Natural Scene Statistics for Image Denoising

Jeffrey S. Perry, Wilson S. Geisler

***Abstract*—A method for denoising digital camera images is described. The method is based on directly measuring the local statistical structure of natural images in a large training set that has been corrupted with noise mimicking digital camera noise. The measured statistics are conditional means of the ground truth pixel value given a local context of input pixels. Each conditional mean is the Bayes optimal (minimum mean squared error) estimate given the specific local context. The conditional means are measured and applied recursively (e.g., the second conditional mean is measured after denoising with the first conditional mean). Each local context vector consists of only three variables, and hence the conditional means can be measured directly without prior assumptions about the underlying probability distributions, and they can be stored in fixed lookup tables. Performance accuracy matches state-of-the-art algorithms on additive white Gaussian noise (AWGN) and multiplicative white Gaussian noise (MWGN). Performance accuracy exceeds state-of-the-art algorithms on realistic camera noise, which is multiplicative and correlated (MCGN). Performance speed greatly exceeds state-of-the-art algorithms because the estimates are obtained by applying only a few fixed tables, each indexed by only a few pixel values.**

***Index Terms*—Bayesian methods, higher order statistics, image denoising, natural scene statistics, statistical learning.**

I. Introduction

PHOTON and sensor noise limit the performance of all imaging systems. Minimizing the effects of this noise is a universal and fundamental image processing task. Here we address the problem of denoising in still digital camera images, using a new approach that combines measurements of natural image statistics with measurements of the noise characteristics of digital cameras.

In general, a noisy image can be represented as an unknown "true" image that has been corrupted by noise. Let $z(\mathbf{x})$ represent the value of a pixel at location $\mathbf{x} = (x, y)$ in the true image. Without loss of generality, the observed value is given by $z_0(\mathbf{x}) = z(\mathbf{x}) + n(\mathbf{x}, \mathbf{z})$, where $n(\mathbf{x}, \mathbf{z})$ is the noise, which may be spatially correlated and/or dependent on the true image values $\mathbf{z}$.

The goal of denoising is to estimate $z(\mathbf{x})$ given the observed context of pixel values $c(\mathbf{x})$ at and around the pixel location. The optimal estimate is given by the standard formula from Bayesian statistical decision theory:

$$\hat{z}_{opt}(\mathbf{x}) = \arg\min_{\hat{z}(\mathbf{x})} \sum_{z(\mathbf{x})} \gamma\left[z(\mathbf{x}), \hat{z}(\mathbf{x})\right] p\left[z(\mathbf{x}) \middle| \mathbf{c}(\mathbf{x})\right] \qquad (1)$$

where $\gamma[z(\mathbf{x}), \hat{z}(\mathbf{x})]$ is the cost function, and $p[z(\mathbf{x})|\mathbf{c}(\mathbf{x})]$ is the posterior probability of the true value given the observed context.

A vast number of different denoising methods have been proposed over the past several decades (for recent summaries see [1, 2]). They can all be viewed as providing some form of sub-optimal approximation to the Bayes optimal estimate given by equation (1).

Most often, the explicit (or implicit) cost function is the squared error between the estimated and true pixel values: $\gamma\left[z(\mathbf{x}), \hat{z}(\mathbf{x})\right] = \left[z(\mathbf{x}) - \hat{z}(\mathbf{x})\right]^2$. This cost function finds the estimate with the minimum mean squared error (MMSE) or equivalently the estimate with the maximum peak signal-to-noise ratio (PSNR). Other cost functions, such as those that are based on perceptual properties of the human visual system [3], are worthy of consideration; however, as is common in the denoising literature, we will focus here on the squared-error cost function. For this cost function, equation (1) becomes:

$$\hat{z}_{opt}(\mathbf{x}) = E\left[z(\mathbf{x}) \middle| \mathbf{c}(\mathbf{x})\right] \qquad (2)$$

In other words, the Bayes optimal estimate is simply the expected value of the true pixel value given the observed context [e.g., 4].

To develop an optimal denoising method for a specific application, one must characterize both the signal (the statistical structure of true images) and the noise (the statistical structure of the noise). The various denoising methods can be distinguished based on assumptions they make about the structure of the signal and noise. Also important is the computational efficiency (speed and complexity). For a given application, the best method will be the one that jointly maximizes the approximation to equation (2) and the computational efficiency.

The earliest principled denoising method is the Wiener filter [5], which is an exact implementation of equation (2), under

This work was supported by NIH Grant EY11747.
J. S. Perry and W. S. Geisler are with the Center for Perceptual systems at the University of Texas at Austin. (512-471-5380, jsp@mail.utexas.edu, geisler@mail.cps.utexas.edu).

the assumption that both the signal and the noise are described by stationary (not necessarily white) Gaussian processes. However, images are generally non-stationary and hence this method does not produce good results for most images (it blurs edges and texture). Subsequently, there have been many attempts to weaken the assumption of global stationarity. Adaptive Wiener filtering methods assume Gaussian noise and signal that is locally stationary; the methods estimate the Gaussian parameters at each pixel location and then apply the Wiener filter with those parameters [e.g., 6]. A closely related approach combines image segmentation and Bayesian MAP estimation [7]. The critical component of these methods is estimating the local Gaussian parameters; the less noisy the estimated parameters the more accurate the denoising. Simple non-iterative methods for estimating the parameters that use only pixels in the immediate neighborhood of the pixel being denoised can be computationally efficient.

Other recent methods do not make explicit formal assumptions about the structure of the noise or signal, but instead exploit heuristic intuitions to average out the noise and leave the signal. One simple and effective method of this type is bilateral filtering [8] which takes the weighted average of pixels in the local neighborhood, where the weights depend jointly on the spatial and gray-level (color) distance of the neighboring pixel from the pixel being denoised. The intuition is that spatially nearby pixels are positively correlated in gray level and can be averaged, but spatially nearby pixels that differ substantially in gray level usually contain strong signals (true image features) and should not be averaged. This method can be computationally efficient.

Related methods are those based on non-local averaging [9]. For example, the NL-Means algorithm [1] searches for pixels whose local neighborhood in the image is similar to the neighborhood of the pixel being denoised. It then averages all these pixels to obtain the estimate. The more similar is the local neighborhood the greater is the weight given to the pixel when computing the average. The intuition is that natural images are statistically regular and hence if two image patches are similar in structure it is likely that the center pixels are similar and hence can be averaged to estimate the true image value. To the extent that this assumption is valid for the kind of noise in an imaging system and for the kinds of images being captured, such averaging could provide a good approximation to the right side of equation (2). Indeed, methods based on non-local averaging provide good results and are currently popular. However, these methods are less computationally efficient because of the need to make the neighborhood similarity measurements.

Another class of methods involves hard or soft thresholding following a linear transform, such as a wavelet or discrete cosine transform [10, 11]. The intuition is that for appropriately chosen kernel shapes, the regular structure of natural images results in a very sparse representation (a few large kernel coefficients, with most near zero), whereas the much more random structure of noise results in a less sparse representation (many coefficients with modest values). Thus, thresholding out the smaller coefficients selectively removes the noise. These methods can be computationally efficient, but can be prone to producing ringing artifacts.

Currently the best performing denoising methods are hybrid methods [12,13]. For example the BM3D method combines non-local averaging, cooperative linear transform thresholding, and Wiener filtering [12].

In summary, most of the existing methods either assume Gaussian image and noise models, or principled heuristics based on qualitative properties of natural images. Further, the parameters of most denoising methods are estimated from the image being denoised.

Here, we propose a method, *Recursive Conditional Means* (*RCM*) denoising, that makes almost no assumptions about the underlying probability distributions, and that learns its estimates from the statistics of a large database of natural images. The fundamental idea is to directly measure the conditional means on the right side of equation (2) recursively for a number of different small neighborhoods (context regions). During denoising, each of these different neighborhoods provides an improved estimate of the denoised image pixel. In previous work we have found that for the task of upsampling (super resolution), this approach is every effective and computationally efficient [14].

