Fast external denoising using pre-learned transformations

Shibin Parameswaran
UC San Diego
sparames@ucsd.edu

Enming Luo
The Facebook
elou@fb.com

Charles-Alban Deledalle
CNRS, Univ Bordeaux
UC San Diego
cdeledalle@eng.ucsd.edu

Truong Q. Nguyen
UC San Diego
tqn001@eng.ucsd.edu

Abstract

We introduce a new external denoising algorithm that utilizes pre-learned transformations to accelerate filter calculations during runtime. The proposed fast external denoising (FED) algorithm shares characteristics of the powerful Targeted Image Denoising (TID) and Expected Patch Log-Likelihood (EPLL) algorithms. By moving computationally demanding steps to an offline learning stage, the proposed approach aims to find a balance between processing speed and obtaining high quality denoising estimates. We evaluate FED on three datasets with targeted databases (text, face and license plates) and also on a set of generic images without a targeted database. We show that, like TID, the proposed approach is extremely effective when the transformations are learned using a targeted database. We also demonstrate that FED converges to competitive solutions faster than EPLL and is orders of magnitude faster than TID while providing comparable denoising performance.

1. Introduction

The advances in camera technology and storage has led to an explosive growth in the amount of images and videos captured for personal and professional uses such as for sharing on social media sites or security purposes. Quite often, data is captured in poor conditions. Hence, there is a constant demand for *efficient* restoration algorithms that can post-process and correct for degradations in images and videos captured by unideal sensors and/or in unideal environmental conditions. Denoising is such an important restoration method that aims to reconstruct underlying clean signals (images or videos) from observed noisy signals.

In the last decade, the field of image denoising has seen tremendous advancements. One of the innovations introduced during this time is patch-based denoising. These methods divide an image into overlapping patches and remove noise by applying a denoising filter on a patch-by-patch basis. Patch-based denoising methods can be broadly divided into two classes – internal and external methods. Internal methods are those algorithms that rely exclusively on the information contained within an image when designing the denoising filter. Popular methods that fall into this category are non-local means (NLM) [3], BM3D [6], k-SVD [8], SAFIR [2], NL-Bayes [13], DDID [12] and LPG-PCA [25]. To denoise a patch in the noisy image (commonly referred to as a *query* patch), these methods rely on information contained in *reference* patches (i.e. other patches that are similar to the query patch) within the noisy image itself. Due to this reason, internal methods often suffer when the noise level in an image is high and also when denoising rare patches due to the lack of an adequate number of reference patches [7]. To ease these problems, external denoising methods were developed that restore noisy patches using information from outside the given noisy image. Methods in this category have taken many different approaches of leveraging external information. For instance, the Expected Patch Log Likelihood (EPLL) algorithm [27] denoises patches using a Gaussian Mixture Model (GMM) prior learned on an external database of clean patches. Other methods include learning mapping functions to map noisy input patches to clean ones using multi-layer perceptrons (MLP) [4], piecewise linear estimators [23, 20] and straightforward adaptations of NLM, BM3D and k-SVD [8, 26, 15].

Recently, Luo *et al.* [14, 15] proposed *targeted* image denoising (TID) that utilizes *targeted* external databases. This follows from the observation that a large generic database does not necessarily contain enough useful information (and may often contain irrelevant information) needed to denoise the noisy image of a specific domain. Hence, instead of using a generic database, TID obtains reference patches from a targeted database that contains images *relevant* to the noisy image. For example, while denoising a face image, a targeted database is made up of patches from other face images whereas while processing a license plate image a targeted database will be constructed using license plate images only. This approach of using targeted databases tailored to particular domains and tasks can be viewed in a similar category as a straightforward adapta-

tion of task-driven image processing methods [16].

Although they alleviate the problems of the internal methods, external (both generic and targeted) methods are either slower to converge to an acceptable solution or are computationally expensive. EPLL [27] is one of the fastest external denoising algorithms but it is slow to produce a good quality solution requiring many iterations with heavily overlapped patches. Additionally, in each iteration of EPLL, one has to calculate the patch log-likelihood of the patch under the learned GMM prior. This involves calculating Mahalanobis distance using the covariance matrices of each mixture component which is computationally more complex than calculating Euclidean distances. The TID filter produces a higher quality image within 2-3 iterations but is computationally expensive. Depending on the size of the image and the database used, TID processing time is usually orders of magnitude slower than EPLL. These issues make these successful external denoising methods unsuitable for processing videos and large images.

In this work, we propose a fast external denoising algorithm that we refer to as FED. This algorithm is efficient during runtime and can be trained with ease and flexibility.

2. Background and Related Work

The proposed approach falls under the external denoising category. An external denoising algorithm uses information from outside the noisy image when estimating the denoising filter. In this section we provide a brief overview of two external denoising algorithms that inspired the development of our method.

2.1. Targeted Image Denoising Filter

Targeted Image Denoising (TID) is an external denoising algorithm that utilizes a *targeted* database of clean patches [14, 15]. A *targeted* database is chosen using prior knowledge of the scene depicted in the image being denoised.

TID designs optimal denoising filters for each noisy patch in the image by utilizing the given targeted database maximally. To this end, for each noisy patch, TID retrieves clean patches similar to the noisy patch and obtains an optimal denoising filter for it by solving a group sparsity minimization problem and using a localized Bayesian prior [15].

