Medical Image Analysis and Processing

Image Noise Filtering – Point Esimation

Emad Fatemizadeh

Distance/online Course: Session 09

Date: 14 March 2021, 24th Esfand 1399

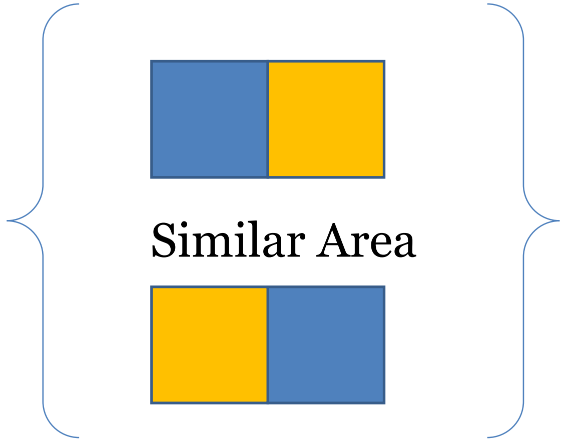
Contents

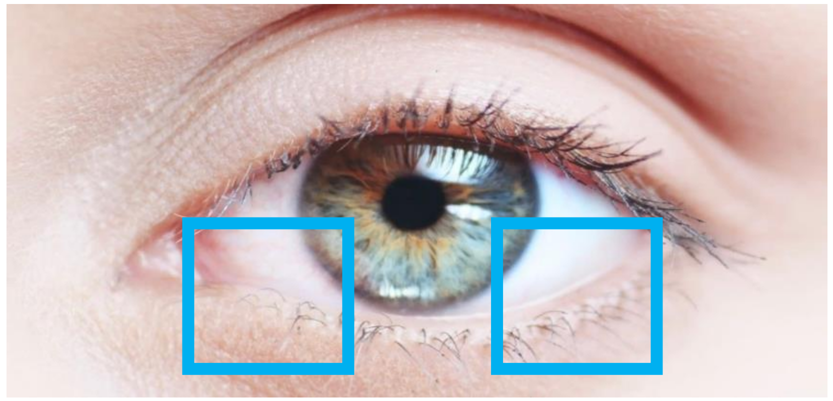
Non Local Mean (NLM) Filtering

- Contents of $S(X)$:
  - Pixel values (vectorized patch)
  - Feature Vectors (any features sets, SIFT/SURF/ORB/BRIEF/….)
  - Probability distribution
  - …
- Distance:
  - Euclidean
  - Divergence measure (between two pdfs)
  - …

NLM Challenges (Patch descriptor):

- For this situation, vectorized patch and Euclidean distance is not working (why?)





NLM Challenges (Patch descriptor):

- Euclidean distance is optimal (ONLY) for gaussian additive noise!

NLM Challenges (Patch distance):

- Divergence measure as patch distance:

$$D_{\mathrm{KL}}(p \,\|\, q) = \int p(x) \ln\left(\frac{p(x)}{q(x)}\right) dx$$

$$H^2(p, q) = 2 \int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 dx$$

$$D_J(p \,\|\, q) = \int (p(x) - q(x))\left(\ln p(x) - \ln q(x)\right) dx$$

$$D^{(\alpha)}(p \,\|\, q) = \frac{4}{1-\alpha^2}\left(1 - \int p(x)^{\frac{1-\alpha}{2}} q(x)^{\frac{1+\alpha}{2}} dx\right)$$

$$D_e(p \,\|\, q) = \int p(x)\left(\ln p(x) - \ln q(x)\right)^2 dx$$

$$D_{\chi^2}(p \,\|\, q) = \frac{1}{2} \int \frac{(p(x) - q(x))^2}{p(x)} dx$$

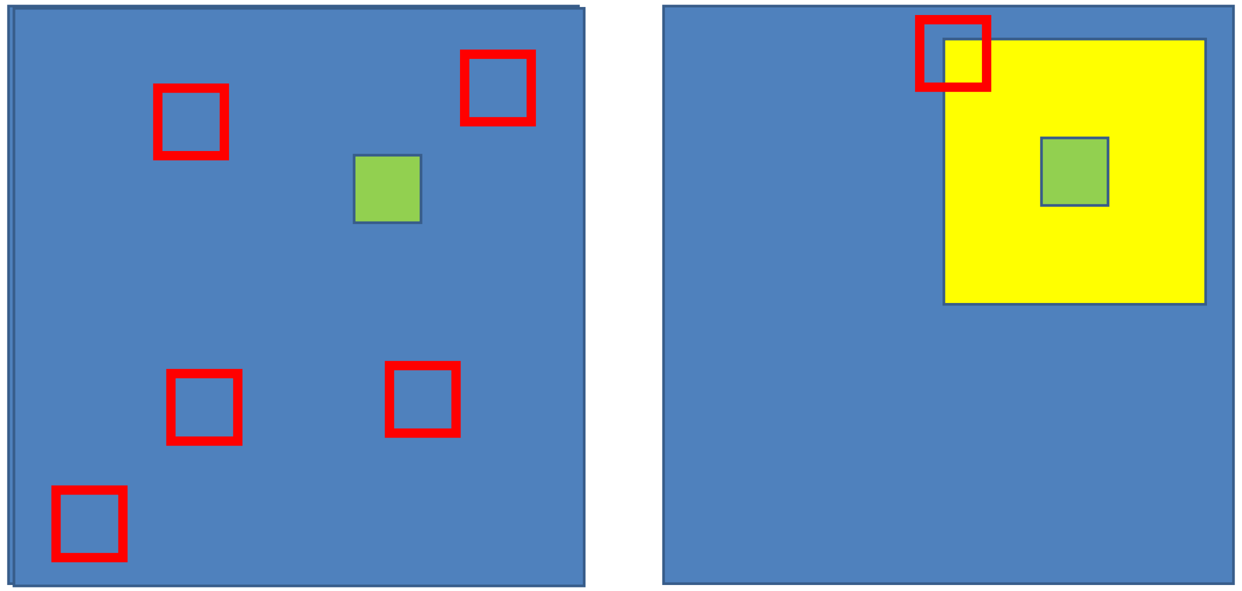
- Many different patches has same pdf

NLM Challenges (Complexity):