Conceptually, our approach is similar to non-local methods in that an estimate is based on the average across a large number of similar neighborhoods. The difference is that RCM neighborhoods are small enough that similar neighborhoods are identical, and hence there is no need for an arbitrary definition of similarity. Also, the neighborhoods are small enough that the conditional means can be learned precisely from a large set of natural images and then stored in tables. Thus, unlike the non-local methods there are never “unique patches” for which a large estimation error is made. We find that RCM denoising is competitive with the best performing denoising methods, and that it is extremely fast computationally because all the relevant statistics are stored in fixed tables that can then be applied to any image.

An important general consideration is that denoising methods are almost always tested by adding constant-variance white Gaussian noise (AWGN) to gamma-compressed ground truth images. However, the noise in most digital cameras is more complex. The noise in the raw image from a CCD or CMOS sensor array is generally statistically independent and multiplicative: the variance of the noise is proportional to the mean intensity falling on the pixel. This noise occurs prior to gamma compression, and thus differs from additive noise following gamma compression. (Note, however, that additive

noise following gamma compression is similar to multiplicative noise prior to gamma compression, explaining in part the popularity of AWGN.) Further, the digital camera noise becomes spatially correlated after the standard image processing in the camera's hardware or firmware, which typically involves color interpolation (demosaicing) and conversion to a standard display format such as sRGB. Thus, even a "lossless" tiff or png camera image contains spatially correlated noise. It is quite possible that methods that work well for AWGN will perform more poorly on noisy digital camera images. Therefore, both for training and testing denoising methods it is critical to consider the actual noise in digital cameras [7,15].

In what follows we first measure and characterize the noise in a high quality digital camera. This camera noise model is then used to simulate the effects of the camera's noise by adding model noise to a large set of ground-truth images to obtain training and test images. Next, we describe the details of RCM denoising. Finally, we compare RCM denoising with other methods, both in terms of mean squared error (MSE/PSNR) and subjective appearance.

II. Digital Camera Noise

A. Noise measurements

Noise was measured in a Nikon D700 camera. Images were captured of a fixed uniform light field created from a tungsten light source. The shutter speed was set at 1/60 sec. The camera's aperture was adjusted in small increments so that images varied from completely black fields to maximally white fields. Measurements were repeated for a range of ISO settings: 200, 400, 640, 800, 1000, 1600, and 3200. Note that ISO represents the gain applied to CCD/CMOS elements, and thus the greater the ISO, the noisier the images. Higher ISO values are typically needed for low light levels or for stop-action (fast shutter speeds).

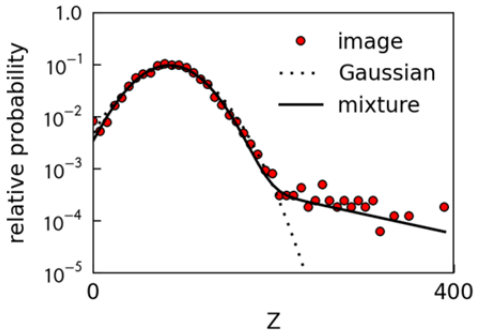


Fig. 1. Noise model fit for 14-bit flat-field red pixel image with mean value of 89.5. A Gaussian noise model does not fit well when the image pixel values are low because of the higher kurtosis of the image's noise distribution. A mixture of a Gaussian and Laplace distribution provides a better fit.

Statistical analysis was carried out separately for the R, G and B pixel locations in the 14-bit raw images. In order to control for the effect of vignetting, only the center 128x128 patch of pixels from each image was analyzed. Figure 1 shows the distribution of R pixel values for one ISO and aperture setting. The distributions often differ slightly from Gaussian and are better described by a mixture of a Gaussian and a Laplace distribution (black curve in Figure 1):

$$f(z) = \alpha \frac{1}{\sqrt{2\pi}\sigma_N} e^{\frac{(z-\bar{z})^2}{2\sigma_N^2}} + (1-\alpha)\frac{1}{\sqrt{2}\sigma_L} e^{-\frac{\sqrt{2}|z-\bar{z}|}{\sigma_L}} \quad (3)$$

where σ_N and σ_L are the standard deviations of the Gaussian and Laplace distributions, and $0 \le \alpha \le 1$. Each measured distribution was fitted with this mixture function. Figure 2 plots the fitted variance parameters as a function of the mean pixel value, for R, G and B pixels, at ISO_{3200}. As can be seen, the variances increase approximately linearly. We used the best fitting linear functions to summarize the camera noise for each ISO setting. The camera noise increases with the ISO setting, and thus the plots in Figure 2 show the highest noise levels we measured.

As a check, the measurements were repeated with a shutter speed of 1/250 s (and larger aperture). As expected, the noise parameters were independent of the shutter speed.

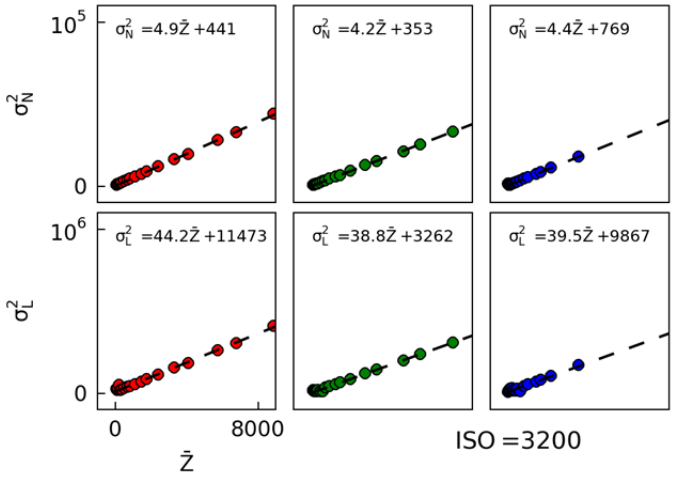


Fig. 2. Measured variances of Gaussian and Laplace distributions from flat field images. The measurements were summarized with linear equations.

B. Noise simulation

The lowest standard ISO setting (low noise) available in the D700 camera is ISO_{200}; thus, we took natural images captured with this ISO to be ground truth images. Training and test images for higher ISO settings were then created by adding simulated camera noise. In particular, for any value $\bar{z}$ in the ground truth image we added a random value ΔZ by sampling from the density function in equation (3) with the standard deviations for the Gaussian and Laplace density functions given by

$$\sigma_N = \Delta\sigma_N(\bar{z}) = \sqrt{\sigma_N^2(\bar{z}; ISO_T) - \sigma_N^2(\bar{z}; ISO_{200})} \quad (4)$$

and,

$$\sigma_L = \Delta\sigma_L(\bar{z}) = \sqrt{\sigma_L^2(\bar{z}; ISO_T) - \sigma_L^2(\bar{z}; ISO_{200})} \quad (5)$$

The value of α was determined by averaging the value of α for all fits across all channels. The average value of α was 0.975, and the standard deviation of α was 0.027.

The variance of the simulated noise increases in proportion to the mean value and hence is "multiplicative" noise. Further, the simulated noise is generated independently for each pixel location. Because the noise is approximately Gaussian we will refer to the simulated noise in raw images as multiplicative white Gaussian noise (MWGN).

The above steps simulate the noise in raw images, but most applications involve denoising images that have been interpolated (demosaiced), converted to a standard color format (usually linear sRGB), gamma compressed, and finally quantized to 8-bits per color channel (24-bit sRGB). We can simulate these cases simply be processing the raw ground truth and the simulated raw test/training images through the standard processing steps [7]. For example, Figure 3a shows a cropped region from an ISO_{200} raw image, after conversion to standard 24-bit sRGB. Figure 3b shows a cropped image of the same scene taken at ISO_{3200}. Figure 3c shows the result of adding simulated noise to the ISO_{200} raw image.

