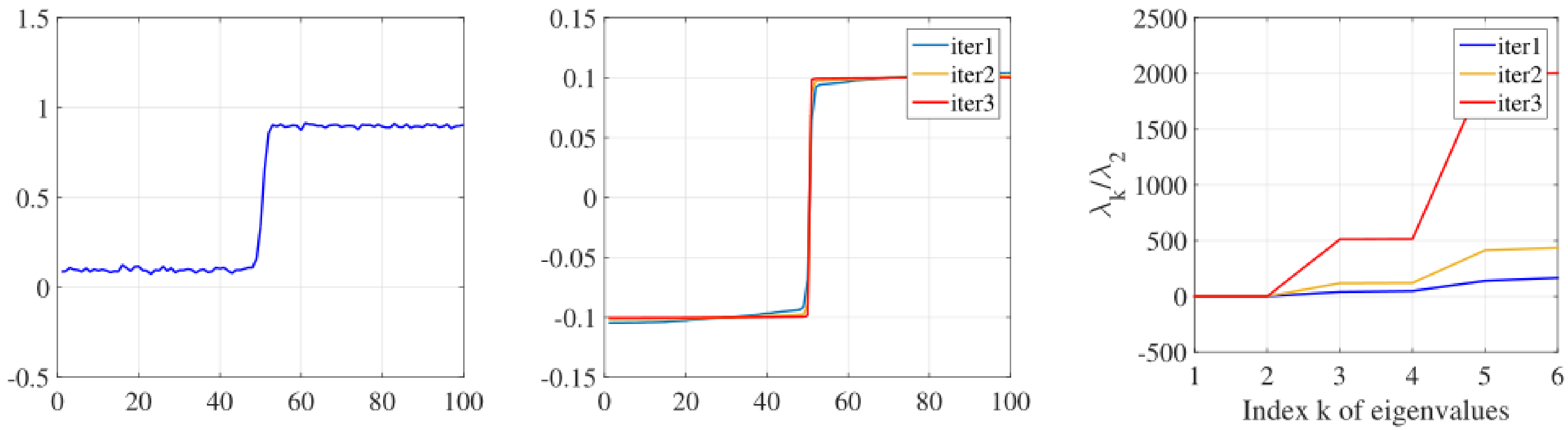
Spectral Domain

- Weights in RGTV is functions of graph signal. Therefore, there is no fixed graph spectrum for RGTV.
- Initialize weights for RGTV like GTV, and then update weights and spectrum. We analyze the gradual transformation of spectrum iteratively.

$$\mathbf{x}^{(n+1)} = \mathbf{U}_{\Gamma}^{(n)} \operatorname{diag}\left(\frac{1}{1+\mu \cdot \lambda_k^{\Gamma}(\mathbf{x}^{(n)})}\right) \mathbf{U}_{\Gamma}^{(n)^T} \mathbf{x}^{(n)}$$

$$\mathbf{U}_{\Gamma}^{(n)} = \mathbf{U}_{\Gamma}(\mathbf{x}^{(n)}).$$

Spectral domain



(a) (b) (c)

Fig. 5. Illustrative experiments of graph spectrum of RGTV on an 1D graph signal. (a) a PWS signal blurred by a Gaussian blur $\sigma_b = 1$ with Gaussian noise $\sigma_n = 0.0001$. (b) is the second eigenvectors of approximate RGTV in each iteration. (c) represents the curves of ratio λ_k/λ_2 with respect to k in each iteration. The graphs are constructed as a 4-neighbour adjacency matrix with weight parameter $\sigma = 0.3$.

Our algorithm

Alternating Iterative algorithm:

$$\begin{cases} \hat{\mathbf{x}} = \mathrm{argmin}_{\mathrm{x}} \ \frac{1}{2} \left\| \mathbf{x} \otimes \widehat{\boldsymbol{k}} - \mathbf{b} \right\|_2^2 + \beta \|\mathbf{x}\|_{RGTV} \\ \hat{\mathbf{k}} = \mathrm{argmin}_{\mathrm{k}} \|\nabla \hat{\mathbf{x}} \otimes \mathbf{k} - \nabla \mathbf{b}\|_2^2 + \mu \|\mathbf{k}\|_2^2 \end{cases} \Longrightarrow \quad \begin{aligned} &\hat{\mathbf{x}} = \mathrm{argmin}_{\mathrm{x}} \left\| \mathbf{x} \otimes \widehat{\boldsymbol{k}} - \nabla \mathbf{b} \right\|_2^2 + \beta \cdot \mathbf{x}^{\mathbf{T}} \mathbf{L}_{\Gamma} \mathbf{x} \\ &\left(\widehat{\mathbf{K}}^T \widehat{\mathbf{K}} + 2\beta \cdot \mathbf{L}_{\Gamma} \right) \hat{\mathbf{x}} = \widehat{\mathbf{K}}^{\mathbf{T}} \mathbf{b} \end{aligned}$$

System of linear equations.
Efficiently solved via conjugate gradient.

Extensions

- Accelerated algorithm for Gaussian blur:

$$\hat{\mathbf{x}} = \operatorname{argmin}_{\mathrm{x}} \frac{1}{2} \| \boxed{(\mathbf{I} + a \cdot \mathbf{L}_\Gamma)} \mathbf{x} - \mathbf{b} \|_2^2 + \beta \| \mathbf{x} \|_{RGTV}$$

$$\hat{\mathbf{x}} = \left(\frac{g(\mathbf{L}_\Gamma)}{g^2(\mathbf{L}_\Gamma) + 2\beta \cdot \mathbf{L}_\Gamma} \right) \mathbf{b}$$

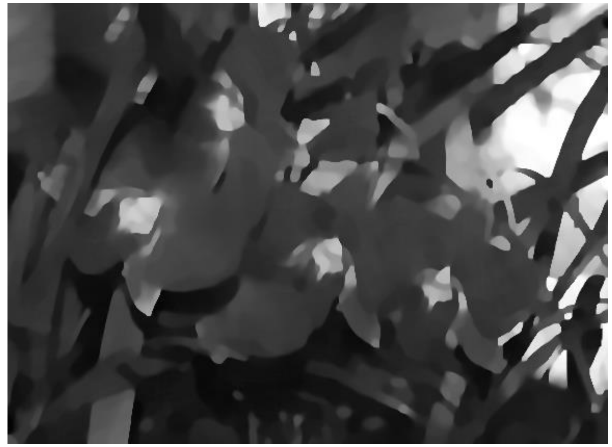
$$= \mathbf{U}_\Gamma \left(\frac{g(\mathbf{\Lambda}_\Gamma)}{g^2(\mathbf{\Lambda}_\Gamma) + 2\beta \cdot \mathbf{\Lambda}_\Gamma} \right) \mathbf{U}_\Gamma^T \mathbf{b}$$

Solve it via **accelerated graph filter** via *Lanczos method* [1]

[1] A. Susnjara, N. Perraudin, D. Kressner, and P. Vandergheynst, “Accelerated filtering on graphs using lanczos method,” arXiv preprint arXiv:1509.04537, 2015.

