Exploiting the Sparse Derivative Prior for Super-Resolution and Image Demosaicing

Marshall F. Tappen Bryan C. Russell William T. Freeman

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
{*mtappen, brussell, billf*}*@ai.mit.edu*

Abstract

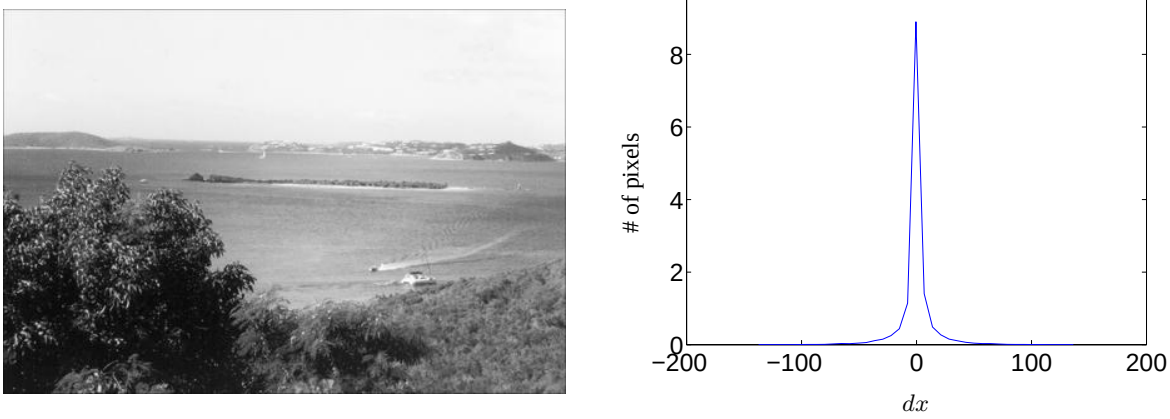
When a band-pass filter is applied to a natural image, the distribution of the output has a consistent, distinctive form across many different images, with the distribution sharply peaked at zero and relatively heavy-tailed. This prior has been exploited for several image processing tasks. We show how this prior on the appearance of natural images can also be used to estimate full-resolution images from incomplete data. The unobserved image pixels are modeled with a factor graph. The constraints in the factor graph are based on the characteristic distribution of image derivatives. We introduce an efficient representation for finding candidate values for patches of the image being estimated, avoiding combinatorial explosion. The usefulness of this approach is demonstrated by applying it to two applications: extracting a high-resolution image from a low-resolution version and estimating a full-color image from an image with one color sample per pixel. We show how the super resolution system produces noticeably sharper images, with few significant artifacts. The demosaicing system produces full-color images with fewer color-fringing artifacts than images from other methods.

1 Introduction

A remarkably consistent property of natural images is the distribution of filter outputs when a band-pass filter is applied to the image. Many researchers [25, 21, 29] have reported that the histogram of the filter outputs is sharply peaked at zero and highly kurtotic, similar to the histogram shown in Figure 1(b). This property has been observed across very many categories of images.

The strength and repeatability of this property make it invaluable as a statistical prior on the structure of natural images. Simoncelli used this prior as the basis of a Bayesian noise-removal algorithm [30]. Simoncelli and Buccigrossi used the regular statistical structure of images for compression [31]. Levin et al. have shown how this prior can enable an image to be decomposed into transparent layers [17]. Zhu and Mumford used the prior to formulate reaction-diffusion equations for noise and clutter removal [39]. Several authors have used this prior for wavelet-based image deconvolution [7, 6].

In this paper, we describe a general strategy for taking advantage of this prior on natural images to estimate full-resolution images from incomplete data. We then show two different examples of how this strategy can be applied. Section 3 discusses the application of the image prior to the problem of estimating high-resolution images from low resolution images, also known as super-resolution. In Section 4, we show



(a) (b)

Figure 1: (a) An example of a natural image. (b) Histogram of the horizontal derivatives of the image in (a). The histogram is sharply-peaked at zero.

how to improve the estimation of true color images from poly-chromatically sampled images, such as those created by CCD's.

In both tasks, the latent image being estimated is modeled well with a graphical model. We use the first derivative as the band-pass filter and use the regular distribution of natural image derivatives to determine the parameters of the model. The prior is enforced using a graphical model represented by discrete states. One limitation of this approach is the potentially large number of states that need to be represented at each node. To overcome this, we demonstrate a novel technique which enables a discrete-valued graphical model to represent the wide range of image values possible at each node. This technique allows for a flexible representation of the hidden variables, while controlling the computational complexity of state estimation, which is a typical problem in learning-based approaches to low-level vision.

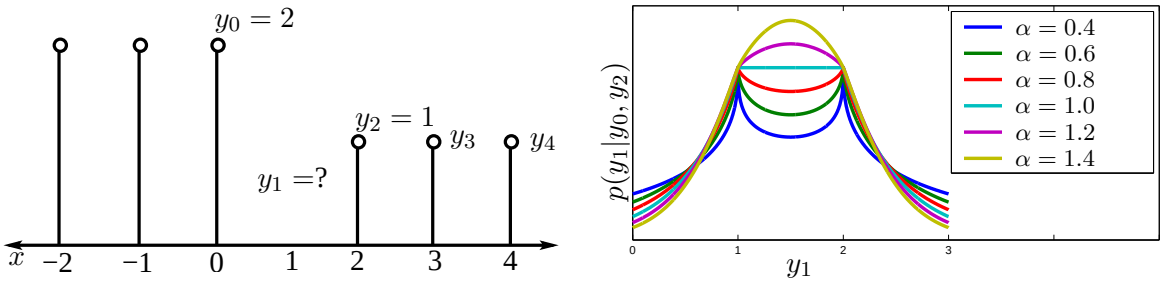
2 Graphical Models and Natural Image Statistics

2.1 Natural Image Statistics

As mentioned, the output of band-pass filters, including derivative filters, applied to a natural image have sharply peaked distributions, with relatively heavy tails [25, 29]. These distributions are often described as sparse distributions because only a small proportion of the outputs are greater than zero. The distribution of the outputs is described well with a generalized Laplacian distribution [21]:

$$p(x) \propto e^{-\frac{|x|^\alpha}{s}} \tag{1}$$

where $0 < \alpha < 1$. We refer to this characteristic distribution of derivatives as the *natural image prior*, because it is consistent across a wide variety of real images. An interpretation of this prior is that this distribution describes images as consisting largely of zero gradient regions interspersed with occasional strong gradient transitions. It seems plausible that in a given image a sharp edge is preferred over a blurry



