1-2015

A Collaborative Adaptive Wiener Filter for Image Restoration Using a Spatial-Domain Multi-patch Correlation Model

Khaled M. Mohamed
University of Dayton

Russell C. Hardie
University of Dayton, rhardie1@udayton.edu

Mohamed and Hardie *EURASIP Journal on Advances in Signal Processing* (2015) 2015:6
DOI 10.1186/s13634-014-0189-3

EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

A collaborative adaptive Wiener filter for image restoration using a spatial-domain multi-patch correlation model

Khaled M Mohamed† and Russell C Hardie*†

Abstract

We present a new patch-based image restoration algorithm using an adaptive Wiener filter (AWF) with a novel spatial-domain multi-patch correlation model. The new filter structure is referred to as a collaborative adaptive Wiener filter (CAWF). The CAWF employs a finite size moving window. At each position, the current observation window represents the reference patch. We identify the most similar patches in the image within a given search window about the reference patch. A single-stage weighted sum of all of the pixels in the similar patches is used to estimate the center pixel in the reference patch. The weights are based on a new multi-patch correlation model that takes into account each pixel's spatial distance to the center of its corresponding patch, as well as the intensity vector distances among the similar patches. One key advantage of the CAWF approach, compared with many other patch-based algorithms, is that it can jointly handle blur and noise. Furthermore, it can also readily treat spatially varying signal and noise statistics. To the best of our knowledge, this is the first multi-patch algorithm to use a single spatial-domain weighted sum of all pixels within multiple similar patches to form its estimate and the first to use a spatial-domain multi-patch correlation model to determine the weights. The experimental results presented show that the proposed method delivers high performance in image restoration in a variety of scenarios.

Keywords: Image restoration; Wiener filter; Correlation model; Patch-based processing

1 Introduction

1.1 Image restoration

During image acquisition, images are subject to a variety of degradations. These invariably include blurring from diffraction and noise from a variety of sources. Restoring such degraded images is a fundamental problem in image processing that has been researched since the earliest days of digital images [1,2]. A wide variety of linear and nonlinear methods have been proposed. Many methods have focused exclusively on noise reduction, and others seek to address multiple degradations jointly, such as blur and noise.

A widely used method for image restoration, relevant to the current paper, is the classic Wiener filter [3]. The standard Wiener filter is a linear space-invariant filter designed to minimize mean squared error (MSE) between the desired signal and estimate, assuming stationary random signals and noise. It is important to note that there are many disparate variations of Wiener filters. These include finite impulse response, infinite impulse response, transform-domain, and spatially adaptive methods. Within each of these categories, a wide variety of statistical models may be employed. Some statistical models are very simple, such as the popular constant noise-to-signal power spectral density model, and others are far more complex. In the case of the empirical Wiener filter [4], no explicit statistical model is used at all. Rather, a pilot or prototype estimate is used in lieu of a parametric statistical model. While all of these methods may go by the name of 'Wiener filter', they can be quite different in their character.

Recently, a form of adaptive Wiener filter (AWF) has been developed and successfully applied to super-resolution (SR) and other restoration applications by one of the current authors [5]. This AWF approach employs a spatially varying weighted sum to form an estimate of

*Correspondence: rhardie@udayton.edu
†Equal contributors
University of Dayton, Department of Electrical and Computer Engineering, 300 College Park, Dayton, 45469-0226 OH, USA

each pixel. The Wiener weights are determined based on a spatially varying spatial-domain parametric correlation model. This particular brand of AWF SR emerged from earlier work, including that in [6-8]. This kind of AWF is capable of jointly addressing blur, noise, and undersampling and is well suited to dealing with a non-stationary signal and noise. The approach is also very well suited to dealing with non-uniformly sampled imagery and missing or bad pixels. This AWF SR method has been shown to provide best-in-class performance for nonuniform interpolation-based SR [5,9-11] and has also been used successfully for demosaicing [12,13] and Nyquist sampled video restoration [14]. Under certain conditions, the method can also be very computationally efficient [5]. The key to this method lies in the particular correlation model used and how it is employed for spatially adaptive filtering.

A different approach to image restoration, also relevant to the current paper, is based on fusing multiple similar patches within the observed image. This patch-based approach is used primarily for noise reduction applications and exploits spatial redundancy that may express itself within an image, locally and/or non-locally. The method of non-local means (NLM), introduced in [15], may be the first method to directly fuse non-local patches from within the observed image based on vector distances for the purpose of image denoising. A notable early precursor to the NLM is the vector detection method [16,17]. In the vector detection approach, a codebook of representative patches from training data is used, rather than patches from the observed image itself [16,17]. A number of NLM variations have been proposed, including [18-26]. The basic NLM method forms an estimate of a reference pixel as a weighted sum of non-local pixels. The weights are based on the vector distances of the patch intensities between various non-local patches and the reference patch. In particular, the center samples of a non-local patches are weighted in proportion to the negative exponential of the corresponding patch distance. The NLM algorithm can be viewed as an extension of the bilateral filter [27-31], which forms an estimate by weighting neighboring pixels based on both spatial proximity and intensity similarity of individual pixels (rather than patches).

Improved performance for noise reduction is obtained with the block matching and 3D filtering (BM3D) approach proposed in [32-34]. The BM3D method also uses vector distances between patches, but the filtering is performed using a transform-domain shrinkage operation. By utilizing all of the samples within selected patches and aggregating the results, excellent noise reduction performance can be achieved with BM3D. Another related patch-based image denoising algorithm is the total least squares method presented in [35]. In this method, each ideal patch is modeled as a linear combination of similar patches from the observed image. Another type of patch-based Wiener denoising filter is proposed in [36], and a globalized approach to patch-based denoising is proposed in [37]. While such patch-based methods perform well in noise reduction, most are not capable of addressing blur and noise jointly. However, there are a few recent methods that do treat both blur and noise and incorporate multi-patch fusion. These include BM3D deblurring (BM3DDEB) [38] and iterative decoupled deblurring-BM3D (IDD-BM3D) [39]. The deblurring in these algorithms is not achieved by patch fusion alone. Rather, the patch fusion component of these algorithms serves as a type of signal model used for regularization. Note that multi-patch methods have also been developed and applied to SR [40-45]. However, the focus of this paper is on image restoration without undersampling/aliasing.

1.2 Proposed method and novel contribution

In this paper, we propose a novel multi-patch AWF algorithm for image restoration. In the spirit of [32], we refer to this new filter structure as a collaborative adaptive Wiener filter (CAWF). It can be viewed as an extension of the AWF in [5], with the powerful new feature of incorporating multiple patches for each pixel estimate. As with other patch-based algorithms, we employ a moving window. At each position, the current observation window represents the reference patch. Within a given search window about the reference, we identify the most similar patches to the reference patch. However, instead of simply weighting just the center pixels of these similar patches, as with NLM, or using transform-domain shrinkage like BM3D, we use a spatial-domain weighted sum of all of the pixels within all of the selected patches to estimate the one center pixel in the reference patch. The weights used are based on a novel spatially varying spatial-domain multi-patch correlation model. The correlation model takes into account each pixel's spatial distance to the center of its corresponding patch, as well as the intensity vector distances among the similar patches. The 'collaborative' nature of the CAWF springs from the fusion of multiple, potentially non-local, patches. One key advantage of the CAWF approach is that it can jointly handle blur and noise. Furthermore, the CAWF method is able to accommodate spatially varying signal and noise statistics.

Our approach is novel in that we use a single-pass spatial-domain weighted sum of all pixels within all of the similar patches to form the estimate each desired pixel. In the case of NLM, only the center pixel of each similar patch is given a weight [15].This is simple and effective for denoising, but deconvolution is not possible within the basic NLM framework, and all of the available information in the patches may not be exploited. While BM3D does fuse all of the pixels in the similar patches, the fusion

in BM3D is based on a wavelet shrinkage operation and not a spatial-domain correlation model [32]. Because of the nature of wavelet shrinkage, the standard BM3D is also unable to perform deblurring [32]. On the other hand, the CAWF structure can jointly address blur and noise and does not employ separate transform-domain inverse filtering as in [38] or iterative processing like that in [39]. To the best of our knowledge, this is the first multi-patch algorithm to use a single-pass weighted sum of all pixels within multiple similar patches to jointly address blur and noise. It is also the first to use a spatial-domain multi-patch correlation model.

The remainder of this paper is organized as follows. The observation model is described in Section 2. The CAWF algorithm is presented in Section 3. This includes the basic algorithm description as well as the new spatial-domain multi-patch correlation model. Computational complexity and implementation are also discussed in Section 3. Experimental results for simulated and real data are presented and discussed in Section 4. Finally, conclusions are offered in Section 5.

2 Observation model

The proposed CAWF algorithm is based on the observation model shown in Figure 1. The model input is a desired 2D discrete grayscale image, denoted $d(n_1, n_2)$, where n_1, n_2 are the spatial pixel indices. Using lexicographical notation, we shall represent all of the pixels in this desired image with a single column vector $\mathbf{d} = [\, d_1, d_2, \ldots, d_N]^T$, where N is the total number of pixels in the image. Next in the observation model, we assume the desired image is convolved with a specified point spread function (PSF), yielding

$$f(n_1, n_2) = d(n_1, n_2) * h(n_1, n_2), \tag{1}$$

where $h(n_1, n_2)$ is the PSF, and $*$ is 2D spatial convolution. In matrix form, this can be expressed as

$$\mathbf{f} = \mathbf{H}\mathbf{d} = [f_1, f_2, \ldots, f_N]^T, \tag{2}$$

where $\mathbf{H}$ is an $N \times N$ matrix containing values of PSF, and the vector $\mathbf{f}$ is the image $f(n_1, n_2)$ in lexicographical form. The PSF can be designed to model different types of blurring, such as diffraction from optics, spatial detection integration, atmospheric effects, and motion blurring. In the experimental results presented in this paper, we use a simple Gaussian PSF.

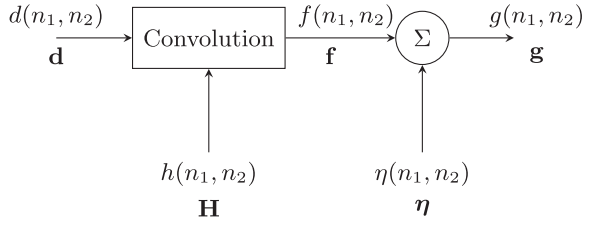

Figure 1 Observation model block diagram.

With regard to noise, our model assumes zero-mean additive Gaussian noise, with noise standard deviation of σ_η. Thus, the observed image is given by

$$g(n_1, n_2) = f(n_1, n_2) + \eta(n_1, n_2), \tag{3}$$

where $\eta(n_1, n_2)$ is a Gaussian noise array. In lexicographic form, this is given by

$$\mathbf{g} = \mathbf{f} + \boldsymbol{\eta}, \tag{4}$$

where $\mathbf{g} = [g_1, g_2, \ldots, g_N]^T$ and $\boldsymbol{\eta} = [\eta_1, \eta_2, \ldots, \eta_N]^T$ are the observed image and noise vectors, respectively. The random noise vector is Gaussian such that $\boldsymbol{\eta} \sim \mathcal{N}\left(\mathbf{0}, \sigma_\eta^2 \mathbf{I}\right)$.

3 Collaborative adaptive Wiener filter

3.1 CAWF overview

The CAWF employs a moving window approach with a moving reference patch and corresponding moving search window, each centered about pixel i, where $i = 1, 2, \ldots, N$. The reference patch spans $K_1 \times K_2 = K$ pixels symmetrically about pixel i. All of the pixels that lie within the span of this reference patch are placed into the reference patch vector defined as $\mathbf{g}_i = [g_{i,1}, g_{i,2}, \ldots, g_{i,K}]^T$. The search window is of size $L_1 \times L_2 = L$ pixels. Let the set $S_i = [\, S_i(1), S_i(2), \ldots\, S_i(L)]^T$ contain the indices of the pixels within the search window.

The next step in the CAWF algorithm is identifying the M patches from the search window that are most similar to the reference. This is done using a simple squared Euclidean distance. That is, we compute $\left\|\mathbf{g}_i - \mathbf{g}_j\right\|_2^2$, for $j \in S_i$. We select the M patches corresponding to the M smallest distances and designate these as 'similar patches'. All of the pixels from the similar patches shall be concatenated into a single $KM \times 1$ column vector, $\tilde{\mathbf{g}}_i = \left[\mathbf{g}_{s_{i,1}}^T, \mathbf{g}_{s_{i,2}}^T, \ldots, \mathbf{g}_{s_{i,M}}^T\right]^T$, where $\mathbf{s}_i = \left[s_{i,1}, s_{i,2}, \ldots s_{i,M}\right]^T$ contains the indices of the similar patches in order from smallest corresponding distance to largest. The minimum distance will always be zero and will correspond to the reference patch itself, such that $s_{i,1} = i$. This selection of similar patches is common to many patch-based algorithms, such as those in [32-34]. Examples of the similar patch selection is illustrated in Figure 2. The red square represents the reference patch, and the green squares represent selected similar patches. Note that there will generally be variability with regard to how similar the selected patches are to the reference and to each other. Some reference patches will have numerous low distance similar patches, and others will not. It is this variability that we wish to capture and account for with our multi-patch correlation model in Section 3.2.

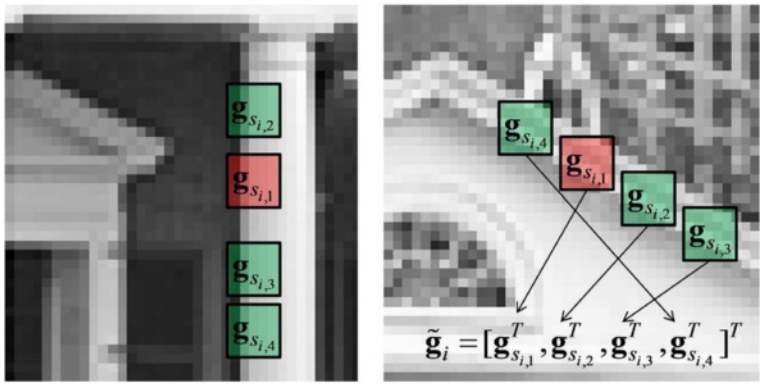

Figure 2 Selection of similar patches (green) from a given reference patch (red) within a search window.

