FAST HIGH-DIMENSIONAL FILTERING USING CLUSTERING

Pravin Nair and Kunal N. Chaudhury

Indian Institute of Science

Proceedings of ICIP, 2017

Outline

- High-dimensional filtering
- Fast approximation.
- Results.
- MATLAB Demo.
- Conclusion.

High-dimensional filtering

Definition

- Given high-dimensional images $\boldsymbol{f} : \Omega \to \mathbb{R}^n$ and $\boldsymbol{p} : \Omega \to \mathbb{R}^\rho$, where $\Omega \subset \mathbb{Z}^d$ is the domain.

- Filter output $\boldsymbol{g} : \Omega \to \mathbb{R}^n$:

$$\boldsymbol{g}(i) = \frac{1}{\eta(i)} \sum_{j \in W} \omega(j)\ \phi(\boldsymbol{p}(i-j) - \boldsymbol{p}(i))\ \boldsymbol{f}(i-j), \qquad (1)$$

$$\eta(i) = \sum_{j \in W} \omega(j)\ \phi(\boldsymbol{p}(i-j) - \boldsymbol{p}(i)). \qquad (2)$$

- $\omega : \mathbb{R}^d \to \mathbb{R}$ and $\phi : \mathbb{R}^\rho \to \mathbb{R}$ are kernels.

Approximation Geometry Application1 Application2 Application3

Examples

- Grayscale images ($d = 2, n = 1$).
- Color images ($d = 2, n = 3$).
- Color videos ($d = 3, n = 3$).
- Flow fields ($d = 3, n = 3$).
- Hyperspectral images ($d = 2$ and n can be of the order of tens or hundreds).
- Patch-based representations ($d = 2$ and n is patch size).

Applications

- Edge preserving smoothing [Tomasi and Maduchi 1998].
- Tone mapping [Durand and Dorsey 2002].
- Denoising [Buades et al. 2005].
- Detail manipulation [Bae et al. 2006; Fattal et al. 2007].
- Hyperspectral image filtering [Peng et al. 2009].
- Flow field denoising [Westenberg et al. 2004].

Taking out nonlinearity

- Given high-dimensional images $\boldsymbol{f} : \Omega \to \mathbb{R}^n$ and $\boldsymbol{p} : \Omega \to \mathbb{R}^\rho$, where $\Omega \subset \mathbb{Z}^d$ is the domain.

- Filter output $\boldsymbol{g} : \Omega \to \mathbb{R}^n$:

$$\boldsymbol{g}(i) = \frac{1}{\eta(i)} \sum_{j \in W} \omega(j)\ \phi(\boldsymbol{p}(i-j) - \boldsymbol{p}(i))\ \boldsymbol{f}(i-j),$$

$$\eta(i) = \sum_{j \in W} \omega(j)\ \phi(\boldsymbol{p}(i-j) - \boldsymbol{p}(i)).$$

- $\omega : \mathbb{R}^d \to \mathbb{R}$ and $\phi : \mathbb{R}^\rho \to \mathbb{R}$ are kernels.

Taking out nonlinearity

- Given high-dimensional images $\boldsymbol{f} : \Omega \to \mathbb{R}^n$ and $\boldsymbol{p} : \Omega \to \mathbb{R}^\rho$, where $\Omega \subset \mathbb{Z}^d$ is the domain.

- Filter output $\boldsymbol{g} : \Omega \to \mathbb{R}^n$:

$$\boldsymbol{g}(i) = \frac{1}{\eta(i)} \sum_{j \in W} \omega(j)\ \phi(\boldsymbol{p}(i-j) - \boldsymbol{p}(i))\ \boldsymbol{f}(i-j),$$

$$\eta(i) = \sum_{j \in W} \omega(j)\ \phi(\boldsymbol{p}(i-j) - \boldsymbol{p}(i)).$$

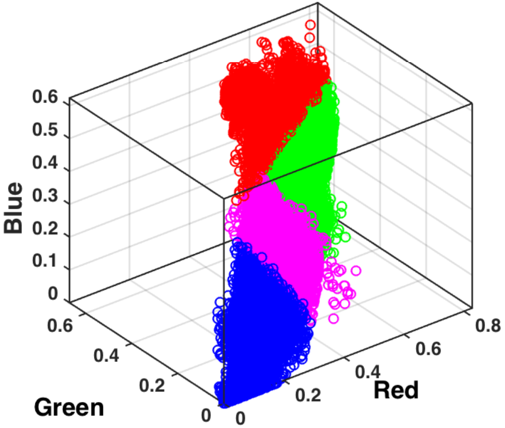
- $\omega : \mathbb{R}^d \to \mathbb{R}$ and $\phi : \mathbb{R}^\rho \to \mathbb{R}$ are kernels.

