GENERALIZED NON-LOCAL MEANS FOR ITERATIVE DENOISING

Enming Luo[a], Shengjun Pan[b], Truong Nguyen[a]

University of California, San Diego
[a] Department of Electrical and Computer Engineering
[b] Department of Computer Science and Engineering
{*eluo, s1pan, tqn001*}*@ucsd.edu*

ABSTRACT

Non-local means (NL-means) filter removes independent and identically distributed (i.i.d.) image noises using self-similarity. In this paper, we derive a *generalized* NL-means (GNL-means), which is specifically used to deal with non-i.i.d. noises in the NL-means filtered images. Inspired by BM3D and LPG-PCA, which perform denoising iteratively, our idea is also to iteratively apply NL-means. However, NL-means can't be applied directly due to the correlated noises in the image filtered by NL-means. We modify the original NL-means to incorporate noise dependence into the weight function, and show how the new weight can be calculated and give a reasonable estimator. We evaluate GNL-means on several benchmark images, and compare it to NL-means and other state-of-the-art *non-local* methods including BM3D and LPG-PCA. Our experimental results demonstrate that, while it is not surprising that BM3D essentially achieves the best denoising effect, GNL-means always performs better than NL-means, and better than LPG-PCA on average.

Index Terms— denoising, non-local means

1. INTRODUCTION

Image denoising is still a vibrant research topic. Its objective is to recover the original clean image from an observed noisy image. Typically a noisy image is modelled as $\boldsymbol{Z} = \boldsymbol{U} + \boldsymbol{V}$, where $\boldsymbol{Z}$ is the observed noisy image, $\boldsymbol{U}$ is the original clean image and $\boldsymbol{V}$ is noise. The most commonly assumed noise $\boldsymbol{V}$ in the literature is Additive White Gaussian Noise (AWGN) $\boldsymbol{V} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$.

The denoising process is to get an estimate of $\boldsymbol{U}$ from $\boldsymbol{Z}$, denoted by $\hat{\boldsymbol{Z}}$. Various methods have been proposed to remove AWGN and many of them could be classified as either *local* or *non-local*. Local methods exploit the local redundancy and estimate the denoised pixel based on the local information. Some of the popular local methods are bilateral filter [1] and directional filtering like steering kernel regression based [2]. Non-local methods achieve denoising by searching similar pixels that are not necessarily within the neighborhood. NL-means [3], LPG-PCA [4] and BM3D [5] are recent non-local methods that produce very impressive results.

NL-means is one of the first non-local methods. It first calculates weights for all pixels/patches in a selected window, where the weights exponentially decay in dissimilarities, and then denoises the current pixel/patch as a weighted average. More about NL-means will be explained in Section 2.

BM3D and LPG-PCA, inspired by the philosophy of NL-means, are regarded as among the most successful denoising methods [6]. BM3D is considered to be the best approach. It combines the non-local principle with classic algorithms, and consists of two iterations using hard-thresholding and Wiener filtering respectively. In each iteration, similar blocks are grouped and transformed into a new domain, and then the corresponding filter is used to separate the true clean signal from noise followed by an inverse 3D transform. LPG-PCA is competitive to BM3D. It combines the non-local principle with principal component analysis (PCA). It groups and converts patches with similar spatial structures into the PCA domain, and applies an LMMSE technique to separate the true clean signal from noise in the new domain. LPG-PCA is then iterated one more time.

Both BM3D and LPG-PCA are iterative. BM3D uses two iterations with different filters and the LPG-PCA procedure is iterated twice. The idea of this paper is similar: we also propose a two-stage generalized NL-means that iterates NL-means. The remainder of this paper is as follows: in Section 2, we explain how NL-means works. In Section 3, we show how NL-means can be extended to deal with non-i.i.d. noises and apply it on top of NL-means. In Section 4, we conduct experiments to demonstrate the performance of our method and compare it to NL-means, LPG-PCA and BM3D.