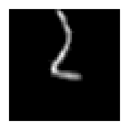
Workflow

Blurry Image



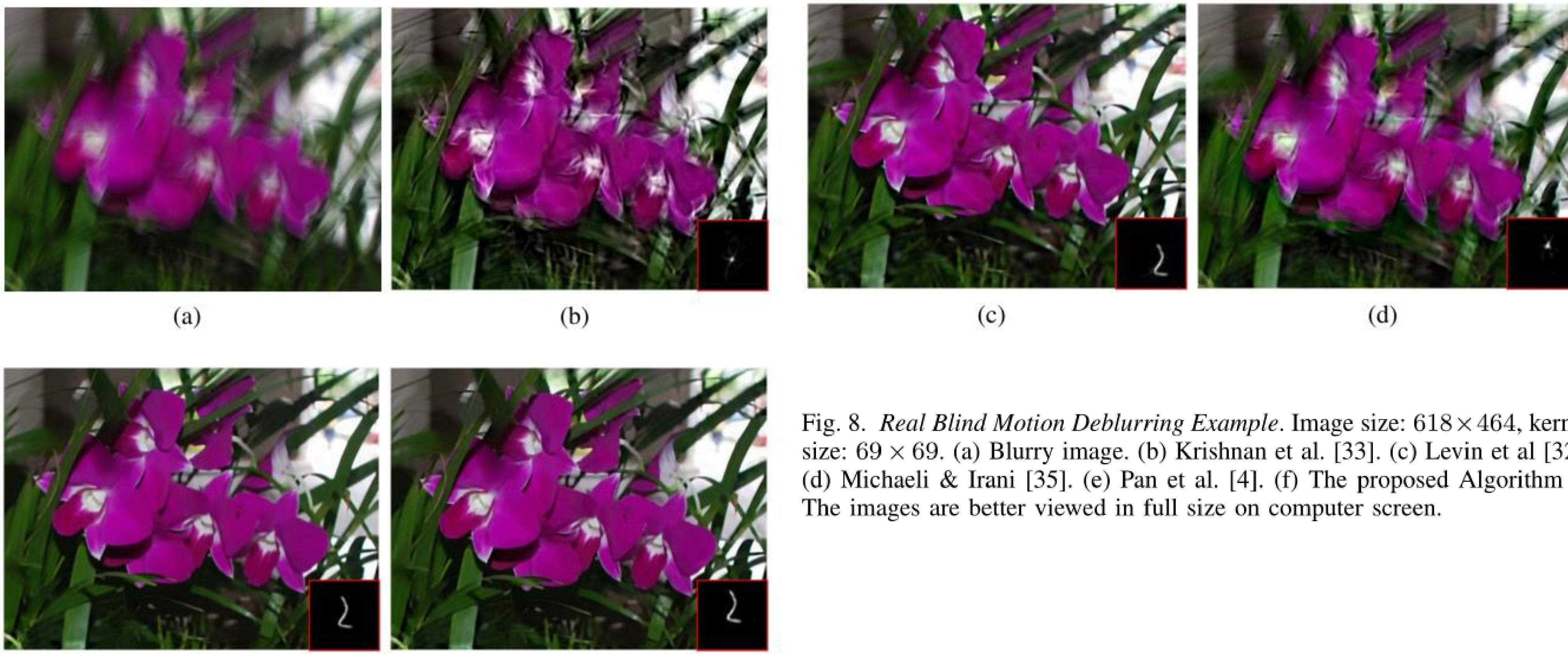
Skeleton Image Reconstruction

Reconstruction



Kernel Estimation

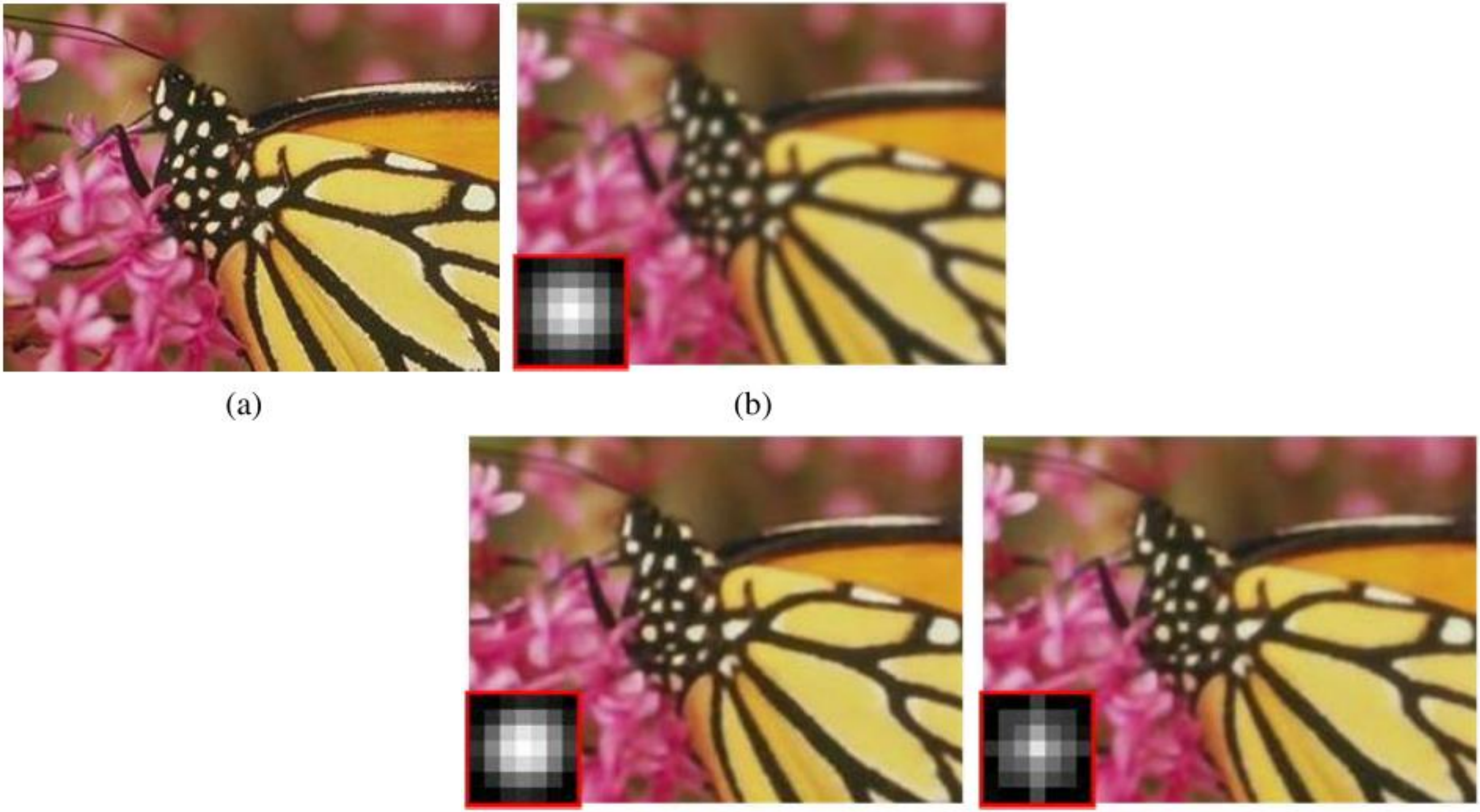
Experimental Results



(a) (b) (c) (d) (e) (f)