With the collection of similar patches defined, we can now express the CAWF output as a weighted sum of the values in $\tilde{\mathbf{g}}_i$. In particular, the CAWF estimate for desired pixel i is given by

$$\hat{d}_i = \mathbf{w}_i^T \tilde{\mathbf{g}}_i, \tag{5}$$

where $\mathbf{w}_i = [w_1, w_2, \ldots w_{KM}]^T$ is a vector of weights. Note that this approach is a one-pass weighted-sum operation that incorporates all of the pixels in $\tilde{\mathbf{g}}_i$ for the estimate of d_i. To minimize the MSE, the Wiener filter weights [5] may be used such that

$$\mathbf{w}_i = \tilde{\mathbf{R}}_i^{-1} \tilde{\mathbf{p}}_i, \tag{6}$$

where $\tilde{\mathbf{R}}_i = E\left\{\tilde{\mathbf{g}}_i \tilde{\mathbf{g}}_i^T\right\}$ is a $KM \times KM$ autocorrelation matrix for the multi-patch observation vector $\tilde{\mathbf{g}}_i$, and $\tilde{\mathbf{p}}_i = E\left\{\tilde{\mathbf{g}}_i d_i\right\}$ is a $KM \times 1$ cross-correlation vector between the desired pixel d_i and $\tilde{\mathbf{g}}_i$. The statistics used to fill $\tilde{\mathbf{R}}_i$ and $\tilde{\mathbf{p}}_i$ are found using the new multi-patch correlation model described in Section 3.2.

3.2 Spatial-domain multi-patch correlation model

The multi-patch correlation model provides the values for $\tilde{\mathbf{R}}_i$ and $\tilde{\mathbf{p}}_i$, so that we may generate the weights in Equation 6. The model attempts to capture the spatial relationship among the pixels within a given patch, which is essential for deconvolution. Furthermore, it also seeks to incorporate knowledge of redundancy among the similar patches. Finally, the model captures the local desired signal variance, as well as the noise variance of each observed pixel.

To begin, first let $\tilde{\mathbf{f}}_i$ be the noise-free version of $\tilde{\mathbf{g}}_i$. In other words, we have $\tilde{\mathbf{g}}_i = \tilde{\mathbf{f}}_i + \tilde{\boldsymbol{\eta}}_i$, where $\tilde{\boldsymbol{\eta}}_i$ is the random noise vector associated with the samples within multi-patch observation vector i. We shall assume the noise is zero mean and uncorrelated with the signal. Furthermore, we will assume the noise samples in $\tilde{\boldsymbol{\eta}}_i$ are independent and identically distributed (i.i.d.) with a noise variance of σ_η^2. In this case, it is straightforward to show that

$$\tilde{\mathbf{R}}_i = E\left\{\tilde{\mathbf{g}}_i \tilde{\mathbf{g}}_i^T\right\} = E\left\{\tilde{\mathbf{f}}_i \tilde{\mathbf{f}}_i^T\right\} + \sigma_\eta^2 \mathbf{I} \tag{7}$$

and

$$\tilde{\mathbf{p}}_i = E\left\{\tilde{\mathbf{g}}_i d_i\right\} = E\left\{\tilde{\mathbf{f}}_i d_i\right\}. \tag{8}$$

Now, the problem reduces to modeling $E\left\{\tilde{\mathbf{f}}_i d_i\right\}$ and $E\left\{\tilde{\mathbf{f}}_i \tilde{\mathbf{f}}_i^T\right\}$.

In our new correlation model, the multi-patch statistics will be expressed in terms of statistics for a single patch and a distance matrix for all of the similar patches. In particular, we use the model

$$E\left\{\tilde{\mathbf{f}}_i \tilde{\mathbf{f}}_i^T\right\} = \hat{\sigma}_{d_i}^2 e^{-\mathbf{D}_i/(\alpha\sigma_\eta)} \otimes \mathbf{R}, \tag{9}$$

where $\otimes$ is a Kronecker product, and $\mathbf{R}$ is a $K \times K$ autocorrelation matrix of a single noise-free patch obtained from a variance-normalized desired image. We will say more about this shortly. The matrix

$$\mathbf{D}_i = \begin{bmatrix} D_{s_{i,1},s_{i,1}} & D_{s_{i,1},s_{i,2}} & \cdots & D_{s_{i,1},s_{i,M}} \\ D_{s_{i,2},s_{i,1}} & D_{s_{i,2},s_{i,2}} & \cdots & D_{s_{i,2},s_{i,M}} \\ \vdots & \vdots & \ddots & \vdots \\ D_{s_{i,M},s_{i,1}} & D_{s_{i,M},s_{i,2}} & \cdots & D_{s_{i,M},s_{i,M}} \end{bmatrix}. \tag{10}$$

is an $M \times M$ distance matrix among the M similar patches. In our notation, the exponential term in Equation 9 is a matrix whose elements are made of the exponential values of the corresponding distance matrix elements. The variable α is a tuning parameter that controls the correlation decay as a function of the distances between similar patches, and σ_η is noise standard deviation. The parameter $\hat{\sigma}_{d_i}^2$ is the estimated desired signal variance associated with the reference patch. By substituting Equation 9 into Equation 7, we get the autocorrelation matrix for $\tilde{\mathbf{g}}_i$ as

$$\tilde{\mathbf{R}}_i = \hat{\sigma}_{d_i}^2 e^{-\mathbf{D}_i/(\alpha\sigma_\eta)} \otimes \mathbf{R} + \sigma_\eta^2 \mathbf{I}. \tag{11}$$

In a similar manner, we model the cross-correlation vector as

$$\tilde{\mathbf{p}}_i = E\left\{\tilde{\mathbf{f}}_i\, d_i\right\} = \hat{\sigma}_{d_i}^2 e^{-[\mathbf{D}_i]_1/(\alpha\sigma_\eta)} \otimes \mathbf{p}, \tag{12}$$

where $\mathbf{p}$ is the $K \times 1$ cross-correlation vector for a single normalized patch, and $[\mathbf{D}_i]_1$ is the first column of the distance matrix $\mathbf{D}_i$. The correlation models in Equations 11 and 12 capture the spatial correlations among pixels within each patch using $\mathbf{R}$ and $\mathbf{p}$. The patch similarities, captured in the distance matrix, are used to 'modulate' these correlations with the Kronecker product to provide the full multi-patch correlation model. In this manner, pixels belonging to patches with smaller inter-patch distances will be modeled with higher correlations among them. Potential changes in the underlying desired image variance are captured in the model with the term $\hat{\sigma}_{d_i}^2$. In addition, a spatially varying noise variance can easily be incorporated if appropriate.

The specific distance metric used to populate the distance matrix $\mathbf{D}_i$ here is a scaled and shifted l^2-norm. This type of metric has been used successfully to quantify similarity between image patches corrupted with additive Gaussian noise [15]. In particular, the distance between patches centered about pixels i and j is given by

$$D_{i,j} = \begin{cases} \frac{\|\mathbf{g}_i - \mathbf{g}_j\|_2}{\sigma_\eta\sqrt{2K}} - D_0 & \frac{\|\mathbf{g}_i - \mathbf{g}_j\|_2}{\sigma_\eta\sqrt{2K}} > D_0 \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where $\|\mathbf{g}_i - \mathbf{g}_j\|_2$ is the l^2-norm distance, K is total number of pixels in each patch, and σ_η is noise standard deviation. The scaling by $\sigma_\eta\sqrt{2K}$ normalizes the distance with respect to K and σ_η, and D_0 can be used as a tuning parameter in the correlation model to adjust for distance due to noise. To see how the scaling works, and understand the potential role of D_0, consider the distance with $D_0 = 0$ defined as

$$\bar{D}_{i,j} = \frac{\|\mathbf{g}_i - \mathbf{g}_j\|_2}{\sigma_\eta\sqrt{2K}}. \tag{14}$$

It can be shown that for identical patches with distance due only to i.i.d. Gaussian noise, the probability density function (pdf) for $\bar{D}_{i,j}$ is that of a scaled Chi random variable and is given by

$$f_{\bar{D}_{i,j}}(x) = \sqrt{K}\chi_K\left(x\sqrt{K}\right). \tag{15}$$

This pdf is plotted in Figure 3 for four values of K. Note that with our scaling, the pdf is not a function of σ_η and the mean is close to 1 for all K. Also, note that D_0 can be used to shift the pdf.

Let us now turn our attention to $\mathbf{R}$, $\mathbf{p}$, and $\hat{\sigma}_{d_i}^2$. Note that $\mathbf{R}$ and $\mathbf{p}$ correspond to a single patch derived from a desired image with zero mean and variance of one, denoted $\bar{d}(n_1, n_2)$. After PSF blurring, the resulting image is denoted $\bar{f}(n_1, n_2)$, following the model shown in Figure 1. Since these statistics are for only a single patch, they can be modeled in a fashion similar to that in [5]. To begin, consider the 2D wide sense stationary (WSS) autocorrelation function model for $\bar{d}(n_1, n_2)$ given by

$$r_{\bar{d}\bar{d}}(n_1, n_2) = \rho^{\sqrt{n_1^2+n_2^2}}, \tag{16}$$

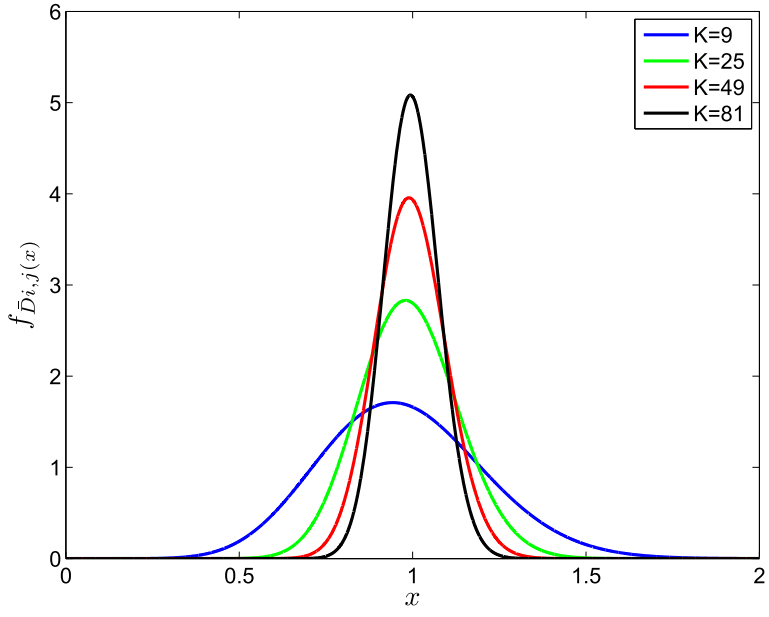

Figure 3 Probability density function for $\bar{D}_{i,j}$ for identical patches with Gaussian noise. The distance here is due exclusively to noise.

where ρ is the one pixel step correlation value. The cross correlation function between $\bar{d}(n_1,n_2)$ and $\bar{f}(n_1,n_2)$ can then be expressed as

$$r_{\bar{d}\bar{f}}(n_1,n_2) = r_{\bar{d}\bar{d}}(n_1,n_2) * h(n_1,n_2). \tag{17}$$

The auto-correlation function for $\bar{f}(n_1,n_2)$ can also be expressed in terms of the desired autocorrelation function as

$$r_{\bar{f}\bar{f}}(n_1,n_2) = r_{\bar{d}\bar{d}}(n_1,n_2) * h(n_1,n_2) * h(-n_1,-n_2). \tag{18}$$

Figure 4 shows an example of $r_{\bar{d}\bar{f}}(n_1,n_2)$ and $r_{\bar{f}\bar{f}}(n_1,n_2)$ for Gaussian blur PSF with standard deviation of 1.5 pixels, and $\rho = 0.7$. As expected, the correlation drops with distance, as controlled by ρ.