The data-adaptive nature of TID makes it very effective. However, designing a new filter for each patch during runtime can lead to excessive computational complexity as the size of the noisy image increases. The complexity of TID is also dependent on the size of the database and the number of similar patches used for designing the filter. This makes TID unsuitable for large images and videos.

2.2. Expected Patch Log Likelihood

Expected Patch Log Likelihood (EPLL) [27] is another successful external denoising algorithm. EPLL models the patch priors using Gaussian Mixture Models from an external database of *generic* patches. These learned patch priors are used to denoise images efficiently. This simple yet powerful algorithm has been shown to outperform both internal denoising algorithms such as BM3D [6] and external denoising algorithms such as external-NLM. The proposed method is similar to the EPLL algorithm, however the denoising filter proposed in this paper is more powerful when a targeted database is available.

2.3. Generic vs. targeted database of patches

As metioned above, TID uses an external database composed of patches from images that are visually similar to the noisy image that are being denoised. Such a database that closely resembles the characteristics of the test (noisy) image is referred to as the targeted database. Whereas, the Gaussian Mixture Models in the original EPLL study is learned from a large database of patches (2×10^6 patches) taken from images in the Berkeley Segmentation Database [18]. A patch database that contains patches from a wide range of images from various domains is termed a generic database. In general, using a targeted database is better than relying on a generic database.

Choosing an appropriate database with *relevant* images can be a challenging problem in itself and is an active research area. This direction of research requires a thorough evaluation of recent developments in image retrieval and grouping algorithms such as deep learning for image retrieval [11] and web scale photo clustering [10]. However, a careful evaluation of database construction methods is out of the scope of this study and we leave it for future work. Although our algorithm is designed to be used with a targeted database, we also include results from experiments using a generic database to give the reader an idea of the performance of our algorithm in this unideal setting.

3. Proposed Method

In this section, we describe the proposed denoising approach. It is based on a whole image denoising formulation that is regularized by targeted patch-based denoising estimate. First, let us define a model for the noisy image as $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{\eta} \in \mathbb{R}^N$ where $\boldsymbol{\eta}$ is the i.i.d. Gaussian noise with zero-mean and variance σ^2, i.e., $\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_N)$. Let us also define a patch extractor operator $\boldsymbol{P}_i \in \mathbb{R}^{d \times N}$ that extracts i-th patch from $\boldsymbol{x}$, i.e., $\boldsymbol{P}_i \boldsymbol{x} \in \mathbb{R}^d$ is a d-dimensional vector obtained by lexicographic ordering of $\sqrt{d} \times \sqrt{d}$ patch extracted from the image $\boldsymbol{x}$.

3.1. Whole image denoising

Given a noisy image $\boldsymbol{y}$, the noise variance σ^2, and a set of targeted image patches $\boldsymbol{A} \in \mathbb{R}^{d \times k}$ of k patches, we propose to optimize the following cost function to estimate the

underlying clean image, $\boldsymbol{x}$.

$$\min_{x,\{\boldsymbol{z}_i\}} \frac{1}{2\sigma^2}\|\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \frac{\beta}{2}\left[\sum_{i=1}^{N}\|\boldsymbol{P}_i\boldsymbol{x}-\boldsymbol{A}\boldsymbol{z}_i\|_2^2 + \lambda\|\boldsymbol{z}_i\|_2^2\right] \quad (1)$$

The above formulation ensures that the estimated image as a whole closely resembles the original noisy image $\boldsymbol{y}$ (first term), and the individual patches of the estimated image can be expressed as a linear combination of the patches in the patch-matrix $\boldsymbol{A}$ with minimum error. The optimization parameter β controls the relative contribution of the patch-by-patch reconstruction strategy of the second term to our overall goal of denoising the whole image, and the parameter λ ensures the existence of feasible solutions for the set of coefficient vectors $\{\boldsymbol{z}_i | i = 1 \dots N\}$.

The solution for the minimization in eq. (1) can be carried out by alternating between: 1. Fixing $\boldsymbol{x}$ and finding optimal set of $\{\boldsymbol{z}_i\}$ using the assumption that $\boldsymbol{z}_i$'s are mutually independent and 2. Fixing $\{\boldsymbol{z}_i\}$ and solving for $\boldsymbol{x}$. We defer the details of the first of these two steps (finding optimal $\{\boldsymbol{z}_i\}$) to Section 3.2 below. Assuming we have the optimal solutions for all coefficient vectors $\{\boldsymbol{z}_i | i = 1 \dots N\}$, we will first discuss solving for the optimal $\boldsymbol{x}$.

Fixing all $\{\boldsymbol{z}_i\}$ and solving for optimal $\boldsymbol{x}$ leads to

$$\widehat{\boldsymbol{x}} = \left(\frac{1}{\sigma^2}\boldsymbol{I}_N + \beta\sum_{i=1}^{N}\boldsymbol{P}_i^T\boldsymbol{P}_i\right)^{-1}\left(\frac{1}{\sigma^2}\boldsymbol{y} + \beta\sum_{i=1}^{N}\boldsymbol{P}_i^T\boldsymbol{A}\boldsymbol{z}_i\right) \quad (2)$$

where the matrix $\boldsymbol{P}_i^T$ performs the complementary operation of $\boldsymbol{P}_i$ of placing a patch back as the i-th patch of the image, and $\sum_{i=1}^{N}\boldsymbol{P}_i^T\boldsymbol{P}_i$ counts the number of estimates obtained for each pixel in the image. Therefore, the solution of the whole image denoising shown in eq. (2) is simply a weighted average of the given noisy image and the image estimate obtained by denoising individual patches independently.