2. NL-means FILTER FOR IMAGE DENOISING

The basic principle of NL-means is simple and intuitive. It searches for similar pixels and estimates the true clean value as a weighted average, where the weights decay exponentially as the similarities decrease. More precisely, let U_i and

This work is supported in part by NSF grant CCF-1065305

V_i be the original pixel value and the added noise, for $i = 1, 2, \ldots, M$, where M is the number of pixels in the image, and hence the observed noisy pixel is $Z_i = U_i + V_i$. NL-means estimates U_i as

$$\hat{Z}_i = \frac{1}{C} \sum_{j \in N_i} W_{ij} Z_j,$$

where N_i is the search window centered at i, which could be as large as the whole image and usually chosen empirically, $\{W_{ij} \mid j \in N_i\}$ are the weights, and $C = \sum_{j \in N_i} W_{ij}$ is a normalization term.

The weight W_{ij} is defined as an increasing function of the similarity, or equivalently a decreasing function in some distance between U_i and U_j, which are usually unknown and Z_i and Z_j are used instead.

There are various distances that could be used. NL-means adopts the squared Euclidean distance and it has been demonstrated to be very effective. More precisely, the distance between two noisy pixels Z_i and Z_j is

$$D_{\text{pixel}}(Z_i, Z_j) \stackrel{\text{def}}{=} (Z_i - Z_j)^2 - 2\sigma^2.$$

Note that $\mathbb{E}\left[D_{\text{pixel}}(Z_i, Z_j)\right] = (U_i - U_j)^2 \geq 0$, however $D_{\text{pixel}}(Z_i, Z_j)$ may be ≤ 0.

To make the distance more robust to noise, NL-means extends the pixel comparison to patch comparison. Based on the observation that in natural images, similar patches tend to have similar centers, NL-means uses the following patch-based squared Euclidean distance:

$$D_{\text{patch}}(Z_i, Z_j) \stackrel{\text{def}}{=} \sum_{k=1}^{d} (\boldsymbol{Z}_i(k) - \boldsymbol{Z}_j(k))^2 - 2d\sigma^2,$$

where $\boldsymbol{Z}_i(k)$ and $\boldsymbol{Z}_j(k)$ are the pixels in the patches centered at the i-th and j-th pixels, respectively, and d is the number of pixels in a patch. Note that $\mathbb{E}\left[D_{\text{patch}}(Z_i, Z_j)\right] = \sum_{k=1}^{d} (\boldsymbol{U}_i(k) - \boldsymbol{U}_j(k))^2$.

NL-means uses an exponential kernel as the weight function:

$$W_{ij} \stackrel{\text{def}}{=} \exp\left(-\frac{\max\left\{D_{\text{patch}}(Z_i, Z_j), 0\right\}}{d\sigma^2 T^2}\right), \qquad (1)$$

where $d\sigma^2$ is for normalization, T is a decay parameter, and $\max$ is used so that the weight is set to 1 when the distance is negative.

The above described process denoises an image pixel by pixel. NL-means further extends it to patchwise implementation. Similar to the pixelwise process, a weight function is defined between two patches, but each patch is denoised as a weighted average of all patches centered in the search window of the first patch.

3. GENERALIZED NL-means

Compared to other non-local methods such as BM3D and LPG-PCA, NL-means has a poorer performance in both objective and subjective measures. As NL-means is a weighted average, it's also faced with a bias-variance dilemma. Several parameters would contribute to this, for example, if we choose a large search window, more similar pixels will help to reduce the variance, but more non-similar pixels, though with small weights, would introduce large bias as a whole. Similarly if we choose a large patch size, the similarity measure becomes more robust to noise, but variance is not reduced much since fewer pixels are given larger weights, etc.