- For $N \times N$ image and $n \times n$ patch size need $O(N^4 n^2)$ operation (full search)
- Solutions:
  - Limit search windows
  - Multiscale algorithm,
  - Patch-based filtering,
  - Pre-classify (Prefilter),
  - …

NLM Challenges (Complexity):

› Limit Search Windows
  - Regular
  - Random (PatchMatch: an advanced search algorithm,)



NLM Challenges (Complexity):

› Multiscale algorithm:

1. Zoom out the image u_0 by a factor 2, by a standard Shannon subsampling procedure. This yields a new image u_1. For convenience, we denote by (i, j) the pixels of u_1 and by $(2i, 2j)$ the even pixels of the original image u_0.
2. Apply NL-means to u_1, so that with each pixel (i, j) of u_1, a list of windows centered in $(i_1, j_1), \ldots, (i_k, j_k)$ is associated.
3. For each pixel of u_0, $(2i + r, 2j + s)$ with $r, s \in \{0, 1\}$, we apply the NL-means algorithm. But instead of comparing with all the windows in a searching zone, we compare only with the nine neighboring windows of each pixel $(2i_l, 2j_l)$ for $l = 1, \cdots, k$.
4. This procedure can be applied in a pyramid fashion by subsampling u_1 into u_2, and so on. In fact, it is not advisable to zoom down more than twice.

By zooming down by just a factor 2, the computation time is divided by approximately 16.

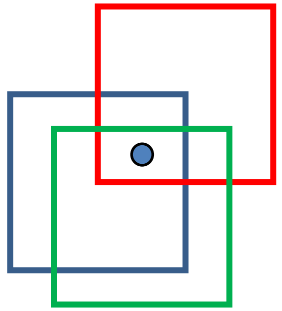
NLM Challenges (Complexity):

- Patch-based filtering

$$\hat{P}(i) = \frac{1}{Z_i}\sum_j K\big(P(i), Q(j)\big)Q(j)$$

- $P(i) \in \mathbb{R}^{n\times n}$: i_{th} patch (fixed), $Q(j) \in \mathbb{R}^{n\times n}$: j_{th} patch (float),
- Z_i: Normalization factor
- Each pixel may falls in several patch:
  - Weighted or unweighted averaging!

$$\hat{f}(X) = \frac{1}{|A_X|}\sum_{j\in A_X} P_j(X), \quad A_X = \{P_j | X \in P_j\}$$



NLM Challenges (Complexity):

- Pre-Classify (Pre-Filter):
  - Skip distance calculations for some float windows, based on:
    - Mean of windows ($\mathrm{mean}(S_g(X)) \not\cong \mathrm{mean}(S_g(Y))$)
    - variance of windows ($(\mathrm{var}(S_g(X)) \not\cong \mathrm{var}(S_g(Y)))$)
    - Simple (calculation) statistical moments of windows
    - Gradient of windows

NLM Challenges (Complexity):

- Dimension Reduction:
  - Distances computed from projections of onto a lower dimensional subspace (e.g. PCA on windows)

NLM Challenges (Complexity):

› Parameter Selection:
  - Patch size: Scale of image size (7×7 or 7×7)
  - Smoothing parameter (h): proportional to noise level (k$\sigma, k \approx 10$),

› Noise Estimation:

$$E\left\{\left\|S_g(X) - S_g(Y)\right\|_2^2\right\} = E\left\{\left\|S_f(X) - S_f(Y)\right\|_2^2\right\} + 2\sigma^2$$

$$\sigma^2 \approx \frac{1}{2} \min_{X \neq Y} \left\{\left\|S_g(X) - S_g(Y)\right\|_2^2\right\}$$

BM3D – A Fashion Scheme!

› Main article:

Image denoising by sparse 3D transform-domain collaborative filtering

› http://www.cs.tut.fi/~foi/GCF-BM3D/

BM3D

› Algorithm has two major steps:

1. Estimates the clean image using hard thresholding during the collaborative filtering.
2. Wiener filtering using noisy and estimated clean image

› Optimal Wiener Filter:

$$W(u,v) = \frac{P_{FF}(u,v)}{P_{FF}(u,v) + P_{NN}(u,v)}$$

$$\hat{F}(u,v) = W(u,v)G(u,v)$$

BM3D

› Key Idea:

› Grouping:

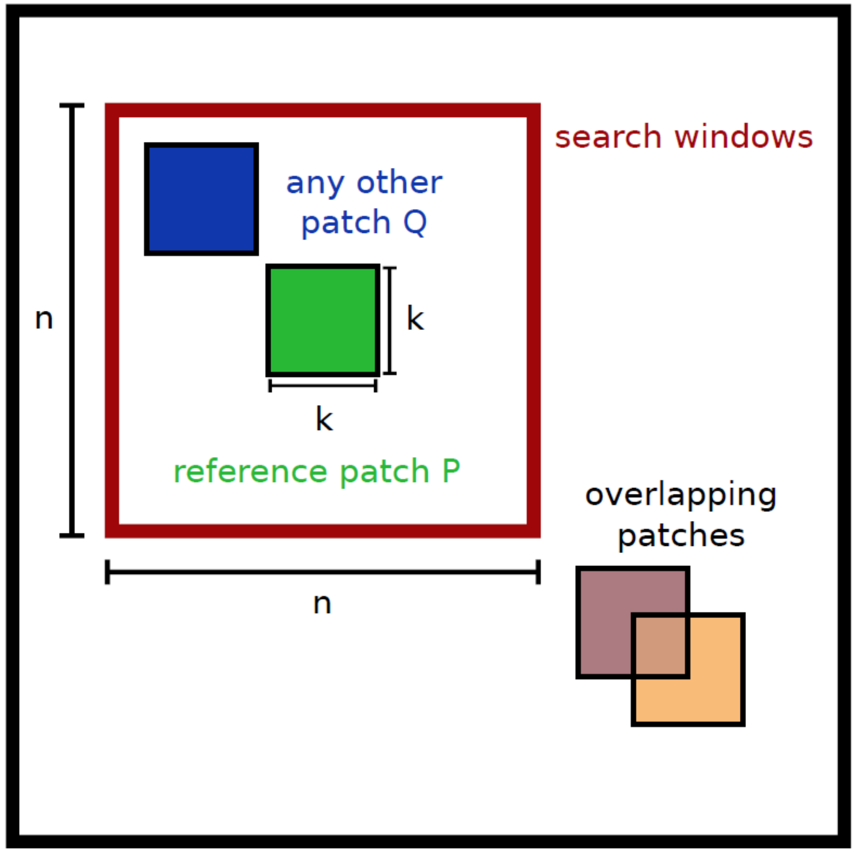
Image patches are grouped together based on similarity to a reference patch using Block-Matching Algorithm.

› Collaborative Filtering:

All image patches in a group are then stacked together to form 3D cylinder-like shapes. Filtering is done on every patch group

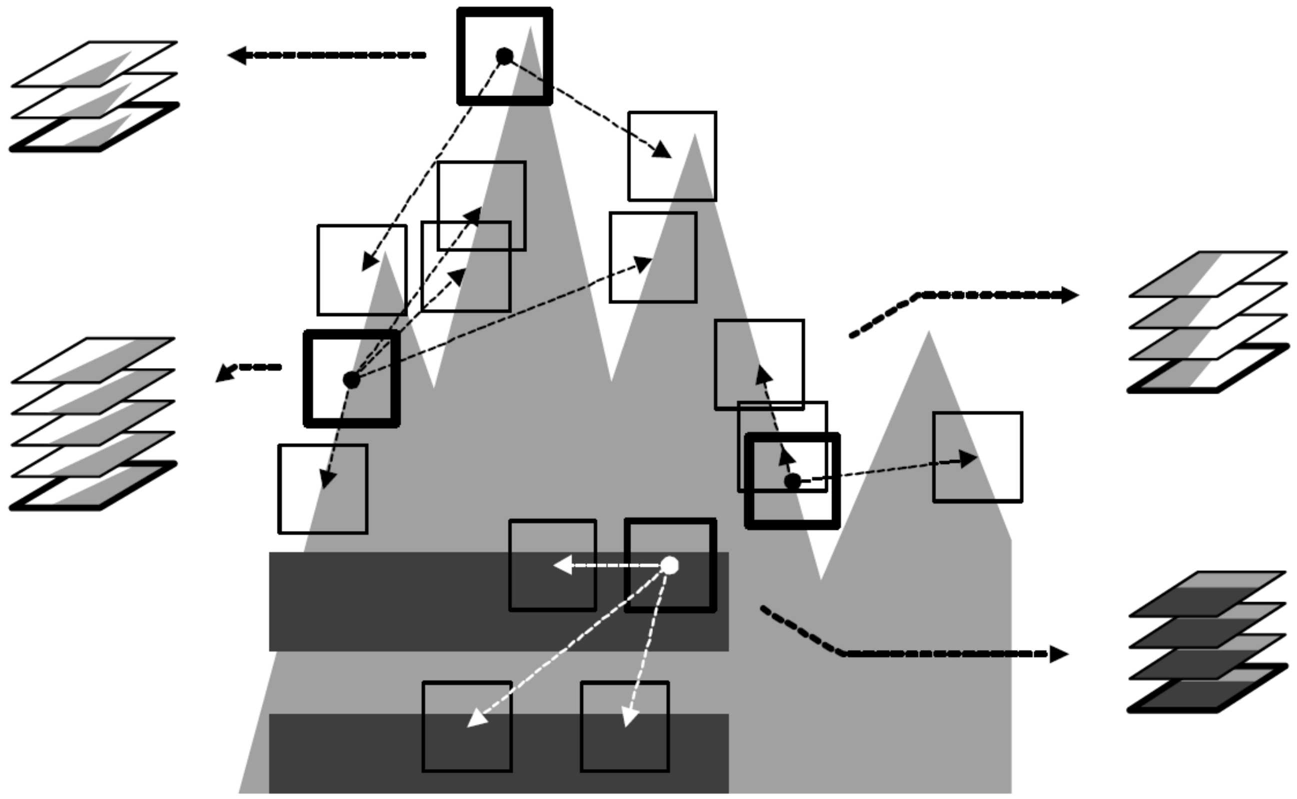
BM3D

› Reference patch and search:



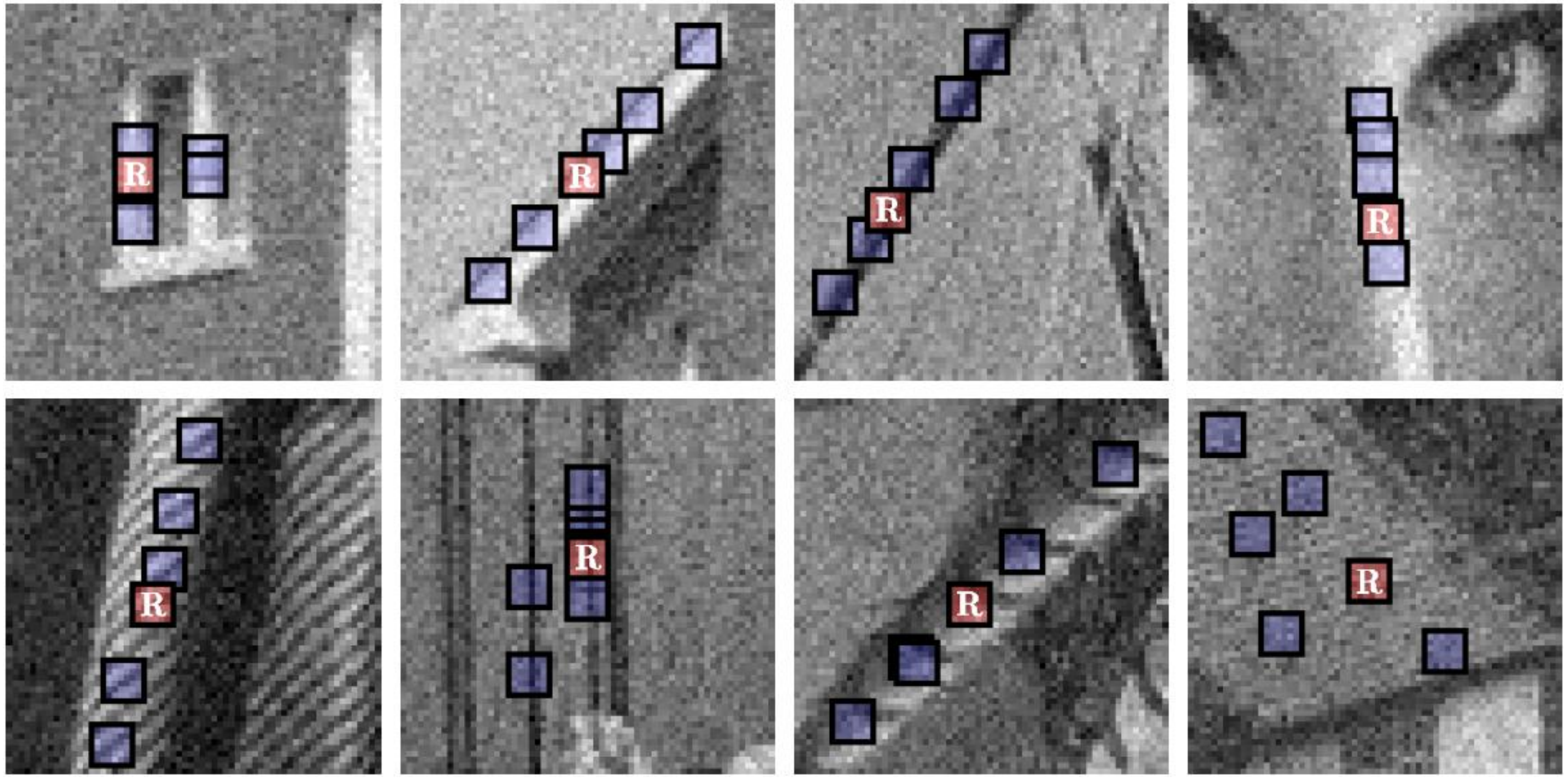
BM3D

› Grouping:



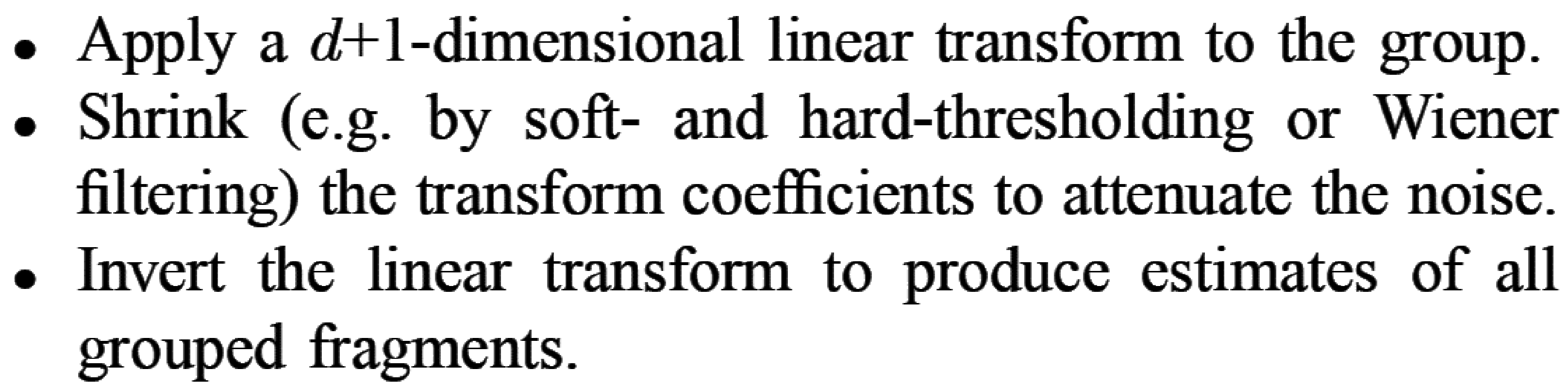
BM3D

› Grouping:



BM3D

› Collaborative Filtering:



- Apply a d+1-dimensional linear transform to the group.
- Shrink (e.g. by soft- and hard-thresholding or Wiener filtering) the transform coefficients to attenuate the noise.
- Invert the linear transform to produce estimates of all grouped fragments.