3.2. Patch denoising

We now focus on the solution for optimal $\{\boldsymbol{z}_i\}$ obtained by fixing $\boldsymbol{x}$. The optimal $\boldsymbol{z}_i$ can be found by solving the following minimization problem for each i separately:

$$\min_{\boldsymbol{z}_i}\|\boldsymbol{P}_i\boldsymbol{x}-\boldsymbol{A}\boldsymbol{z}_i\|_2^2 + \lambda\|\boldsymbol{z}_i\|_2^2 \quad (3)$$

Here, the second term is added for regularization purposes and to ensure a unique solution. The solution of eq. (3) has a closed-form expression which leads to estimate $\boldsymbol{A}\widehat{\boldsymbol{z}}_i$ as:

$$\boldsymbol{A}\widehat{\boldsymbol{z}}_i = \boldsymbol{A}\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{I}_k\right)^{-1}\boldsymbol{A}^T\boldsymbol{P}_i\boldsymbol{x} \quad (4)$$

If we let $(\boldsymbol{U}, \boldsymbol{D}, \boldsymbol{V})$ be the singular value decomposition (SVD) of $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$, then eq. (4) simplifies to:

$$\boldsymbol{A}\widehat{\boldsymbol{z}}_i = \boldsymbol{U}\frac{\boldsymbol{D}^2}{\boldsymbol{D}^2+\lambda\boldsymbol{I}_d}\boldsymbol{U}^T\boldsymbol{P}_i\boldsymbol{x} \quad (5)$$

where the division must be read as an element-wise division of diagonal elements. By noticing that $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{U}\boldsymbol{D}^2\boldsymbol{U}^T$, then we can rewrite eq. (5) as:

$$\boldsymbol{A}\widehat{\boldsymbol{z}}_i = \underbrace{\boldsymbol{U}\frac{\boldsymbol{S}}{\boldsymbol{S}+\lambda\boldsymbol{I}_d}\boldsymbol{U}^T}_{\text{Denoising Filter}}\underbrace{\boldsymbol{P}_i\boldsymbol{x}}_{\text{Patch}} \quad (6)$$

where $\boldsymbol{U}$ and $\boldsymbol{S}$ are obtained via the eigen-decomposition of $\boldsymbol{A}\boldsymbol{A}^T$. Repeating the above steps for all $i = \{1 \dots N\}$ will provide denoised estimates for all of the patches in a given noisy image. Here, the parameter λ is chosen such that it is proportional to the noise variance in $\boldsymbol{y}$ (which is assumed to be known *a priori*.)

3.2.1 Proposed choice for $\boldsymbol{A}$ matrix

Note that in eq. (3), we have opted for an ℓ_2 constraint over an ℓ_1 constraint to facilitate a closed form solution to our optimization problem. This choice is driven to ensure computational efficiency. However, the ℓ_2-norm does not promote sparsity and hence the reconstruction accuracy depends heavily on the quality and relevance of patches in matrix $\boldsymbol{A}$. Therefore, it is not ideal to use the entire targeted database in place of $\boldsymbol{A}$. A valid alternate option is to set matrix $\boldsymbol{A} = [\boldsymbol{p}_1, \dots, \boldsymbol{p}_m]$ where $\boldsymbol{p}_{1:m}$ are the m closest neighbors of $\boldsymbol{P}_i\boldsymbol{x}$. According to Luo et al. [15], this option leads to the optimal usage of the provided external database and will yield a filter similar to the TID filter [14, 15] (the main difference being the absence of the whole image denoising part in TID). On the contrary, it adds an undesirable amount of computational complexity during runtime.

To alleviate these issues, we propose to find a set of anchor patches, $\{\boldsymbol{a}_1 \dots \boldsymbol{a}_k\}$, to represent the entire targeted database. If the targeted database can be clustered in k groups, then each anchor patch can be thought of as the representative of one of these clusters. Then, a patch dictionary is created for each of the anchor patches using their respective m nearest neighbors. By taking the eigenvalue decompositions of these k patch matrices, we can calculate the corresponding $\boldsymbol{U}_i$ and $\boldsymbol{S}_i$ matrices, for all $i \in 1 \dots k$. During runtime, a noisy query patch, $\boldsymbol{q}$, can be denoised using $\boldsymbol{U}_i$ and $\boldsymbol{S}_i$ that correspond to the anchor patch, $\boldsymbol{a}_i$, that is most similar to $\boldsymbol{q}$. Since the anchor points are independent of the noisy image, finding anchor patches and calculating their corresponding denoising filters can be carried out off-line; thus avoiding delays during runtime.