Fig. 8. *Real Blind Motion Deblurring Example*. Image size: 618×464, kernel size: 69×69. (a) Blurry image. (b) Krishnan et al. [33]. (c) Levin et al [32]. (d) Michaeli & Irani [35]. (e) Pan et al. [4]. (f) The proposed Algorithm 1. The images are better viewed in full size on computer screen.

Experimental Results



(a) (b)

(c) (d)

Fig. 11. *Gaussian Blur Blind Deblurring Example*. Image size: 768×512, kernel size: 7×7, $\sigma_b = 1.85$. (a) Ground-truth image. (b) Blurry image. (c) The proposed Algorithm 1. (d) The proposed Algorithm 3.

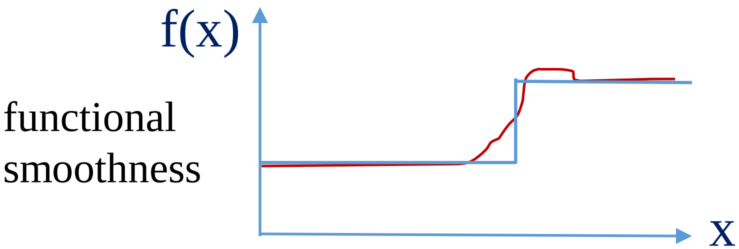
Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV: 3D Point Cloud Denoising
- Deep GLR
- Ongoing & Future Work

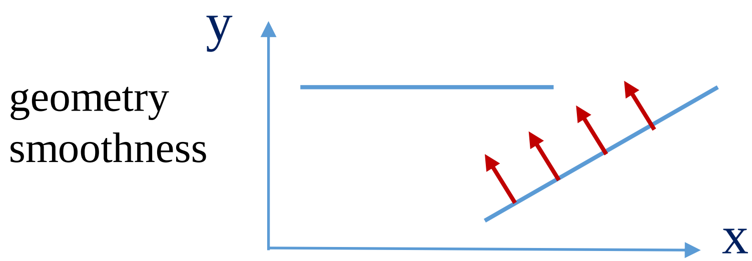
GTV for Point Cloud Denoising

- Acquisition of point cloud introduces noise.
- Point cloud is irregularly sampled 2D manifold in 3D space.
- Not appropriate to apply GTV directly on 3D coordinates [1].
    - only **a singular 3D point has zero GTV value**.

$$\sum_i \left| f(x_i) - f(x_{i-1}) \right|$$



$$\sum_i \left| y_i - y_{i-1} \right| \longrightarrow \sum_i \left\| \mathbf{n}_i - \mathbf{n}_{i-1} \right\|_1$$



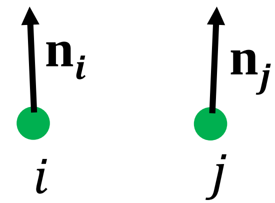
- **Proposal**: Apply GTV is to the surface normals of 3D point cloud—**a generalization of TV to 3D geometry.**

[1] Y. Schoenenberger, J. Paratte, and P. Vandergheynst, "**Graph-based denoising for time-varying point clouds**," in *IEEE 3DTV-Conference*, 2015, pp. 1–4

Algorithm Overview

- Use graph total variation (GTV) of surface normals over the K-NN graph:

$$||\mathbf{n}||_{\mathrm{GTV}} = \sum_{i,j \in \mathcal{E}} w_{i,j} ||\mathbf{n}_i - \mathbf{n}_j||_1$$



$$w_{i,j} = \exp\left(-\frac{||\mathbf{p}_i - \mathbf{p}_j||_2^2}{\sigma_p^2}\right)$$

- Denoising problem as l2-norm fidelity plus GTV of surface normals:

$$\min_{\mathrm{p},\mathrm{n}} \|\mathrm{q} - \mathrm{p}\|_2^2 + \gamma \sum_{i,j \in E} \|\mathrm{n}_i - \mathrm{n}_j\|_1$$

- Surface normal estimation of $\mathbf{n}_i$ is a nonlinear function of $\mathbf{p}_i$ and neighbors.

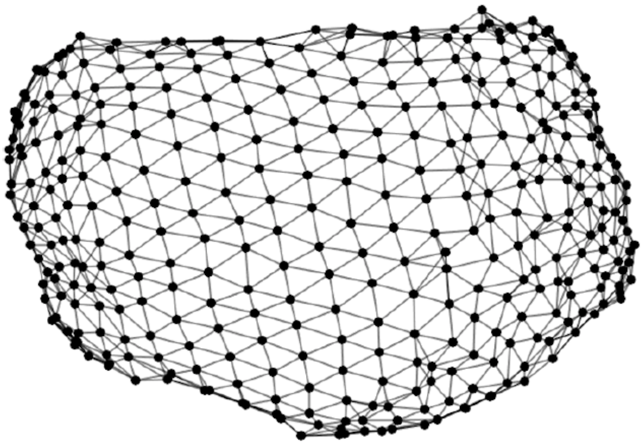
Proposal:

1. Partition point cloud into **two independent classes** (say red and blue).
2. When computing surface normal for a red node, use only neighboring blue points.
3. Solve convex optimization for red (blue) nodes alternately.

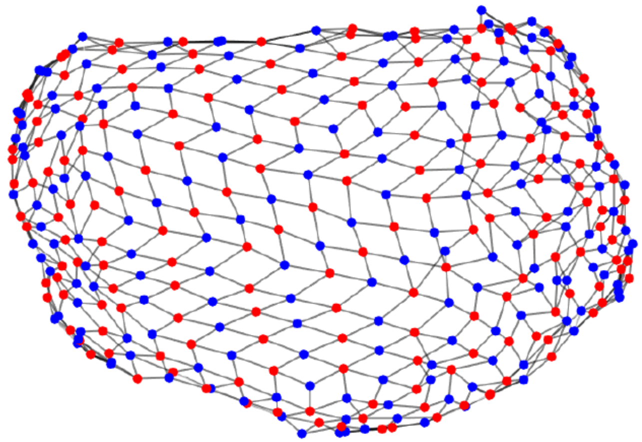
[1] C. Dinesh, G. Cheung, I. V. Bajic, C. Yang, "Fast 3D Point Cloud Denoising via Bipartite Graph Approximation & Total Variation," accepted to *IEEE 20th International Workshop on Multimedia Signal Processing*, Vancouver, Canada, August 2018.