Many papers have proposed to improve NL-means by decreasing the bias and/or the variance. For example, [7] proposed to decrease the bias using an adaptive search window, [8] set the parameters locally to find a bias-variance tradeoff, etc. Motivated by the iterative approaches in BM3D and LPG-PCA, we also propose a two-stage denoising process that iterates NLmeans. Given an image with i.i.d. Gaussian noises, the first stage is to use NL-means itself to obtain a filtered image. For the second stage, a direct application of NL-means is not feasible since the noises in the filtered image no long have the same variance and furthermore they are correlated. We modify NL-means to deal with non-i.i.d. noises, and apply it to the filtered image from the first stage.

3.1. Generalized weights

Recall that in NL-means the weight is calculated using the patch-based squared Euclidean distance D_{patch}. Notice that each pair of pixels $\boldsymbol{Z}_i(k)$ and $\boldsymbol{Z}_j(k)$ are independent Gaussian. Let

$$\Delta_{ij}(k) \stackrel{\text{def}}{=} \boldsymbol{Z}_i(k) - \boldsymbol{Z}_j(k).$$

Then $\Delta_{ij}(k)$ is also Gaussian and $\text{Var}(\Delta_{ij}(k)) = 2\sigma^2$. It follows that $D_{\text{patch}}(Z_i, Z_j)$ can be rewritten as

$$2\sigma^2 \sum_{k=1}^{d} \left[\frac{\Delta_{ij}(k)^2}{2\sigma^2} - 1\right] = 2\sigma^2 \sum_{k=1}^{d} \left[\frac{\Delta ij(k)^2}{\text{Var}(\Delta_{ij}(k))} - 1\right],$$

and hence W_{ij} can be reformulated as a generalized weight:

$$W_{ij}^G \stackrel{\text{def}}{=} \exp\left(-\frac{\max\left\{\sum_{k=1}^{d}\left[\frac{\Delta ij(k)^2}{\text{Var}(\Delta_{ij}(k))} - 1\right], 0\right\}}{dT^2/2}\right).$$

Note that the new formulation is still a well-defined weight function since it decreases in each $\Delta_{ij}(k) = \boldsymbol{Z}_i(k) - \boldsymbol{Z}_j(k)$. Particularly, if the patches centered at i and j are identical, then $\Delta_{ij}(k) = 0$ and $W_{ij}^G = 1$.

NL-means can be generalized by replacing W_{ij} with W_{ij}^G. We call this Generalized NL-means (GNL-means). Note that, comparing to NL-means, GNL-means does not require $\boldsymbol{Z}_i(k)$ and $\boldsymbol{Z}_j(k)$ to be independent, as long as we know how to calculate $\text{Var}(\Delta_{ij}(k))$.

3.2. Calculation of variances

Our idea is to further denoise the image by applying GNL-means to the image filtered by NL-means. Recall that $\hat{\boldsymbol{Z}}_i$ is the estimate of the true pixel value $\boldsymbol{U}_i$ by NL-means. Let

$$\hat{\Delta}_{ij}(k) \stackrel{\text{def}}{=} \hat{\boldsymbol{Z}}_i(k) - \hat{\boldsymbol{Z}}_j(k).$$

To apply GNL-means, we need to calculate $\mathrm{Var}(\hat{\Delta}_{ij}(k))$. In NL-means, since $\boldsymbol{Z}_i(k)$ and $\boldsymbol{Z}_j(k)$ are independent, $\mathrm{Var}(\Delta_{ij}(k))$ is trivially $2\sigma^2$. However since NL-means estimates each pixel with a weighted average over pixels in its search window, $\hat{\boldsymbol{Z}}_i(k)$ and $\hat{\boldsymbol{Z}}_j(k)$ could become correlated.