In practice, the fidelity of the data matrix $\boldsymbol{A}$ can be improved by introducing weight matrix, i.e, by replacing $\boldsymbol{A}$ with $\boldsymbol{A}\boldsymbol{W}^{1/2}$ [14, 15]. Here the weight matrix $\boldsymbol{W} = \frac{1}{\alpha}\,\text{diag}\{w_1, w_2, \dots, w_n\}$ where $w_i = \exp\left(-\frac{\|\boldsymbol{a}-\boldsymbol{p}_i\|^2}{h^2}\right)$ for some user-tunable bandwidth parameter h, and α is a normalization parameter so that the weights add up to 1. This modification also diminishes the adverse effect of irrelevant

neighbors (that are far away from the anchor patches) on the learned denoising filters. Matrices $\boldsymbol{U}$ and $\boldsymbol{S}$ are obtained off-line from the eigen-decomposition of $\boldsymbol{AWA}^T$.

3.3. Offline training and iterations

Thanks to anchor patches, the process of designing denoising filters is decoupled from the actual denoising step. This enables us to split the proposed algorithm into an off-line training stage and online denoising stage. The proposed denoising algorithm is summarized in Algorithm 1.

Algorithm 1 Fast External Denoising

TRAINING PHASE (Offline)

Input:

$\boldsymbol{D}$:	Patch database
k:	No. of anchors,
m:	Max no. of neighbors,
h:	bandwidth parameter

Output:

$\boldsymbol{a}_1 \dots \boldsymbol{a}_k$:	Anchor patches
$\{(\boldsymbol{U}_1, \boldsymbol{S}_1) \dots\}$:	Eigen-decomposition

1: Find k anchor patches from $\boldsymbol{D}$ ▷ *k-means clustering*
2: **for each** $\boldsymbol{a}_i \in \{\boldsymbol{a}_1 \dots \boldsymbol{a}_k\}$ **do**
3: Find m nearest neighbors from $\boldsymbol{D} \rightarrow (\boldsymbol{p}_1 \dots \boldsymbol{p}_m)$
4: Form matrix $\boldsymbol{A} = [\boldsymbol{p}_1 \dots \boldsymbol{p}_m]$
5: Form weights: $\boldsymbol{W} = \frac{1}{\alpha}\,\text{diag}\{\boldsymbol{w}_1, \dots, \boldsymbol{w}_m\}$
where $w_j = \exp\left(-\frac{\|\boldsymbol{a}_i - \boldsymbol{p}_j\|^2}{h^2}\right)$ and $\alpha = \Sigma_j w_j$.
6: Compute eigen-decomposition of weighted matrix:

$$[\boldsymbol{U}_i, \boldsymbol{S}_i] = \text{eig}\left(\boldsymbol{AWA}^T\right)$$

7: **return** $\{\boldsymbol{a}_1 \dots \boldsymbol{a}_k\}$ and $\{(\boldsymbol{U}_1, \boldsymbol{S}_1) \dots (\boldsymbol{U}_k, \boldsymbol{S}_k)\}$

DENOISING PHASE (runtime)

Input:

$\boldsymbol{y}$:	noisy image
σ^2:	noise variance
$\boldsymbol{a}_1 \dots \boldsymbol{a}_k$:	Anchor patches
$\{(\boldsymbol{U}_1, \boldsymbol{S}_1) \dots\}$:	Eigen-decomposition

Output:

$\hat{\boldsymbol{x}}$:	Denoised image estimate

1: **for each** noisy patch, $\boldsymbol{q} \in \boldsymbol{y}$, **do**
2: Find the index i of the closest patch $\boldsymbol{a}_i$ to $\boldsymbol{q}$
3: Compute the shrinkage matrix:

$$\boldsymbol{\Lambda} = \left(\text{diag}\left(\boldsymbol{S}_i + \sigma^2 \boldsymbol{I}_d\right)\right)^{-1} \text{diag}(\boldsymbol{S}_i)$$

4: Perform patch denoising $\hat{\boldsymbol{p}} = \boldsymbol{U}_i \boldsymbol{\Lambda} \boldsymbol{U}_i^T \boldsymbol{q}$
5: Perform whole image denoising using eq. (2)
6: **return** denoised image $\hat{\boldsymbol{x}}$

For best results, each query patch should be matched to an anchor patch that most closely resembles the underlying clean patch. Since at first, the matching is carried out with noisy patch itself, it is recommended that the runtime algorithm is repeated for more than one iteration. Each subsequent iteration uses patches from the previous image estimate to find anchor patches and performs the whole image denoising with progressively larger values of β in eq. (2).

4. Experimental Results

We present denoising results on three grayscale image datasets containing text images, face images and license plate images, respectively. We compare the performance of the proposed approach against leading fast denoising algorithms in both internal and external denoising categories. To represent the internal denoising category, we choose BM3D [6] as it is the fastest and most popular internal denoising algorithm. For external denoising, we compare the performance with EPLL [27] trained on both generic and targeted priors. We also compare our results with TID to demonstrate the trade-off between speed-up vs. quality achieved by the proposed method. We tested these algorithms in low-, mid- and high-noise settings by varying the variance (σ^2) of the additive Gaussian noise from $\left(\frac{20}{255}\right)^2$ to $\left(\frac{80}{255}\right)^2$.

Algorithm settings and parameters

In all of our experiments, the size of the patches used is set to 8×8 and Euclidean metric is used to measure patch-wise (dis)similarity. For BM3D[1] and EPLL[2], we used the implementations provided by their respective authors.