Bipartite Graph Approx. & Normal Def’n

Step 1: bipartite graph approx. of k-NN graph.

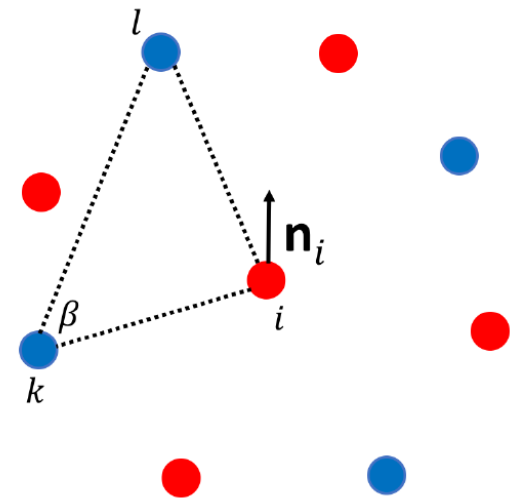


(a) original graph



(b) bipartite graph

Normal vector estimation at a red node



Step 2: define red nodes’ normals using blue nodes.

$$\mathbf{n}_i = \frac{[\mathbf{p}_i - \mathbf{p}_k] \times [\mathbf{p}_k - \mathbf{p}_l]}{||[\mathbf{p}_i - \mathbf{p}_k] \times [\mathbf{p}_k - \mathbf{p}_l]||_2}$$

$$\mathbf{n}_i = \mathbf{A}_i \mathbf{p}_i + \mathbf{b}_i$$

$\mathbf{A_i}$ is a constant matrix and $\mathbf{b_i}$ is a constant vector with respect to $\mathbf{p_i}$

[1] J. Zeng, G. Cheung, A. Ortega, "**Bipartite Approximation for Graph Wavelet Signal Decomposition**," *IEEE Transactions on Signal Processing*, vol.65, no.20, pp.5466-5480, October 2017.

Convex Optimization Formulation

- After computing normals for each red node, construct a new k-NN graph for red nodes only.
- For a red node graph, objective is a l2 -l1 -norm minimization w/ linear constraints:

$$\min_{\mathrm{p},\mathrm{m}} \|\mathrm{q}-\mathrm{p}\|_2^2 + \gamma \sum_{i,j \in E_r} \|\mathrm{m}_{i,j}\|_1$$

$$\left.\begin{aligned} \mathrm{m}_{i,j} &= \mathrm{n}_i - \mathrm{n}_j \\ \mathrm{n}_i &= \mathrm{A}_i \mathrm{p}_i + \mathrm{b}_i \end{aligned}\right\} \rightarrow \mathrm{m} = \mathrm{Bp} + \mathrm{v}$$

Solution:

- ADMM: $\min_{\mathrm{p},\mathrm{m}} \|\mathrm{q}-\mathrm{p}\|_2^2 + \gamma \sum_{i,j \in E_r} \|\mathrm{m}_{i,j}\|_1 + \frac{\rho}{2}\|\mathrm{Bp}+\mathrm{v}-\mathrm{m}+\mathrm{u}\|_2^2 + \mathrm{const}$

- p-minimization: $(2\mathbf{I} + \rho \mathbf{B}^T \mathbf{B})\mathbf{p}^{k+1} = 2\mathbf{q} + \rho \mathbf{B}^T(\mathbf{m}^k - \mathbf{u}^k - \mathbf{v}),$
- m-minimization: $\min_{\mathbf{m}} \frac{\rho}{2}\|\mathbf{B}\mathbf{p}^{k+1} + \mathbf{v} - \mathbf{m} + \mathbf{u}^k\|_2^2 + \gamma \sum_{i,j \in \mathcal{E}_1} w_{i,j}\|\mathbf{m}_{i,j}\|_1,$

Proximal gradient descent

- Alternately update red and blue graphs until convergence.

Experimental Setup

- 4 competing local methods: **APSS** [1], **RIMLS** [2], **AWLOP** [3], **MRPCA** [4]
- 7 pint cloud datasets used: Bunny, Gargoyle, DC, Daratrch, Anchor, Lordquas, Fandisk, Laurana
- Metrics: **point to point error** (C2C) and **point to plane error** (C2P)
- Gaussian noise with zero mean, standard deviation σ of 0.1 and 0.3.

[1] G. Guennebaud and M. Gross, "**Algebraic point set surfaces**," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, p. 23, 2007.

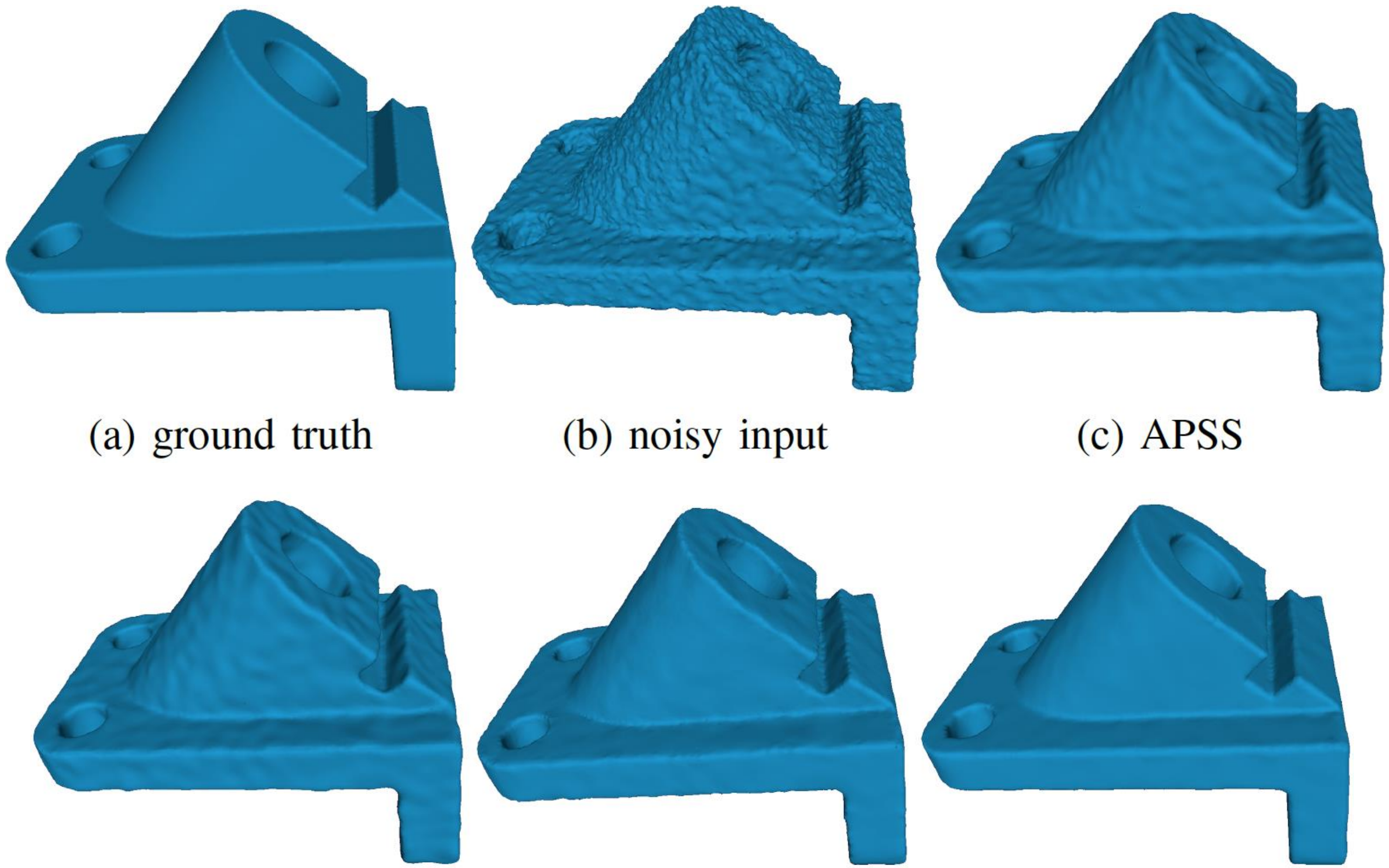
[2] A. C. Oztireli, G. Guennebaud, and M. Gross, "**Feature preserving point set surfaces based on non-linear kernel regression**," in *Computer Graphics Forum*, vol. 28, no. 2, 2009, pp. 493–501.