BM3D

› Group Characteristic:

1. intra-fragment correlation which appears between the pixels of each grouped fragment;
2. inter-fragment correlation which appears between the corresponding pixels of different fragments result of the similarity between grouped fragments.

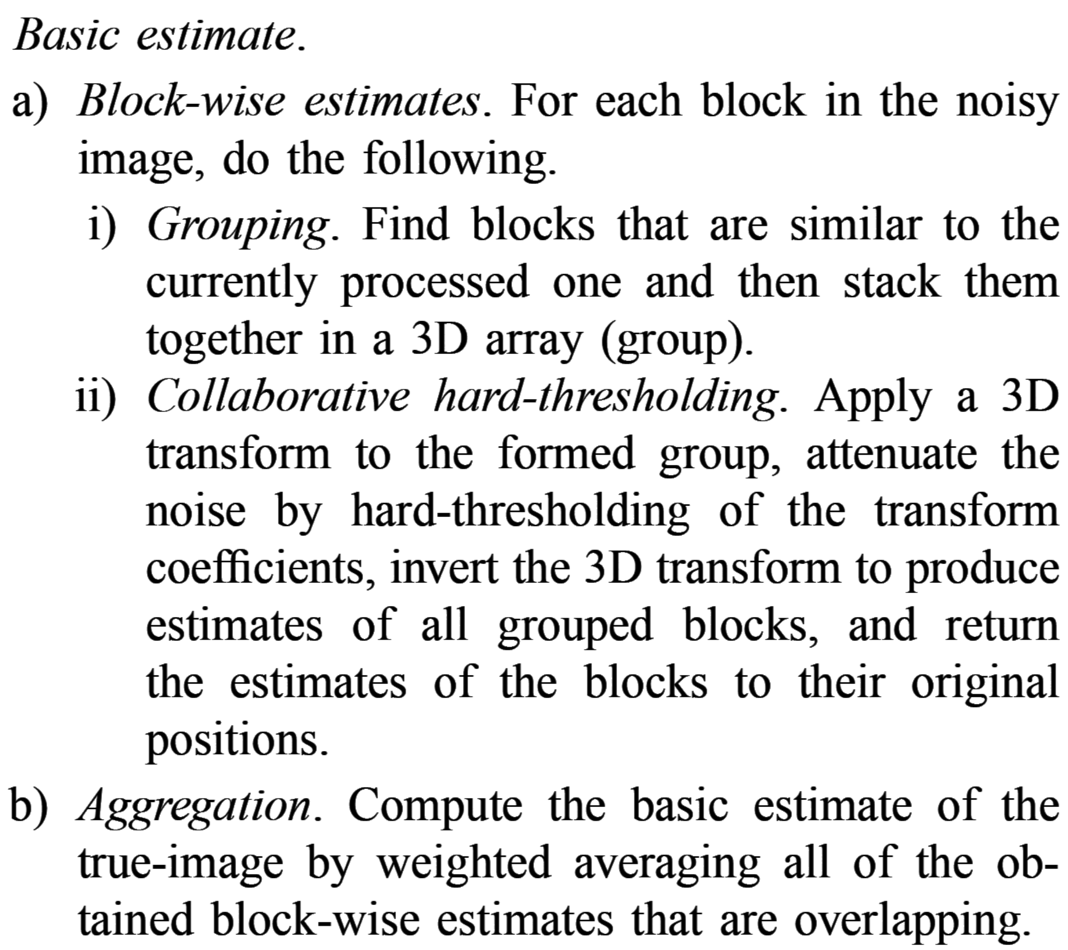
BM3D

› General Procedure:

1. Find blocks that are similar to the reference one (Block-Matching) and stack them together to form a 3D array (group);
2. Perform collaborative filtering of the group and return the obtained 2D estimates of all grouped blocks to their original locations.

BM3D

› Step #1



Basic estimate.

a) *Block-wise estimates*. For each block in the noisy image, do the following.