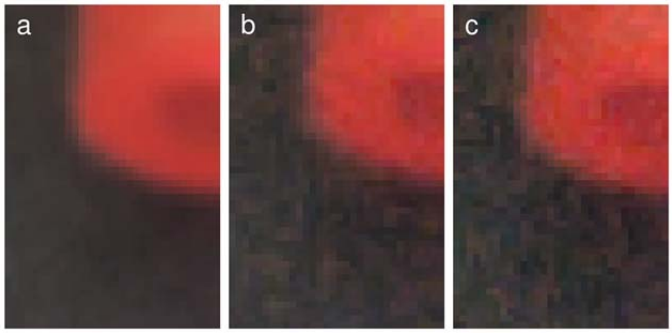


Fig. 3. Synthesized noise. (a) Cropped region of an image taken with camera ISO_{200}. (b) The same image taken with ISO_{3200}. (c) The region in *a* with noise added to create a synthesized ISO_{3200} image.

The noise in the real and simulated sRGB images is clearly spatially correlated. We will call this multiplicative correlated Gaussian noise (MCGN). Figure 4 compares the synthesized noise in a sRGB camera image with that of standard additive white Gaussian noise (AWGN) of similar noise power. Note that the white noise was (as is standard) added after gamma compression. The RCM method of denoising can be applied to any kind of noise. The emphasis here is on denoising sRGB images (MCGN), but we also consider multiplicative and additive white Gaussian noise (MWGN and AWGN).

III. RCM Denoising

A. Recursive conditional means and variances

As mentioned earlier, the key concept of RCM denoising is to measure conditional means for different local contexts. The conditional mean for each context provides the Bayes optimal (MMSE) estimate given that context. The number of variables constituting each local context is chosen to be small so that the conditional means can be measured accurately from training images without making assumptions about the underlying probability distributions. By measuring and applying the conditional means recursively the effective size of the context is expanded.

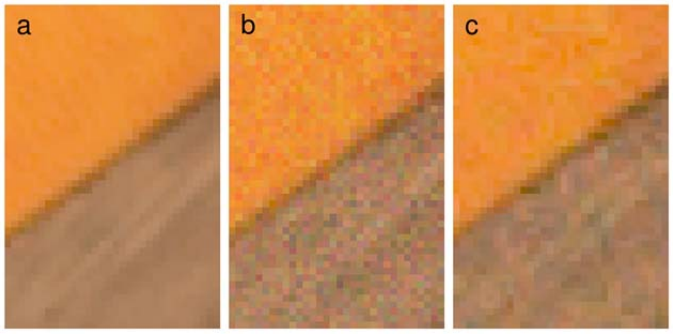


Fig. 4. Synthesized ISO noise versus traditional AWGN (additive white Gaussian noise). (a) Detail of original image. (b) Same detail with synthesized AWGN, MSE across entire image=55.8. (c) Same detail with synthesized MCGN (ISO_{3200}), MSE across entire image=56.3.

To be more precise, let $z_0(\mathbf{x})$ represent the input image, let $\mathbf{c}_i(\mathbf{x})$ be the context vector used to obtain the i^{th} recursively estimated image $z_i(\mathbf{x})$, and let $z(\mathbf{x})$ represent the ground truth image. Thus, the Bayes optimal estimate of $z(\mathbf{x})$ on iteration i is given by

$$z_i(\mathbf{x}) = E\left[z(\mathbf{x}) \middle| \mathbf{c}_i(\mathbf{x}) \right] \tag{6}$$

where the context vector on iteration i is obtained from image $z_{i-1}(\mathbf{x})$. The number of iterations n is set based on when the estimation accuracy reaches asymptote.

For symmetry, we typically also estimate images with all the context vectors rotated by 90 deg. This slows the computation speed, but does not require estimating additional tables. This second set of estimated images can be written as

$$z_i^{\perp}(\mathbf{x}) = E\left[z(\mathbf{x}) \middle| \mathbf{c}_i^{\perp}(\mathbf{x}) \right] \tag{7}$$

The last estimated images, $z_n(\mathbf{x})$ and $z_n^{\perp}(\mathbf{x})$, can be combined using their reliabilities:

$$z_n^*(\mathbf{x}) = \frac{\rho_n(\mathbf{x}) z_n(\mathbf{x}) + \rho_n^{\perp}(\mathbf{x}) z_n^{\perp}(\mathbf{x}) - \rho u}{\rho_n(\mathbf{x}) + \rho_n^{\perp}(\mathbf{x}) - \rho} \tag{8}$$

Where the mean of the prior is $u = E[z(\mathbf{x})]$, and reliabilities are $\rho_n(\mathbf{x}) = 1/Var[z(\mathbf{x})|\mathbf{c}_n(\mathbf{x})]$, $\rho_n^{\perp}(\mathbf{x}) = 1/Var[z(\mathbf{x})|c_n^{\perp}(\mathbf{x})]$, and $\rho = 1/Var[z(\mathbf{x})]$ (see Appendix A). Note that if the reliabilities of the two estimates are approximately equal and are much larger than ρ, then equation (8) reduces to the simple average:

$$z_n^*(\mathbf{x}) = \frac{z_n(\mathbf{x}) + z_n^{\perp}(\mathbf{x})}{2} \tag{9}$$

As it turned out, for the context vectors used here, we found that the reliabilities are approximately equal and much larger than ρ; thus, all the results reported here are for equation (9).

To apply this estimation method, one must specify the conditional means, and if it is necessary to combine estimates with equation (8) rather than equation (9), then one must also specify the conditional variances. The approach taken here is to directly measure the conditional means and variances from a large set of training images. Let $\{z(\mathbf{x},1), \ldots, z(\mathbf{x},k)\}$ be a set of k ground truth images indexed by j, and let $\{z_0(\mathbf{x},1), \ldots, z_0(\mathbf{x},k)\}$ be the corresponding training images. The sample conditional mean for a specific context vector $\mathbf{c}_i$ is given by

$$m_i[\mathbf{c}_i] = \frac{\sum_{(\mathbf{x},j)\in\Omega(\mathbf{c}_i)} z(\mathbf{x},j)}{N(\mathbf{c}_i)} \tag{10}$$

where $\Omega(\mathbf{c}_i)$ is the set of locations in the training images with context $\mathbf{c}_i$, and $N(\mathbf{c}_i)$ is the total number of locations in the set. Similarly, the sample variance (if needed) is given by

$$v_i[\mathbf{c}_i] = \frac{\sum_{(\mathbf{x},j)\in\Omega(\mathbf{c}_i)} z^2(\mathbf{x},j)}{N(\mathbf{c}_i)} - m_i^2[\mathbf{c}_i] \tag{11}$$

The values of $m_i[\mathbf{c}_i]$ and $v_i[\mathbf{c}_i]$ can be stored as tables (which we do here), or potentially summarized with descriptive functions. If the set of training images is sufficiently large, then $E[z(\mathbf{x})|\mathbf{c}_i(\mathbf{x})] \cong m_i[\mathbf{c}_i(\mathbf{x})]$, and $Var[z(\mathbf{x})|\mathbf{c}_i(\mathbf{x})] \cong v_i[\mathbf{c}_i(\mathbf{x})]$. In practice, we have found that the tables are the same for the rotated context vectors, and hence the same tables can be used for both orientations.

B. Ground truth, training and test images

The ground truth, training, and test images were obtained from a database of 1204 high resolution (4284 x 2844) 14-bit raw images captured with a calibrated Nikon D700 digital camera, at its lowest standard ISO setting (ISO_{200}). Care was taken to minimize clipping. From the 1204 images, 803 were randomly selected to be training images, and the remaining 401 were used as test images. The 803 training images provided approximately 10^{10} samples for learning each table of conditional means.