Now consider a $K_1 \times K_2 = K$ patch from $\bar{f}(n_1,n_2)$, denoted $\bar{\mathbf{f}} = [\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_K]^T$, and the corresponding desired pixel value, $\bar{d}$. The $K \times K$ autocorrelation matrix for $\bar{\mathbf{f}}$ is $\mathbf{R} = E\left\{\bar{\mathbf{f}}\bar{\mathbf{f}}^T\right\}$, and the $K \times 1$ cross-correlation vector is $\mathbf{p} = E\left\{\bar{\mathbf{f}}\bar{\mathbf{d}}\right\}$. Expressing all of the terms in the autocorrelation matrix, we obtain

$$\mathbf{R} = E\left\{\bar{\mathbf{f}}\bar{\mathbf{f}}^T\right\} = \begin{bmatrix} E\left\{\bar{f}_1\bar{f}_1\right\} & E\left\{\bar{f}_1\bar{f}_2\right\} & \cdots & E\left\{\bar{f}_1\bar{f}_K\right\} \\ E\left\{\bar{f}_2\bar{f}_1\right\} & E\left\{\bar{f}_2\bar{f}_2\right\} & \cdots & E\left\{\bar{f}_2\bar{f}_K\right\} \\ \vdots & \vdots & \ddots & \vdots \\ E\left\{\bar{f}_K\bar{f}_1\right\} & E\left\{\bar{f}_K\bar{f}_2\right\} & \cdots & E\left\{\bar{f}_K\bar{f}_K\right\} \end{bmatrix}. \tag{19}$$

Assuming a 2D WSS model, as in [5], this matrix can be populated from Equation 18 as follows

$$\mathbf{R} = \begin{bmatrix} r_{\bar{f}\bar{f}}(0,0) & r_{\bar{f}\bar{f}}(\Delta(1,2)) & \cdots & r_{\bar{f}\bar{f}}(\Delta(1,K)) \\ r_{\bar{f}\bar{f}}(\Delta(2,1)) & r_{\bar{f}\bar{f}}(0,0) & \cdots & r_{\bar{f}\bar{f}}(\Delta(2,K)) \\ \vdots & \vdots & \ddots & \vdots \\ r_{\bar{f}\bar{f}}(\Delta(K,1)) & r_{\bar{f}\bar{f}}(\Delta(K,2)) & \cdots & r_{\bar{f}\bar{f}}(0,0) \end{bmatrix}, \tag{20}$$

where $\Delta(m,n) = [\Delta_x(m,n), \Delta_y(m,n)]$, and $\Delta_x(m,n)$ and $\Delta_y(m,n)$ are the x and y distances between $\bar{f}_m$ and $\bar{f}_n$ in $\bar{f}(n_1,n_2)$ in units of pixels. In a similar fashion, we can populate $\mathbf{p}$ using Equation 17 as follows

$$\mathbf{p} = E\{\bar{\mathbf{f}}\bar{d}\} = \begin{bmatrix} E\{\bar{f}_1\bar{d}\} \\ E\{\bar{f}_2\bar{d}\} \\ \vdots \\ E\{\bar{f}_K\bar{d}\} \end{bmatrix} = \begin{bmatrix} r_{\bar{d}\bar{f}}\left(\Delta\left(1, \frac{K+1}{2}\right)\right) \\ r_{\bar{d}\bar{f}}\left(\Delta\left(2, \frac{K+1}{2}\right)\right) \\ \vdots \\ r_{\bar{d}\bar{f}}\left(\Delta\left(K, \frac{K+1}{2}\right)\right) \end{bmatrix}, \tag{21}$$

where $\bar{f}_{\frac{K+1}{2}}$ corresponds to the spatial position of $\bar{d}$.

The final term needed for the correlation model is $\hat{\sigma}^2_{d_i}$, which corresponds to the underlying desired signal variance associated with the reference patch. Since $\mathbf{R}$ and $\mathbf{p}$ are based on a desired signal with unit variance, we scale these by an estimate of the desired signal variance for each reference patch in the observed image to obtain the appropriate values. To obtain this estimate, we first compute the sample variance estimate of the pixels in $\tilde{\mathbf{g}}_i$ and we denoted this as $\hat{\sigma}_{g_i}$. We then subtract the noise variance to give an estimate of the noise-free observed signal variance as

$$\hat{\sigma}^2_{f_i} = \hat{\sigma}^2_{\mathrm{g_i}} - \sigma^2_\eta. \tag{22}$$

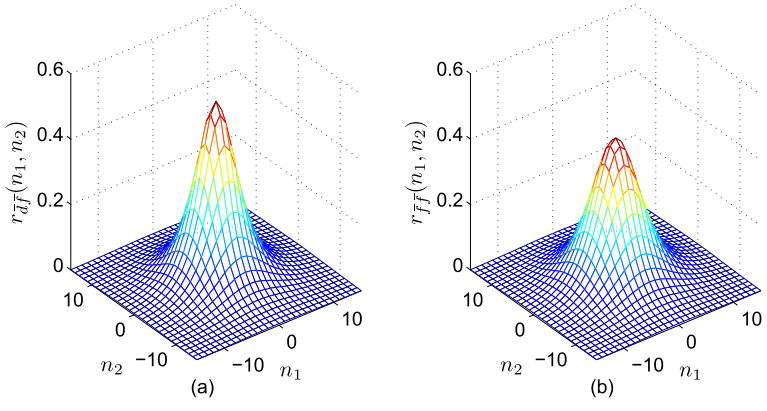

Figure 4 Single-patch spatial correlation for Gaussian blur with a standard deviation of 1.5 pixels. (a) Cross-correlation $r_{\bar{d}\bar{f}}(n_1,n_2)$ used to populate $\mathbf{p}$. **(b)** Autocorrelation $r_{\bar{f}\bar{f}}(n_1,n_2)$ used to populate $\mathbf{R}$.

Note that in practice, we do not allow the value in Equation 22 to go below a specified minimum value. Using Equation 17, it can be shown that the relationship between $\sigma_{d_i}^2$ and $\sigma_{f_i}^2$ is given by [5]

$$\sigma_{d_i}^2 = \frac{1}{C(\rho)}\sigma_{f_i}^2, \tag{23}$$

where

$$C(\rho) = \sum_{-\infty}^{\infty}\sum_{-\infty}^{\infty} \rho^{\sqrt{n_1^2+n_2^2}}\, \tilde{h}(n_1, n_2), \tag{24}$$

and $\tilde{h}(n_1, n_2) = h(n_1, n_2) * h(-n_1, -n_2)$. Thus, our desired signal variance estimate is $\hat{\sigma}_{d_i}^2 = \hat{\sigma}_{f_i}^2 / C(\rho)$.

By substituting Equations11 and 12 into Equation 6, and dividing through by $\hat{\sigma}_{d_i}^2$, the CAWF weight vector can be computed as

$$\mathbf{w}_i = \tilde{\mathbf{R}}_i^{-1}\tilde{\mathbf{p}}_i = \left[e^{-\mathbf{D}_i/(\alpha\sigma_\eta)} \otimes \mathbf{R} + \frac{\sigma_\eta^2}{\hat{\sigma}_{d_i}^2}\mathbf{I} \right]^{-1} e^{-[\mathbf{D}_i]_1/(\alpha\sigma_\eta)} \otimes \mathbf{p}. \tag{25}$$

Note that the DC response of the CAWF filter is not guaranteed to be one (i.e., the weights may not sum to 1). To prevent artifacts when processing an image that is not zero mean, we normalize the weights to sum to one by dividing the weight vector by the sum of the weights for each i before computing the weighted sum in Equation 5. From Equation 25, it is clear that CAWF weights adapt spatially based on the local signal variance, the variance of the noise, and the distance matrix among the similar patches. There are two tuning parameters in the correlation model, ρ, which controls the correlation between samples within a given patch expressed in $\mathbf{R}$ and $\mathbf{p}$, and α which controls the correlation between patches. We have found that the algorithm is not highly sensitive to these tuning parameters, and good performance can be obtained for a wide range of images using a specified fixed value for these. Note that in addition to providing an estimate of the desired image, an estimate of the MSE itself can be readily generated based on the correlation model. This estimated MSE is given by [9]

$$\hat{J}_i = E\left\{ \left(d_i - \hat{d}_i \right)^2 \right\} = \hat{\sigma}_{d_i}^2 - 2\mathbf{w}_i^T\tilde{\mathbf{p}}_i + \mathbf{w}_i^T\tilde{\mathbf{R}}_i\mathbf{w}_i. \tag{26}$$

A block diagram showing all of the key steps in the CAWF filter is shown in Figure 5.

To better understand the workings of the collaborative correlation model in determining the filter weights, consider Examples 1 and 2 shown in Figures 6 and 7, respectively. These examples are for the case of blur with a Gaussian PSF with a 1 pixel standard deviation and Gaussian noise of standard deviation 20. The patch size

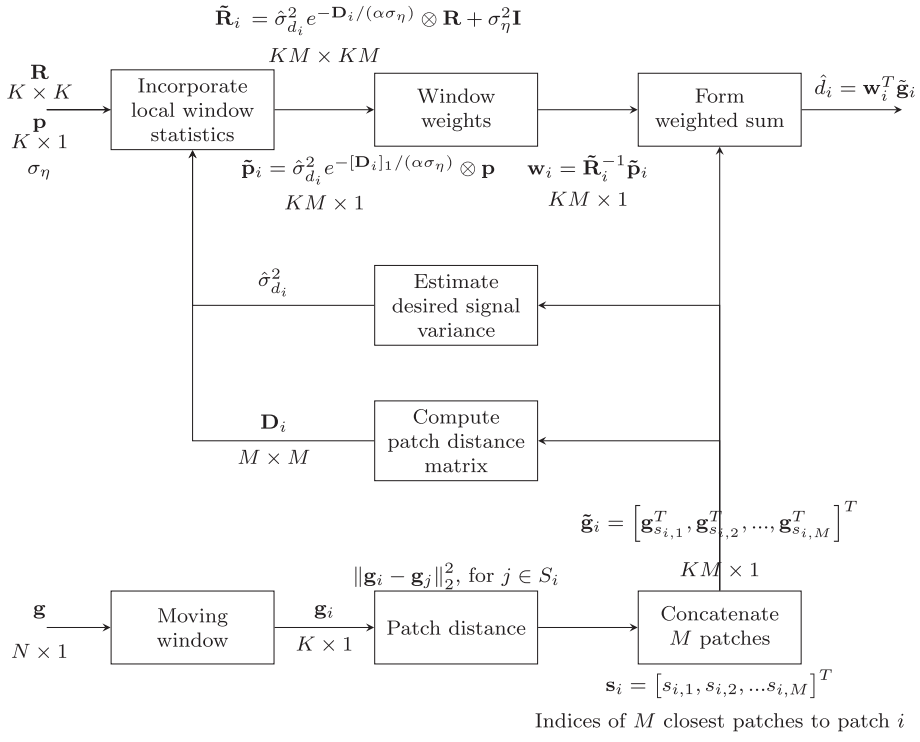

Figure 5 Block diagram illustrating the proposed CAWF algorithm. The algorithm employs a weighted sum of the samples from similar patches. The Wiener weights are computed based on the new multi-patch correlation model as shown.

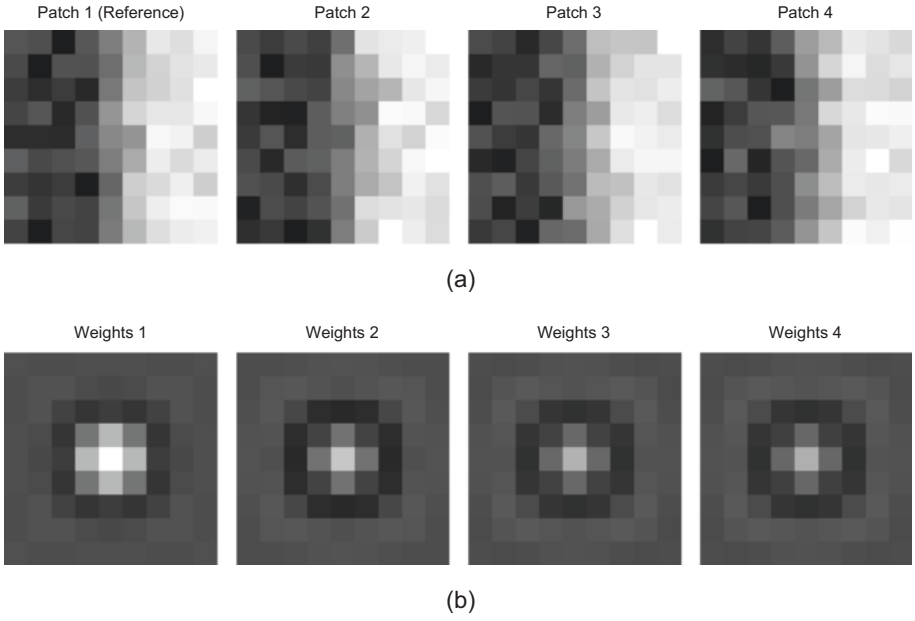

Figure 6 Example 1 showing a group of similar patches with small inter-patch distances. (a) Spatial domain similar patches; **(b)** CAWF weights corresponding to the four patches.

is $K = 9 \times 9 = 81$, and there are $M = 4$ patches. Example 1 in Figure 6 shows patches for an edge region where the multiple patches have a very similar underlying structure, and hence small inter-patch distances. In contrast, Example 2 in Figure 7 shows the case of dissimilar patches with large inter-patch distances. In Figures 6 and 7, (a) shows the spatial domain patches, and (b) shows the CAWF weights corresponding to these patches. Note that in Example 1, pixels in all of the patches get significant weight as shown in Figure 6b. In contrast, for the dissimilar patches in Example 2, only the reference patch pixels get significant weight, as shown in Figure 7b. Figure 8 provides additional insight by comparing the filter weights, summed over patches, for Examples 1 and

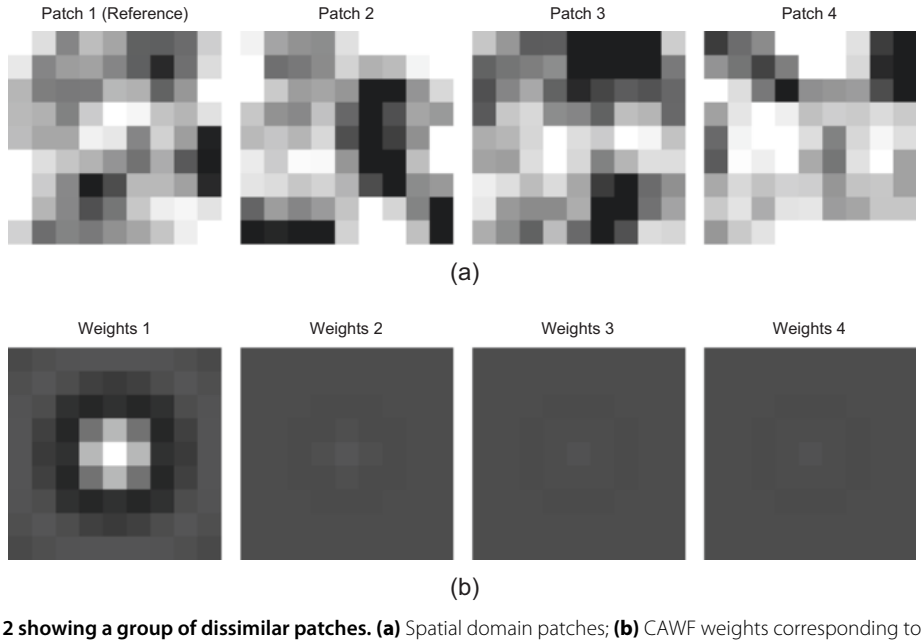

Figure 7 Example 2 showing a group of dissimilar patches. (a) Spatial domain patches; **(b)** CAWF weights corresponding to the four patches.