The proposed algorithm is repeated for 3 iterations with a patch overlap of 4, 6 and 7 pixels in the first, second and third iterations, respectively (i.e. stride lengths $N_s = [4, 2, 1]$). For EPLL and BM3D, we use the default parameters and number of iterations prescribed by the original authors in their corresponding implementations. In the cases of EPLL and TID, we also show the results obtained and time taken by these algorithms when the patch overlap parameter and the number of iterations are matched to FED. For the three-iterations-EPLL, a 200 component GMM is learned on the targeted database (tar-EPLL3) and the parameters λ and β that gave the best performance on the validation sets were selected. These corresponded to $\lambda = \frac{N}{\sigma^2}$ and $\beta = \frac{1}{\sigma^2}[1, 4, 16]$.

FED parameter selection: number of anchors, size of $\boldsymbol{A}$ matrix and weighting

We use k-means clustering algorithm to identify a predetermined number (k) of anchor patches from the database. Other methods relying on dictionary learning [1, 17] or high

[1] BM3D: http://www.cs.tut.fi/~foi/GCF-BM3D/
[2] EPLL: https://people.csail.mit.edu/danielzoran/epllcode.zip

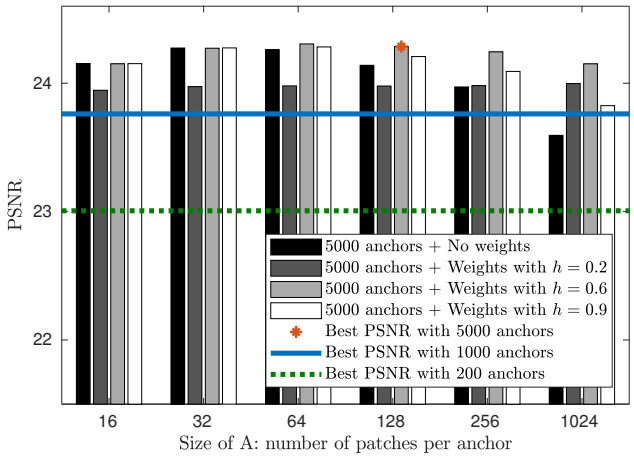


Figure 1. The effect of size of patch matrix, number of anchors, and weighting of the patch matrix

dimensional regression trees [9] can also be used for finding anchor patches from the database. For simplicity, we chose to cluster the database into k clusters and use the means of each cluster as our anchor patches. The number of anchor patches k is a parameter that can be set via cross validation.

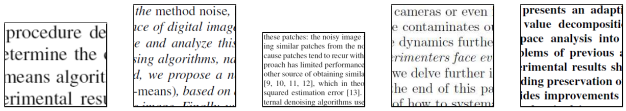
Once the anchor patches are identified, the next parameter that we have to set is the number of neighbors (m) for each anchor patch. The chosen neighbors form the respective data matrices $\boldsymbol{A}_i$ for the anchors $\boldsymbol{a}_i$. As mentioned above, we can weigh each neighbor selected for forming the $\boldsymbol{A}$ matrix with a weight based on its similarity to its corresponding anchor patch.

Figure 1 illustrates the average denoising performance obtained on the validation datasets of text, face and license image datasets with varying number of anchor patches (k), the effect of the weighting and different number of neighbors (m) included in an anchor's $\boldsymbol{A}$ matrix. The bar graph shows the variations in performance when k is set to 5000 and m is varied from 16 to 1024 and the $h = [0.2, 0.6, 0.9]$ in the weighting matrix (used in $\boldsymbol{AWA}^T$) or with no weighting. For brevity, we only display the best results obtained for $k = 200$ and $k = 1000$ with above mentioned values of m and h. For the three datasets with targeted databases, the best average PSNR is obtained when $k = 5000$, $m = 128$ and $h = 0.6$.

In the following, we will present the denoising results and discuss the characteristics of the datasets we used.

4.1. Text denoising

Text image dataset [15] contains images cropped from documents. These images have a clean white background with black text and display simple edge structures and almost no texture. The test image to be denoised and database images vary in font styles and sizes. Some examples from



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

Figure 2. Sample images from text image dataset [15]. Experiments are conducted on noise corrupted versions of image 2(a). The images 2(b)–2(e) are four examples out of nine clean text images that are used as the targeted database.

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

Figure 3. Sample images from face image dataset [19]. Figure 3(a) shows one of ten test images that were used in our experiments. The images 3(b)–3(e) are four examples out of 80 face images that make up the targeted database.

the text image dataset are shown in Figure 2. Out of 14 images in our dataset, the targeted database is created using nine images, four are used as a validation set and one image is used for testing. Since the denoising experiments reported for this dataset are conducted on only one image, we average our results over 5 independent trials with different noise realizations for each noise level. This is a simple setting to show the effect of a near perfect targeted database.

Table 1 reports the quantitative results in terms of PSNR and SSIM [21] of the proposed FED algorithm compared to other competing fast denoising algorithms. FED algorithm trained on a targeted database provides much higher PSNR and SSIM than BM3D and EPLL in all the noise level settings tested. The EPLL algorithm trained on the targeted dataset and run with default parameters (5 iterations) performed better than generic EPLL. Still, the proposed algorithm outperformed targeted EPLL consistently by a margin ranging from 5-7 dB in low-,mid- and high-noise settings. The best performing algorithm on this dataset, the TID algorithm, is three orders of magnitude slower. The PSNR gains of TID over the proposed FED drop quickly from 5dB for $\sigma = \frac{20}{255}$ to under 1dB for $\sigma = \frac{40}{255}$. We argue that compromising a small amount of quality to gain such an enormous speed-up is highly warranted. Note that such computational complexity makes TID unsuitable for denoising large images/videos, whereas FED is a natural fit for these domains.