[3]H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. R. Zhang, "**Edge-aware point set resampling**," *ACM Transactions on Graphics*, vol. 32, no. 1, p. 9, 2013.

[4] E. Mattei and A. Castrodad, "**Point cloud denoising via moving RPCA**," in *Computer Graphics Forum*, vol. 36, no. 8, 2017, pp. 123–137.

Experimental Results – Visual Comparison

Anchor model (σ=0.3)

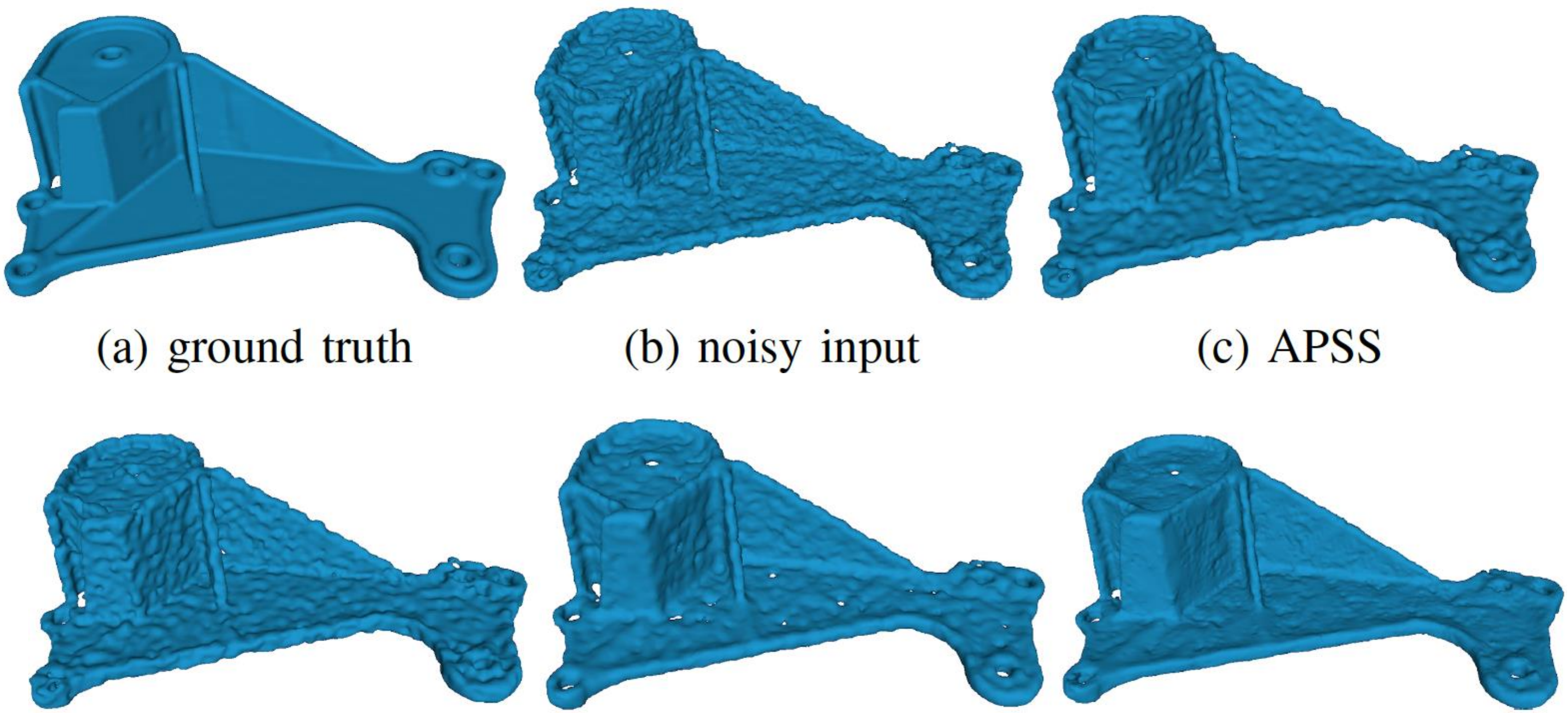


(a) ground truth (b) noisy input (c) APSS

(d) RIMLS (e) MRPCA (f) proposed

Experimental Results – Visual Comparison

Daratech model (σ=0.3)



(a) ground truth (b) noisy input (c) APSS

(d) RIMLS (e) MRPCA (f) proposed

Experimental Results – Numerical Comparison

TABLE I
C2C OF DIFFERENT MODELS, WITH GAUSSIAN NOISE ($\sigma = 0.1$)

Model	Noise	APSS	RIMLS	AWLOP	MRPCA	Prop.
Bunny	0.157	0.135	0.143	0.153	0.141	**0.128**
Gargoyle	0.154	0.133	0.143	0.151	0.144	**0.131**
DC	0.154	0.130	0.140	0.148	0.136	**0.128**
Daratech	0.156	0.134	0.137	0.156	0.134	**0.132**
Anchor	0.156	0.134	0.139	0.152	0.130	**0.127**
Lordquas	0.155	0.130	0.143	0.153	0.132	**0.126**
Fandisk	0.159	0.148	0.148	0.157	0.138	**0.136**
Laurana	0.150	0.136	0.139	0.147	**0.130**	**0.130**