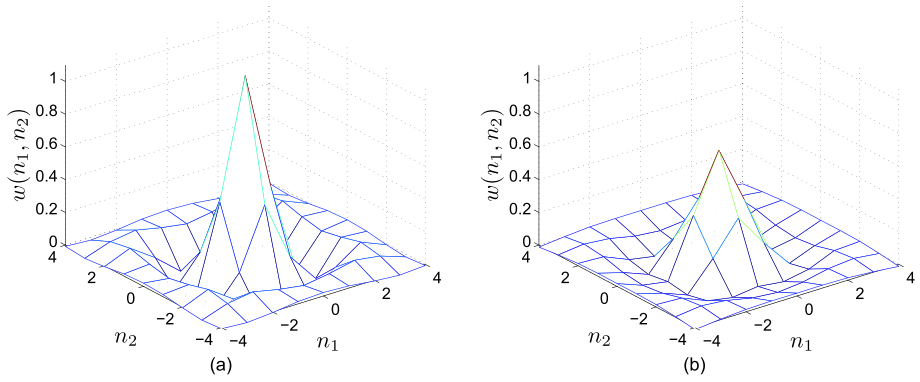

Figure 8 Mesh plots of the CAWF weights summed over patches. (a) Example 1 with similar patches and **(b)** Example 2 with dissimilar patches. Because of the high patch similarity, the weights for Example 1 represent a more aggressive deconvolution than in Example 2.

2 side by side. When many similar patches are available, more noise reduction is possible. In turn, this allows the deconvolution to be more aggressive. This can be seen in the more aggressive deconvolution weights in Figure 8a, compared with those in Figure 8b. On the other hand, if no low-distance patches can be found for the reference, the CAWF algorithm essentially reverts to a single-patch AWF filter, giving non-zero weights only to the reference patch. Because less noise reduction can be achieved this way, the filter automatically becomes less aggressive in its deconvolution, so as to not exaggerate noise.

These examples show how the distance matrix plays an interesting role in the joint deblurring/denoising aspect of the CAWF. This type of spatially adaptive deconvolution is unlike anything we have seen before in other multi-patch restoration algorithms. A similar process occurs with noise only. In that case, the CAWF balances the amount of spatial low-pass filtering employed. When good patch matches are found, the CAWF relies more on patch fusion for noise reduction. When no good matches are found, it resorts to more spatial smoothing. All of this is also balanced by the estimate of the local desired signal variance, $\hat{\sigma}^2_{d_i}$. Generally, more smoothing is done in low signal variance areas. One last point of interest with regard to these examples relates to the structure of the multi-patch autocorrelation matrix $\tilde{\mathbf{R}}_i$. These matrices are shown in Figure 9 for Examples 1 and 2. Note that based on Equation 11, these are both block matrices made up of a 4×4 grid of scaled 81×81 $\mathbf{R}$ submatrices. The $\mathbf{R}$ matrix itself has an apparent 9×9 substructure, due to the column-ordered lexicographical representation of each patch. Note that the off diagonal blocks of $\tilde{\mathbf{R}}_i$ for Example 2 in Figure 9b are essentially zero. In most cases, the inter-block correlations will lie between Examples 1 and 2.

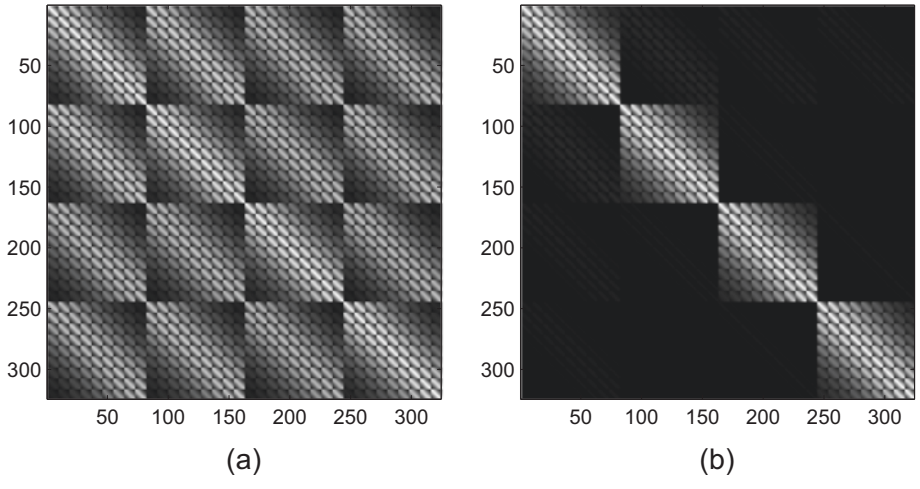

Figure 9 Multi-patch autocorrelation matrix $\tilde{\mathbf{R}}_i$. (a) Example 1 with similar patches and **(b)** Example 2 with dissimilar patches.

3.3 Multi-pixel estimation and aggregation

In the CAWF algorithm described in Section 3.1, one pixel is estimated for each reference patch. However, in a manner similar to that in [5], it is possible to estimate multiple desired pixels from each multi-patch observation vector $\tilde{\mathbf{g}}_i$. In fact, all of the desired pixels corresponding to $\tilde{\mathbf{g}}_i$ can be estimated. Let this full $KM \times 1$ vector of desired pixels be denoted $\tilde{\mathbf{d}}_i$. If all multi-patch observation vectors are used in this fashion, many estimates of each desired pixel are obtained. These can be aggregated by a simple average. In the case of noise only, we have observed that aggregation yields improved results. For joint deblurring and denoising with any significant amount of blur, the aggregation does not appear to provide any advantage. However, this multi-pixel estimation approach can be used to reduce the computational complexity, since not every multi-patch observation vector must be processed in order to form a complete image estimate.

To perform the multi-pixel estimation, the CAWF filter output is expressed as

$$\hat{\tilde{\mathbf{d}}}_i = \mathbf{W}_i^T \tilde{\mathbf{g}}_i, \tag{27}$$

where $\hat{\tilde{\mathbf{d}}}_i$ is the estimate of $\tilde{\mathbf{d}}_i$, and $\mathbf{W}_i$ is a $KM \times KM$ matrix of weights. The weight matrix is given by

$$\mathbf{W}_i = \tilde{\mathbf{R}}_i^{-1} \tilde{\mathbf{P}}_i, \tag{28}$$

where

$$\tilde{\mathbf{P}}_i = E\left\{\tilde{\mathbf{f}}_i \tilde{\mathbf{d}}_i^T\right\} = \hat{\sigma}_{d_i}^2 e^{-\mathbf{D}_i/(\alpha\sigma_\eta)} \otimes \mathbf{P}, \tag{29}$$

$\mathbf{P} = E\left\{\bar{\mathbf{f}}\bar{\mathbf{d}}^T\right\}$ is a $K \times K$ normalized cross-correlation matrix, and $\bar{\mathbf{d}}$ is the $K \times 1$ desired vector corresponding to $\bar{\mathbf{f}}$.

3.4 Computational complexity and implementation

Here, we briefly address the computational complexity of the CAWF filter by tracking the number of floating point operations (flops), where a flop is defined as one multiply plus add operation. The first action of the CAWF filter is finding similar patches. This requires computing L distances of K dimensional vectors (note that L is the search window size, and K is the patch size in pixels). The next step is computing the distance matrix based on Equation 13 for the M selected patches. This requires computing $M^2/2$ scaled and shifted distances for K dimensional vectors. The Kronecker products for $\tilde{\mathbf{R}}_i$ and $\tilde{\mathbf{p}}_i$ require $(KM)^2$ and KM multiplies, respectively. However, the main computational burden of the CAWF filter comes next with the computation of the weights in Equation 6. This can be done using Cholesky factorization, which requires $(KM)^3/3$ flops to perform LU decomposition for the $KM \times KM$ autocorrelation matrix $\tilde{\mathbf{R}}_i$. Computing the weights from the LU decomposition requires $2(KM)^2$ flops using forward and backward substitution. The final weighted sum operation is accomplished with KM flops. Since the dominant term in the computational complexity is the Cholesky factorization, we might conclude that the complexity of the CAWF filter is $O((KM)^3)$. Thus, the complexity of the CAWF algorithm goes up significantly with larger windows sizes, K, and more similar patches, M. However, an important thing to note is that the CAWF algorithm is completely parallel at the output pixel level. Unlike most variational image restoration methods, each output pixel can be computed independently and in parallel. Also, the CAWF approach requires only one pass over the data.

To put the CAWF computational complexity into context, consider that the AWF method employed here, with a spatially varying signal-to-noise ratio (SNR) estimate, may be viewed as a special case of the CAWF with $M = 1$. Thus, increasing M for CAWF causes a corresponding increase in complexity according to $O((KM)^3)$. The NLM method shares the same distance computations and comparisons and CAWF. However, in contrast to CAWF, NLM only requires L flops per output in the weighted sum, since it only weights the center sample of each patch in the search window. Although significantly simpler computationally, NLM does not fully exploit all of the information in the patches and it cannot perform deconvolution. Also, AWF is not able to exploit multi-patch information.

For pure denoising application, we have found that good results can be obtained with CAWF for $M = 10$, and $K = 3 \times 3 = 9$ for light noise and $K = 5 \times 5 = 25$ for moderate to heavy noise. In the case of joint deblurring and denoising, a larger window size is needed for adequate deconvolution. We have found that $K = 9 \times 9 = 81$ is a reasonable choice for light to moderate blurring. Our implementation uses MATLAB with no parallel acceleration or mex files, and processing is done on a PC with Intel®Xeon®Processor 3.7 GHz. CAWF processing time for a pure denoising application with a 512×512 image using $K = 9$ and $M = 10$ is 155 s. For context, the AWF processing takes 33 s, and NLM takes 3.2 s.

4 Experimental results

In this section, we demonstrate the efficacy of the proposed CAWF algorithm using images with a variety of simulated degradations and using real video frames. We also present a parameter sensitivity analysis. The filter parameters used for all of the experimental results are listed in Table 1. Note that for a given scenario, the same parameters are used for processing all of the test images.

4.1 Simulated data

In this section, we present quantitative results using simulated data. We consider two cases: noise only and blur with noise. For each case, we consider four specific scenarios

Table 1 CAWF parameters used in experimental results

Parameter name	Variable	Case 1: noise only		Case 2: blur and noise
		Selected value $\sigma_\eta < 20$	Selected value $\sigma_\eta \geq 20$	Selected value
Patch size	K	$3 \times 3 = 9$	$5 \times 5 = 25$	$9 \times 9 = 81$
Search window size	L	$17 \times 17 = 289$	$11 \times 11 = 121$	$9 \times 9 = 81$
Number of patches	M	10	10	8
Autocorrelation decay	ρ	0.65	0.70	0.65
Patch similarity decay	α	2.00	1.40	1.20
Distance offset	D_0	0.25	0.50	0.00
Aggregation	N/A	Averaging	Averaging	None

and use six test images. Also, for each case, we compare against state-of-the art methods for which MATLAB implementations are publicly available.

The test images are shown in Figure 10. These are 8-bit uncompressed images with a high level of detail. We use two quantitative performance metrics to evaluate the restorations. The first is the commonly used peak signal-to-noise ratio (PSNR), defined as

$$\mathrm{PSNR}\left(\mathbf{d},\hat{\mathbf{d}}\right) = 10\,\log_{10}\left(\frac{255^2}{\frac{1}{N}\sum_{i=1}^{N}\left(d_i-\hat{d}_i\right)^2}\right). \quad (30)$$

We also use the structural similarity (SSIM) index [46], which many argue is more consistent with subjective perception than PSNR. When reporting PSNR, we also include the improvement in PSNR (ISNR) for the reader's convenience. This is given by

$$\mathrm{ISNR} = \mathrm{PSNR}\left(\mathbf{d},\hat{\mathbf{d}}\right) - \mathrm{PSNR}\left(\mathbf{d},\mathbf{g}\right). \quad (31)$$

4.1.1 Additive Gaussian noise

In our first case, we consider additive Gaussian noise with no PSF blur (i.e., $h(n_1,n_2) = \delta(n_1,n_2)$). We consider four different noise standard deviations. The denoising benchmark methods are NLM [15], Globalized NLM (GLIDE-NLM) [37], PLOW [36], BM3D [32], and the single patch AWF [5]. Note that the NLM implementation is from [37], and AWF used is the same as CAWF with no aggregation and $M = 1$.