(a) (b)

Figure 2: A simple example showing the influence of the exponent α in Equation 1 on interpolated image sharpness. (a) A simple interpolation problem where y_1 must be interpolated from y_0 and y_2. (b) The probability curves for y_1. Each curve represents the probability of y for one value of α. For $\alpha > 1$, the most likely values of y_1 is 1.5, leading to a softer image edge. For $\alpha < 1$, y_1 is most likely 1 or 2, either of which gives a sharper edge transition.

edge. This can be seen by examining the simple interpolation problem shown in Figure 2. The samples at y_0 and y_2 are observed, but y_1 must be interpolated from the observed values.

To estimate y_1, we assume that the probability of some value of y_1, given y_0 and y_2, is proportional to the probability of derivatives that would be induced by setting y_1 to that value. If we assume that the derivatives in the signal are distributed according to Equation 1, then

$$p(y_1|y_0, y_2) \propto e^{-\left(\frac{|1-y|^\alpha + |2-y|^\alpha}{s}\right)} \tag{2}$$

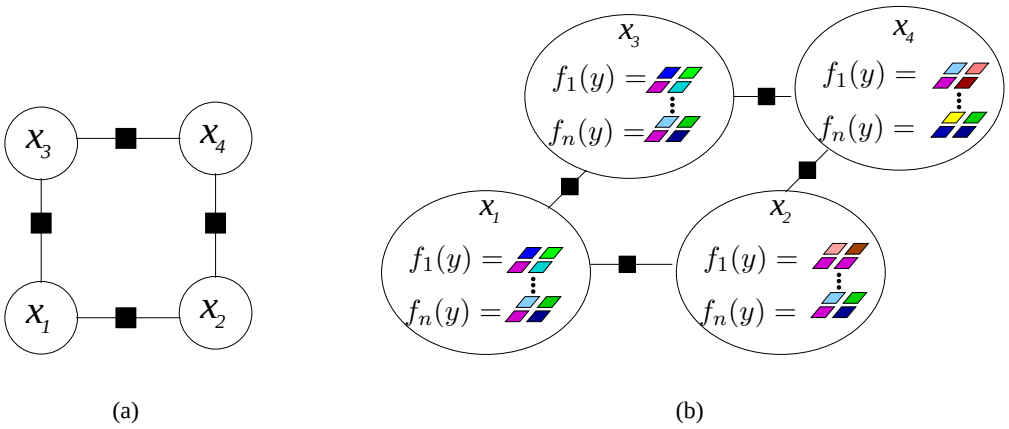
Examining equation 2, shown in Figure 2(b), shows that an extremum of $p(y_1|y_0, y_2)$ will occur at $y_1 = 1.5$, regardless of the value of α. However, for $\alpha < 1$, the extremum will be a minimum, causing the most probable value of y_1 to be either 1 or 2. For $\alpha > 1$, $p(y_1 = 1.5)$ will be maximum, making it the best estimate for y_1. The fact that $\alpha < 1$ for natural images is important because it imposes a "sharpness prior". If the distribution is sparse, it is preferable to have a single strong derivative, which would appear as a strong edge in an image, rather than a pair of smaller derivatives, which would appear as a fuzzy edge.

The relationship between α and the convexity or concavity of the probability curve is important to any application relying on this natural image prior. In [17], Levin et al. show how the probability must be less than one in order to successfully decompose an image into transparent layers.

2.2 Applying Image Statistics

Graphical models are a natural choice because the constraints on groups of pixels provided by the sparse image prior are conveniently expressed in a graphical model. For the problems described in Sections 3 and 4, our goal is to infer the pixel values of a full-resolution image.

We chose to use a factor graph to express the graphical model [16]. An example of a factor graph is shown in Figure 3. Each node labelled x_i represents a group of one or more full-resolution pixels that must be estimated. Each x_i does not necessarily represent just a single pixel, the state of x_i could instead represent the values of multiple pixels, as shown in Figure 3(b). The filled in squares represent the local functions, or



(a) (b)

Figure 3: (a) An example of a simple factor graph. (b) The x_i nodes in the graph represent states of groups of pixels, not a single pixel. In this graph, each candidate state of a node represents a value of four pixels. These candidate groups of pixels are computed from local image data by a set of different linear regressions, $f_1(\cdot) \ldots f_n(\cdot)$.

constraints, that relate the various nodes to each other.

The probability distribution expressed by a graph is the product of the constraint functions. We denote the constraint functions between nodes as $\psi(\cdot)$, which is a function of the nodes connected to that constraint. The joint probability of any configuration of the nodes is

$$P(x_1, x_2, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{M} \psi_i(\cdot) \tag{3}$$

where $\psi_i(\cdot)$ are functions of the nodes in the graph and C is a normalization constant.

The probability distribution for a graph in Figure 3(a) is

$$P(x_1, x_2, x_3, x_4) = \psi(x_1, x_2)\psi(x_2, x_4)\psi(x_3, x_4)\psi(x_1, x_3) \tag{4}$$

We utilize the properties of natural images by basing the constraints between two neighboring candidates on the image derivatives that the two image candidates would induce if they were chosen for the estimated image. If we expect the latent image to appear natural, then the derivatives between its pixels should also have a sparse distribution. Consequently, if x_i and x_j represented the value of two neighboring pixels being estimated, the general form of $\psi(x_i, x_j)$ will be

$$\psi(x_i, x_j) \propto e^{\frac{|x_i - x_j|^\alpha}{s}} \tag{5}$$

2.3 Choosing Candidates

Along with specifying the form of compatibility functions, the form of each variable in the graphical model must also be chosen. We model the nodes in the graphical model, which represent the pixel values in the image being estimated, as discrete random variables.

