Different Implementations of the NLM Denoising Algorithm

Robu Petru-Razvan Verzotti Matteo-Alexandru Voaides-Negustor Robert-Ionut

Motivation

- Image noise is commonly modeled as additive white Gaussian noise.
- Denoising should remove noise while preserving textures and edges.
- Non-Local Means is effective but expensive at scale.

$$y = x + \eta, \quad \eta \sim \mathcal{N}(0, \sigma)$$

Non-Local Means

Patch-based weighted average

$$z(p) = \frac{1}{C(p)} \sum_{q \in \Omega} w(p,q)\, y(q), \quad C(p) = \sum_{q \in \Omega} w(p,q)$$

Standard similarity weight

$$w_i = \exp\left(-\frac{\|\mathbf{y} - \mathbf{x}_i\|^2}{2h_r^2} \right)$$

- Complexity: $\mathcal{O}(mnd)$ (or $\mathcal{O}(mD^2d)$ with a window).

Monte Carlo NLM: Sampling and Estimators

- Sample reference patches with $I_j \sim \text{Bernoulli}(p_j)$.
- Use sampled weights to approximate NLM efficiently.

Unbiased estimators

$$A = \frac{1}{n}\sum_{j=1}^{n} x_j w_j \frac{I_j}{p_j}, \quad B = \frac{1}{n}\sum_{j=1}^{n} w_j \frac{I_j}{p_j}$$

$$Z = \frac{A}{B}$$

- A and B are unbiased; Z is biased but concentrates as n grows.

Monte Carlo NLM: Stochastic Approximation

- Concentration bounds control the deviation $|Z - z|$.

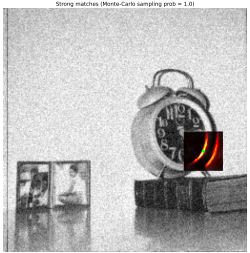
Simplified probability bound (uniform sampling)

$$\mathbb{P}(|Z - z| > \varepsilon) \leq 2 \exp\left(- \frac{n\varepsilon^2}{2\left(\rho \sum_{j=1}^{n} \alpha_j^2 + \varepsilon/6\right)} \right)$$

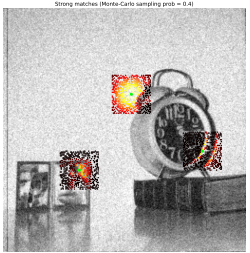
- $\alpha_j = \frac{w_j}{\sum_{k=1}^{n} w_k}$ and $\rho = \frac{1-p}{p}$.
- Result: the error probability decays exponentially with the window size.

Spatial Sampling (Semi-Local NLM)

$$w_j = w_j^r \cdot w_j^s, \quad w_j^s = \exp\left(-\frac{(d_2^j)^2}{2h_s^2}\right) \cdot \mathbb{I}\{d_\infty^j \leq \rho\}$$