We considered three kinds of training and test images. For the first kind (MCGN) we performed the following sequence of steps: (1) addition of simulated camera noise to ground truth raw images, (2) AHD interpolation (demosaicing), (3) conversion to linear sRGB, (4) gamma compression, (5) quantization to 24-bit (8 bits per channel) sRGB.

For the second kind (MWGN) the steps were: (1) AHD interpolation, (2) conversion to linear sRGB, (3) addition of MWGN, (4) gamma compression, (5) quantization to 24-bit sRGB.

For the third kind (AWGN) the steps were: (1) AHD interpolation, (2) conversion to linear sRGB, (3) gamma compression, (4) quantization to 24-bit sRGB, (5) addition of AWGN. The average noise power (mean squared error from the ground truth images) for the second and third kinds of images was set to match that of the first kind.

For more details about the natural images see [14] and *www.cps.utexas.edu/natural_scenes*, where all the ground truth images are available.

C. Context vectors

Once the ground truth and training images have been specified, the only remaining steps are to specify the context vectors and measure the tables. In RCM denoising, all context vectors consist of three 8-bit variables.

For gray scale images there are five context vectors:

$$\begin{aligned}
\mathbf{c}_1(\mathbf{x}) &= \left[z_0(x-1,y), z_0(x,y), z_0(x+1,y)\right] \\
\mathbf{c}_2(\mathbf{x}) &= \left[z_1(x,y-1), z_1(x,y), z_1(x,y+1)\right] \\
\mathbf{c}_3(\mathbf{x}) &= \left[z_2(x-1,y), z_2(x,y), z_2(x+1,y)\right] \\
\mathbf{c}_4(\mathbf{x}) &= \left[z_3(x,y-1), z_3(x,y), z_3(x,y+1)\right] \\
\mathbf{c}_5(\mathbf{x}) &= \left[z_4(\mathbf{x}), a_4(\mathbf{x}), \sigma_4(\mathbf{x})\right]
\end{aligned} \tag{12}$$

The first four contexts consist of the pixel location being estimated and two immediately neighboring pixel locations. The contexts alternate between the horizontal and vertical directions. As a result of applying tables for these contexts recursively, there are effectively 9 pixels in $z_0(\mathbf{x})$ (the 3x3 neighborhood) that contribute to each estimated pixel in $z_2(\mathbf{x})$, and 25 pixels (the 5x5 neighborhood) contributing to each estimated pixel in $z_4(\mathbf{x})$ (see Fig. 5). Tables for these first four context vectors can be applied very efficiently—they are effectively 1x3 kernels that can be applied successively to the image, in place, with minimal buffering.

The final context vector consists of the pixel location being estimated, the average $a_4(\mathbf{x})$ of the values in the surrounding 9x9 neighborhood of pixel locations (80 locations), and the standard deviation $\sigma_4(\mathbf{x})$ of the surrounding 80 pixel values from the regression plane (see Appendix B). The rationale for this last context vector is that if the ground truth image is locally planar at some location, then the MMSE estimate is the average of the values in the neighborhood. The standard deviation from the regression plane measures how close the neighborhood is to being planar, allowing the table to know when to put the most weight on the local average.

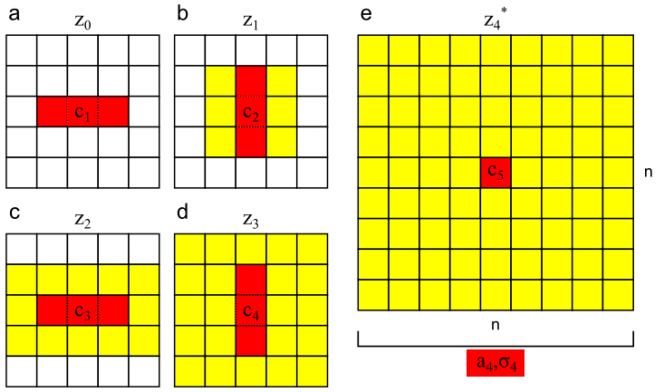


Fig. 5. Denoising kernels. The input image is z_0, and the recursively estimated images are z_1 to z_5. (a)-(d) The red pixels show the explicit context vector used from that image to obtain the next image. The yellow pixels show the additional pixels effectively contributing to the estimate. (e) The context for estimating z_5 consists of the center (red) pixel, the average of the surrounding (yellow) pixels, a_4, and the standard deviation of those pixels from the regression plane, σ_4.

For color images, the algorithm first converts the images from RGB to a perceptual color space (e.g., Rec 709 YCbCr, or a simple opponent color space [12]), applies the tables, and then converts back to RGB. Converting to a perceptual color space significantly reduces visible color artifacts in the denoised images. Perceptual color spaces represent each pixel with a luminance value (e.g., Y) and two color-opponent values (e.g., Cb, Cr). Using the same context vectors described above, we learn one set of tables for the luminance values and another set for the color-opponent values (the same tables are used for both color-opponent values). For the color-opponent channels, the fifth context vector uses an 11x11 neighborhood rather than 9x9. For color images, the algorithm also uses a final context vector consisting of the estimated RGB values in $z_5(\mathbf{x})$:

$$\mathbf{c}_6(\mathbf{x}) = \left[R_5(\mathbf{x}), G_5(\mathbf{x}), B_5(\mathbf{x})\right] \tag{13}$$

IV. Results

A. Estimation tables

RCM denoising uses local contexts having only three elements. This makes it possible to visualize the statistical rules implicit in the MMSE estimates. For example, Fig. 6a-c shows the optimal estimates for context $\mathbf{c}_1$ (see Fig. 5a). In each plot, the horizontal and vertical axes give the values of the two context pixels surrounding the center context pixel (the location being denoised). The color scale gives the MMSE estimate of the gray level of the center pixel (i.e., the directly measured conditional mean). The upper plot is for when the value of center context pixel is 64, the middle plot when the value is 128, and the bottom plot when the value is 192. In general, the MMSE estimates are smooth but non-trivial functions of the context vector. When the surrounding context pixels are nearly equal (i.e., near the main diagonal) and are lower in value than the center context pixel, then the estimate is strongly reduced. Similarly when the surrounding context pixels are nearly equal and greater than the center context pixel, then the estimate is strongly increased. On the other hand, when the two surrounding pixels differ there is greater likelihood of structure in the ground truth image and hence more weight is put on the center context pixel. The tables for contexts $\mathbf{c}_2$ to $\mathbf{c}_4$ are qualitatively similar to that for $\mathbf{c}_1$, but differ in detail.