4.2. Face image denoising

Face image denoising experiments were conducted on a subset of images taken from FEI face dataset [19]. It contains one image per person and has no overlap between the set of images used as the noisy images (test set), validation

Table 1. Comparison of average PSNR, SSIM and time taken by each of the algorithms to denoise a text image of size 107×104 pixels. The speed-up is calculated with respect to the proposed FED algorithm.

	$\sigma\times$ 255	**BM3D** 2 iterations $N_s = [6, 4]$	**EPLL** 5 iterations $N_s = [1]$	**tar-EPLL** 5 iterations $N_s = [1]$	**tar-EPLL3** 3 iterations $N_s = [4, 2, 1]$	**TID** 3 iterations $N_s = [4, 2, 1]$	**FED** 3 iterations $N_s = [4, 2, 1]$
PSNR:	20	28.64	28.04	30.38	29.92	39.37	35.65
	40	23.14	22.95	25.51	24.96	33.00	31.95
	60	19.27	20.07	22.67	22.09	29.31	28.80
	80	17.59	18.27	20.80	20.11	26.52	25.92
SSIM:	20	0.9856	0.9796	0.9910	0.9878	0.9966	0.9950
	40	0.9434	0.9303	0.9704	0.9533	0.9850	0.9843
	60	0.8392	0.8705	0.9368	0.9053	0.9645	0.9607
	80	0.7583	0.7920	0.8873	0.8282	0.9229	0.9053
Time (seconds):	-	0.12s	7.06s	6.83s	1.55s	2273.01s	1.39s
Speed up:	-	x0.08	x5.09	x4.91	x1.12	x1635.26	x1.00

Table 2. Same as Table 1 but for 10 face images of size 90×65 from FEI face dataset [19].

	$\sigma\times$ 255	**BM3D** 2 iterations $N_s = [6, 4]$	**EPLL** 5 iterations $N_s = [1]$	**tar-EPLL** 5 iterations $N_s = [1]$	**tar-EPLL3** 3 iterations $N_s = [4, 2, 1]$	**TID** 3 iterations $N_s = [4, 2, 1]$	**FED** 3 iterations $N_s = [4, 2, 1]$
PSNR:	20	31.37	31.40	31.99	31.15	32.26	32.11
	40	27.63	27.86	28.32	27.09	28.51	28.20
	60	25.70	25.68	26.08	24.67	26.09	25.60
	80	24.37	24.29	24.56	23.00	24.27	23.73
SSIM:	20	0.9054	0.9048	0.9160	0.8860	0.9201	0.9164
	40	0.8176	0.8136	0.8283	0.7554	0.8273	0.8094
	60	0.7576	0.7477	0.7612	0.6381	0.7524	0.7193
	80	0.6973	0.6859	0.6964	0.5483	0.6741	0.6288
Time (seconds):	-	0.05	2.74	2.73	0.87	878.20	0.71
Speed up:	-	x0.07	x3.84	x3.83	x1.23	x1236.90	x1.00

(a) Original (b) Noisy ($\sigma = \frac{30}{255}$) (c) BM3D (28.96, 0.8572) (d) EPLL (28.82, 0.8518) (e) tar-EPLL (29.18, 0.8652) (f) tar-EPLL3 (28.42, 0.8311) (g) TID (29.17, 0.8608) (h) FED (28.98, 0.8533)

Figure 4. Visual and objective comparison of denoising performance of a face image. The objective evaluation metrics of each case is shown in parenthesis in (PSNR, SSIM) format.

set and the targeted database. Out of the 100 images of distinct individuals, we used 10 individuals for creating noisy images, 10 for validation purposes and the targeted database contained images of a different set of 80 individuals. Since there is no overlap between the individuals in these different partitions, the images to be denoised are not a perfect match to the ones used in the targeted database. This setting also displays variations in lighting, contrast, gender, etc. Some examples from this dataset are shown in Figure 3.

The results obtained on the face images are shown in Ta-

Table 3. Same as Table 1 but for 10 license images of average size 44×92 cropped from the Caltech cars (1999) dataset [22].

	$\sigma \times 255$	**BM3D** 2 iterations $N_s = [6, 4]$	**EPLL** 5 iterations $N_s = [1]$	**tar-EPLL** 5 iterations $N_s = [1]$	**tar-EPLL3** 3 iterations $N_s = [4, 2, 1]$	**TID** 3 iterations $N_s = [4, 2, 1]$	**FED** 3 iterations $N_s = [4, 2, 1]$
PSNR:	20	26.15	25.99	26.87	26.55	25.27	25.87
	40	21.66	21.86	22.98	22.53	23.11	22.83
	60	19.16	19.52	20.70	20.13	21.02	20.89
	80	17.68	17.87	18.98	18.32	19.48	19.41
SSIM:	20	0.9330	0.9366	0.9476	0.9432	0.9223	0.9260
	40	0.8270	0.8452	0.8803	0.8706	0.8666	0.8640
	60	0.6984	0.7306	0.8009	0.7869	0.7885	0.7954
	80	0.6022	0.6322	0.7336	0.7102	0.7349	0.7480
Time (seconds):	-	0.03	1.74	1.75	0.46	311.97	0.45
Speed up:	-	x0.07	x3.90	x3.91	x1.02	x693.27	x1.00

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

Figure 5. Sample images from the license plate dataset created from Caltech Cars (1999) [22]. Figure 5(a) shows one of ten test images that were used in our experiments. The images 5(b)–5(e) show four examples out of 90 images from the targeted database.