Using a discrete representation requires choosing the number of states to be represented at each node. The most complete representation would require one state for each possible combination of intensity-levels in the image patch. Unfortunately, this makes performing inference in the model computationally intractable because the number of states for the model would be huge. The cost of exact inference grows exponentially in the number of states and even approximate inference algorithms, such as belief propagation, are $\mathcal{O}(n^2)$ in the number of states. This representation problem affects many learning-based vision algorithms. The range of possible values for each node is very large, so algorithms must be able to find a small number of states that represent the most likely values of the hidden variables.

In [11], Freeman et al. avoid this problem by choosing a small number of plausible image values as the set of possible states for each node. Local image information is used to select a candidate set of image patches which represent the hidden image data at each location in the image. Using patches reduces the number of states for each random variable to a reasonable number, but requires that a large database of image patches be stored. In addition, selecting candidate values requires searching this database for each variable.

Instead of selecting patches from a large database, we generate candidates directly from the observed image data. The candidates are produced by a set of functions that generate candidate pixel patches from the observed data. These local functions can be seen as mapping low-resolution image data to the high-resolution "scene data" [35]. Given the observed image information around some point, l, the candidate value for the hidden information at that point, h, is modeled as a linear function of l:

$$h = \mathbf{T}l \tag{6}$$

where $\mathbf{T}$ is the matrix relating the low-resolution image information to a candidate high-resolution patch. Non-linear functions can be computed similarly by expanding l to include non-linear functions of the image data, such as polynomials.

To generate multiple candidates, we find a set of matrices, $\mathbf{T}_1 \ldots \mathbf{T}_N$, each of which interpolates the same observed information to a different candidate for the hidden image patch. This is illustrated in Figure 3(b). The candidate states at each node are computed by a set of linear regressions of the observed image data, $f_1(y) \ldots f_n(y)$.

Given a training set of observed patches, l, and the corresponding hidden image patches, h, the interpolators are found with a simple clustering algorithm:

1. Use k-means clustering to initially assign each training example to one of N clusters.
2. For each cluster, j, set $\mathbf{T}_j$ to be the least-squares solution to $h_j = \mathbf{T}_j l_j$, where h_j and l_j are the set of training examples assigned to cluster j. Note that if there are K examples in cluster j, then h_j and l_j will each have K rows.
3. Assign each example to the cluster where the corresponding $\mathbf{T}_j l$ best predicts h
4. Repeat step 2 until the reconstruction error $||h - \mathbf{T}_j l||$ drops below a threshold.

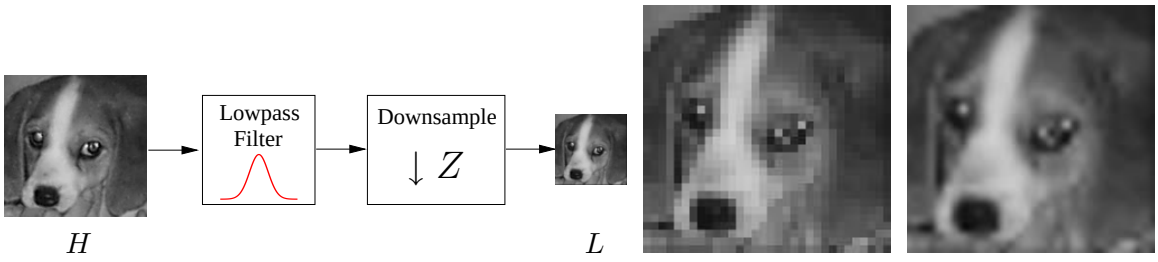


Figure 4: (a) A system diagram of a low resolution image L generated from a high resolution image H. (b) Example of pixel replication to increase resolution. (c) Example of bicubic interpolation to increase resolution.

3 Super Resolution

In this section, we adapt the presented framework to the super resolution problem. Specifically, we describe modifications to the model presented in Section 2.2 and describe in detail the algorithm to find solutions.

3.1 Problem Description

Super resolution takes as input a single low-resolution image and produces as output an image with higher resolution, usually by a factor of two or more[1]. We approach the problem by assuming a model for the degradation of the high resolution image. Specifically, we assume that a low resolution image L is generated from a high resolution image H by first convolving H with a low-pass filter, to reduce aliasing, and then downsampling to the desired size. This process is illustrated in Figure 4(a). We now wish to perform the inverse operation, which is an underconstrained problem [3, 6, 28, 33]. Two simple solutions are pixel replication (Figure 4(b)) and bicubic interpolation (Figure 4(c)). Pixel replication produces jagged edges and "blocky" artifacts while bicubic interpolation produces an overly smooth result. The challenge of this problem is to add visually plausible high frequency information in the higher resolution image to sharpen the edges.

3.2 Previous Work

Functional interpolation methods, such as bicubic and cubic spline interpolation, approximate an unknown continuous function by a set of local functions, which can then be discretely sampled to the desired resolution [14, 13, 27]. While these methods are usually fast and easy to implement, the resulting images often have blurred edges. Image sharpening techniques have been proposed to ameliorate the results from functional interpolation methods [12, 22, 34]. These methods result in sharper images, but may contain haloing artifacts. An alternate solution involves deconvolving the blurring filter [5, 36, 6, 7, 24]. While the results are quite good, deconvolution methods, as well as image sharpening methods, only enhance features that are

[1]Some authors refer to extracting a single high resolution image from multiple low resolution frames as super resolution. Here, we deal with only single frames, which is sometimes called "image interpolation".

present in the low resolution image. Learning-based methods use prior information to enhance the solution space [28, 2, 18, 32, 11, 10, 5]. While many of these methods sharpen the low resolution image, a few of these also add additional features not appearing in the low resolution image [2, 18, 11, 10].