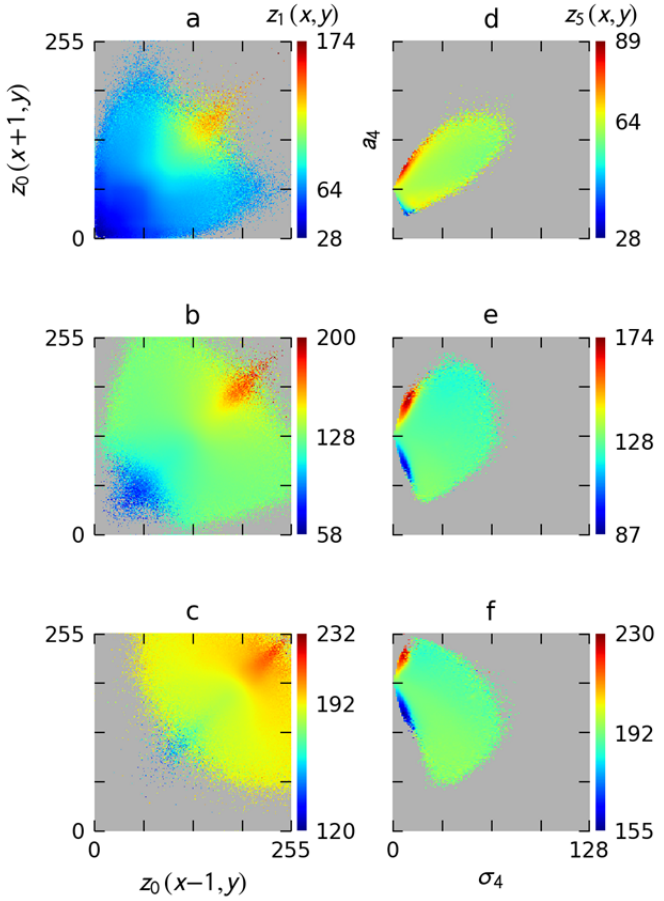


Fig. 6 Estimation tables for RCM denoising. Horizontal and vertical axes give the two context variables that are not the center context pixel. The color axis gives the MMSE estimate. (a-c) Tables for context $\mathbf{c}_1$ when the center context pixel $z_0(x,y)$ has a value of (a) 64, (b) 128, and (c) 192. (d-f) Tables for context $\mathbf{c}_5$ when center context pixel $z_4(x,y)$ has a value of (d) 64, (e) 128, and (f) 192.

Fig. 6d-f shows the optimal estimates for context $\mathbf{c}_5$ (see Fig. 5e). In each plot, the vertical axis gives the average gray level of the context pixels surrounding the center pixel (the pixel location being estimated), and the horizontal axis gives the standard deviation of the context pixel values from the regression plane. Again, the upper, middle and lower plots are for when the value of center context pixel is 64, 128 and 192, respectively. When the standard deviation is low, the region is closer to planar and more weight is put on the average (the estimates change more as the average changes; note the bigger changes in color), whereas when the standard deviation is high the region is less planar and more weight is put on the center context pixel (the estimated value from the previous contexts).

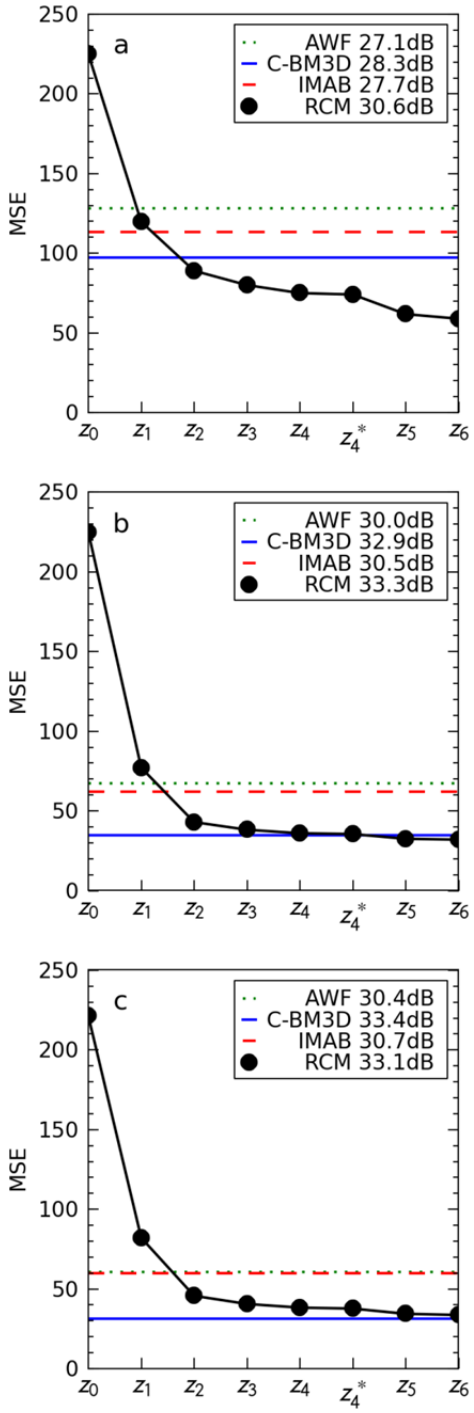


Fig. 7 Quantitative performance of denoising algorithms on 401 RGB test images. The vertical axis gives the average MSE of the denoised image. The horizontal axis indicates the successively denoised images produced by the proposed RCM algorithm. The black circles show the MSE of the successively denoised images by the RCM algorithm. The colored horizontal lines show the final MSE of the comparison algorithms. Also shown are the average final values of PSNR. (a) Results for realistic camera-image noise (MCGN). (b) Results for realistic raw-camera-image noise (MWGN). (c) Results for additive white Gaussian noise (AWGN).

B. Quantitative performance

To assess the performance of RCM denoising, we compared it with two standard algorithms, adaptive Wiener filtering (AWF; using the MATLAB® R2102a *wiener2* function) and ImageMagick® adaptive blur (IMAB; www.imagemagick.org, version 6.7), and with a state-of-the-art algorithm (C-BM3D [12]). Three different kinds of noise were tested: multiplicative correlated Gaussian noise (MCGN; i.e., realistic camera-image noise), multiplicative white Gaussian noise (MWGN; i.e., realistic raw-camera-image noise), and additive white Gaussian noise (AWGN). MCGN was added to each image by interpolating between the variance parameters for each measured ISO value between 200 and 3200 in order to achieve a noise level that corresponded to $\sigma = 15$ (see Fig. 2). The MWGN and AWGN were also set to have this same average noise level. The value of the noise standard deviation given to the three comparison algorithms was also 15.

Performance was compared on the 401 test RGB images and on several standard RGB images in the image processing literature. The black circles in Fig. 7 show the average MSE for the 401 test images after each step of RCM denoising. Recall that the result of applying the table for each context is a partially denoised image (indicated on the horizontal axis). The horizontal lines show the average MSE of the comparison algorithms. Also shown in the figure is the average PSNR of the final estimates.

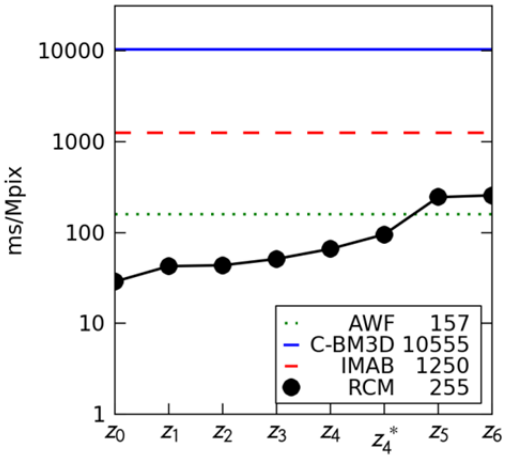


Fig. 8 Computation time of denoising algorithms.

Fig. 7a shows the results for noise that realistically mimics the noise in camera images (MCGN). The MSE drops rapidly as the successive tables are applied, dropping down to that of the AWF and IMAB algorithms after 1 step and below that of C-BM3D after 2 steps. Fig. 7b shows the results for noise that realistically mimics the noise in raw camera images (MWGN). Again, the MSE drops rapidly as the successive tables are applied, dropping below that of AWF and IMAB after 2 steps and below that of C-BM3D after 5 steps. Finally, Fig. 7c shows the results for additive white noise (AWGN). The MSE drops below AWF and IMAB at 2 steps and approaches but does not drop below C-BM3D. Denoising is generally easier with statistically independent noise and hence the final MSE is lower for MWGN and AWGN.