Table 4. Same as Table 1 but for 100 test images of size 321×481 of the BSDS dataset [18].

	$\sigma \times 255$	**BM3D** 2 iters N_s=[6, 4]	**EPLL** 5 iters [1]	**EPLL3** 3 iters [4, 2, 1]	**FED** 3 iters [4, 2, 1]
PSNR:	30	27.57	27.66	26.95	26.87
	40	26.30	26.43	25.56	25.30
	50	25.45	25.51	24.51	23.95
SSIM:	30	0.7607	0.7717	0.7257	0.7155
	40	0.7101	0.7175	0.6494	0.6308
	50	0.6704	0.6729	0.5843	0.5532
Time (s):	-	1.66s	83.38s	19.34s	8.99s
Speed up:	-	x0.18	x9.27	x2.15	x1.00

ble 2. Quantitatively, FED obtains comparable results as those obtained from BM3D and different versions of EPLL. However, FED is almost 4 times faster than the better-performing EPLL variants. In addition, a visual comparison of results obtained on one of the images, shown in Figure 4, indicates that FED algorithm provides a more visually pleasing denoised estimate.

4.3. License plate denoising

License plate dataset was created by cropping license plates from the Caltech Cars (1999) dataset [22]. This dataset was originally collected for testing object category discovery algorithms and contains images of rear views of cars from Caltech parking lots. Therefore, the license plate images, which are only a small part of the car images, naturally contain small amounts of noise and display large variations in lighting and contrast. We include the results from this setting to demonstrate the performance of the proposed FED algorithm on realistic settings with targeted datasets containing similar but not identical images.

Table 3 shows the quantitative results obtained on this dataset using the different denoising algorithms. The PSNR and SSIM values of BM3D and EPLL are better than FED for almost all noise settings. However, FED does better under the highest noise setting we tested. Although quantitatively slightly under par, Figure 6 indicates that FED results are *consistently* better visually than other estimates.

4.4. Generic image denoising on BSDS dataset

As a limiting case, we also include the results obtained by FED on a set of generic images with a generic database. We use the Berkeley Segmentation Dataset (BSDS) [18]. The database of patches was created by randomly sampling 2 million patches from BSDS training set images. The denoising experiments are conducted on the BSDS test set consisting of 100 images. The results reported in Table 4 are averaged over all 100 images for each noise setting.

Since FED is designed to work well with targeted databases, as expected, BM3D and EPLL achieve better PSNR and SSIM measures in this dataset. However, FED results are comparable and are obtained in one-ninth of the time taken by EPLL. For visual comparison, we have included the results of one of the test images in Figure 7.

5. Conclusion

In this paper, we presented a new external denoising algorithm that is more efficient than the current state-of-the-art methods. The proposed algorithm is faster during run-

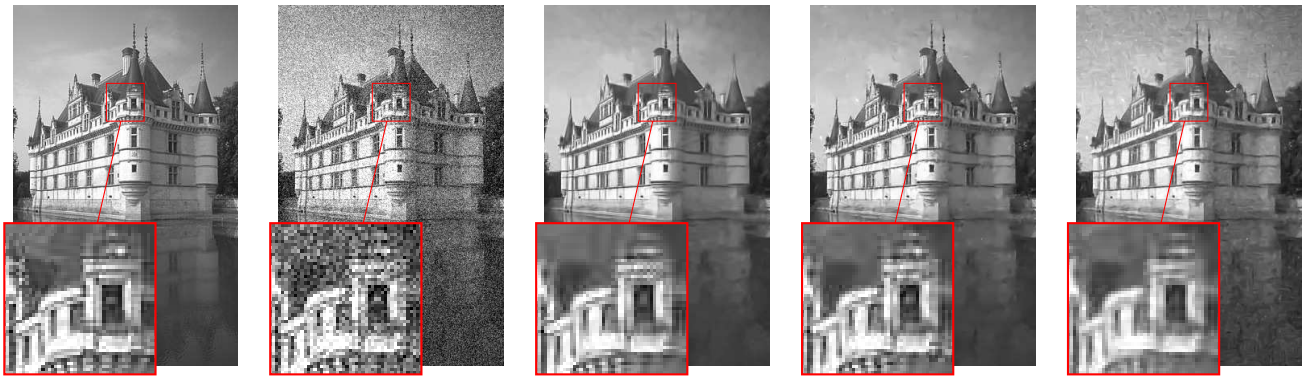
(a) Original

(b) Noisy ($\sigma = \frac{20}{255}$) (c) BM3D (25.77, 0.9497) (d) EPLL (25.51, 0.9508) (e) tar-EPLL (26.84, 0.9624) (f) tar-EPLL3 (26.47, 0.9586) (g) TID (25.37, 0.9525) (h) FED (26.47, 0.9541)