TABLE II
C2C OF DIFFERENT MODELS, WITH GAUSSIAN NOISE ($\sigma = 0.3$)

Model	Noise	APSS	RIMLS	AWLOP	MRPCA	Prop.
Bunny	0.329	0.235	0.251	0.315	0.243	**0.231**
Gargoyle	0.304	0.220	0.232	0.288	0.218	**0.214**
DC	0.305	0.213	0.230	0.302	0.212	**0.207**
Daratech	0.313	0.264	0.268	0.293	0.262	**0.246**
Anchor	0.317	0.225	0.231	0.281	0.216	**0.210**
Lordquas	0.307	0.212	0.228	0.284	0.208	**0.203**
Fandisk	0.406	0.352	0.343	0.390	0.331	**0.319**
Laurana	0.318	0.239	0.249	0.266	0.242	**0.231**

TABLE III
C2P ($\times 10^{-3}$) OF DIFFERENT MODELS, WITH GAUSSIAN NOISE ($\sigma = 0.1$)

Model	Noise	APSS	RIMLS	AWLOP	MRPCA	Prop.
Bunny	9.91	4.62	5.14	7.95	4.66	**4.61**
Gargoyle	9.67	4.59	5.97	8.46	4.56	**4.48**
DC	9.66	4.37	4.82	7.63	3.98	**3.71**
Daratech	9.93	2.93	4.05	9.54	3.01	**2.85**
Anchor	9.87	3.32	4.10	8.43	2.18	**2.05**
Lordquas	9.72	3.33	5.81	9.10	3.79	**3.11**
Fandisk	9.88	6.70	7.05	8.93	4.86	**4.39**
Laurana	9.23	5.16	5.86	7.70	5.13	**5.01**

TABLE IV
C2P ($\times 10^{-2}$) OF DIFFERENT MODELS, WITH GAUSSIAN NOISE ($\sigma = 0.3$)

Model	Noise	APSS	RIMLS	AWLOP	MRPCA	Prop.
Bunny	6.442	1.256	1.704	5.634	1.373	**1.128**
Gargoyle	6.096	1.512	1.954	5.004	1.540	**1.499**
DC	6.130	1.349	1.738	6.097	1.391	**1.201**
Daratech	6.116	3.422	3.483	4.881	3.212	**2.215**
Anchor	6.354	1.930	2.160	3.991	1.714	**1.597**
Lordquas	6.234	1.846	2.558	4.928	1.768	**1.644**
Fandisk	7.297	3.180	2.640	6.093	1.720	**1.702**
Laurana	5.890	1.392	1.800	2.307	1.464	**1.211**

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT
- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV
- Deep GLR
- Ongoing & Future Work

Unrolling Graph Laplacian Regularizer

- Recall MAP formulation of denoising problem with quadratic graph Laplacian regularizer:

$$\min_{x} \|\mathrm{y} - \mathrm{x}\|_2^2 + \mu \, \mathrm{x}^T \mathrm{L} \mathrm{x}$$

fidelity term

smoothness prior

- Solution is system of linear equations:

$$\left(\mathrm{I} + \mu \mathrm{L}\right) \mathrm{x}^* = \mathrm{y}$$

linear system of eqn's w/ sparse, symmetric PD matrix

Bilateral weights:

$$w_{i,j} = \exp\left(\frac{-\|x_i - x_j\|_2^2}{\sigma_1^2} \right) \exp\left(\frac{-\|l_i - l_j\|_2^2}{\sigma_2^2} \right)$$

Q: what is the “most appropriate” graph?

[1] J. Pang, G. Cheung, "**Graph Laplacian Regularization for Inverse Imaging: Analysis in the Continuous Domain**," *IEEE Transactions on Image Processing*, vol. 26, no.4, pp.1770-1785, April 2017.

Unrolling Graph Laplacian Regularizer

- **Deep Graph Laplacian Regularization:**
  1. Learn features **f**'s using CNN.
  2. Compute distance from features.
  3. Compute edge weights using Gaussian kernel.
  4. Construct graph, solve QP.

$$w_{ij} = \exp\left(-\frac{\mathrm{dist}(i,j)}{2\epsilon^2}\right),$$

$$\mathrm{dist}(i,j) = \sum_{n=1}^{N} \left(\mathbf{f}_n(i) - \mathbf{f}_n(j)\right)^2.$$

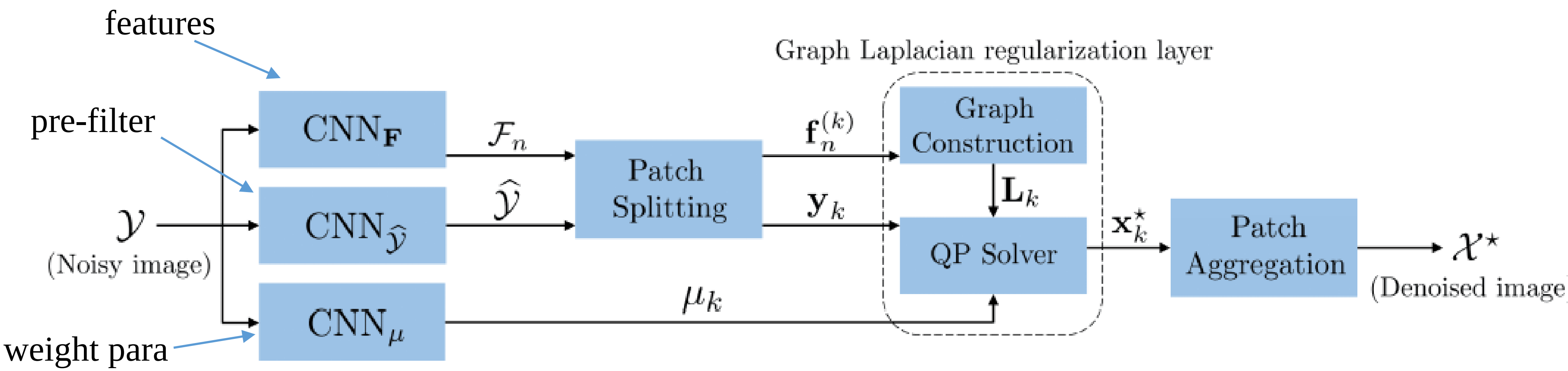


Fig. 1. Block diagram of the proposed GLRNet which employs a graph Laplacian regularization layer for image denoising.

[1] M. McCann et al., "**Convolutional Neural Networks for Inverse Problems in Imaging**," *IEEE SPM*, Nov. 2017.

[2] K. Gregor and Y. LeCun, "**Learning fast approximations of sparse coding,**" in Proc. 27th Int. Conf. Machine Learning, 2010..