The PSNR comparison is provided in Table 2, and the SSIM comparison is in Table 3. Note that CAWF provides the highest PSNR results in Table 2 in all but one instance,

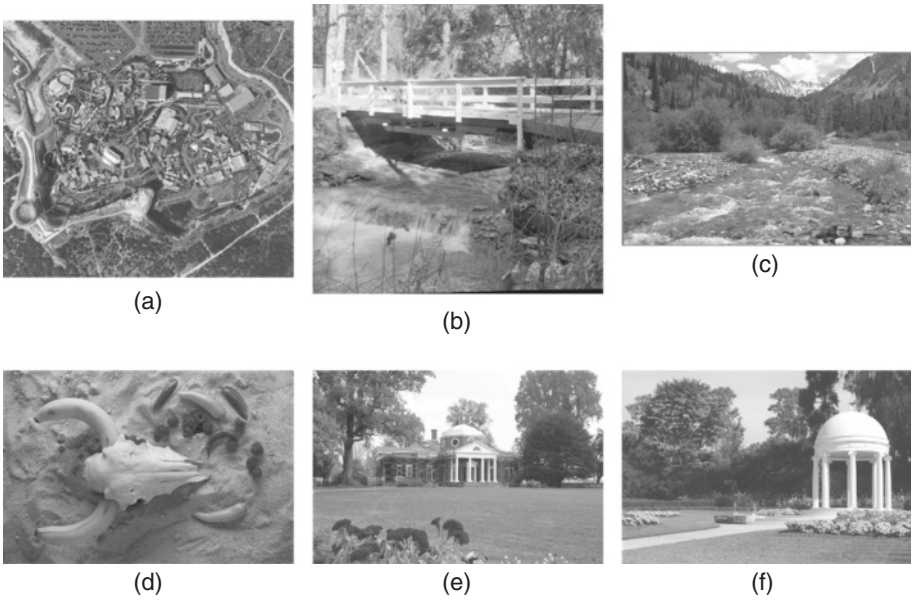

Figure 10 All truth images used in simulated data experimental results. (a) Aerial (491 × 434); **(b)** bridge (512 × 512); **(c)** river (Kodak 11) (768 × 512); **(d)** bones (512 × 768 rotated); **(e)** building (768 × 512); and **(f)** gazebo (768 × 512).

Table 2 PSNR comparison for additive Gaussian noise

Image	Method	PSNR (ISNR)			
		$\sigma_\eta = 10$	$\sigma_\eta = 20$	$\sigma_\eta = 30$	$\sigma_\eta = 40$
Aerial	Corrupted	28.13	22.11	18.59	16.09
491 × 434	NLM	30.14 (2.01)	25.56 (3.45)	23.62 (5.03)	22.35 (6.26)
	GLIDE-NLM	30.25 (2.12)	25.82 (3.71)	23.79 (5.19)	22.58 (6.49)
	PLOW	28.76 (0.63)	25.30 (3.19)	23.68 (5.09)	22.64 (6.55)
	BM3D	30.61 (2.48)	26.48 (4.37)	24.38 (5.79)	22.92 (6.82)
	AWF	30.19 (2.06)	26.09 (3.98)	24.05 (5.46)	22.73 (6.63)
	CAWF	*30.67 (2.54)*	*26.55 (4.44)*	*24.50 (5.90)*	*23.13 (7.04)*
Bridge	Corrupted	28.14	22.11	18.59	16.09
512^2	NLM	30.56 (2.42)	26.36 (4.24)	24.69 (6.09)	23.62 (7.52)
	GLIDE-NLM	30.64 (2.51)	26.55 (4.43)	24.82 (6.22)	23.73 (7.64)
	PLOW	30.00 (1.87)	26.70 (4.58)	25.23 (6.63)	24.25 (8.16)
	BM3D	*31.17 (3.04)*	27.27 (5.16)	25.46 (6.87)	24.31 (8.21)
	AWF	30.58 (2.45)	26.87 (4.76)	25.09 (6.50)	23.90 (7.80)
	CAWF	31.11 (2.97)	*27.31 (5.19)*	*25.50 (6.91)*	*24.33 (8.23)*
River	Corrupted	28.13	22.11	18.59	16.09
(Kodak 11)	NLM	29.95 (1.82)	25.46 (3.35)	23.63 (5.04)	22.47 (6.37)
768 × 512	GLIDE-NLM	29.84 (1.71)	25.71 (3.60)	23.78 (5.19)	22.62 (6.53)
	PLOW	28.53 (0.40)	24.70 (2.58)	22.94 (4.35)	22.10 (6.01)
	BM3D	30.37 (2.24)	26.16 (4.05)	24.16 (5.57)	22.86 (6.77)
	AWF	29.96 (1.82)	25.90 (3.78)	23.90 (5.31)	22.61 (6.52)
	CAWF	*30.48 (2.35)*	*26.32 (4.20)*	*24.35 (5.76)*	*23.11 (7.02)*
Bones	Corrupted	28.13	22.11	18.59	16.09
512 × 768	NLM	30.36 (2.23)	26.65 (4.54)	25.50 (6.91)	24.72 (8.63)
	GLIDE-NLM	30.48 (2.35)	26.86 (4.75)	25.50 (6.91)	24.78 (8.68)
	PLOW	29.33 (1.20)	26.54 (4.43)	25.60 (7.01)	24.94 (8.85)
	BM3D	30.85 (2.72)	27.18 (5.07)	25.81 (7.22)	25.05 (8.96)
	AWF	30.46 (2.33)	27.13 (5.02)	25.60 (7.01)	24.51 (8.41)
	CAWF	*30.93 (2.79)*	*27.46 (5.35)*	*26.03 (7.44)*	*25.11 (9.02)*
Building	Corrupted	28.13	22.11	18.59	16.09
768 × 512	NLM	30.61 (2.48)	26.35 (4.24)	24.50 (5.91)	23.30 (7.21)
	GLIDE-NLM	30.32 (2.19)	26.43 (4.32)	24.64 (6.05)	23.45 (7.36)
	PLOW	29.25 (1.12)	25.04 (2.93)	23.32 (4.73)	22.51 (6.42)
	BM3D	30.97 (2.84)	26.91 (4.79)	24.88 (6.29)	23.62 (7.53)
	AWF	30.46 (2.33)	26.51 (4.40)	24.51 (5.92)	23.17 (7.08)
	CAWF	*31.12 (2.98)*	*27.05 (4.94)*	*25.09 (6.50)*	*23.84 (7.74)*
Gazebo	Corrupted	28.13	22.11	18.59	16.09
768 × 512	NLM	31.11 (2.98)	26.71 (4.59)	24.93 (6.34)	23.77 (7.68)
	GLIDE-NLM	31.11 (2.98)	26.97 (4.86)	25.05 (6.46)	23.95 (7.86)
	PLOW	29.82 (1.68)	25.69 (3.58)	24.20 (5.61)	23.46 (7.36)
	BM3D	31.57 (3.44)	27.40 (5.29)	25.48 (6.89)	24.29 (8.19)
	AWF	30.78 (2.64)	26.88 (4.77)	24.91 (6.32)	23.59 (7.50)
	CAWF	*31.60 (3.47)*	*27.49 (5.37)*	*25.55 (6.96)*	*24.30 (8.21)*

Table 3 SSIM for additive Gaussian noise

Image	Method	SSIM			
		$\sigma_\eta = 10$	$\sigma_\eta = 20$	$\sigma_\eta = 30$	$\sigma_\eta = 40$
Aerial	Corrupted	0.9719	0.9046	0.8224	0.7388
491 × 434	NLM	0.9753	0.9058	0.8412	0.7872
	GLIDE-NLM	0.9747	0.9058	0.8427	0.7916
	PLOW	0.9734	0.9252	0.8760	0.8282
	BM3D	0.9783	0.9340	0.8835	0.8287
	AWF	0.9762	0.9304	0.8804	0.8322
	CAWF	*0.9787*	*0.9351*	***0.8859***	*0.8365*
Bridge	Corrupted	0.9542	0.8512	0.7365	0.6305
512^2	NLM	0.9611	0.8651	0.7925	0.7379
	GLIDE-NLM	0.9621	0.8757	0.8010	0.7363
	PLOW	0.9587	0.9051	0.8485	0.7943
	BM3D	*0.9692*	*0.9126*	*0.8539*	0.7963
	AWF	0.9652	0.9061	0.8477	0.7927
	CAWF	0.9676	0.9109	0.8523	*0.7985*
River	Corrupted	0.9512	0.8429	0.7252	0.6196
(Kodak 11)	NLM	0.9525	0.8343	0.7588	0.7042
768 × 512	GLIDE-NLM	0.9548	0.8461	0.7617	0.7007
	PLOW	0.9505	0.8732	0.7921	0.7299
	BM3D	0.9597	0.8792	0.8066	0.7437
	AWF	0.9574	0.8823	0.8142	0.7512
	CAWF	*0.9609*	*0.8873*	*0.8207*	*0.7634*
Bones	Corrupted	0.9259	0.7735	0.6224	0.4979
512 × 768	NLM	0.9329	0.7946	0.7269	0.6821
	GLIDE-NLM	0.9379	0.8163	0.7351	0.6725
	PLOW	0.9240	0.8295	0.7587	0.7047
	BM3D	0.9417	0.8404	0.7641	0.7091
	AWF	0.9389	0.8459	0.7669	0.6970
	CAWF	*0.9439*	*0.8551*	*0.7839*	*0.7277*
Building	Corrupted	0.9153	0.7660	0.6349	0.5318
768 × 512	NLM	0.9395	0.8370	0.7808	0.7386
	GLIDE-NLM	0.9426	0.8416	0.7835	0.7385
	PLOW	0.9379	0.8577	0.7823	0.7262
	BM3D	0.9482	0.8748	0.8141	0.7637
	AWF	0.9398	0.8583	0.7797	0.7072
	CAWF	*0.9502*	*0.8788*	*0.8190*	*0.7665*
Gazebo	Corrupted	0.9130	0.7678	0.6428	0.5435
768 × 512	NLM	0.9561	0.8653	0.8090	0.7649
	GLIDE-NLM	0.9578	0.8712	0.8117	0.7686
	PLOW	0.9525	0.8698	0.8046	0.7559
	BM3D	0.9620	0.8963	*0.8397*	*0.7940*
	AWF	0.9452	0.8740	0.7950	0.7230
	CAWF	*0.9611*	*0.8988*	0.8387	0.7837

with BM3D generally providing the next highest PSNR values. Looking at Table 3, we see that CAWF still performs well, but BM3D does provide a higher SSIM in 5 of 24 scenarios. These results also show that CAWF consistently outperforms AWF. This demonstrates the advantage of using multiple patches within this framework. It is also interesting to note that AWF itself does quite well compared with some of the benchmark methods on these data, especially in the SSIM metric.

Selected regions of interest (ROIs) from images bridge and river for the noise-only case with $\sigma_\eta = 30$ are shown in Figures 11 and 12, respectively. We find that BM3D tends to do a better job in smooth areas, and CAWF generally appears better in high-detail texture areas. Note that more branches on the small trees are visible in the CAWF estimate in Figure 11f, compared with that for BM3D in Figure 11e. Also, the texture in the tree foliage appears to be better preserved with CAWF processing in Figure 12f, compared with that for BM3D in Figure 12e.

The results in Figure 13 show how the CAWF method can produce an estimate of the MSE on a pixel-by-pixel basis. Figure 13a shows an ROI from the image

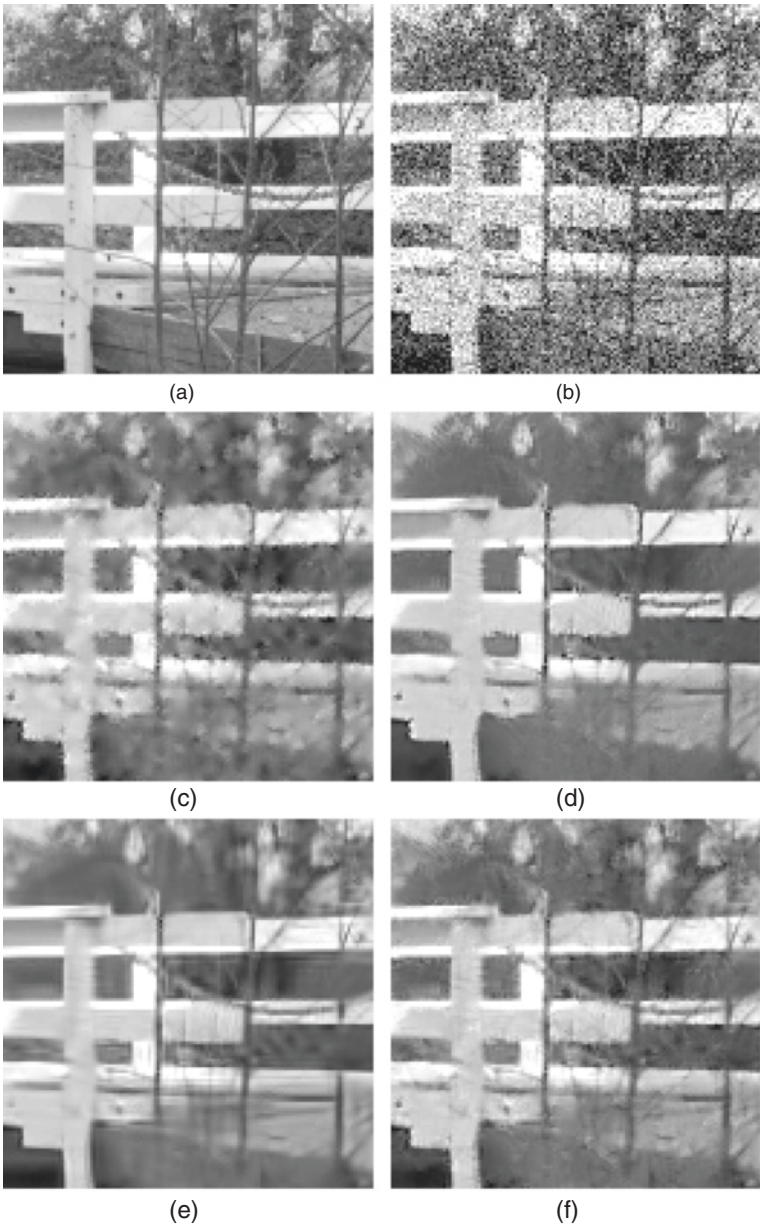

Figure 11 Region of interest from the image bridge with $\sigma_\eta = 30$ Gaussian noise. (a) Truth image, **(b)** noisy image, **(c)** AWF, **(d)** GLIDE-NLM, **(e)** BM3D, and **(f)** CAWF.

aerial with noise of standard deviation 10. The CAWF estimate image is shown in Figure 13b. The estimated MSE, computed according to Equation 26, is shown in Figure 13c. The average squared error over 100 noise realizations is shown in Figure 13d. Aside from some of the small high frequency structures, the estimated MSE is appears similar to the average squared error. The ability to provide an estimate of the MSE is another distinctive feature of the CAWF method among other multi-patch methods.