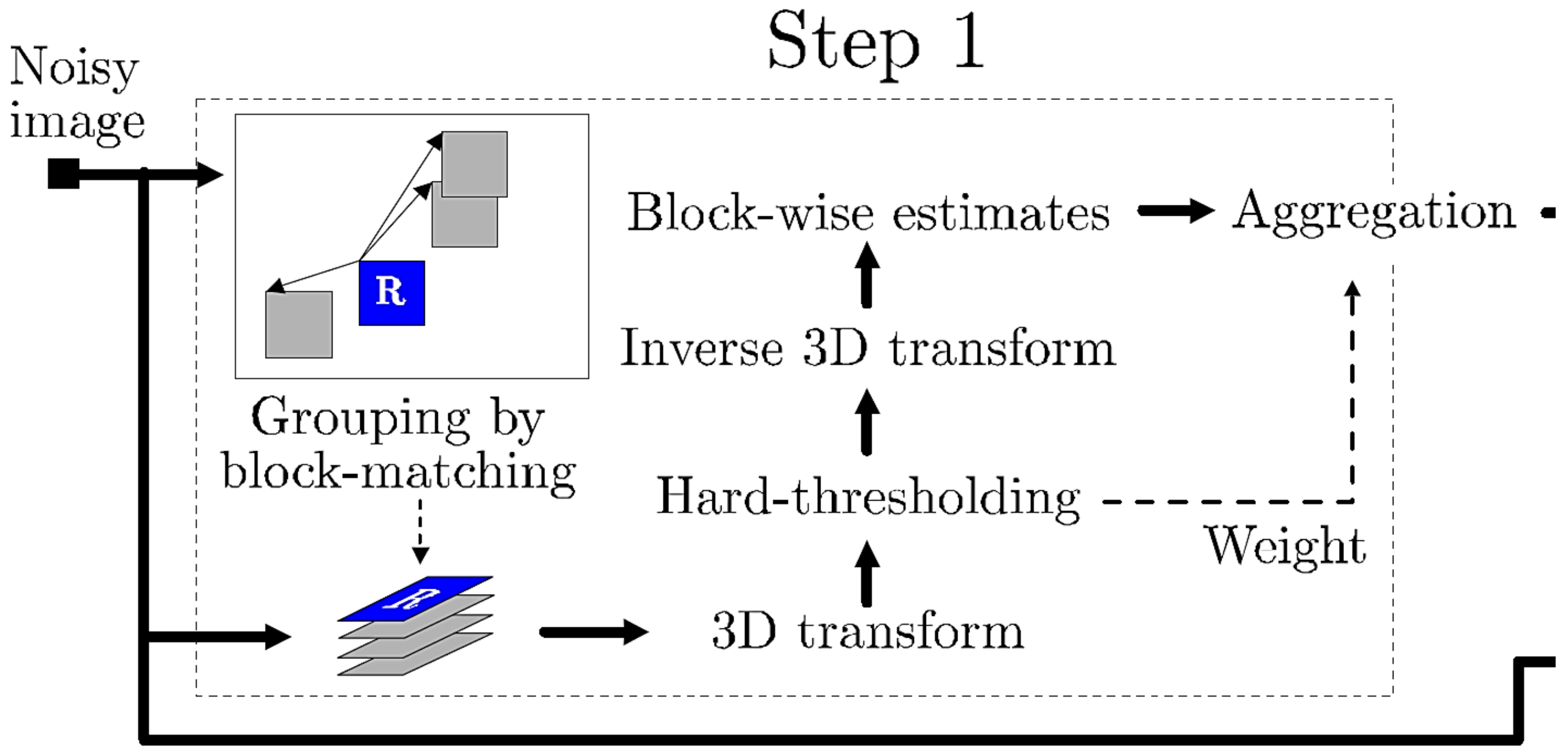
   i) *Grouping*. Find blocks that are similar to the currently processed one and then stack them together in a 3D array (group).

   ii) *Collaborative hard-thresholding*. Apply a 3D transform to the formed group, attenuate the noise by hard-thresholding of the transform coefficients, invert the 3D transform to produce estimates of all grouped blocks, and return the estimates of the blocks to their original positions.

b) *Aggregation*. Compute the basic estimate of the true-image by weighted averaging all of the obtained block-wise estimates that are overlapping.

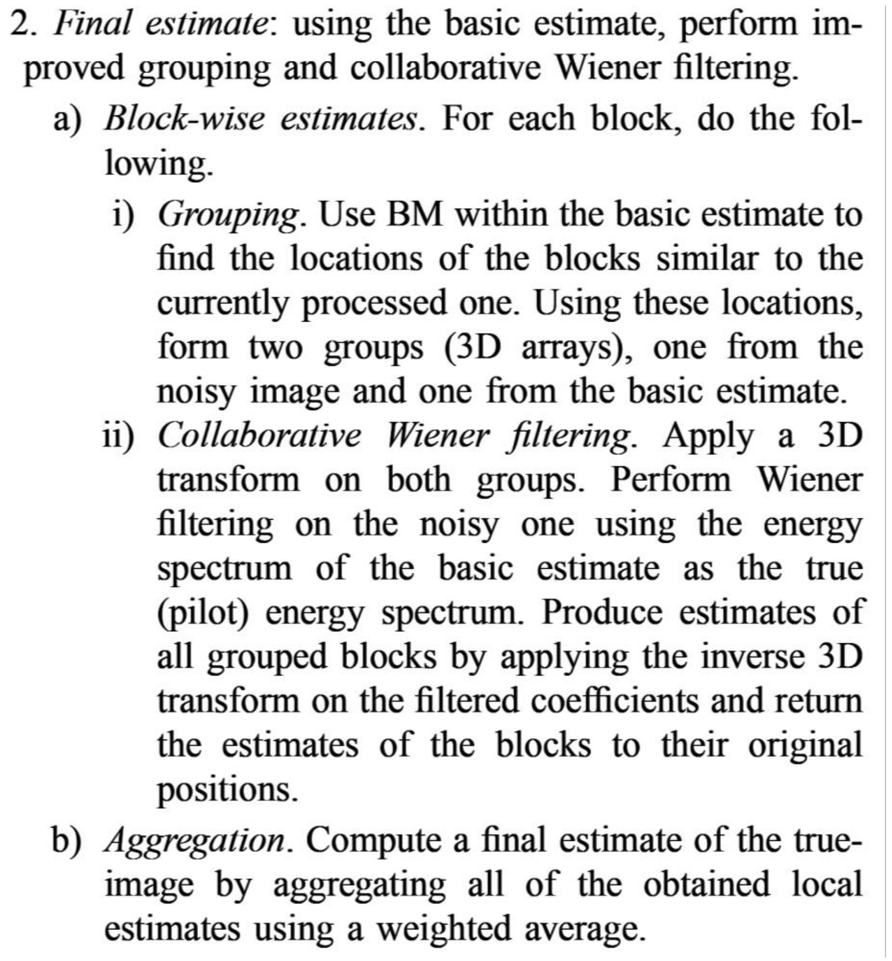
BM3D

- Step #1


Step 1
Noisy image
R
Grouping by block-matching
R
3D transform
Hard-thresholding
Inverse 3D transform
Block-wise estimates
Aggregation
Weight