3.3 Training and Algorithm

For super resolution, we consider Equation 3 which outlines how to incorporate the natural image prior and local constraints. In this problem, the local constraint is the reconstruction of the low resolution image from the inferred high resolution pixels. We derive the following equation which takes into account the natural image prior and reconstruction constraint:

$$\Pr\left(\{x\} \mid \{y\}\right) = \frac{1}{C} \underbrace{\prod_{(i,j)} \exp\left(-\frac{1}{2}\left(\frac{|D_i * \hat{x}_j|}{\sigma_N}\right)^{\alpha}\right)}_{\text{Natural image prior}} \cdot \underbrace{\prod_{i=1}^{MN} \exp\left(-\frac{1}{2}\left(\frac{W * \tilde{x}_i - y_i}{\sigma_R}\right)^2\right)}_{\text{Reconstruction constraint}}. \tag{7}$$

where $*$ is the convolution operator, $\{x\}$ are the latent random variables and $\{y\}$ are the observed variables. For the natural image prior, D_i is a directional derivative kernel, $\hat{x}_j$ is a patch of pixels from the high resolution image of size equal to the derivative kernel D_i, σ_N is the standard deviation parameter of the natural image prior, and α is the natural image prior. The product ranges over all directional derivative kernels and over all appropriately sized patches. For the reconstruction constraint, W is a Gaussian kernel, y_i is a low resolution pixel, $\tilde{x}_i$ is a patch of high resolution pixels that subsamples to the low resolution pixel y_i of size equal to the Gaussian kernel W, and σ_R is the standard deviation parameter of the reconstruction constraint. The above equation is for a low resolution image of size $M \times N$. Also, the first product contains the natural image prior constraint while the second product is the reconstruction constraint.

Consider the simple case of zooming by a factor of two. We start by converting Equation 7 into a factor graph representation, as discussed in Section2.2 (we describe the factor graph representation for Equation 7 below). To solve the factor graph, we use the max-product belief propagation (BP) algorithm [8, 37] to compute the maximum a posteriori (MAP) estimate of the random variables. Instead of considering all possible intensities for the pixels within each patch, which is intractable, we assume for each low resolution pixel a small set of candidate 2×2 high resolution patches. The sets of candidate patches for each spatial location are the possible states of the latent variables, thereby making the problem tractable.

To illustrate how to represent the above equation, let us consider the one-dimensional case. Figure 5(a) shows the factor graph, where the transparent circles represent random variables (x_i are the latent variables and y_i are the observed variables), the solid squares represent the natural image prior constraint, and the solid circles represent the reconstruction constraint. We will assume a derivative kernel of $[-1, 0, 1]$ and a three-tap Gaussian kernel. For tractability, we apply the high resolution patch constraint above by requiring each latent variable to represent a patch of two pixels, as illustrated in Figure 5(e). We notate the two pixels a and b of a patch candidate of latent node x_i in our model as x_i^a and x_i^b respectively, shown in Figure 5(f).

To derive the propagation equations for the factor graph, we need to consider three propagation cases: latent node to constraint node, derivative constraint node to latent node, and reconstruction constraint node to latent node, illustrated in Figure 5(b)-(d). Let μ_i be an S-tuple representing messages passed between

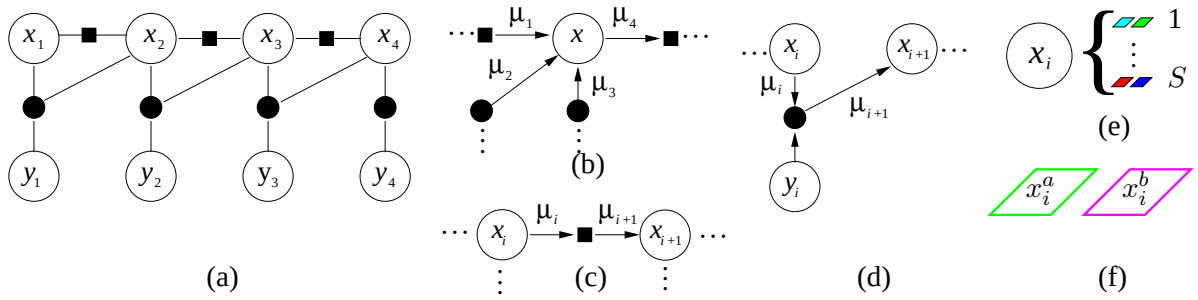


Figure 5: (a) Factor graph for 1D super resolution example. The random variables are represented by the transparent nodes where the x_i are the latent variables and the y_i are the observed variables, the solid squares represent the derivative constraint, and the solid circles represent the reconstruction constraint. In (b)-(d) we show message propagation for the three possible cases: (b) latent node to constraint node; (c) derivative constraint node to latent node; (d) reconstruction constraint node to latent node. The messages are computed via Equations 8, 9, and 10 respectively. In all of these graphs, it is important to note that the latent nodes x_i represent patches, not individual pixels. In (e), we pictorally show that for a given latent node, there are S candidate patches. In (f), we show in detail the two pixels a and b of a patch candidate for random variable x_i.

latent node i and a constraint node, where S is the total number of states for latent node i. Each component of μ_i corresponds to one of the S states of node i. To compute the message sent from latent node x_i at a particular setting to a neighboring constraint node, we take the product of all incoming message at that setting of x_i except from the target constraint node. Using Figure 5(b), we write this explicitly as follows:

$$\mu_4(x_i) \leftarrow \mu_1(x_i)\mu_2(x_i)\mu_3(x_i). \tag{8}$$

To compute the message sent from a derivative constraint node to a latent node x_{i+1} at a particular setting, we incorporate the natural image prior, as discussed in Section 2.2. Using Figure 5(c), the message is computed as follows:

$$\mu_{i+1}(x_{i+1}) \leftarrow \max_{x_i} \mu_i(x_i) \prod_{p\in\{a,b\}} \exp\left(-\frac{1}{2}\left(\frac{|x_{i+1}^p - x_i^p|}{\sigma_N}\right)^\alpha\right). \tag{9}$$

To compute the message sent from a reconstruction constraint node to a latent node x_{i+1} at a particular setting, we enforce the high to low resolution constraint, as discussed in Section 3.1. Using Figure 5(d), the message is computed as follows:

$$\mu_{i+1} \leftarrow \max_{x_i} \mu_i(x_i) \exp\left(-\frac{1}{2}\left(\frac{w^T x' - y_i}{\sigma_R}\right)^2\right) \tag{10}$$

where w is a three-tap Gaussian kernel and $x' = (x_i^a, x_i^b, x_{i+1}^a)^T$. For the two-dimensional message propagation equations, see the Appendix.

With the propagation equations in hand, we can now describe the algorithm. We follow the procedure as outlined in Section 2.3 to produce candidate patches, run BP to find the candidates with highest belief, and then construct the output image. The overall algorithm proceeds as follows:

Figure 6: Gallery of test images used in this paper. All images are of size 256×256, with the exception of image 5, which is of size 128×128.