The black circles in Fig. 8 show the cumulative computation time in milliseconds per megapixel, at each successive step of the RCM algorithm. The final black circle and the horizontal

lines show the total computation time of the algorithms. The algorithms were run on a single 3.1GHz Intel processor. Not surprisingly, given its simplicity and its use of tables, the RCM algorithm runs very quickly. The slowest step is application of the table for context $\mathbf{c}_5$ (see Fig. 5e). The next slowest step is repeating the first four steps with the orthogonal context vectors to obtain $z_4^{\perp}(\mathbf{x})$. Interestingly, this step provides symmetry (which is good to have), but has a relatively minor effect on MSE (see Fig. 7). Thus under demanding conditions it could be dropped.

C. *Qualitative performance*

Many of the standard test images in the image processing literature already contain substantial camera noise and thus one way to compare denoising algorithms is simply to denoise the original image. For example, Fig. 9 shows a cropped region of the original standard peppers image together with the results of the AWF, C-BM3D and RCM algorithms. Qualitatively the quality of C-BM3D and RCM are similar with a little less chromatic aliasing for the RCM algorithm.

Fig. 10 shows results for synthesized camera noise (MCGN). The upper image is cropped from one of the test images in our data set. The bottom image is of a human-made object and not in our data set. The values of PSNR are for the entire image. As can be seen RCM denoising removes much more of correlated spatio-chromatic noise. In the bottom image RCM denoise softens the edges slightly. If desired this effect can be lessened by eliminating or reducing the size of the surround in the context vector $\mathbf{c}_5$ (see Fig. 5e), with a modest effect on the MSE (see Fig. 7a). For example, Fig. 11 shows the RCM estimates without context vector $\mathbf{c}_5$.

Fig. 12 shows results for noise that mimics adding independent multiplicative noise prior to gamma compression (MWGN). Recall that this noise is like the noise in raw camera images and is quite similar to additive independent noise following gamma compression (AWGN). The upper image is cropped from one of the test images in our data set. The middle image contains human made objects and is not in our data set. The bottom image is cropped from the standard Lena image. Again the PSNR values are for the whole image. In general the C-BM3D and RCM results are similar in quality. C-BM3D produces slightly smoother contours (e.g., the contour inside the window of the middle figure), but removes some image texture (e.g., the brick structure on the left in the middle figure).

D. *Limitations, Extensions, and Applications*

Although RCM denoising is fast and simple computationally, it does require substantial memory to store the fixed tables. The memory requirements are not significant for most personal computers, but may be an issue for image-processing hardware or firmware in devices such as digital cameras. We note, however, that each table is just a list of 8-bit numbers (unsigned bytes) and hence could be stored and retrieved like an image. Also, the tables are relatively smooth and regular (see Fig. 6), and thus it should be possible to closely approximate them in some way; e.g., fitting them with a sum of appropriate basis functions. (Note that the noisy pixels in Fig. 6 correspond to extremely rare context vectors. We find that quadrupling the training set smooth's those regions, but has negligible effect on performance precisely because those context vectors are so rare.)

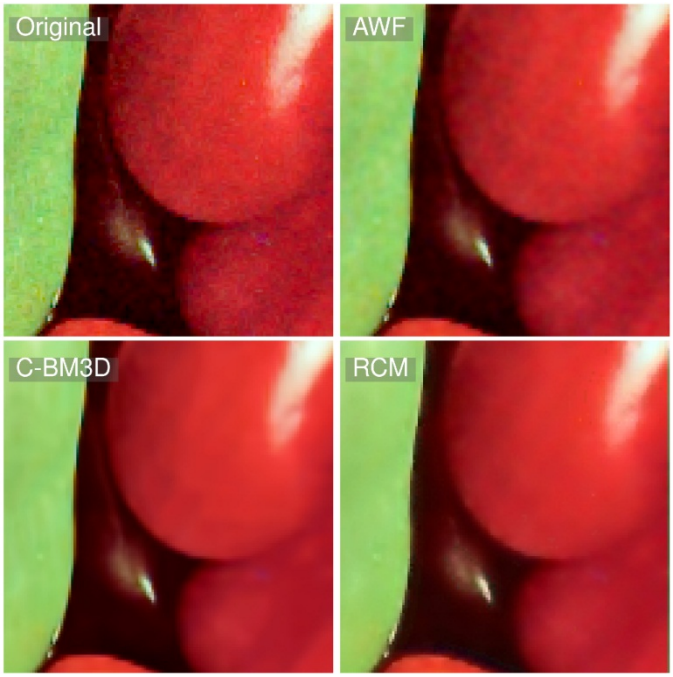
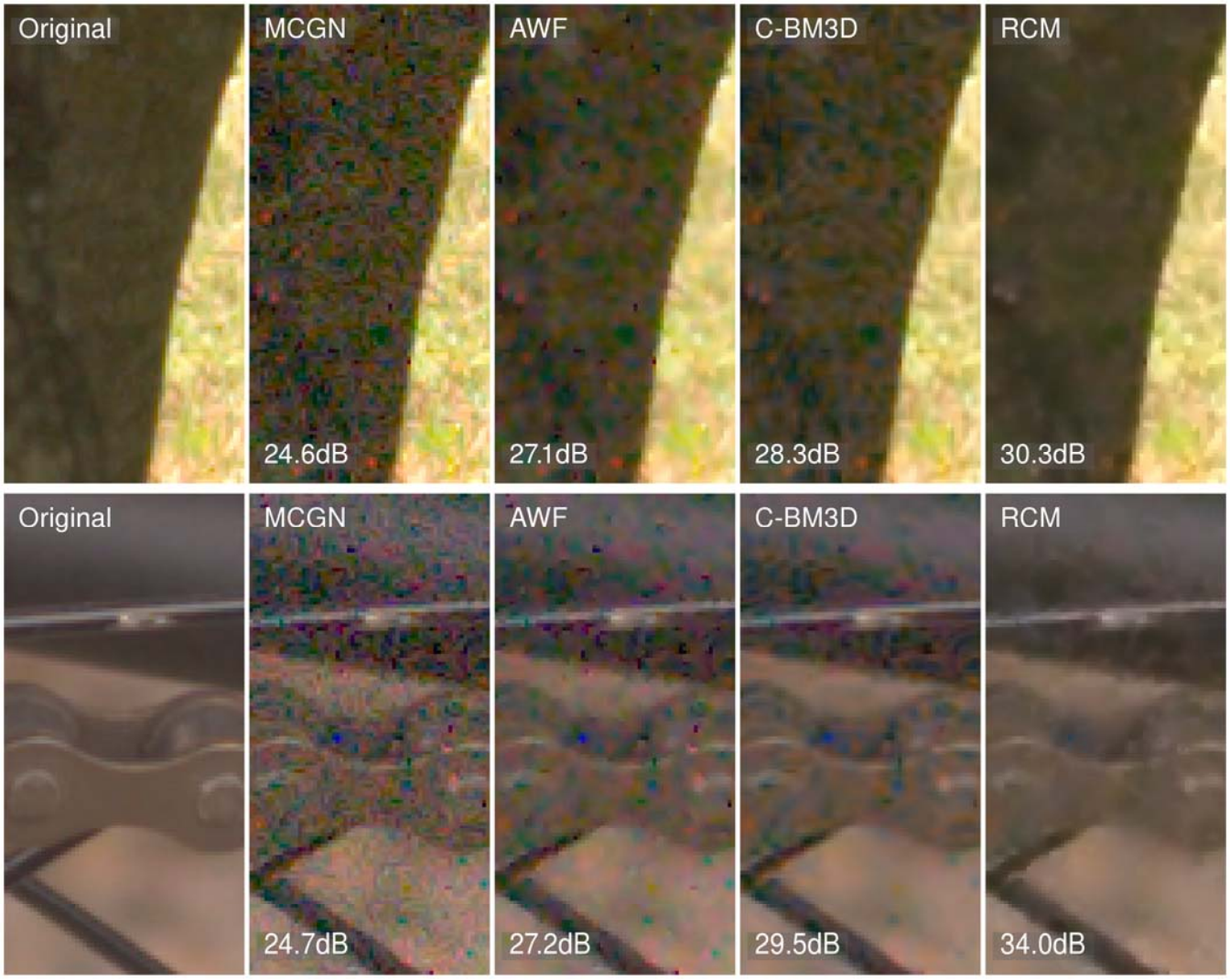


Fig. 9 Denoising algorithms applied to the standard peppers image. No additional noise was added to the original image.