4.1.2 Gaussian blur plus Gaussian noise

We consider four scenarios of Gaussian blur plus Gaussian noise, and these are listed in Table 4. The benchmark methods in this case must be able to address both blur and noise. We use L0-Abs [47], TVMM [48], BM3DDEB [38], IDD-BM3D [39], and AWF [5]. Note that for IDD-BM3D, the tuning parameters are selected from those used in [39]. In particular, we use the tuning parameters from Scenario 4 in [39], as these produce the highest PSNR values in the current experiments.

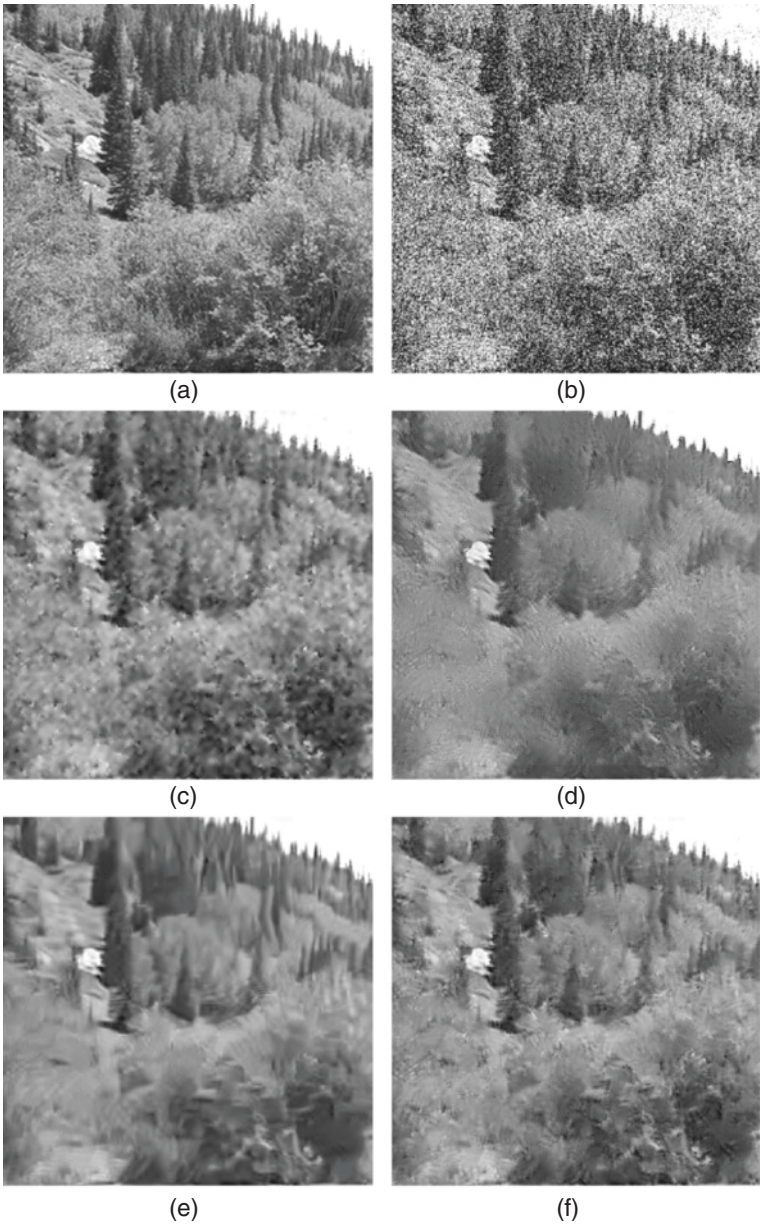

Figure 12 Region of interest from the image river with $\sigma_\eta = 30$ Gaussian noise. (a) Truth image, **(b)** noisy image, **(c)** AWF, **(d)** GLIDE-NLM, **(e)** BM3D, and **(f)** CAWF.

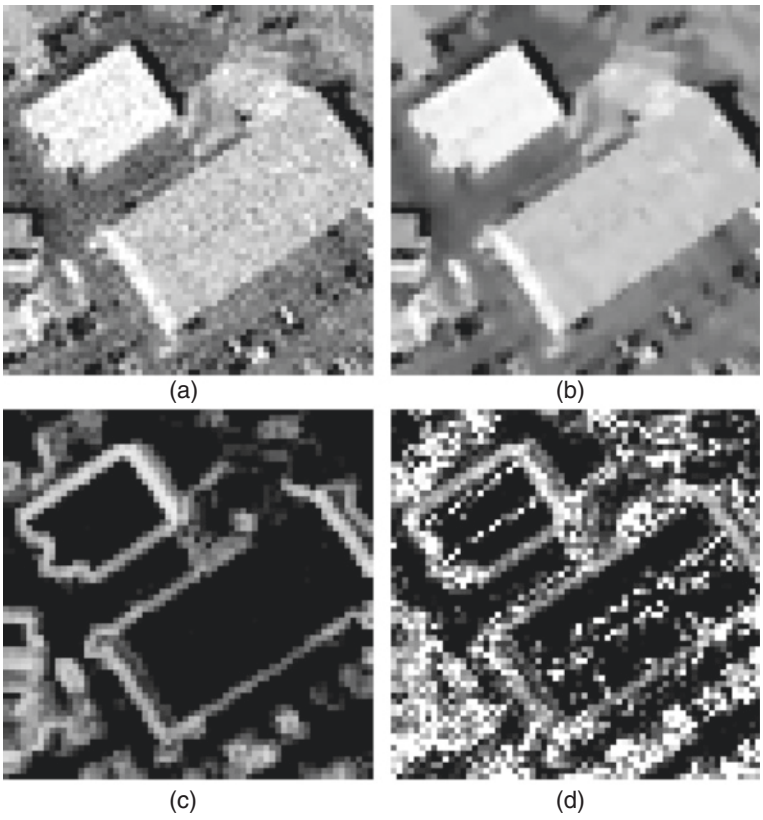

Figure 13 Predicted MSE for the image aerial with $\sigma_\eta = 10$ Gaussian noise. (a) Noisy image, **(b)** non-aggregated CAWF image estimate, **(c)** CAWF estimate of MSE, $\hat{J}_i$, and **(d)** average squared error over 100 noise realizations.

The PSNR comparison is provided in Table 5, and the SSIM comparison is in Table 6. Here, IDD-BM3D provides the highest PSNR in 15 of 24 instances, and CAWF provides the highest in 9 of 24. In terms of SSIM, IDD-BM3D provides the highest values in 5 of 24 instances and CAWF in 19 of 24. We see a similar situation with AWF as we did in the noise-only experiments. The AWF method does well compared to many of the benchmark methods, but CAWF consistently outperforms it. Selected ROIs from the images aerial and bones for Scenario III in Table 4 are shown in Figures 14 and 15, respectively. Again, CAWF appears to do a good job restoring image detail, based on subjective evaluation and SSIM. Note that a larger blur kernel generally demands a larger restoration filter window. The variational benchmark methods, like IDD-BM3D, are not restricted to local processing like CAWF and AWF. Thus, they may have an advantage in high levels of blur. However, the iterative nature of these methods also means that a full parallel implementation may not be possible.

Table 4 Scenarios of Gaussian blur and Gaussian noise

Scenario	PSF	σ_η
I	Gaussian std. = 1.0	10
II	Gaussian std. = 1.5	10
III	Gaussian std. = 1.0	20
IV	Gaussian std. = 1.5	20

4.2 Real data

Real video frames have been acquired of an outdoor natural scene on the campus of the University of Dayton using an Imaging Source 8 bit grayscale camera (DMK 23U618) with Sony ICX618ALA sensor. A short exposure time is used, proving a low SNR. A sequence of 500 frames is acquired for the static scene. This allows us to form a temporal average as a type of reference with which to compare the noise reduction estimates. Since this real noise will have both a signal-dependent and signal-independent component, we apply an Anscombe transform to stabilize the local noise variance prior to applying all denoising methods [49]. After the transform and scaling, an effective constant noise standard deviation of $\sigma_\eta = 9.11$ is estimated and used for all methods.

Table 5 PSNR comparison for Gaussian blur plus Gaussian noise

Image	Method	PSNR (ISNR)			
		Scenario I	Scenario II	Scenario III	Scenario IV
Aerial	Corrupted	22.64	20.79	19.98	18.88
491 × 434	L0-Abs	24.02 (1.38)	21.88 (1.10)	21.80 (1.82)	20.36 (1.48)
	TVMM	24.18 (1.54)	22.18 (1.40)	21.94 (1.95)	20.49 (1.60)
	BM3DDEB	24.41 (1.76)	22.56 (1.78)	22.54 (2.56)	21.32 (2.44)
	IDD-BM3D	*24.71 (2.06)*	*22.69 (1.90)*	22.71 (2.73)	21.32 (2.44)
	AWF	24.25 (1.61)	22.33 (1.55)	22.64 (2.66)	21.29 (2.41)
	CAWF	24.47 (1.83)	22.56 (1.78)	*22.83 (2.85)*	*21.43 (2.55)*
Bridge	Corrupted	24.19	22.80	20.76	20.07
512^2	L0-Abs	25.63 (1.44)	24.08 (1.28)	23.52 (2.77)	22.53 (2.47)
	TVMM	25.73 (1.53)	24.27 (1.47)	23.57 (2.82)	22.84 (2.77)
	BM3DDEB	26.02 (1.82)	24.71 (1.91)	24.47 (3.72)	23.62 (3.55)
	IDD-BM3D	*26.23 (2.03)*	*24.79 (2.00)*	*24.52 (3.76)*	*23.63 (3.56)*
	AWF	25.78 (1.59)	24.43 (1.63)	24.39 (3.63)	23.45 (3.38)
	CAWF	25.94 (1.74)	24.56 (1.77)	24.48 (3.72)	23.48 (3.41)
River	Corrupted	22.14	20.66	19.69	18.79
(Kodak 11)	L0-Abs	23.22 (1.08)	21.43 (0.77)	21.45 (1.76)	20.33 (1.54)
768 × 512	TVMM	23.13 (1.00)	21.52 (0.86)	21.14 (1.45)	20.33 (1.53)
	BM3DDEB	23.24 (1.11)	21.78 (1.12)	21.72 (2.03)	20.90 (2.11)
	IDD-BM3D	*23.63 (1.49)*	21.90 (1.24)	21.95 (2.26)	20.96 (2.17)
	AWF	23.29 (1.15)	21.78 (1.12)	22.06 (2.36)	21.06 (2.27)
	CAWF	23.48 (1.34)	*21.93 (1.28)*	*22.21 (2.52)*	*21.15 (2.36)*
Bones	Corrupted	24.80	23.78	21.01	20.56
512 × 768	L0-Abs	26.07 (1.27)	25.14 (1.36)	24.65 (3.64)	24.17 (3.62)
	TVMM	25.94 (1.15)	25.11 (1.33)	24.52 (3.51)	24.24 (3.68)
	BM3DDEB	26.33 (1.54)	25.48 (1.71)	25.23 (4.22)	24.79 (4.23)
	IDD-BM3D	*26.46 (1.66)*	*25.51 (1.74)*	25.26 (4.24)	*24.80 (4.24)*
	AWF	26.31 (1.51)	25.37 (1.59)	25.25 (4.24)	24.69 (4.14)
	CAWF	26.44 (1.64)	25.43 (1.66)	*25.28 (4.27)*	24.67 (4.12)
Building	Corrupted	22.60	21.27	19.96	19.18
768 × 512	L0-Abs	23.88 (1.28)	22.24 (0.97)	22.25 (2.29)	21.22 (2.04)
	TVMM	23.82 (1.22)	22.42 (1.16)	22.11 (2.15)	21.57 (2.39)
	BM3DDEB	23.86 (1.26)	22.53 (1.27)	22.49 (2.53)	21.73 (2.55)
	IDD-BM3D	*24.25 (1.65)*	*22.69 (1.43)*	22.75 (2.79)	21.85 (2.66)
	AWF	23.89 (1.28)	22.52 (1.26)	22.76 (2.80)	21.85 (2.66)
	CAWF	24.05 (1.45)	22.65 (1.38)	*22.89 (2.93)*	*21.92 (2.74)*
Gazebo	Corrupted	23.29	21.90	20.31	19.55
768 × 512	L0-Abs	24.82 (1.53)	23.11 (1.21)	23.02 (2.71)	21.93 (2.38)
	TVMM	24.80 (1.50)	23.41 (1.50)	23.34 (3.03)	22.17 (2.61)
	BM3DDEB	24.91 (1.61)	23.45 (1.55)	23.39 (3.08)	22.52 (2.97)
	IDD-BM3D	*25.29 (2.00)*	*23.65 (1.75)*	23.68 (3.37)	*22.66 (3.11)*
	AWF	24.83 (1.53)	23.37 (1.47)	23.58 (3.26)	22.58 (3.03)
	CAWF	24.99 (1.69)	23.51 (1.61)	*23.70 (3.39)*	22.66 (3.10)