In general, we explain how to calculate $\mathrm{Var}(\hat{Z}_p - \hat{Z}_q)$ for any given p, q. Recall that, for any $i = 1, 2, \ldots, M$,

$$\hat{Z}_i = \sum_{\ell \in N_i} W'_{i\ell} Z_\ell = \sum_{\ell \in N_i} W'_{i\ell}(U_\ell + V_\ell),$$

where $W'_{i\ell}$ is normalized weight. For simplicity, let $\Delta U_{pq} \stackrel{\text{def}}{=} \sum_{\ell \in N_p} W'_{p\ell} U_\ell - \sum_{\ell \in N_q} W'_{q\ell} U_\ell$. We have

$$\begin{aligned}
&(\hat{Z}_p - \hat{Z}_q) - \Delta U_{pq} \\
&= \sum_{\ell \in N_p} W'_{p\ell} V_\ell - \sum_{\ell \in N_q} W'_{q\ell} V_\ell \\
&= \sum_{\ell \in N_p \setminus N_q} W'_{p\ell} V_\ell - \sum_{\ell \in N_q \setminus N_p} W'_{q\ell} V_\ell + \sum_{\ell \in N_p \cap N_q} (W'_{p\ell} - W'_{q\ell}) V_\ell.
\end{aligned}$$

Then the variance $\mathrm{Var}(\hat{Z}_p - \hat{Z}_q)$ is

$$\sum_{\ell \in N_p \setminus N_q} W'^2_{p\ell}\sigma^2 + \sum_{\ell \in N_q \setminus N_p} W'^2_{q\ell}\sigma^2 + \sum_{\ell \in N_p \cap N_q} (W'_{p\ell} - W'_{q\ell})^2 \sigma^2.$$

It follows that

$$\mathrm{Var}(\hat{Z}_p - \hat{Z}_q) = \sigma^2 S_p + \sigma^2 S_q - 2 \sum_{\ell \in N_p \cap N_q} W'_{p\ell} W'_{q\ell} \sigma^2,$$

where

$$S_p = \sum_{\ell \in N_p} W'^2_{p\ell}, \text{ and } S_q = \sum_{\ell \in N_q} W'^2_{q\ell}.$$

Given any i, j, the above equation can be used to compute $\mathrm{Var}(\hat{\Delta}_{ij}(k))$ for all $1 \le k \le d$. The computation could be accelerated by pre-computing S_p for all $p = 1, 2, \ldots, M$. However, it seems that the summation over the intersection $N_p \cap N_q$ has to be calculated individually for each pair of windows N_p and N_q. To increase the efficiency, we use the following estimator $\widehat{\mathrm{Var}}(\hat{Z}_p - \hat{Z}_q) = \sigma^2 S_p + \sigma^2 S_q$. In our experiments this estimator works slightly worse than the exact variance, but much faster.

3.3. GNL-means algorithm

Our GNL-means consists of two stages. The first stage is simply to use the original NL-means to filter the noisy image.

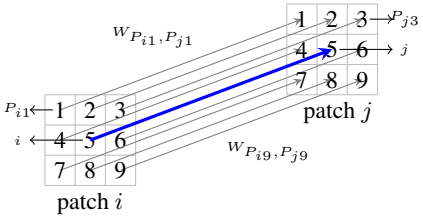

Fig. 1. Patchwise weights

The weights $\boldsymbol{W} = (W_{ij})_{M \times M}$ by NL-means, where W_{ij} is defined in Equation (1), are retained for later use.

If patchwise NL-means is used, the weights are calculated for patches instead of pixels, and all pixels within the same patch are denoised simultaneously. The number of estimates for each pixel is the number of patches that cover it; the final denoised value is the average of all these estimates.

It is easy to see that in patchwise NL-means the final denoised value of a pixel is still a weighted average. The actual weights are the average for the patches that cover the pixel. Note that the centers of such patches are exactly the pixels within the patch centered at this pixel. More precisely, let $\boldsymbol{P}_i = (P_{i1}, P_{i2}, \ldots, P_{ik})$ be the patch centered at i. Then the patchwise weight is

$$W^{\text{patch}}_{ij} \stackrel{\text{def}}{=} \frac{1}{d} \sum_{k=1}^{d} W_{P_{ik},P_{jk}}.$$

For illustration, an example is shown in Figure 1 for $d = 9$. The corresponding patchwise weight is

$$W^{\text{patch}}_{ij} = \left(W_{P_{i1},P_{j1}} + W_{P_{i2},P_{j2}} + \cdots + W_{P_{i9},P_{j9}}\right)/9.$$

In the second stage, as described in Algorithm 1, we first compute the variance $\mathrm{Var}(\hat{Z}_p - \hat{Z}_q)$ for all pairs of pixels p and q. Note that we don't need to compute $\mathrm{Var}(\hat{Z}_p - \hat{Z}_q)$ if q is not in the search window N_p (or p is not in the search window N_q by symmetry). Then we compute the generalized weights and denoise the pixels by taking the new weighted average.