Previous approaches and their limitations

- Only handle grayscale images [Durand and Dorsey 2002; Chen et al. 2007; Yang et al. 2009; Chaudhury et al. 2011].
- Still not sufficiently fast for real-time applications [Paris and Durand 2009; Adams et al. 2009; Adams et al. 2010].
- Adaptive mainfolds-proceedure is complex and error bounds are not available [Gastal et al. 2011]. **(state-of-the-art)**.

Fast approximation

Pixel distribution



Data clustering

Let the *range* of the guide $\boldsymbol{p}$ to be

$$\Theta = \{\boldsymbol{p}(i) : i \in \Omega\}.$$

- Partition the set Θ into subsets (clusters) $\mathscr{C}_1, ..., \mathscr{C}_K$ and each point from Θ belongs to exactly one cluster.
- Denote the centre (representative) of cluster $\mathscr{C}_\ell$ by $\boldsymbol{\mu}_\ell$.
- Clustering error is given by

$$\mathcal{E}_K = \sum_{\ell=1}^{K} \sum_{\boldsymbol{p}(i) \in \mathscr{C}_\ell} \|\boldsymbol{p}(i) - \boldsymbol{\mu}_\ell\|^2. \tag{3}$$

Fast approximation

For $1 \leq \ell \leq K$, define

$$\boldsymbol{h}_\ell(i) = \sum_{j \in W} \omega(j)\, \phi(\boldsymbol{p}(i-j) - \boldsymbol{\mu}_\ell)\boldsymbol{f}(i-j), \tag{4}$$

$$\zeta_\ell(i) = \sum_{j \in W} \omega(j)\, \phi(\boldsymbol{p}(i-j) - \boldsymbol{\mu}_\ell). \tag{5}$$

where $\boldsymbol{p}(i) \in \mathscr{C}_\ell$.
For $1 \leq \ell \leq K$, define $\boldsymbol{f}_\ell : \Omega \to \mathbb{R}^n$ and $\phi_\ell : \Omega \to \mathbb{R}$ to be

$$\boldsymbol{f}_\ell(i) = \phi(\boldsymbol{p}(i) - \boldsymbol{\mu}_\ell)\boldsymbol{f}(i) \quad \text{and} \quad \phi_\ell(i) = \phi(\boldsymbol{p}(i) - \boldsymbol{\mu}_\ell).$$

Clusterwise convolution

We can then write (4) and (5) as

$$\boldsymbol{h}_\ell(i) = (\omega * \boldsymbol{f}_\ell)(i) = \sum_{j \in W} \omega(j)\boldsymbol{f}_\ell(i-j), \tag{6}$$

$$\zeta_\ell(i) = (\omega * \phi_\ell)(i) = \sum_{j \in W} \omega(j)\phi_\ell(i-j). \tag{7}$$

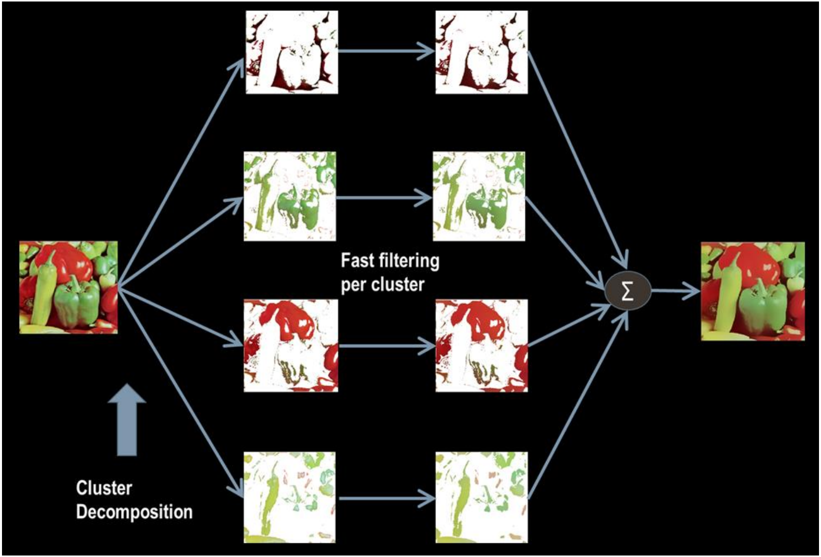
We define the approximation of (1) to be the image $\hat{\boldsymbol{g}}_K : \Omega \to \mathbb{R}^n$ given by

$$\hat{\boldsymbol{g}}_K(i) = \frac{\boldsymbol{h}_\ell(i)}{\zeta_\ell(i)} \qquad \big(\boldsymbol{p}(i) \in \mathscr{C}_\ell\big). \tag{8}$$

Is complexity reduced?