Table 6 SSIM comparison for Gaussian blur plus Gaussian noise

Image	Method	SSIM			
		Scenario I	Scenario II	Scenario III	Scenario IV
Aerial	Corrupted	0.8891	0.7794	0.8176	0.7117
491 × 434	L0-Abs	0.9087	0.8050	0.7821	0.6448
	TVMM	0.9215	0.8325	0.7885	0.6604
	BM3DDEB	0.9198	0.8571	0.8400	0.7710
	IDD-BM3D	*0.9292*	0.8591	0.8406	0.7577
	AWF	0.9199	0.8519	0.8517	0.7733
	CAWF	0.9273	*0.8662*	*0.8649*	*0.7957*
Bridge	Corrupted	0.8836	0.7975	0.7745	0.6929
512^2	L0-Abs	0.8902	0.8044	0.7425	0.6524
	TVMM	0.8995	0.8156	0.7323	0.6689
	BM3DDEB	0.9099	0.8579	0.8294	0.7796
	IDD-BM3D	*0.9173*	0.8573	0.8252	0.7657
	AWF	0.9080	0.8503	0.8344	0.7712
	CAWF	0.9155	*0.8637*	*0.8462*	*0.7892*
River	Corrupted	0.8373	0.7198	0.7246	0.6161
(Kodak 11)	L0-Abs	0.8328	0.7159	0.6670	0.5560
768 × 512	TVMM	0.8452	0.7219	0.6344	0.5421
	BM3DDEB	0.8487	0.7713	0.7377	0.6745
	IDD-BM3D	0.8672	0.7767	0.7463	0.6666
	AWF	0.8573	0.7736	0.7676	0.6878
	CAWF	*0.8707*	*0.7949*	*0.7837*	*0.7105*
Bones	Corrupted	0.8231	0.7323	0.6681	0.5884
512 × 768	L0-Abs	0.8051	0.7287	0.6652	0.6181
	TVMM	0.7954	0.7192	0.6427	0.6172
	BM3DDEB	0.8320	0.7768	0.7420	0.7064
	IDD-BM3D	0.8421	0.7753	0.7389	0.6969
	AWF	0.8358	0.7699	0.7526	0.7000
	CAWF	*0.8476*	*0.7866*	*0.7615*	*0.7108*
Building	Corrupted	0.8119	0.7162	0.6609	0.5753
768 × 512	L0-Abs	0.8356	0.7491	0.7142	0.6397
	TVMM	0.8454	0.7592	0.6936	0.6615
	BM3DDEB	0.8520	0.7913	0.7633	0.7138
	IDD-BM3D	*0.8662*	0.7953	0.7715	0.7130
	AWF	0.8509	0.7848	0.7721	0.7147
	CAWF	0.8607	*0.7974*	*0.7816*	*0.7268*
Gazebo	Corrupted	0.8279	0.7413	0.6819	0.6030
768 × 512	L0-Abs	0.8651	0.7867	0.7527	0.6788
	TVMM	0.8741	0.8057	0.7785	0.6826
	BM3DDEB	0.8805	0.8249	0.8001	0.7500
	IDD-BM3D	*0.8898*	*0.8282*	0.8064	0.7504
	AWF	0.8796	0.8190	0.8035	0.7481
	CAWF	0.8801	0.8238	*0.8072*	*0.7561*

Figure 16a shows a 500 frame temporal average image. This image represents a near noise-free image of the scene that can be used as a reference. A single noisy frame is shown in Figure 16b. The processed single frame outputs for GLIDE-NLM, AWF, BM3D, and CAWF are shown in Figure 16c-f, respectively. The PSNR values, relative to the temporal average, for observed GLIDE-NLM, AWF, BM3D, and CAWF outputs are 31.78, 36.04, 36.33, 36.47, and 36.65, respectively. The corresponding SSIM values are 0.7525, 0.8929, 0.8975, 0.8973, and 0.9059. These results appear to be consistent with the results obtained with the simulated data.

4.3 Parameter and distance metric sensitivity

In this section, we investigate the sensitivity of the CAWF algorithm to some of the key tuning parameters listed in Table 1 and the distance metric used in the correlation model. We begin with the number of patches M. A plot of PSNR versus the number of similar patches is shown in Figure 17 for CAWF using the image bones with noise

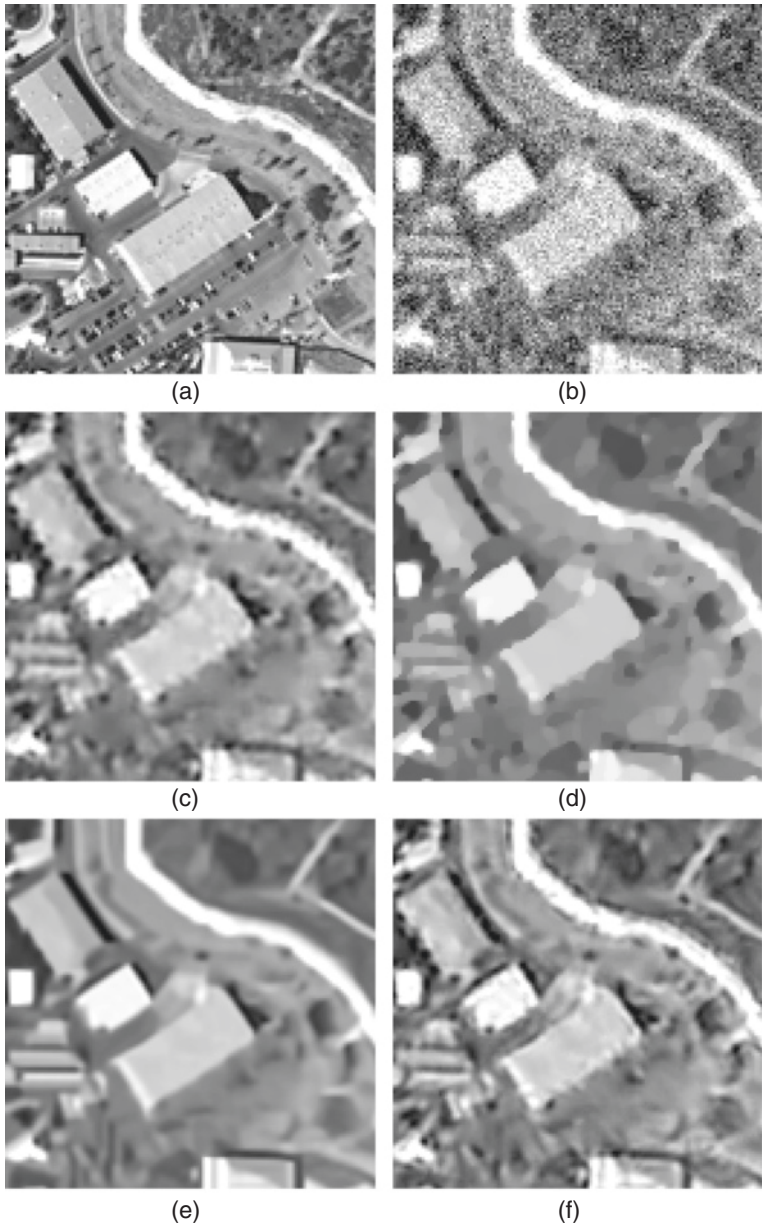

Figure 14 Region of interest from the image aerial for blur and noise Scenario III. (a) Truth image, **(b)** degraded image, **(c)** AWF, **(d)** TVMM, **(e)** IDD-BM3D, and **(f)** CAWF.

only and $\sigma_\eta = 40$. This plot is representative of many of the restoration scenarios. One can see a 'knee' in the curve near $M = 10$. Since increasing M has a significant impact on computational complexity, we have elected to use $M = 10$ for our denoising applications. Note that deconvolution requires a larger window size than denoising. Thus, to manage computational complexity, we compensate with a somewhat lower $M = 8$, as shown in Table 1.

Next, we examine the autocorrelation decay constant, ρ, and the patch similarity decay, α, for the image aerial with additive Gaussian noise. We have evaluated CAWF PSNR for ρ ranging from 0.6 to 0.75, with all other parameter values as listed in Table 1. The maximum change in PSNR as a function of ρ, for noise levels ranging from $\sigma_\eta = 10$ to $\sigma_\eta = 40$, is only 0.13%. Similarly, we have evaluated PSNR values for α ranging from 1.0 to 2.0. The maximum change in PSNR as a function of α is observed to be 0.23%. Thus, we conclude that the CAWF method is not highly sensitive to these tuning parameters within these operating ranges.

Finally, we explore CAWF performance using different distance metrics in the correlation model. Our standard

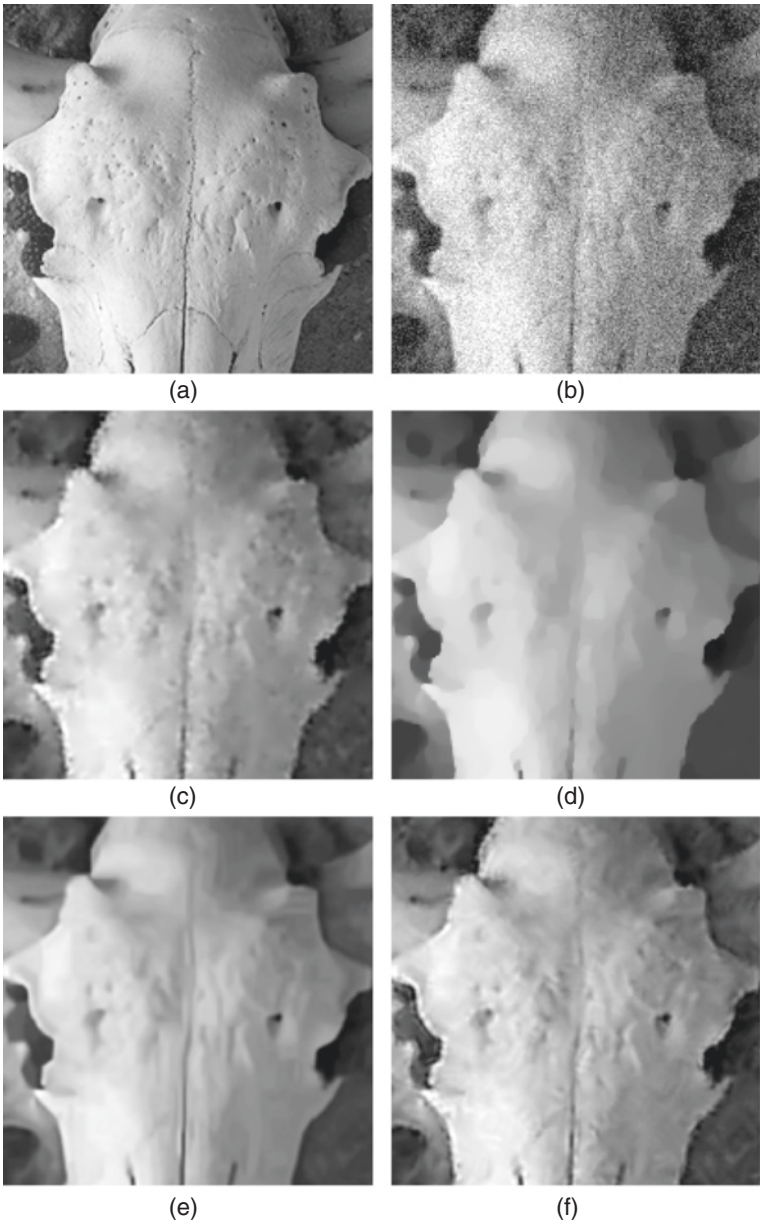

Figure 15 Region of interest from the image bones for blur and noise Scenario III. (a) Truth image, **(b)** degraded image, **(c)** AWF, **(d)** TVMM, **(e)** IDD-BM3D, and **(f)** CAWF.

metric uses the l^2-norm as defined in Equation 13. To test distance metric sensitivity, we compare distances with the l^1-, l^2-, and l^{10}-norms. For aerial with $\sigma_\eta = 10$, the PSNRs are 30.67, 30.67, and 30.60, respectively (with tuned scaling parameters). For aerial with $\sigma_\eta = 40$, the PSNRs are 23.01, 23.13, and 23.05, respectively (also with tuned scaling parameters). As with the other parameters, we do not see a strong sensitivity to the choice of distance metric. However, the l^2-norm generally provides the best results.

5 Conclusions

We have proposed a novel CAWF method for image restoration, which can be thought of an extension of the AWF [5] using multiple patches. For each reference window, M similar patches are identified. The output is formed as a single-pass weighted sum of all of the pixels from the multiple selected patches. Wiener weights are used to provide a minimum MSE estimate for this filter structure. A key aspect of the method is the new spatial-domain multi-patch correlation model, presented

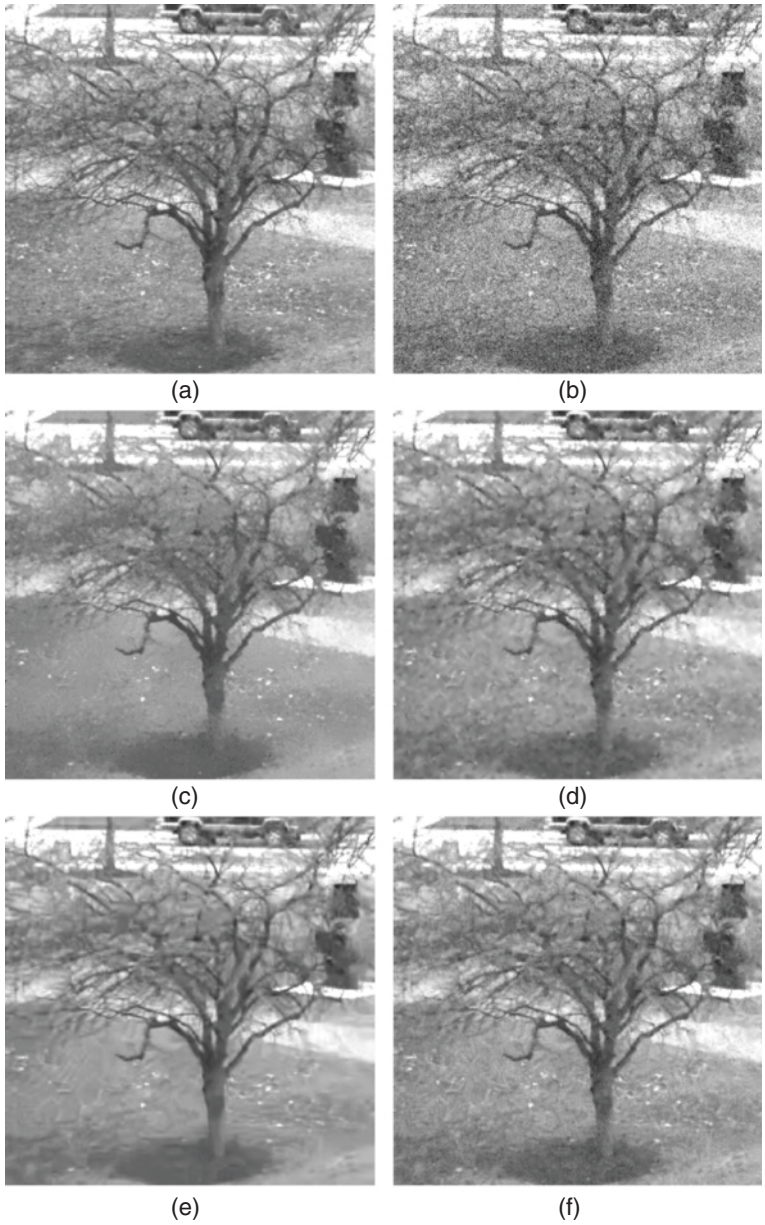

Figure 16 Region of interest from the real image data. (a) 500 frame temporal average image (reference image), **(b)** single observed frame, **(c)** GLIDE-NLM, **(d)** AWF, **(e)** BM3D, and **(f)** CAWF.