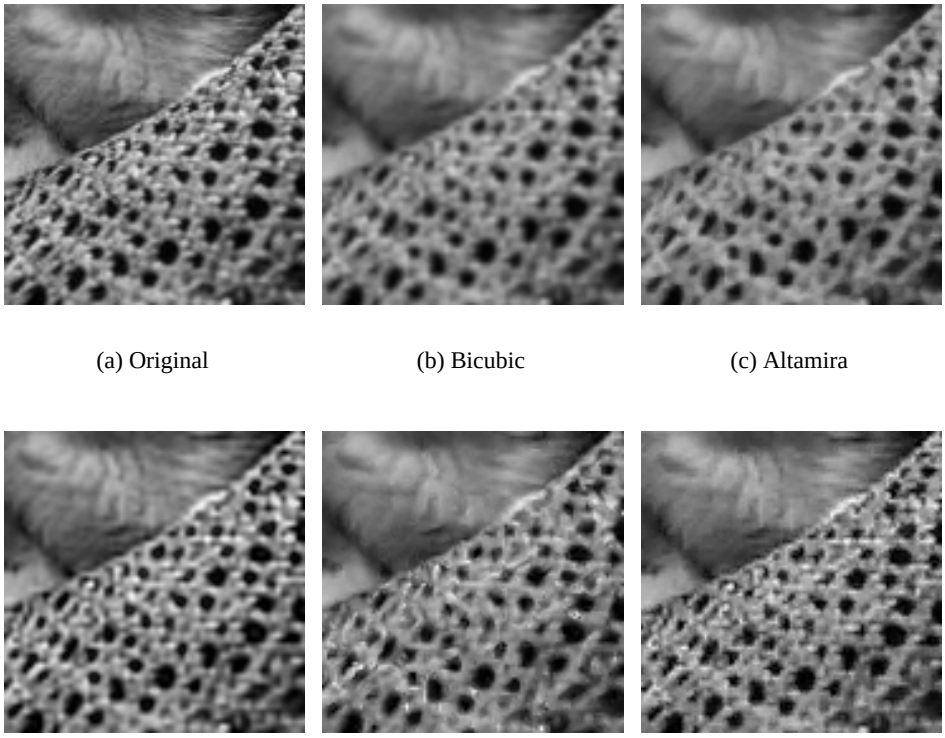
1. For each pixel p in the low resolution image:
   (a) Extract the 3×3 window of pixels centered at p. This is the local evidence.
   (b) Vectorize the pixels in the 3×3 window to form l.
   (c) Using the set of trained linear interpolators $\mathbf{T}_1 \ldots \mathbf{T}_N$ and l, linearly interpolate to obtain a set of high resolution candidate patches $h_1 \ldots h_N$.
2. With the candidate high resolution patches and observed low resolution image in hand, run BP using the derived propagation equations in the Appendix.
3. For each node, insert into the corresponding position the high resolution patch with the highest belief.

We train the set of linear interpolators by considering a set of natural images. We use a 3×3 Gaussian kernel and subsample to get a low/high resolution pair. We then extract for each low resolution pixel the corresponding 3×3 low resolution local evidence patch and 2×2 high resolution patch. With these low and high resolution patches, we train the set of linear interpolators as outlined in Section 2.3.

For the experiments in this paper, we set $\alpha = 0.7$, $\sigma_N = 1$, and $\sigma_R = 0.01$ and ran BP for 5 iterations. For training, nine 432×576 pixel grayscale natural images were used, generating roughly 500,000 low/high resolution patch pairs, and 16 linear interpolators were trained.

3.4 Results

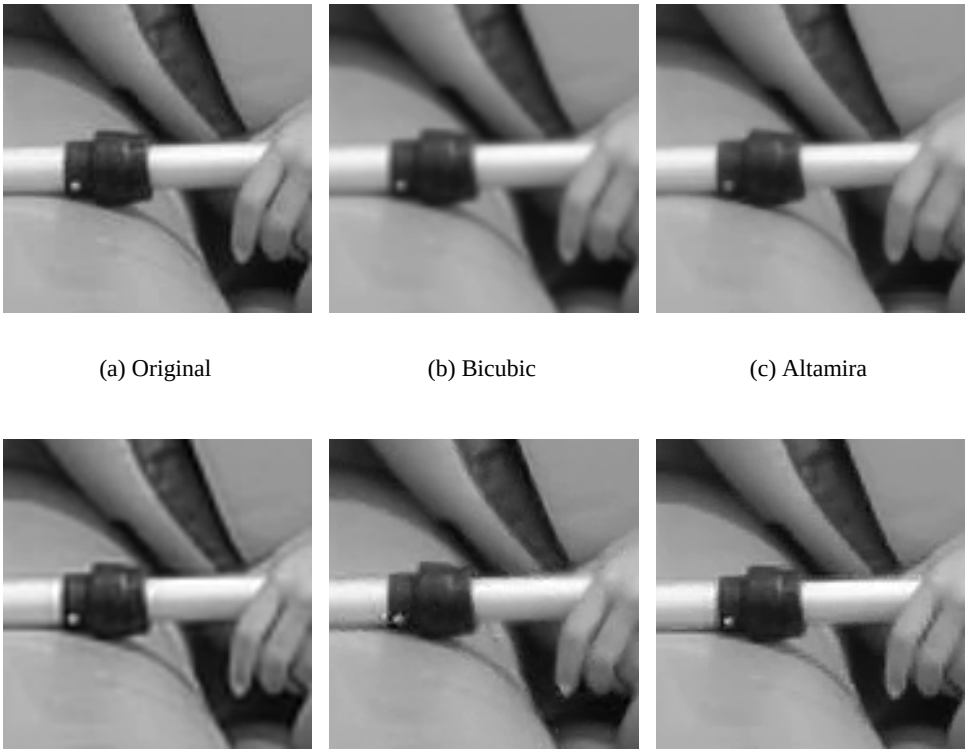
To evaluate our super resolution algorithm, we (1) decimated a test image by filtering with a 3×3 Gaussian kernel and subsampled as described above and (2) super resolved back to the original dimensions. We compared the natural image prior algorithm against the original image, bicubic interpolation, Freeman et al. fast example-based super resolution algorithm—a memory-intensive approach using graphical models [10], Photoshop Altamira plug-in—an undisclosed proprietary algorithm [1], and Greenspan et al. nonlinear enhancement algorithm—an image sharpening algorithm (here, we used band-pass filtering, $c = 0.4$, and $s = 5$) [12]. We tested our algorithm on the set of images shown in Figure 6, none of which were used for training.