As an immediate practical application, RCM denoising can be used to denoise standard 24-bit sRGB images and 8-bit gray scale images. Another obvious application would be the denoising of raw camera images. We have not yet implemented this application, but comparison of Figs. 7a and 7b suggests that the RCM denoising may be more effective if applied to the raw image, before demosaicing, which creates correlated noise (note the lower final MSE in Fig. 7b). Of course, in many practical situations the raw image is not available.

RCM denoising gets most of its power by measuring very local statistical structure. The output of the 4th recursive step, z_4, is quite accurate (Fig. 7), and has a support region of only 5x5 pixels (Fig. 5). Other successful denoising algorithms generally have a larger support region, and thus it is possible that some hybrid approaches could produce better performance, assuming they capture more large-scale information than our simple regression-plane measure (Fig. 5e). How much better image denoising can get is a matter of



natural scene taken from the CPS natural image database. The bottom image is not a natural scene and is not contained in the database.

some debate [2,17]. We note, however, that attempts to set bounds on maximum possible denoising performance have assumed AWGN.

V. Conclusion

RCM denoising is remarkably effective given its conceptual and computational simplicity. In this approach, recursive conditional means are measured directly (by simple averaging) from a large training set of natural images, for small (3 element) context vectors. Denoising with the resulting fixed set of tables exceeds state-of-the-art algorithms in accuracy for realistic camera noise and matches them for additive white Gaussian noise. RCM denoising is much faster than state-of-the-art algorithms. This speed allows the method to be applied to very large images and, with further optimization, should allow it to be applied to video in real time.

The recursive conditional means approach has also proved to be very effective for the task of upsampling (super resolution, [14]), and thus it is likely to be effective for a number of other basic image processing tasks.

Acknowledgements

We thank Alan Bovik for comments on the manuscript.

Appendix A

Here we derive the rule used for combining multiple estimates of the true pixel value z into a single estimate. This rule is optimal given certain assumptions, and is closely related to standard rules for cue combination [16]. Let $\mathbf{c}_1,\ldots,\mathbf{c}_n$ represent n sets of known pixel values (sources of information) that are to be used in estimating a single unknown pixel value z. Using Bayes' rule, the posterior probability of z given the known pixel values can be written as:

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=\frac{p\left(\mathbf{c}_1,\cdots,\mathbf{c}_n\left|z\right.\right)p\left(z\right)}{p\left(\mathbf{c}_1,\cdots,\mathbf{c}_n\right)} \tag{A1}$$

Our first assumption is that given the true value of z, $\mathbf{c}_i$ is independent of $\mathbf{c}_j$ for all i and j (i.e., the sources of information are statistically independent). In this case, we have

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=\frac{p\left(z\right)\prod_{i=1}^{n}p\left(\mathbf{c}_i\left|z\right.\right)}{p\left(\mathbf{c}_1,\cdots,\mathbf{c}_n\right)} \tag{A2}$$

Applying Bayes' rule again,

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=\frac{\prod_{i=1}^{n}p\left(z\left|\mathbf{c}_i\right.\right)}{p^{n-1}\left(z\right)}\frac{\prod_{i=1}^{n}p\left(\mathbf{c}_i\right)}{p\left(\mathbf{c}_1,\cdots,\mathbf{c}_n\right)}$$

or,

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=K_{\mathbf{c}}\frac{\prod_{i=1}^{n}p\left(z\left|\mathbf{c}_i\right.\right)}{p^{n-1}\left(z\right)} \tag{A3}$$

where $K_{\mathbf{c}}=\dfrac{\prod_{i=1}^{n}p\left(\mathbf{c}_i\right)}{p\left(\mathbf{c}_1,\cdots,\mathbf{c}_n\right)}$.

Our second assumption is that the prior and posterior probability distributions are Gaussian. In this case, we have

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=\frac{K_{\mathbf{c}}\exp\left(\sum_{i=1}^{n}-0.5\left(z-u_i\right)^2\rho_i\right)\prod_{i=1}^{n}\frac{\sqrt{\rho_i}}{\sqrt{2\pi}}}{\exp\left(\sum_{i=1}^{n-1}-0.5\left(z-u\right)^2\rho\right)\prod_{i=1}^{n-1}\frac{\sqrt{\rho}}{\sqrt{2\pi}}} \tag{A4}$$

Where u_i and ρ_i are the conditional mean and reliability of posterior probability distribution of z given $\mathbf{c}_i$, and u and ρ are the mean and reliability of the prior probability distribution of z. (Note that reliability is the inverse of the variance.) Rearranging, we have

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=$$
$$\frac{K_{\mathbf{c}}\prod_{i=1}^{n}\frac{\sqrt{\rho_i}}{\sqrt{2\pi}}}{\prod_{i=1}^{n-1}\frac{\sqrt{\rho}}{\sqrt{2\pi}}}\exp\left(0.5\left(n-1\right)\left(z-u\right)^2\rho-0.5\sum_{i=1}^{n}\left(z-u_i\right)^2\rho_i\right)$$

Consider the term inside the exponential. Because this term is quadratic in z it follows that $p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)$ is Gaussian. Expanding, collecting terms, and completing the square, shows that $p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)$ is of the form:

$$p\left(z\left|\mathbf{c}_1,\cdots,\mathbf{c}_n\right.\right)=$$
$$K\exp\left(-0.5\left(-\left(n-1\right)\rho+\sum_{i=1}^{n}\rho_i\right)\left[z-\frac{\left(-\left(n-1\right)ru+\sum_{i=1}^{n}\rho_iu_i\right)}{\left(-\left(n-1\right)\rho+\sum_{i=1}^{n}\rho_i\right)}\right]^2\right) \tag{A5}$$

where K is the constant required to make the right side of the equation a probability distribution.

The minimum mean squared error (MMSE) estimate is the expected value of the posterior probability distribution, and the maximum *a posteriori* (MAP) estimate is the mode of the posterior probability distribution. It follows from eq. (A5) that the MMSE and MAP estimates are the same and are given by

$$\hat{z}=\frac{\left(\rho u+\sum_{i=1}^{n}\left(\rho_iu_i-\rho u\right)\right)}{\left(\rho+\sum_{i=1}^{n}\left(\rho_i-\rho\right)\right)} \tag{A6}$$

Finally, note that the entropy of $p\left(z\left|\mathbf{c}_i\right.\right)$ must be less than or equal to the entropy of $p(z)$. Thus, $\rho\le\rho_i$, and hence the denominator is always positive.

In using eq. (A6), we set $u=\bar{z}$ (the sample mean of the prior probability distribution of z), $\rho=1/\sigma^2$ (one over the sample variance of the prior probability distribution of z), $u_i=z_i$ (the sample mean of the posterior probability distribution of z given $\mathbf{c}_i$), and $\rho_i=1/\sigma_i^2$ (one over the sample variance of the posterior probability distribution of z given $\mathbf{c}_i$). Eq. (A6) may not be valid if the two strong assumptions above do not hold, and thus should be regarded as an approximation.