Unrolling Graph Laplacian Regularizer

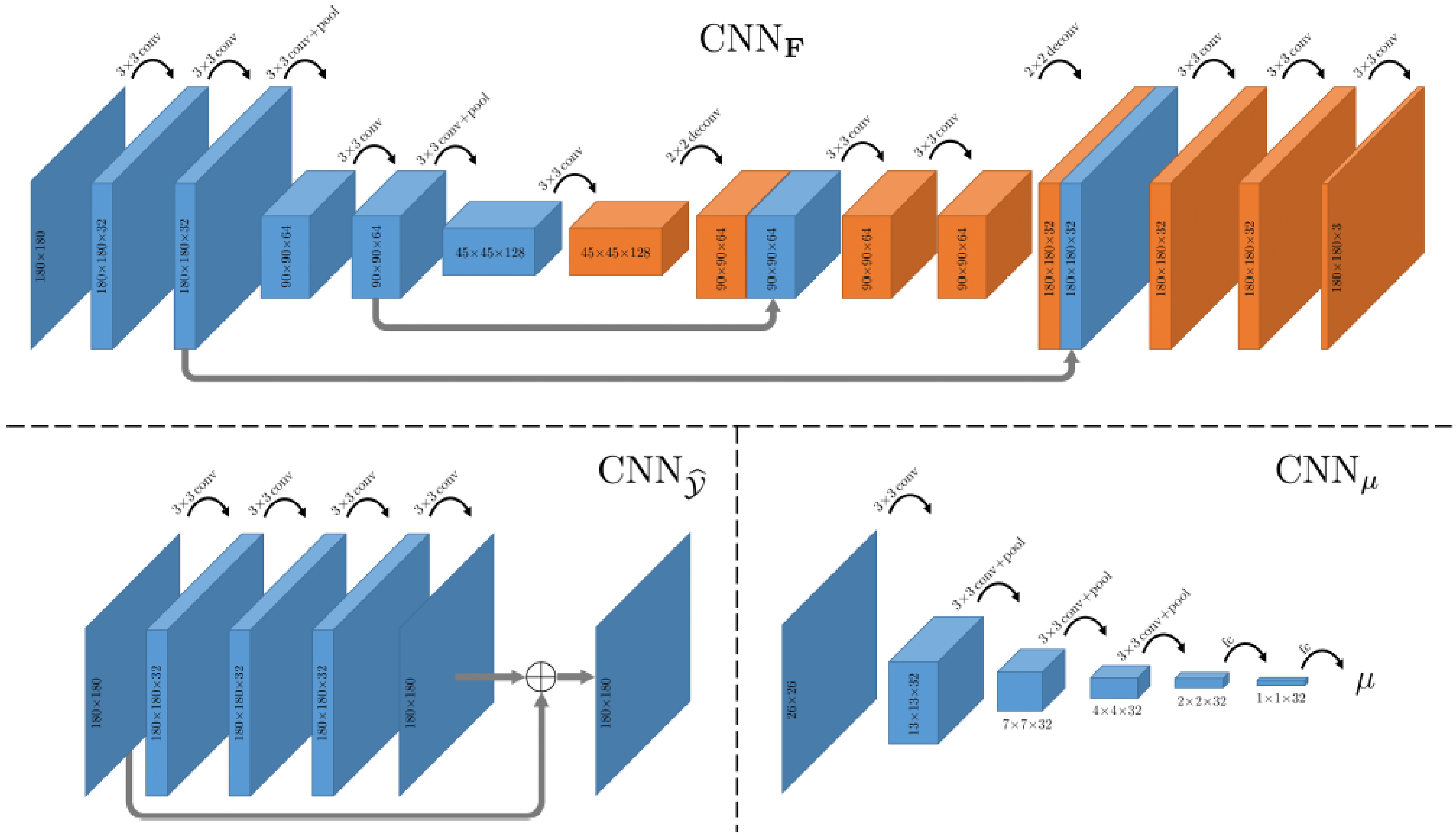


Fig. 3. Network architectures of CNN$_{\mathbf{F}}$, CNN$_{\widehat{\mathcal{Y}}}$ and CNN$_{\mu}$ in the experiments. Data produced by the decoder of CNN$_{\mathbf{F}}$ is colored in orange.

[1] J. Pang et al., "**Deep Graph Laplacian Regularization**," submitted to *ECCV*, Feb 2018.

Unrolling Graph Laplacian Regularizer

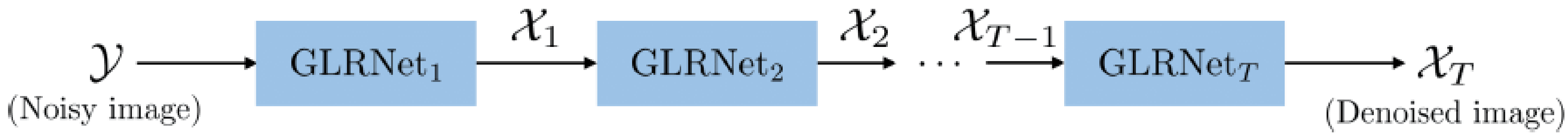


Fig. 2. Block diagram of the overall DeepGLR framework.

- **Graph Model** *guarantees numerical stability of solution*:

$$\left(\mathrm{I} + \mu \mathrm{L}\right) \mathrm{x}^* = \mathrm{y}$$

- **Thm 1**: condition number κ of matrix satisfies [1]:

$$\kappa \leq 1 + 2\,\mu\, d_{\max},$$

maximum node degree

- **Observation**: By restricting search space of CNN to degree-bounded graphs, we achieve robust learning.

[1] J. Pang et al., "**Deep Graph Laplacian Regularization**," submitted to *ECCV*, Feb 2018.

Experimental Results – Numerical Comparison

- Trained on AWGN on 5 images, patches of size 26-by-26.
- Batch size is 4, model is trained for 200 epochs.
- Trained for both known and blind noise variance.

Table 1. Average PSNR (dB) and SSIM values of different methods for Gaussian noise removal. The best results for each metric is highlighted in boldface.

Noise	Metric	Method						
		BM3D	WNNM	OGLR	DnCNN-S	DnCNN-B	DeepGLR-S	DeepGLR-B
25	PSNR	29.95	30.28	29.78	**30.41**	30.33	30.26	30.21
	SSIM	0.8496	0.8554	0.8463	**0.8609**	0.8594	0.8599	0.8557
40	PSNR	27.62	28.08	27.68	28.10	28.13	**28.16**	28.04
	SSIM	0.7920	0.8018	0.7949	0.8080	0.8091	**0.8125**	0.8063
50	PSNR	26.69	27.08	26.58	27.15	27.18	**27.25**	27.12
	SSIM	0.7651	0.7769	0.7539	0.7809	0.7811	**0.7852**	0.7807

[1] Kai Zhang et al, "Beyond a GCNNaussian denoiser: Residual learning of deep for image denoising," *TIP* 2017.
[2] Marc Lebrun et al, "The noise clinic: a blind image denoising algorithm," *IPOL* 2015.