(a) Original (b) Bicubic (c) Altamira

(d) Nonlinear enhancement (e) Example-based (f) Natural image prior

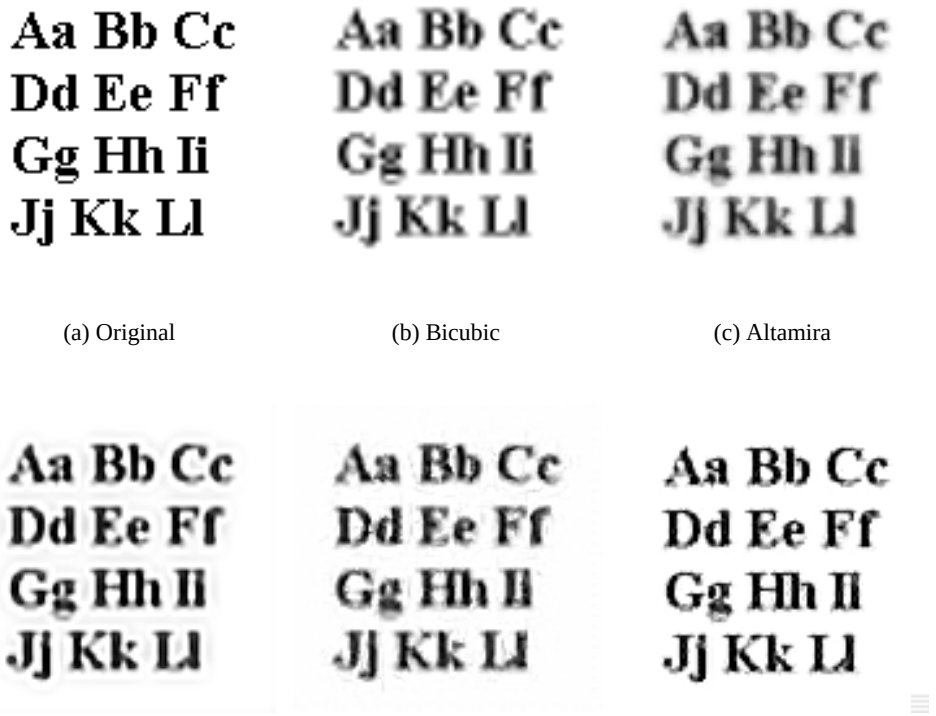
Figure 7: 128×128 textured region cropped from image 2, decimated to 64×64 and then super resolved. (a) True high resolution; (b) Bicubic interpolation; (c) Altamira; (d) Greenspan et al. nonlinear enhancement; (e) Freeman et al. example-based; (f) our natural image prior based algorithm. Notice that our natural image prior algorithm clearly out-performs the bicubic interpolation and Altamira algorithms. Also, the example-based algorithm produces noisy artifacts, which our algorithm overcomes. The nonlinear enhancement algorithm produces a sharp image as well, but at the expense of "haloing" artifacts.



(a) Original (b) Bicubic (c) Altamira

(d) Nonlinear enhancement (e) Example-based (f) Natural image prior

Figure 8: 128×128 bar region cropped from image 1, decimated to 64×64 and then super resolved. (a) True high resolution; (b) Bicubic interpolation; (c) Altamira; (d) Greenspan et al. nonlinear enhancement; (e) Freeman et al. example-based; (f) our natural image prior based algorithm. As in Figure 7, our algorithm produces a sharp image with minimal noise and "haloing" artifacts.



(a) Original (b) Bicubic (c) Altamira

(d) Nonlinear enhancement (e) Example-based (f) Natural image prior

Figure 9: 128×128 synthetic font image (not included in the test gallery), decimated to 64×64 and then super resolved [MSE in brackets]. (a) True high resolution; (b) Bicubic interpolation [0.0345]; (c) Altamira [0.0294]; (d) Greenspan et al. nonlinear enhancement [0.0740]; (e) Freeman et al. example-based [0.0599]; (f) our natural image prior based algorithm [0.0133]. As in Figures 7 and 8, we see that our algorithm produces a sharp result. Moreover, notice that the nonlinear enhancement algorithm has significant "haloing" artifacts around the fonts. These artifacts do not appear in our outputs.

A comparison of the outputs for the different super resolution algorithms are shown in Figures 7, 8, and Figure 9. Here, we show cropped sections of two natural images from the test gallery, in addition to a synthetic image of fonts. In Figure 10, we show the mean-squared error (MSE) of the images in the test gallery for the different super resolution algorithms. Notice that the presented algorithm results in the lowest MSE for all of the test images, followed by the Altamira plug-in.

In all of the images, the bicubic-interpolated method results in overly smooth outputs. Our method clearly outperforms this, producing sharper results. The example-based algorithm produces a sharp image as well, but at the expense of perceptually distracting artifacts. This is due to the database of patches that the example-based method uses to obtain candidates for each latent node. Since the database comprises patches that directly come from a set of training images, the patches tend to be noisy and dependent on the content of the training images, which our algorithm overcomes through the linear interpolators. The interpolators reduce the noise, but still provide enough variability in the higher frequencies. Moreover, our method is more efficient in time and memory usage since we do not have to search or store a database of patches– we simply interpolate. The Altamira algorithm produces sharp images, but appears to over-compensate in