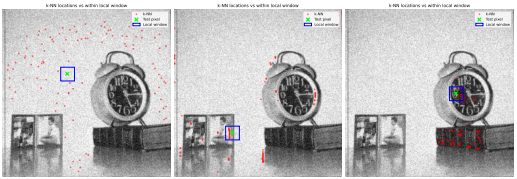
Weight values in search window

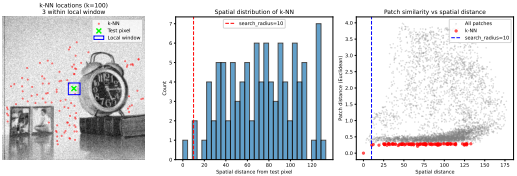


Weights for sampled center pixels

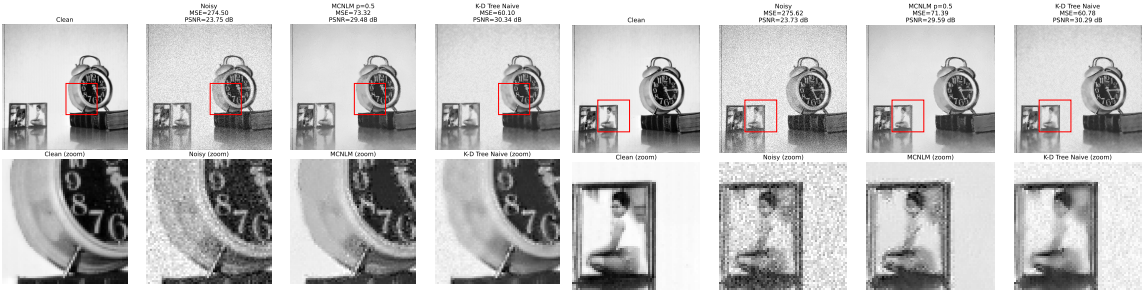
KD-Tree NLM: Patch Search

- Build a KD-Tree of patches; query K nearest neighbors.
- Faster similarity search than brute-force within a window.
- Trade-off: accuracy vs speed and parallelization overhead.





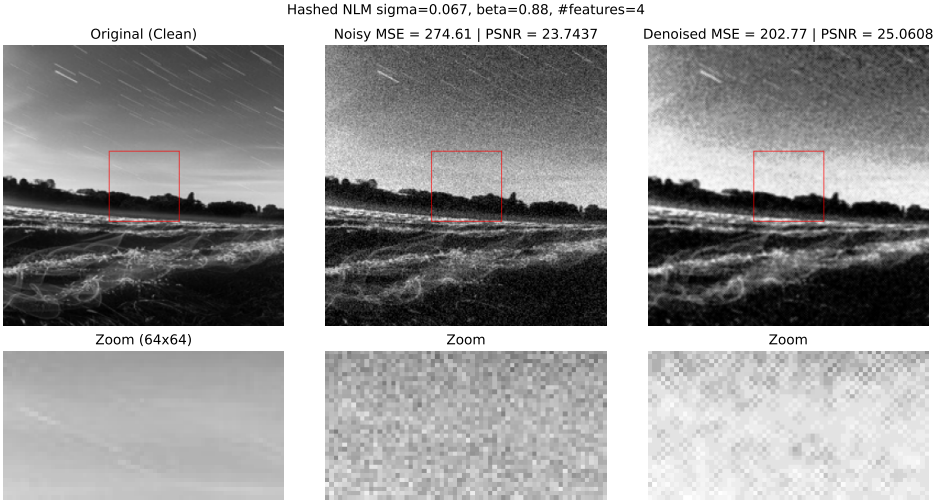
KD-Tree NLM: Results



- KD-Tree improves detail in some textured regions, but can add noise in flat areas.

Hashed NLM

- Approximate patch matching via hashing to speed up search.
- Produces competitive denoising with distinctive artifacts.



Hashed NLM

- Features of this space are 4-dimensional vectors that contain the direct neighbours of our pixel.
- Because the weighting kernel is axis-aligned we can use separable convolution.
- Locality introduced by adding position as additional dimensions.

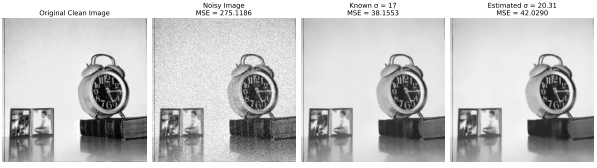
$$H_1(g) = \sum_{j=1}^{|X|} \delta(g - f_j)$$

$$H_f(g) = \sum_{j=1}^{|X|} \delta(g - f_j) \cdot I[x_j]$$

$$\tilde{I}_i = \frac{(w_K * H_f)(\mathbf{f}_i)}{(w_K * H_1)(\mathbf{f}_i)} = \frac{H'_f(\mathbf{f}_i)}{H'_1(\mathbf{f}_i)}.$$

Noise Estimation via FFT

- Estimate σ from high-frequency components.
- FFT $\rightarrow$ mask high frequencies $\rightarrow$ inverse FFT.
- Use $\sigma = \text{std}(\textit{noise})$ for denoising.



Conclusion

- MCNLM reduces NLM cost while preserving image structure.
- Spatial weighting and KD-Tree search are viable alternatives.
- Hashed NLM and FFT noise estimation provide additional practical speedups.

Thanks!