- W is set of neigbourhood data positions per pixel. Cardinality of W is $|W| \sim 1$ to 10 millions.
- Brute-force implementation : $O(|W|)$ per pixel.
- If ω is box or gaussian, computation complexity for one convolution is $O(1)$.
- Total complexity per pixel for approximation is $O(K)$ where K is number of clusters i.e. not dependent on W.

Diagramatic representation

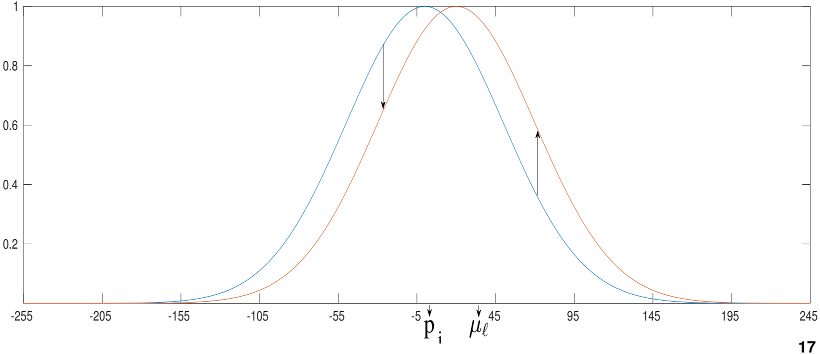


Geometric interpretation

$$\text{Let} \quad \phi(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x}\|^2}{2\sigma_r^2}\right), \boldsymbol{x} \in \mathbb{R}^\rho \text{ with } L = \frac{1}{\sigma_r}\exp(-1/2).$$

Here kernel is approximated as

$$\exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{p}(i)\|^2}{2\sigma_r^2}\right) \rightarrow \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{\mu}_\ell\|^2}{2\sigma_r^2}\right). \tag{9}$$



Theorem (Approximation error)

Assume kernel ϕ is Lipschitz continuous, for some $L > 0$,

$$|\phi(\boldsymbol{x}) - \phi(\boldsymbol{y})| \leq L\|\boldsymbol{x} - \boldsymbol{y}\| \qquad (\boldsymbol{x}, \boldsymbol{y} \in \mathbf{R}^{\rho}),$$

$$\sum_{i \in \Omega} \|\hat{\boldsymbol{g}}_K(i) - \boldsymbol{g}(i)\|^2 \leq CLn|W|^2 \mathcal{E}_K, \ \textit{for } C > 0. \tag{10}$$

Corollary (Arbitrary accuracy)

If ϕ is Lipschitz continuous and $\mathcal{E}_K$ vanishes as $K \to \infty$, then for $i \in \Omega$,

$$\lim_{K \to \infty} \hat{\boldsymbol{g}}_K(i) = \boldsymbol{g}(i). \tag{11}$$

[1] "Fast high-dimensional filtering using clustering", Pravin Nair and Kunal N. Chaudhury, ICIP 2017.

Results

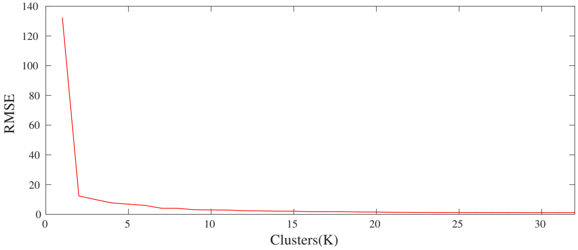
Edge-preserving filter ($\sigma_s = 10$, $\sigma_r = 30$, $K = 16$)

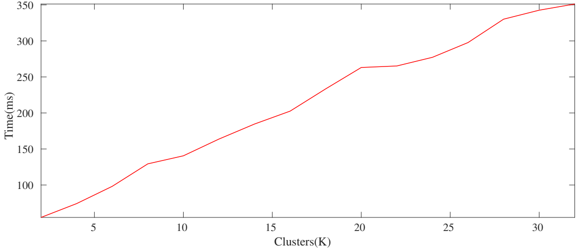
Input.