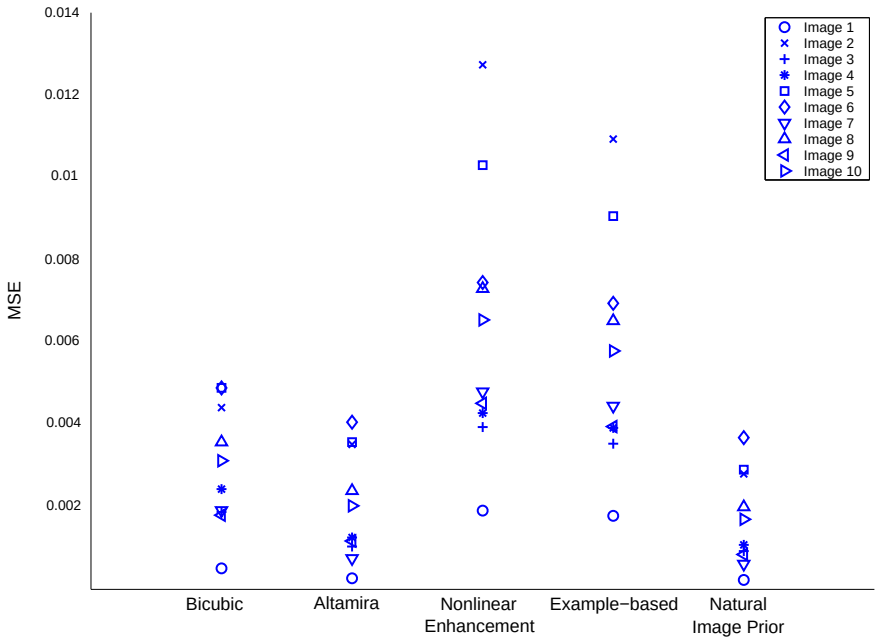


Figure 10: Plot of mean-squared error (MSE) for super resolution. Notice that our natural image prior based algorithm has the lowest MSE in comparison to the other methods shown here. While MSE is not always a good measure of image quality, for this problem we feel the MSE correlates reasonably well with the image quality for the different methods, as shown in Figures 7, 8, and 9.

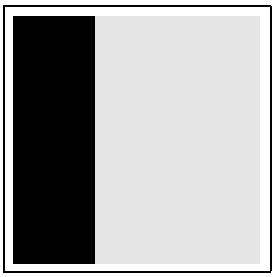
certain areas, resulting in a higher error. The Greenspan et al. nonlinear enhancement sharpens the edges, but produces ringing artifacts as can be seen in the outline of the fonts in Figure 9(d).

We compared the results when α is set to 0.7 and 2. While the former setting results in slightly lower MSE, there is not a significant improvement in visual quality over the latter setting. This may be because the interpolators are capturing well the derivative statistics. Further study of the interpolators will shed light on this issue.

4 Demosaicing CCD Output

A similar problem to super-resolution is that of estimating a full-color image from samples of only one color band. Typical CCD's are only able to sample one color band at each pixel in the sensor. This is known as poly-chromatic sampling because the samples at neighboring pixels represent the intensity of different color bands. Figure 11(c) shows the poly-chromatically sampled version of Figure 11(a), using the sampling pattern in Figure 11(b) [4]. To obtain the full-color image, with the value of three color bands per pixel, the value of the other two color bands at each pixel must be interpolated from the observed samples. This problem, known as demosaicing, is similar to the super-resolution problem in that we are trying to estimate hidden information at every pixel location in the image, except now we are trying to estimate color values instead of high-resolution information.

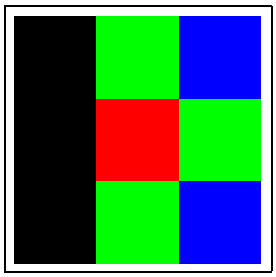
While the missing color values in each band could simply be interpolated from the observed values in that band, that ignores the correlation between color band values. A change in one color band is usually



(a) Input image

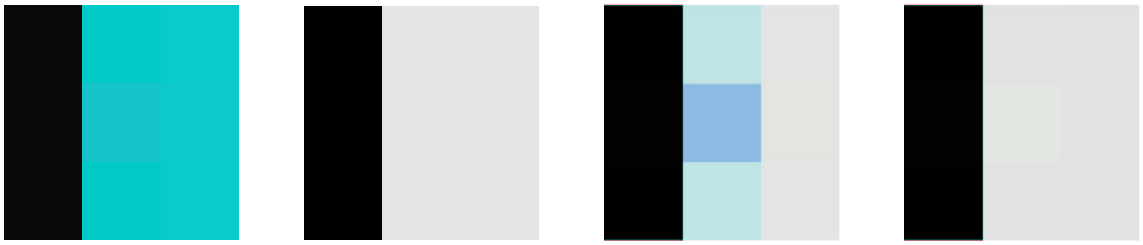
R	G	R	G	R
G	B	G	B	G
R	G	R	G	R
G	B	G	B	G
R	G	R	G	R

(b) Sampling Pattern



(c) Sampled input image

Figure 11: An example of poly-chromatic sampling and the candidate color values produced by our method. (a) The original input image. (b) The sampling pattern used (c) The input image sampled according to the pattern shown in Figure 11(b).



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

Figure 12: If the interpolation matrix best suited for the edge in (a) is applied to (b), then (c) results. However, if the best interpolation matrix for (b) is used, then (d) can be achieved.

correlated with a change in the other bands also. In order to take advantage of this correlation, researchers have proposed using all of the samples in a neighborhood around the pixel being estimated to interpolate the unobserved color bands. The interpolated color values, h, are calculated as the linear combination of the observed color samples, l:

$$h = \mathbf{T}l \tag{11}$$

where l is created by unwrapping a local $K \times K$ neighborhood into a $K^2 \times 1$ vector and $\mathbf{T}$ is a $2 \times K^2$ matrix containing the interpolation coefficients. For the results shown here, we used $K = 3$.

In [20], Brainard shows how $\mathbf{T}$ can be found using Bayesian methods. Researchers have used learning methods to find $\mathbf{T}$ from both test patterns [38] and real imagesc [15, 23]. In [23], the system also performs non-linear interpolation by expanding l to include its squared terms.