(i) Noisy ($\sigma = \frac{50}{255}$) (j) BM3D (19.48, 0.7977) (k) EPLL (19.64, 0.8295) (l) tar-EPLL (21.36, 0.8801) (m) tar-EPLL3 (20.99, 0.8734) (n) TID (22.72, 0.9101) (o) FED (21.99, 0.8833)

(p) Noisy ($\sigma = \frac{80}{255}$) (q) BM3D (17.07, 0.6766) (r) EPLL (16.97, 0.6862) (s) tar-EPLL (18.72, 0.7885) (t) tar-EPLL3 (18.26, 0.7826) (u) TID (19.43, 0.7840) (v) FED (19.81, 0.8183)

Figure 6. Visual and objective comparison of denoising performance of the same license image under different noise levels. The objective evaluation metrics of each case is shown in parenthesis in (PSNR, SSIM) format.



(a) Original (b) Noisy ($\sigma = \frac{30}{255}$) (c) BM3D (28.59, 0.8300) (d) EPLL (28.54, 0.8262) (e) Generic FED (27.25, 0.7323)

Figure 7. Visual and objective comparison of denoising performance of one of the BSDS test images using a generic database created from BSDS training set. The objective evaluation metrics of each case is shown in parenthesis in (PSNR, SSIM) format.

time and achieves better performance when used with targeted databases than EPLL, the current state-of-the-art efficient external denoising algorithm. It is also orders of magnitude faster than the powerful state-of-the-art TID algorithm without compromising much in terms of quality. This balance between speed and quality was achieved by transferring computationally demanding steps of designing optimal filters to an offline training step. Specifically, the information from a targeted database is extracted and stored in pre-learned transformations that are used directly during runtime. The proposed approach is extremely powerful when the transformation matrices are learned using a targeted database. However, when trained on a generic dataset the algorithm is unable to reconstruct texture details leading to over-smoothing, as can be observed in Figure 7. This can be avoided to a limited extent by increasing the number of anchor points so that detailed texture patches are properly represented. Another approach is using sophisticated algorithms (e.g. dictionary learning) in place of k-means to identify a more representative set of anchor patches. Future work will focus on a thorough comparison of our approach to deep learning based methods [5, 24] and making the algorithm more robust to database mismatch.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. k -svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, Nov 2006. 4

[2] J. Boulanger, J.-B. Sibarita, C. Kervrann, and P. Bouthemy. Non-parametric regression for patch-based fluorescence microscopy image sequence denoising. In *Biomedical Imaging: From Nano to Macro, 2008. ISBI 2008. 5th IEEE International Symposium on*, pages 748–751. IEEE, 2008. 1

[3] A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005. 1

[4] H. Burger, C. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1

[5] Y. Chen, W. Yu, and T. Pock. On learning optimized reaction diffusion processes for effective image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5261–5269, 2015. 8

[6] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, August 2007. 1, 2, 4

[7] C.-A. Deledalle, V. Duval, and J. Salmon. Non-local methods with shape-adaptive patches (nlm-sap). *Journal of Mathematical Imaging and Vision*, 43(2):103–120, 2012. 1

[8] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, December 2006. 1

[9] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006. 5

[10] Y. Gong, M. Pawlowski, F. Yang, L. Brandy, L. Bourdev, and R. Fergus. Web scale photo hash clustering on a single machine. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2

[11] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision*, pages 241–257. Springer, 2016. 2

[12] C. Knaus and M. Zwicker. Dual-domain image denoising. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 440–444. IEEE, 2013. 1

[13] M. Lebrun, A. Buades, and J.-M. Morel. A nonlocal bayesian image denoising algorithm. *SIAM Journal on Imaging Sciences*, 6(3):1665–1688, 2013. 1

[14] E. Luo, S. H. Chan, and T. Q. Nguyen. Image denoising by targeted external databases. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2450–2454. IEEE, May 2014. 1, 2, 3

[15] E. Luo, S. H. Chan, and T. Q. Nguyen. Adaptive image denoising by targeted databases. *IEEE Transactions on Image Processing*, 24(7):2167–2181, July 2015. 1, 2, 3, 5

[16] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, 2012. 2

[17] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009. 4

[18] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings 8th International Conference on Computer Vision*, volume 2, pages 416–423, July 2001. 2, 7

[19] C. E. Thomaz and G. A. Giraldi. A new ranking method for principal components analysis and its application to face image analysis. *Image and Vision Computing*, 28(6):902–913, June 2010. 5, 6

[20] Y.-Q. Wang and J.-M. Morel. Sure guided gaussian mixture image denoising. *SIAM Journal on Imaging Sciences*, 6(2):999–1034, 2013. 1

[21] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004. 5

[22] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 101–108 vol.2, 2000. 7

[23] G. Yu, G. Sapiro, and S. Mallat. Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity. *IEEE Transactions on Image Processing*, 21(5):2481–2499, 2012. 1

[24] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *arXiv preprint arXiv:1608.03981*, 2016. 8

[25] L. Zhang, W. Dong, D. Zhang, and G. Shi. Two-stage image denoising by principal component analysis with local pixel grouping. *Pattern Recognition*, 43(4):1531–1549, April 2010. 1

[26] Y. Zhou. Explore the power of external data in denoising task. 1

[27] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *International Conference on Computer Vision*, pages 479–486. IEEE, November 2011. 1, 2, 4