Experimental Results – Numerical Comparison

- DeepGLR has average PSNR of 0.34 dB higher than CDnCNN [1].
- Model-based provides robustness against overfitting.

Table 2. Evaluation of different methods for low-light image denoising. The best results for each metric, except for those tested on the training set, are highlighted in boldface.

Metric	Noisy	Method					
		CBM3D	MC-WNNM	CDnCNN (train)	CDnCNN	CDeepGLR (train)	CDeepGLR
PSNR	20.36	26.08	26.23	33.43	31.26	32.31	**31.60**
SSIM Y	0.5198	0.8698	0.8531	0.9138	0.8978	0.9013	**0.9028**
SSIM R	0.2270	0.6293	0.5746	0.8538	0.8218	0.8372	**0.8297**
SSIM G	0.4073	0.8252	0.7566	0.8979	0.8828	0.8840	**0.8854**
SSIM B	0.1823	0.5633	0.5570	0.8294	0.7812	0.8138	**0.7997**

[1] Kai Zhang et al, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *TIP* 2017.
[2] Marc Lebrun et al, "The noise clinic: a blind image denoising algorithm," *IPOL* 2015.

Experimental Results – Visual Comparison

- trained on Gaussian noise, tested on low-light images in (RENOIR).
- Competing methods: DnCNN [1], noise clinic [2].
- outperformed DnCNN by 5.52 dB, and noise clinic by 1.87 dB.

DnCNN

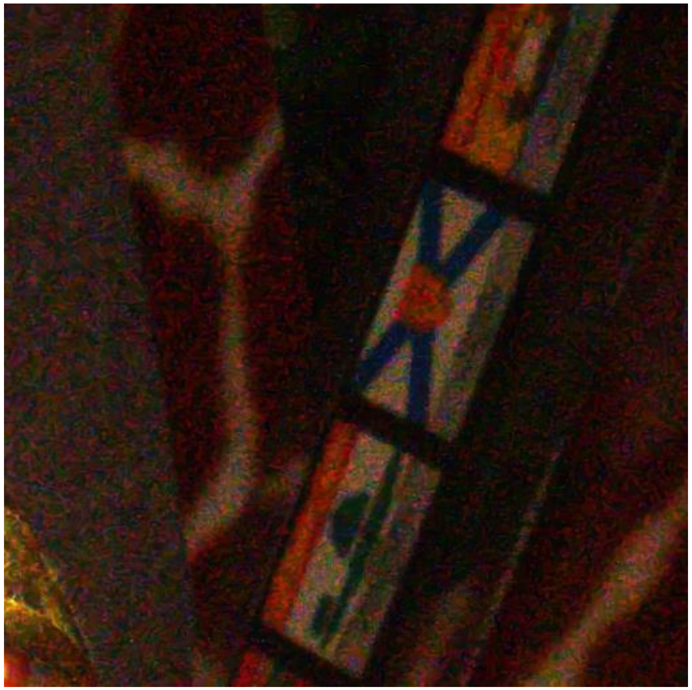
clinic

DeepGLR

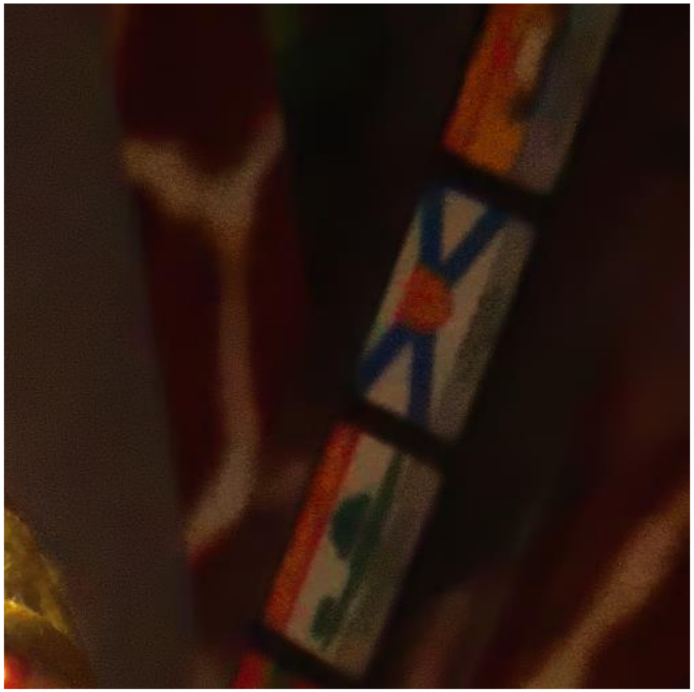
[1] Kai Zhang et al, “Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising,” *TIP* 2017.
[2] Marc Lebrun et al, “The noise clinic: a blind image denoising algorithm,” *IPOL* 2015.

Experimental Results – Visual Comparison

- trained on Gaussian noise, tested on low-light images in (RENOIR).
- Competing methods: DnCNN [1], noise clinic [2].
- outperformed DnCNN by 5.52 dB, and noise clinic by 1.87 dB.



DnCNN



clinic

DeepGLR

[1] Kai Zhang et al, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *TIP* 2017.
[2] Marc Lebrun et al, "The noise clinic: a blind image denoising algorithm," *IPOL* 2015.

Experimental Results – Visual Comparison

- trained on Gaussian noise, tested on low-light images in (RENOIR).
- Competing methods: DnCNN [1], noise clinic [2].
- outperformed DnCNN by 5.52 dB, and noise clinic by 1.87 dB.

DnCNN

clinic

DeepGLR

[1] Kai Zhang et al, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *TIP* 2017.
[2] Marc Lebrun et al, "The noise clinic: a blind image denoising algorithm," *IPOL* 2015.

Outline

- GSP for Image Compression
  - Optimality of GFT
  - Generalized GFT
  - Signed GFT

- GSP for Inverse Imaging
  - Graph Laplacian Regularizer
  - Reweighted Graph TV: 3D Point Cloud Denoising

- Deep GLR

- Ongoing & Future Work

Summary

- Variants of GFTs for optimal decorrelation
  - GFT, GGFT, SGFT
  - Selection of statistical model vs. encoding cost of side information
- GSP for Inverse Imaging
  - PWS-promoting Graph Laplacian Regularizer, RGTV
  - Spectral interpretation of GTV, RGTV
- Graph-based model restricts search space of DNN.
  - Robustness against overfitting.

Ongoing & Future Work

- Unrolling of graph-based convex optimization.
  - Unrolling of ADMM, proximal gradient with GTV prior, convex set constraints.
  - Learn (sparse) connectivity, edge weights.
  - Learn features from RGBD images for depth inpainting / denoising.

- Metric learning for edge weight computation for graph-based binary classifiers.

- Model-guided learning safeguard against worst-case / adversary noise?

Q&A

- Email: gene.cheung@ieee.org
- Homepage: http://research.nii.ac.jp/~cheung/