Bilateral output.
(13sec)

Approximation.
(223ms, RMSE=1.78)

Accuracy and timing analysis





Edge-preserving filter ($\sigma_s = 10$, $\sigma_r = 30$, $K = 16$)

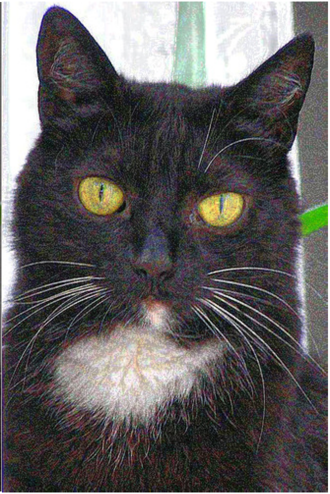
Input image(876×584).

Bilateral output. (Time=42.4sec)

Approximation. (Time=1.229sec, RMSE=1.76)

Detail enhancement

Input image.



Detail enhancement output.

Denoising

- Gaussian Noise of $\sigma = 50$.
- Patch of radius 3 pixels is reduced to 6 dimensions using PCA.

Noisy input(640×427). ($PSNR = 14.8dB$)

Denoised output. ($PSNR = 28.2dB$, Time=1.9sec)

[1] Gastal et al. 2011.

Denoising with extra information

- Patch of radius 3 pixels reduced to 6 dimensions using PCA + 1D infrared data
- Box filter radius is 8 pixels
- $\sigma_r = 0.2$ for color and $\sigma_r = 0.04$ for infrared.

Noisy low light input.

[1] Gastal et al. 2011.

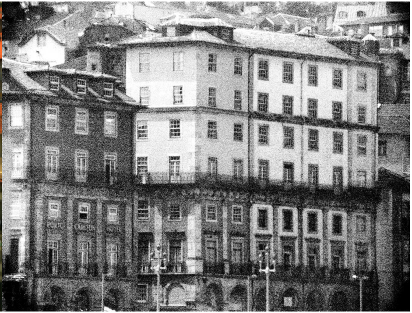
NLM output.

Denoising with extra infrared data.

Hyperspectral filtering

Parameters : $\sigma_s = 5$, $\sigma_r = 0.4$ and $K = 16$.

Hyperspectral input($1340 \times 1017 \times 33$).



440nm channel input.

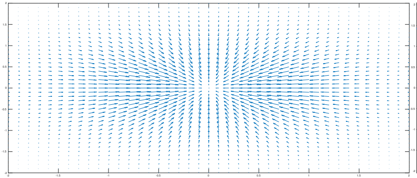
[1] Foster et al. 2006.

Bilateral output. (Time=27min)

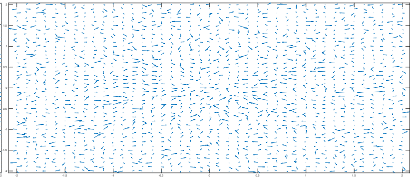
Approximation. (Time=1min)

2D vector field denoising

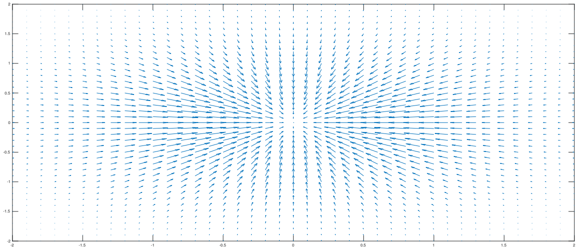
- x and y form rectangular grid.
- $z = \exp\left(-(x^2 + y^2)\right)$.
- $[x, y] \leftarrow [z_x, z_y]$.



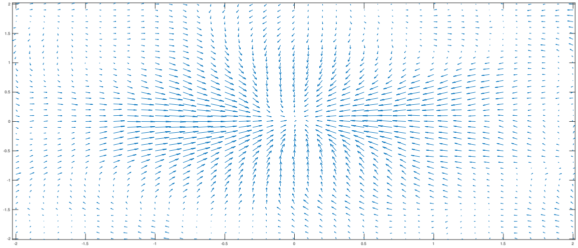
Input flow field.



Noisy flow field.



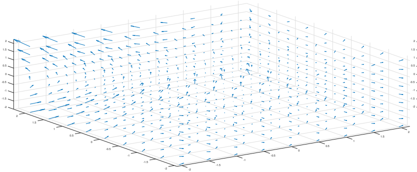
Input flow field.