APPENDIX B

Here we give the formulas used for planar regression. The equation of a plane is $f\left(x,y\right)=Ax+By+C$. Consider a square block of pixels of odd dimension n, where the center pixel is taken to be the origin. The least squares estimates of the parameters of the plane are

$$A=\frac{\sum_{x,y}z\left(x,y\right)x}{k},\ B=\frac{\sum_{x,y}z\left(x,y\right)y}{k},\ C=\frac{\sum_{x,y}z\left(x,y\right)}{n^2} \tag{B1}$$

where,

$$k = \frac{n^2}{3}\left(\frac{n-1}{2}\right)\left(\frac{n-1}{2}+1\right) \tag{B2}$$

The standard deviation from the best fitting plane is given by

$$\sigma = \sqrt{\frac{1}{n^2}\sum_{x,y}\left[Ax + By + C - z(x,y)\right]^2} \tag{B3}$$

REFERENCES

[1] Buades A., Coll B. & Morel J.M. (2010) Image denoising methods: A new non-local method. *SIAM Review*. 52, 113-147.

[2] P. Chatterjee & P. Milanfar (2010) Is denoising dead? *IEEE Trans. on Image Processing*. 19, 895-911.

[3] Wang Z., Bovik A.C., Sheikh H.R., & Simoncelli E.P. (2004) Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Processing*.

[4] Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York: Springer.

[5] Wiener, N. (1949). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. New York: Wiley.

[6] D. T. Kuan, A. A. Sawchuk, T. C. Strand, and P. Chavel (1985) Adaptive noise smoothing filter for images with signal dependent noise, *IEEE Trans. PAMI*, vol. 7, pp. 165–177.

[7] Liu C., Szeliski R., Kang, S.B., Zitnick C.L. & Freeman W.T. (2008) Automatic estimation and removal of noise from a single image. *IEEE Trans. Pattern Anal. & Mach. Intell*. 30, 299-314.

[8] Tomasi C. & Manduchi R. (1998) Bilateral filtering for gray and color images. *Proceedings IEEE Conference in Computer Vision*, Bombay, India.

[9] A. Efros and T. Leung (1999) Texture synthesis by non-parametric sampling, *Proceedings of the IEEE International Conference on Computer Vision,* 2, Corfu, Greece, 1033–1038.

[10] R. R. Coifman and D. Donoho (1995) *Translation-invariant de-noising*, in Wavelets and Statistics, Lecture Notes in Statist., Springer-Verlag, New York, pp. 125–150.

[11] J. Portilla, V. Strela, M.J. Wainwright, and E.P. Simoncelli (2003) Image Denoising Using Scale Mixtures of Gaussians in the Wavelet Domain, *IEEE Trans. Image Processing*, 12, pp. 1338-1351.

[12] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian (2007). Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Trans. Image Process*., 16, pp. 2080–2095.

[13] L. Zhang, W. Dong, D. Zhang & G. Shi (2010) Two stage denoising by principle components analysis with pixel grouping. *Pattern Recognition*, 43, 1531-1549.

[14] Geisler, W. S., & Perry, J. S. (2011). Statistics for optimal point prediction in natural images. *Journal of Vision*, 11(12):14, 1–17, http://www.journalofvision.org/content/11/12/14, doi:10.1167/11.12.14.

[15] Danielyan A., Vehvilanien M., Foi A., Katkovnik V. & Egiazarian K. (2009) *Proc. Local and Non-Local Approx. in Image Process.* 125-129.

[16] Oruc¸, I., Maloney, L. T., & Landy, M. S. (2003) Weighted linear cue combination with possibly correlated error. *Vision Research*, 43, 2451–2468.

[17] Levin, A., & Nadler, B. (2011) Natural image denoising: Optimality and inherent bounds. *IEEE Conference and Pattern Recognition and Computer Vision (CVPR)*. 2833-2940.

Fig. 11 Detail of results from RCM algorithm from images containing additional multiplicative correlated Gaussian noise. In these examples, the RCM algorithm was modified to exclude the 5th recursive step in which the estimates for z_5 are computed using the average and standard deviation from the regression plane.

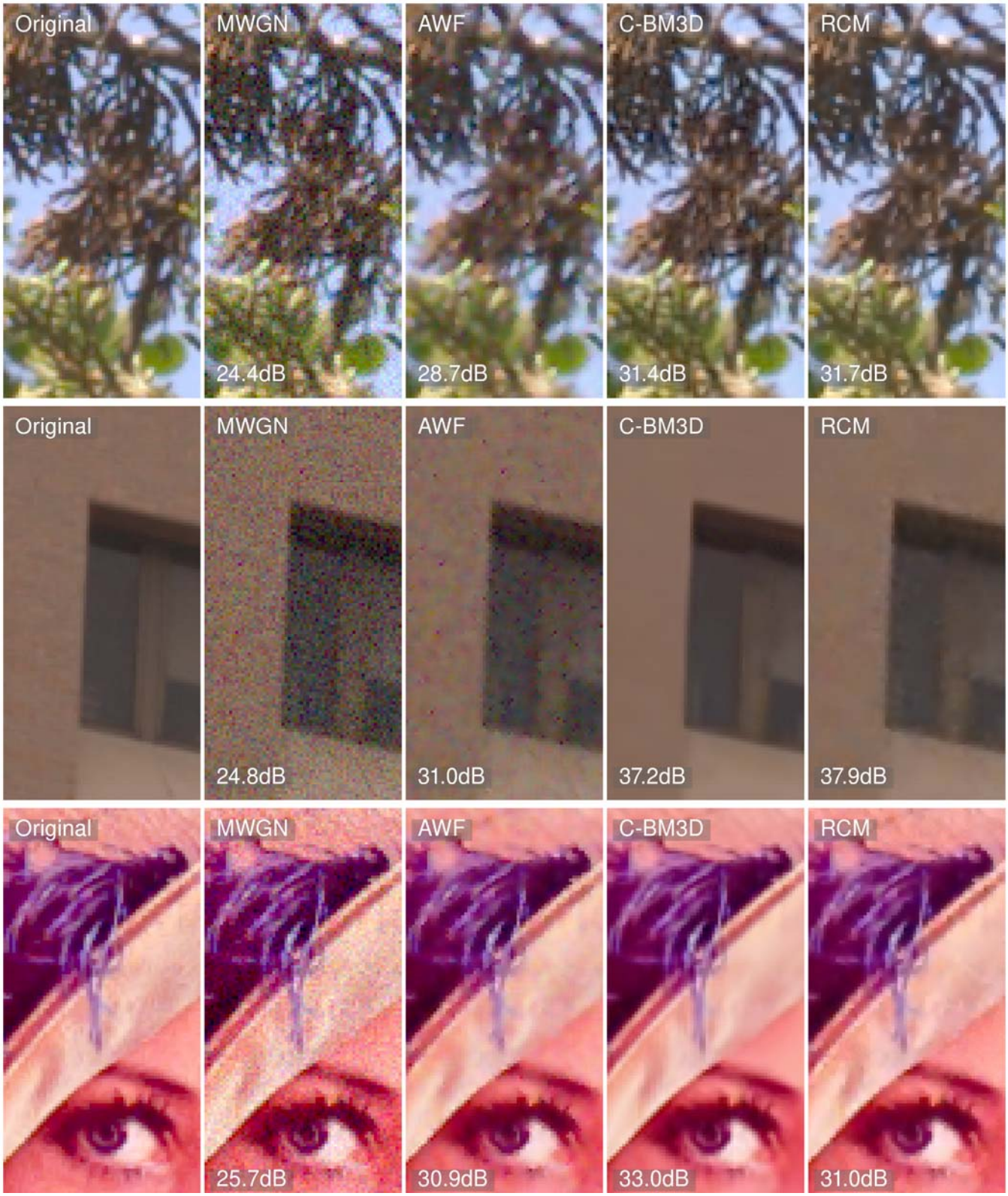


Fig. 12 Details of results from various algorithms from images containing additional multiplicative white Gaussian noise. The top image is from a natural scene taken from the CPS natural image database. The middle image is not a natural scene and is not contained in the database. The bottom image is the standard Lena image. The standard deviation of the added noise was 15.