Analogous to the patchwise NL-means, the second stage can also denoise patches instead of pixels. The multiple estimates for each pixel can be finally averaged to build the final output. Similarly, the final weight of pixel i is

$$W^{G,\text{patch}}_{ij} \stackrel{\text{def}}{=} \frac{1}{d} \sum_{k=1}^{d} W^{G}_{P_{ik},P_{jk}}.$$

4. SIMULATION RESULTS AND DISCUSSION

In this section we present the simulation results on several benchmark images and compare the denoising performance to other methods.

Algorithm 1 GNL-means algorithm

Input: $\boldsymbol{Z}$: image with AWGN
Output: $\tilde{\boldsymbol{Z}}$: denoised image

1. $\hat{\boldsymbol{Z}} \leftarrow$ filtered $\boldsymbol{Z}$ using NL-means
2. **if** NL-means is pixelwise **then**
3. Weight matrix: $\boldsymbol{W} \leftarrow (W_{ij})_{M\times M}$
4. **else** {NL-means is patchwise}
5. Weight matrix: $\boldsymbol{W} \leftarrow (W_{ij}^{\text{patch}})_{M\times M}$
6. **end if**
7. New variance vector: $\boldsymbol{S} \leftarrow (\sigma^2 S_p)_{M\times 1}$
8. Variance matrix: $\boldsymbol{V} \leftarrow \left(\widehat{\text{Var}}(\hat{Z}_p - \hat{Z}_q)\right)_{M\times M}$
9. **if** *pixelwise* **then**
10. New weight matrix: $\boldsymbol{W}^{\text{new}} \leftarrow (W_{ij}^{G})_{M\times M}$
11. **else** {*patchwise*}
12. New weight matrix: $\boldsymbol{W}^{\text{new}} \leftarrow (W_{ij}^{G,\text{patch}})_{M\times M}$
13. **end if**
14. Weighted average: $\tilde{Z}_i \leftarrow \sum_{j\in N_i} W_{ij}^{\text{new}} \hat{Z}_j$
15. Output $\tilde{\boldsymbol{Z}} = (\tilde{Z}_i)_{M\times 1}$

	Noise		$0 < \sigma \le 15$	$15 < \sigma \le 30$
NL-means	Patch		3×3	5×5
	Window		21×21	21×21
	Decay		0.4	0.4
GNL-means	Patch	Stage 1	5×5	7×7
		Stage 2	3×3	3×3
	Window	Stage 1	21×21	21×21
		Stage 2	21×21	21×21
	Decay	Stage 1	0.5	0.4
		Stage 2	1.3	1.0

Table 1. Parameters for NL-means and GNL-means

4.1. Parameter selection

Buades et al. [9] provide some heuristic and empirical ways for NL-means: the patch size d and search window size N increase as the noise standard deviation σ increases, the decaying parameter T decreases as the patch size increases. We basically follow the same strategy. However, since the second step performs better with smaller bias from the first step, we choose the parameters differently than the one-step NL-means, for example the patch size is larger. Table 1 gives the detailed parameters for both steps. Note that the parameters for NL-means are also shown in Table 1. They are from the authors' webpage [9], and claimed to work well for various images.

4.2. Results

We implement patchwise GNL-means in Matlab and compare it with patchwise NL-means, LPG-PCA and BM3D. The source codes for BM3D and LPG-PCA are obtained from the authors' websites. BM3D has two profiles and for fair comparison we use the normal profile which produces better results. We also note that LPG-PCA excludes the boundary pixels of width 20 prior to objective testing such as PSNR and SSIM, thus the same applies to other methods.