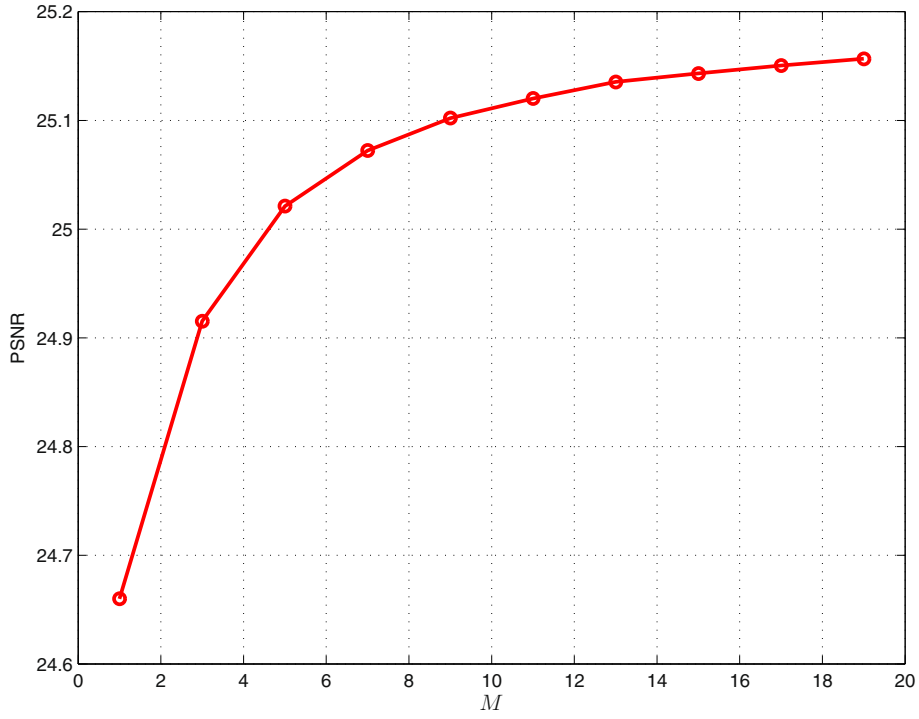

Figure 17 PSNSR versus the number of similar patches *M* for CAWF. Degradation is noise only with Bones image and $\sigma_\eta = 40$.

in Section 3.2. This model attempts to capture the spatial correlation among the samples within a given patch and also the correlations among the patches.

The CAWF is able to jointly perform denoising and deblurring. We believe this type of joint restoration is advantageous, compared with decoupling these operations. The CAWF algorithm is also capable of adapting to local signal and noise variance. Bad or missing pixels can easily be accommodated by leaving them out of the multi-patch observation vector and corresponding correlation statistics. The weights will adapt in a non-trivial way to the missing pixels [5,9,10].

In simulated and real data for Gaussian noise, the CAWF outperforms the benchmark methods in our experiments in Sections 4.1.1 and 4.2, both in PSNR and in SSIM. With blur and noise, CAWF produces the highest SSIM in more cases than the benchmark methods. However, IDD-BM3D does provide a higher PSNR in more instances. Our results show that the CAWF method consistently outperforms the AWF. This clearly demonstrates that incorporating multiple patches within this filter structure is advantageous. From the results in Section 4.3, we also conclude that CAWF performance is not highly sensitive to the tuning parameter values within a given operating range.

We believe the single-pass weighted-sum structure of the CAWF method is conceptually simple and versatile. It is also highly parallel. In principle, each output pixel can be computed in parallel. We have demonstrated that the method provides excellent performance in image restoration with noise and blur and noise. This method may be beneficial in numerous other applications as well, including those where its predecessor, the AWF, is successful [5,9-14]. We think there may also be an opportunity for further improvements in the parametric correlation model that could boost filter performance. Thus, we hope this approach will be of interest to the signal and image processing community.

Competing interests
The authors declare that they have no competing interests.

Received: 22 July 2014 Accepted: 23 December 2014
Published online: 27 January 2015

References

1. G Demoment, Image reconstruction and restoration: overview of common estimation structures and problems. IEEE Trans Acoustics, Speech Signal Process. **37**(12), 2024–36 (1989)
2. MI Sezan, AM Tekalp, Survey of recent developments in digital image restoration. Opt Eng. **29**(5), 393–404 (1990)
3. AK Jain, *Fundamentals of Digital Image Processing*. (Prentice Hall, New Jersey, 1989)
4. SP Ghael, AM Sayeed, RG Baraniuk, in *Proc of SPIE*. Improved wavelet denoising via empirical Wiener filtering, vol. 3169 (SPIE San Diego, 1997), pp. 389–99
5. RC Hardie, A fast image super-resolution algorithm using an adaptive Wiener filter. IEEE Transactions on Image Process. **16**, 2953–64 (Dec. 2007)

6. KE Barner, AM Sarhan, RC Hardie, Partition-based weighted sum filters for image restoration. IEEE Trans Image Process. **8**, 740–745 (May 1999)
7. M Shao, KE Barner, RC Hardie, Partition-based interpolation for image demosaicing and super-resolution reconstruction. Opt Eng. **44**, 107003–1–107003–14 (Oct 2005)
8. B Narayanan, RC Hardie, KE Barner, M Shao, A computationally efficient super-resolution algorithm for video processing using partition filters. IEEE Trans Circuits Syst Video Technol. **17**, 621–34 (May 2007)
9. RC Hardie, KJ Barnard, R Ordonez, Fast super-resolution with affine motion using an adaptive Wiener filter and its application to airborne imaging. Opt Express, 1926208–31 (Dec 2011)
10. RC Hardie, KJ Barnard, Fast super-resolution using an adaptive Wiener filter with robustness to local motion. Opt Express. **20**, 21053–73 (Sep 2012)
11. B Narayanan, RC Hardie, E Balster, Multiframe adaptive Wiener filter super-resolution with JPEG2000-compressed images. EURASIP J Adv Signal Process. **2014**(1), 55 (2014)
12. RC Hardie, DA LeMaster, BM Ratliff, Super-resolution for imagery from integrated microgrid polarimeters. Opt Express. **19**, 12937–60 (Jul 2011)
13. BK Karch, RC Hardie, Adaptive Wiener filter super-resolution of color filter array images. Opt Express, 2118820–41 (Aug 2013)
14. M Rucci, RC Hardie, KJ Barnard, 53. Appl Opt, C1–13 (May 2014)
15. A Buades, B Coll, JM Morel, A review of image denoising algorithms, with a new one. Multiscale Model Simul. **4**, 490–530 (2005)
16. KE Barner, GR Arce, J-H Lin, On the performance of stack filters and vector detection in image restoration. Circuits Syst Signal Process. **11**, No. 1 (Jan 1992)
17. KE Barner, RC Hardie, GR Arce, in *Proceedings of the 1994 CISS*. On the permutation and quantization partitioning of $\mathbf{R}^N$ and the filtering problem (New Jersey, Princeton, Mar 1994)
18. A Buades, B Coll, JM Morel, in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Comput Soc Conference on, vol. 2*. A non-local algorithm for image denoising (IEEE, June 2005), pp. 60–65
19. C Kervrann, J Boulanger, Optimal spatial adaptation for patch-based image denoising. IEEE Trans Image Process. **15**(10), 2866–78 (2006)
20. A Buades, B Coll, J-M Morel, Nonlocal image and movie denoising. Int J Comput Vision. **76**, 123–39 (Feb 2008)
21. Y Han, R Chen, Efficient video denoising based on dynamic nonlocal means. Image Vision Comput. **30**, 78–85 (Feb 2012)
22. Tasdizen, Principal neighborhood dictionaries for nonlocal means image denoising. IEEE Trans Image Process. **18**, 2649–60 (July 2009)
23. H Bhujle, S Chaudhuri, Novel speed-up strategies for non-local means denoising with patch and edge patch based dictionaries. IEEE Trans Image Process. **23**, 356–365 (Jan 2014)
24. Y Wu, B Tracey, P Natarajan, JP Noonan, SUSAN controlled decay parameter adaption for non-local means image denoising. Electron Lett. **49**, 807–8 (June 2013)
25. WL Zeng, XB Lu, Region-based non-local means algorithm for noise removal. Electron Lett. **47**, 1125–7 (September 2011)
26. WF Sun, YH Peng, WL Hwang, Modified similarity metric for non-local means algorithm. Electron Lett. **45**, 1307–9 (Dec 2009)
27. C Tomasi, R Manduchi, in *Proceedings of the 1998 IEEE International Conference on Computer Vision, Bombay*. Bilateral filtering for gray and color images (IEEE India, 1998)
28. H Kishan, CS Seelamantula, *Sure-fast bilateral filters. Acoustics, Speech and Signal Processing (ICASSP) 2012 IEEE International Conference on*. (IEEE, Kyoto, 2012), pp. 1129–32
29. W Kesjindatanawaj, S Srisuk, in *Communications and Information Technologies (ISCIT), 2013 13th International Symposium on*. Deciles-based bilateral filtering (IEEE Surat Thani, 2013), pp. 429–33
30. X Changzhen, C Licong, P Yigui, *An adaptive bilateral filtering algorithm and its application in edge detection. Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on. vol. 1*. (IEEE, Changsha City, 2010), pp. 440–443
31. H Peng, R Rao, SA Dianat, Multispectral image denoising with optimized vector bilateral filter. IEEE Trans Image Process. **23**, 264–73 (Jan 2014)
32. K Dabov, A Foi, V Katkovnik, K Egiazarian, Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Trans Image Process. **16**, 2080–95 (Aug 2007)
33. K Dabov, A Foi, K Egiazarian, Video denoising by sparse 3d transform-domain collaborative filtering. Proc 15th Eur Signal Process Conference. **1**, 7 (2007)
34. M Maggioni, G Boracchi, A Foi, K Egiazarian, Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms. IEEE Trans Image Process. **21**(9), 3952–66 (2012)
35. K Hirakawa, T Parks, Image denoising using total least squares. IEEE Trans Image Process. **15**(9), 2730–42 (2006)
36. P Chatterjee, P Milanfar, Patch-based near-optimal image denoising. IEEE Trans Image Process. **21**, 1635–49 (April 2012)
37. H Talebi, P Milanfar, Global image denoising. IEEE Trans Image Process. **23**, 755–768 (Feb 2014)
38. K Dabov, A Foi, V Katkovnik, K Egiazarian, in *SPIE Electronic Imaging*. Image restoration by sparse 3d transform-domain collaborative filtering, vol. 6812 (San Jose, Jan 2008)
39. A Danielyan, V Katkovnik, K Egiazarian, BM3D frames and variational image deblurring. IEEE Trans Image Process. **21**, 1715–28 (April 2012)
40. K Nasrollahi, TB Moeslund, Super-resolution: a comprehensive survey. Mach Vision Appl. **25**(6), 1423–68 (Aug 2014)
41. M Protter, M Elad, Super-resolution with probabilistic motion estimation. IEEE Trans Image Process. **18**(8), 1899–904 (2009)
42. M Protter, M Elad, H Takeda, P Milanfar, Generalizing the nonlocal-means to super-resolution reconstruction. IEEE Trans Image Process. **18**(1), 36–51 (2009)
43. MH Cheng, HY Chen, JJ Leou, Video super-resolution reconstruction using a mobile search strategy and adaptive patch size. Signal Process. **91**, 1284–97 (2011)
44. B Huhle, T Schairer, P Jenke, W Straber, Fusion of range and color images for denoising and resolution enhancement with a non-local filter. Comput Vision Image Understanding. **114**, 1336–45 (2012)
45. K-W Hung, W-C Siu, Single image super-resolution using iterative Wiener filter. Proc IEEE Int Conference Acoustics, Speech and Signal Process, 1269–72 (2012)
46. Z Wang, A Bovik, H Sheikh, E Simoncelli, Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process. **13**, 600–12 (April 2004)
47. J Portilla, Image restoration through l0 analysis-based sparse optimization in tight frames. Image Process (ICIP), 2009 16th IEEE Int Conference on, 3909–912 (Nov 2009)
48. J Oliveira, JM Bioucas-Dias, MA Figueire, Adaptive total variation image deblurring: A majorization-minimization approach. Signal Process. **89**, 1683–93 (Sep 2009)
49. M Makitalo, F Foi, Optimal inversion of the generalized Anscombe transformation for Poisson-Gaussian noise. IEEE Trans Image Process. **22**(1), 91–103 (2013)