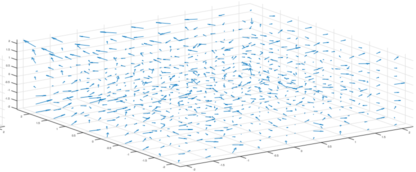
Denoised flow field.

3D vector field denoising

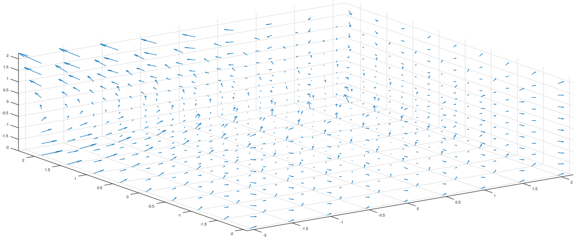
- x, y and z form $3D$ rectangular grid.
- $u = x^2 - ze^y$, $v = y^3 - xze^y$ and $w = z^4 - xe^y$.
- $[x, y, z] \leftarrow [u, v, w]$.



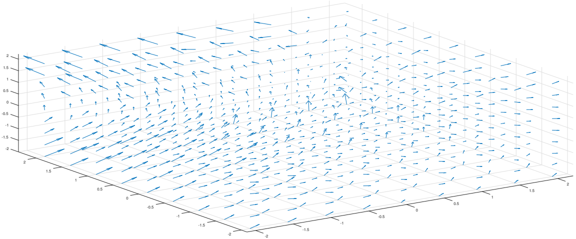
Input flow field.



Noisy flow field.



Input flow field.



Denoised flow field.

Improvement by interpolation

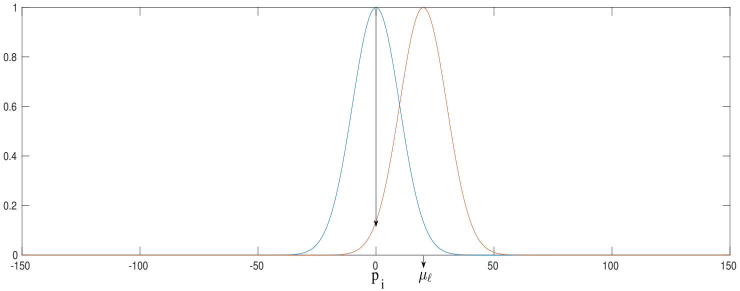
Input image.

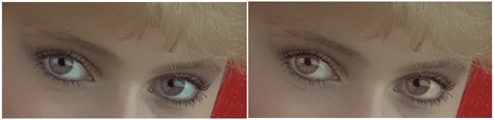
Direct implementation.

Approximation($K = 8$).

Interpolation($K = 8$).

2) Artefacts caused by very narrow range kernel

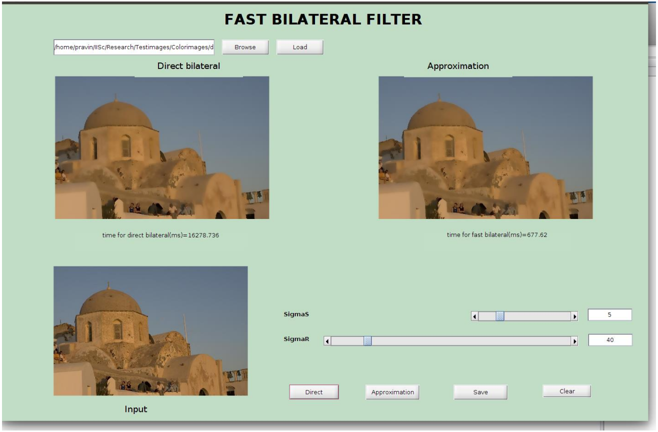




Direct bilateral. Approximation algorithm.

Figure: $\sigma_s = 10$ and $\sigma_r = 0.04$. Check the loss of color in filtered image.

./demoUI.m


FAST BILATERAL FILTER
/home/pravin/IISc/Research/Testimages/Colorimages/d
Browse
Load
Direct bilateral
Approximation
time for direct bilateral(ms)=16278.736
time for fast bilateral(ms)=677.62
SigmaS
5
SigmaR
40
Direct
Approximation
Save
Clear
Input

Conclusions

- Proposed a simple algorithm for high dimensional filtering.
- Detailed accuracy analysis performed.
- Applied the algorithm in various applications.
- Showed reduction in time both theoretically and empirically.
- Scope of improvement discussed.

Thanks for listening.