BM3D

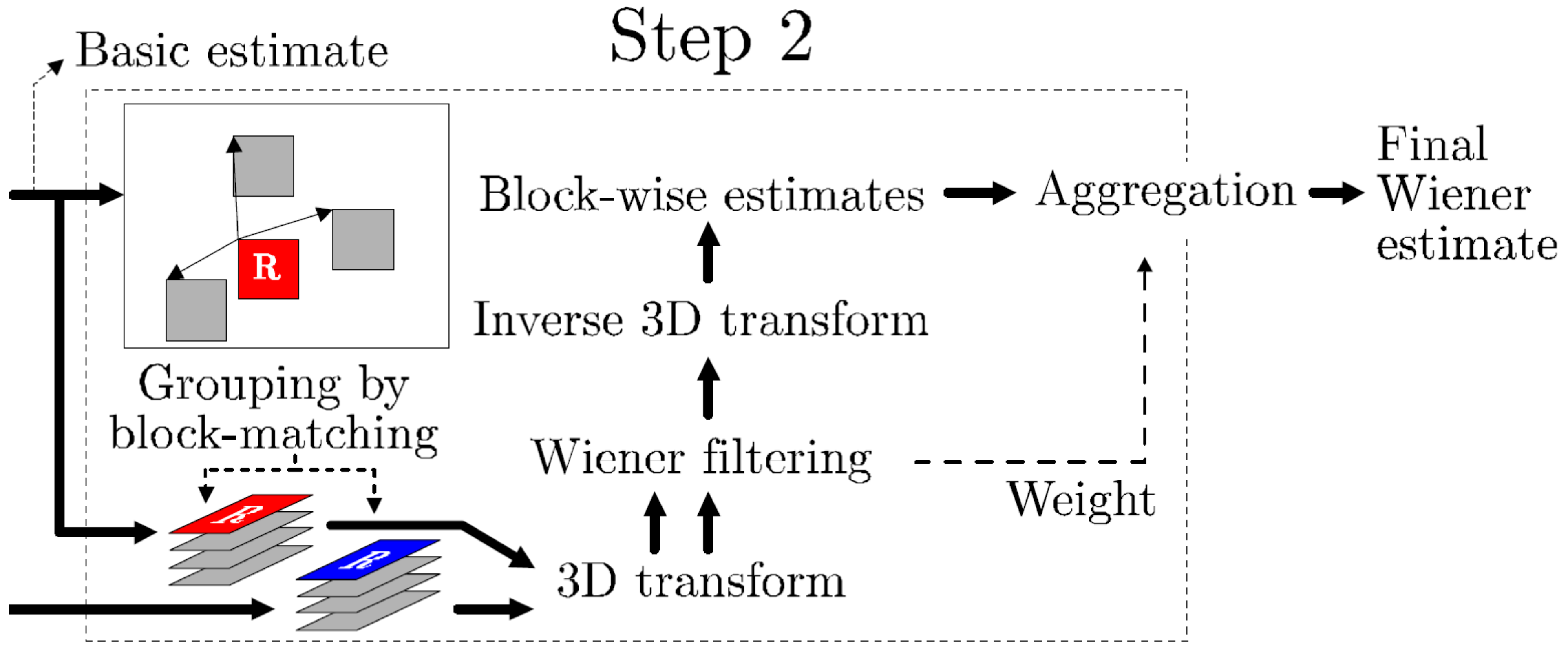
› Step #2



2. *Final estimate*: using the basic estimate, perform improved grouping and collaborative Wiener filtering.
   a) *Block-wise estimates*. For each block, do the following.
      i) *Grouping*. Use BM within the basic estimate to find the locations of the blocks similar to the currently processed one. Using these locations, form two groups (3D arrays), one from the noisy image and one from the basic estimate.
      ii) *Collaborative Wiener filtering*. Apply a 3D transform on both groups. Perform Wiener filtering on the noisy one using the energy spectrum of the basic estimate as the true (pilot) energy spectrum. Produce estimates of all grouped blocks by applying the inverse 3D transform on the filtered coefficients and return the estimates of the blocks to their original positions.
   b) *Aggregation*. Compute a final estimate of the true-image by aggregating all of the obtained local estimates using a weighted average.