GNL-means is tested on six benchmark images *Lena*, *House*, *Cameraman*, *Monarch*, *Peppers*, and *Barbara*. Table 2 gives the PSNR and SSIM from GNL-means as well as those from NL-means, LPG-PCA and BM3D.

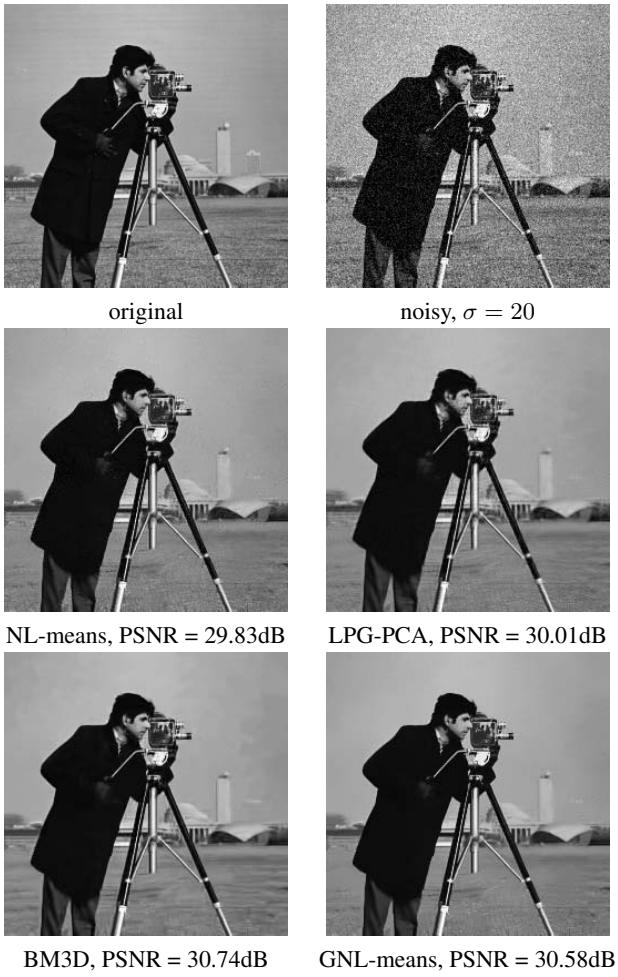

Fig. 2. Comparison of Visual Quality

In Table 2 the best results are shown in bold face, while the second best are shown in blue. As observed, it is not surprising that BM3D achieves the best results. GNL-means is a big improvement over NL-means, and has better results than LPG-PCA on average. Our method follows BM3D and performs competitively well.

For visual quality comparison, we only show the denoised results for one image due to limited space. See magnified Figure 2. GNL-means removes the noise significantly but reconstructs some fine details, even though the overall PSNR is slightly lower than BM3D. For other images and also results for more noise variances, please go to the webpage:

		NL-means	LPG-PCA	BM3D	GNL-means
Lena	$\sigma = 10$	32.84 (0.9053)	33.69 (0.9260)	**33.92** (**0.9277**)	33.32 (0.9179)
	$\sigma = 20$	29.39 (0.8355)	29.95 (0.8582)	**30.26** (**0.8693**)	29.81 (0.8550)
	$\sigma = 30$	27.18 (0.7632)	27.77 (0.7988)	**28.32** (**0.8226**)	27.83 (0.8035)
House	$\sigma = 10$	34.67 (0.8835)	35.79 (0.9112)	**36.21** (**0.9146**)	35.50 (0.8960)
	$\sigma = 20$	31.88 (0.8253)	32.55 (0.8485)	**33.29** (**0.8581**)	32.80 (0.8492)
	$\sigma = 30$	29.79 (0.7627)	30.68 (0.8056)	**31.81** (**0.8335**)	31.00 (0.8188)
Cameraman	$\sigma = 10$	33.65 (0.9161)	33.93 (0.9363)	**34.41** (**0.9402**)	34.11 (0.9337)
	$\sigma = 20$	29.83 (0.8576)	30.01 (0.8790)	**30.74** (**0.8996**)	30.58 (0.8903)
	$\sigma = 30$	27.82 (0.7908)	27.94 (0.8269)	**28.70** (**0.8635**)	28.57 (0.8542)
Monarch	$\sigma = 10$	33.17 (0.9336)	33.79 (0.9546)	33.88 (**0.9572**)	**33.93** (0.9540)
	$\sigma = 20$	29.25 (0.8839)	29.84 (0.9100)	**30.15** (**0.9220**)	30.06 (0.9132)
	$\sigma = 30$	26.98 (0.8225)	27.61 (0.8629)	**28.15** (**0.8877**)	27.76 (0.8689)
Peppers	$\sigma = 10$	33.61 (0.8987)	34.24 (0.9186)	**34.74** (**0.9241**)	34.39 (0.9188)
	$\sigma = 20$	30.46 (0.8428)	30.89 (0.8656)	**31.54** (**0.8851**)	31.14 (0.8713)
	$\sigma = 30$	28.13 (0.7763)	28.60 (0.8146)	**29.51** (**0.8479**)	28.85 (0.8281)
Barbara	$\sigma = 10$	32.78 (0.9243)	**34.79** (0.9529)	34.59 (**0.9535**)	33.85 (0.9442)
	$\sigma = 20$	29.55 (0.8645)	30.79 (0.8967)	**31.02** (**0.9075**)	30.24 (0.8882)
	$\sigma = 30$	27.23 (0.7858)	28.46 (0.8391)	**28.92** (**0.8591**)	28.00 (0.8270)

Table 2. PSNR and SSIM (value in the parenthesis) results

`http://videoprocessing.ucsd.edu/~eluo/projects/denoising`.

5. CONCLUSION

This paper proposed Generalized NL-means filter (GNL-means) that can be used for both i.i.d. and non-i.i.d. noises, which implies the nomenclature "generalized". We applied it to denoise the NL-means filtered image, whose noises are non-i.i.d., and we also showed how to estimate the non-i.i.d. noise variances. This process is a two-stage or iterative image denoising using NL-means and the experimental results demonstrate that the proposed method yields competitive performance as the state-of-the-art denoising methods. In the future we would test GNL-means directly on noisy images with non-i.i.d. noises, for example Multiplicative White Gaussian Noise (MWGN).

6. REFERENCES

[1] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the Sixth International Conference on Computer Vision*, ser. ICCV '98. Washington, DC, USA: IEEE Computer Society, 1998.

[2] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *Image Processing, IEEE Transactions on*, vol. 16, no. 2, pp. 349 –366, feb. 2007.

[3] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Simul*, vol. 4, pp. 490–530, 2005.

[4] L. Zhang, W. Dong, D. Zhang, and G. Shi, "Two-stage image denoising by principal component analysis with local pixel grouping," *Pattern Recogn.*, vol. 43, pp. 1531–1549, April 2010.

[5] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, and S. Member, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE TRANS. IMAGE PROCESS*, vol. 16, p. 2007, 2007.

[6] A. Buades, B. Coll, and J.-M. Morel, "Self-similarity-based image denoising," *Commun. ACM*, vol. 54, pp. 109–117, May 2011.

[7] C. Kervrann and J. Boulanger, "Unsupervised patch-based image regularization and representation," in *Proc. Eur. Conf. Comp. Vis. (ECCV06)*, 2006, pp. 555–567.

[8] V. Duval, J.-F. Aujol, and Y. Gousseau, "A bias-variance approach for the nonlocal means." *SIAM J. Imaging Sciences*, vol. 4, no. 2, pp. 760–788, 2011.

[9] A. Buades, B. Coll, and J. M. Morel, "Non-local means denoising." [Online]. Available: `http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/`