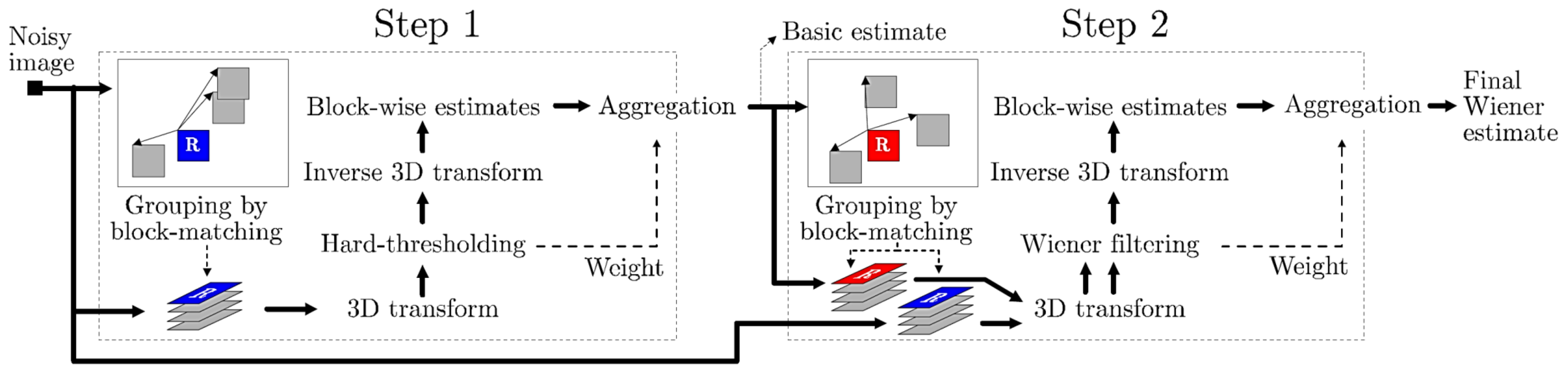
BM3D

› Step #2


Step 2
Basic estimate
R
Grouping by block-matching
Block-wise estimates
Aggregation
Final Wiener estimate
Inverse 3D transform
Wiener filtering
Weight
3D transform

BM3D

› Block Diagram:



BM3D - Mathematics

› Image Model:

$$z(x) = y(x) + \eta(x),\ x \in X,\ \eta(\cdot) \sim \mathcal{N}(0, \sigma^2)$$

› Block Matching (x_R and x):

$$d(Z_{x_R}, Z_x) = \frac{\left\| \Upsilon\left(\mathcal{T}_{2\mathrm{D}}^{\mathrm{ht}}(Z_{x_R})\right) - \Upsilon\left(\mathcal{T}_{2\mathrm{D}}^{\mathrm{ht}}(Z_x)\right) \right\|_2^2}{\left(N_1^{\mathrm{ht}}\right)^2},$$

› where $\mathcal{T}_{2D}^{ht}$ is a 2D transform (DCT, DFT, …) and Υ is hard threshold:

$$\Upsilon(\lambda, \lambda_{thr}) = \begin{cases} \lambda, & \text{if } |\lambda| > \lambda_{thr} \\ 0, & \text{otherwise.} \end{cases}$$

› Matching Criteria:

$$S_{x_R}^{\mathrm{ht}} = \left\{ x \in X : d(Z_{x_R}, Z_x) \leq \tau_{\mathrm{match}}^{\mathrm{ht}} \right\}$$

BM3D - Mathematics

- Collaborative Filtering (Denoising in 3D transform domain):

$$\widehat{\mathbf{Y}}^{\mathrm{ht}}_{S^{\mathrm{ht}}_{x_R}} = \mathcal{T}^{\mathrm{ht}^{-1}}_{3\mathrm{D}} \left(\Upsilon \left(\mathcal{T}^{\mathrm{ht}}_{3\mathrm{D}} \left(\mathbf{Z}_{S^{\mathrm{ht}}_{x_R}} \right) \right) \right)$$

- Weights of local estimate:

$$\omega_{x_R} = \begin{cases} \frac{1}{N_{har}}, & \text{if } N_{har} \geq 1 \\ 1, & \text{otherwise,} \end{cases}$$

- N_{har}: # of non-zero transform coefficients after hard-thresholding

BM3D - Mathematics

- Aggregation:
- We have overcomplete representation of the estimated image due to the overlap between the blocks

$$\widehat{y}(x) = \frac{\sum_{x_R \in X} \sum_{x_m \in S_{x_R}} \omega_{x_R} \widehat{Y}^{x_R}_{x_m}(x)}{\sum_{x_R \in X} \sum_{x_m \in S_{x_R}} \omega_{x_R} \chi_{x_m}(x)}, \quad \forall x \in X,$$

- $\chi_{x_m}: \mathrm{X} \rightarrow \{0,1\}$ is characteristic function of the square support of a block located at $x_m \in X$

BM3D - Mathematics

› Step #2 Grouping and Collaborative Filtering:

$$S^{\text{wie}}_{x_R} = \left\{ x \in X : \frac{\left\| \widehat{Y}^{\text{basic}}_{x_R} - \widehat{Y}^{\text{basic}}_{x} \right\|_2^2}{\left(N_1^{\text{wie}}\right)^2} < \tau^{\text{wie}}_{\text{match}} \right\}$$

$$\mathbf{W}_{S^{\text{wie}}_{x_R}} = \frac{\left| \mathcal{T}^{\text{wie}}_{3\text{D}}\left(\widehat{\mathbf{Y}}^{\text{basic}}_{S^{\text{wie}}_{x_R}} \right) \right|^2}{\left| \mathcal{T}^{\text{wie}}_{3\text{D}}\left(\widehat{\mathbf{Y}}^{\text{basic}}_{S^{\text{wie}}_{x_R}} \right) \right|^2 + \sigma^2} \Longrightarrow \widehat{\mathbf{Y}}^{wie}_{S^{\text{wie}}_{x_R}} = \mathcal{T}^{\text{wie}^{-1}}_{3\text{D}} \left(\mathbf{W}_{S^{\text{wie}}_{x_R}} \mathcal{T}^{\text{wie}}_{3\text{D}} \left(\mathbf{Z}_{S^{\text{wie}}_{x_R}} \right) \right)$$

BM3D - Mathematics

› Final Aggregation as before with this weight function:

$$w_{x_R}^{\text{wie}} = \sigma^{-2} \left\| \mathbf{W}_{S_{x_R}^{\text{wie}}} \right\|_2^{-2}$$

BM3D - Mathematics

› Final Aggregation as before with this weight function:

$$w_{x_R}^{\text{wie}} = \sigma^{-2} \left\| \mathbf{W}_{S_{x_R}^{\text{wie}}} \right\|_2^{-2}$$

The End

- AnY QuEsTiOn?