Each of these algorithms assumes that $\mathbf{T}$ is constant for the whole image, which implies that the correlation between color bands is also the same at every point. However, this correlation between color bands varies according to the structure of the image. The correlation between red and green bands will be very different for a cyan edge than for a white edge. If the interpolator best suited to the cyan edge in Figure 12(a) is applied to the white edge in 12(b), then the results, shown in Figure 12(c), will be incorrect. On the

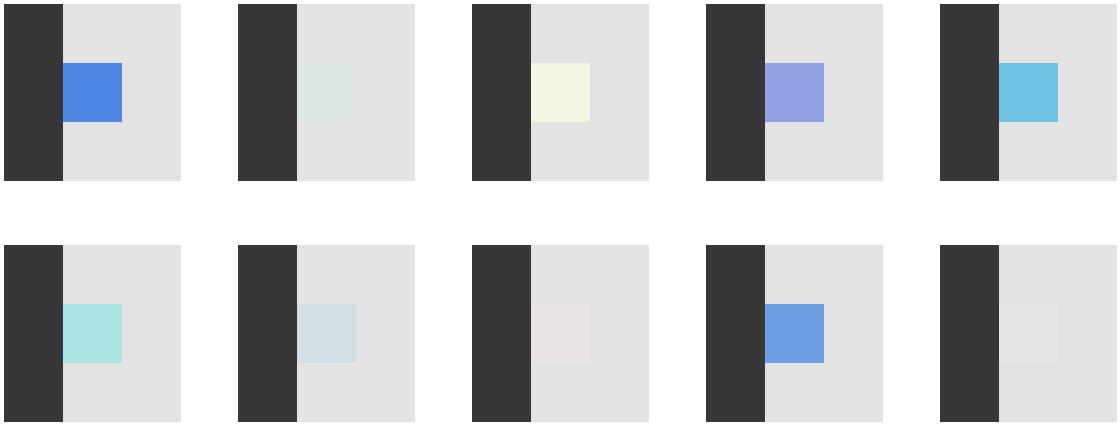


Figure 13: A portion of the candidate color values produced by the set of 20 candidate interpolators. The center pixel shows the candidate full-color value for that pixel.

other hand, if the interpolator best matched to the white edge is used, then the results, shown in Figure 12(d) are excellent.

This can be avoided by allowing the correlation between color bands to vary from pixel to pixel. To do so, the colors at each pixel are interpolated by one of a set of interpolation matrices, each of which implies a different correlation between color bands.

Effectively, this modifies equation 11 so that:

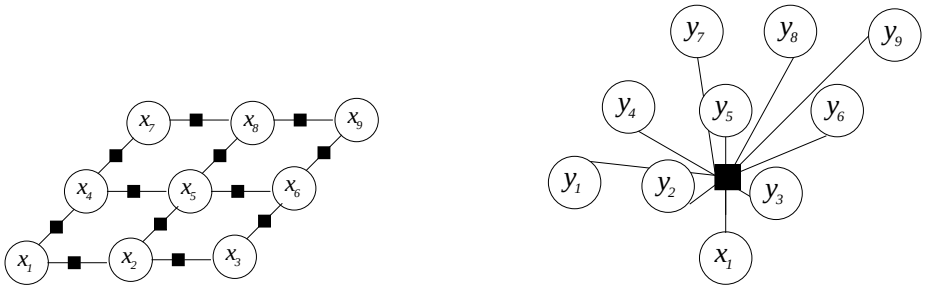
$$h = \mathbf{T}_j l \tag{12}$$

where $\mathbf{T}_j$ is one of N possible sets of interpolation coefficients. By allowing the interpolator to vary, the correlation between color bands can vary. A portion of the set of possible interpolators is shown in Figure 13. The center pixels show the candidate values produced for the center pixel in the image shown in Figure 12(b). The color of the center pixel varies according to the correlation between color channels assumed by each interpolator. The color of the center pixel varies as the correlation between color bands assumed by each matrix varies.

4.1 Model

The system is based on the factor graph shown in Figure 14. Nodes labelled y_i denote observed variables, while the state of nodes denoted x_i must be estimated. There is one x_i node associated with each pixel in the observed image. The state of a hidden node is the best interpolator, $\mathbf{T}_j$, to use at that point. The set of possible interpolators is chosen according to the method described in section 2.3. Using the set of interpolators as the state for the variables in the graphical model is vital because representing the image value directly would require a large two-dimensional state variable at each node. With interpolators, only a small number of states are needed at each node.

As shown in Figure 14(a), each unobserved node is constrained by its direct neighbors. The constraint function between two neighboring interpolators, $\psi(x_i, x_j)$, is based on the sparse image prior of the derivatives between the pixel values that would be produced by each these interpolators. To create the constraint



(a) Constraints between neighboring hidden nodes

(b) Constraint between hidden and observed nodes

Figure 14: The factor graph used for demosaicing. The graph has constraints between hidden nodes and constraints relating hidden nodes to observed image data. For clarity, the two constraints are depicted separately.

function, we use the marginal statistics of the three color bands and assume that the derivatives of each color band have the same distribution. For two neighboring candidate interpolators, $\mathbf{T}_i$ and $\mathbf{T}_j$, the constraint relating the interpolators is based on the neighboring pixels produced by the two interpolators. If p is produced by $\mathbf{T}_i$ and p' is produce by $\mathbf{T}'_j$, then

$$\psi(\mathbf{T}_i, \mathbf{T}'_j) = \exp\left(\frac{-|p_r - p'_r|^\alpha - |p_g - p'_g|^\alpha - |p_b - p'_b|^\alpha}{s}\right) \tag{13}$$

where p_r is the value of the red color band at p and the rest of the color bands are labelled accordingly. For the results shown here, we use $\alpha = 0.7$ and $s = 0.1$.

This model also includes constraint functions between each interpolator and the local observed image information, shown in Figure 14(b). Each y_i node corresponds to a pixel in the observed images. The node's state is the sample of the single color band observed at that point. The constraint between the observed image data and the hidden nodes is modeled as a multivariate gaussian

$$\psi(\mathbf{T}_n, l) = \mathcal{N}(l; \mu_n, \mathbf{\Sigma}_n) \tag{14}$$

where l is a vector containing the observed samples from a 3×3 local patch. The values of μ_n and $\mathbf{\Sigma}_n$ are found when the set of interpolators, $\mathbf{T}_1 \ldots \mathbf{T}_N$ are chosen.

4.2 Demosaicing the Image

Given a sampled input image and N candidate interpolators, we estimate the two missing color-bands at each pixel with the following steps:

1. For each pixel p:

   (a) For each interpolator, $\mathbf{T}_n$, calculate $P(\mathbf{T}_n|l)$, the likelihood that $\mathbf{T}_n$ is the best interpolator, given the local image information.

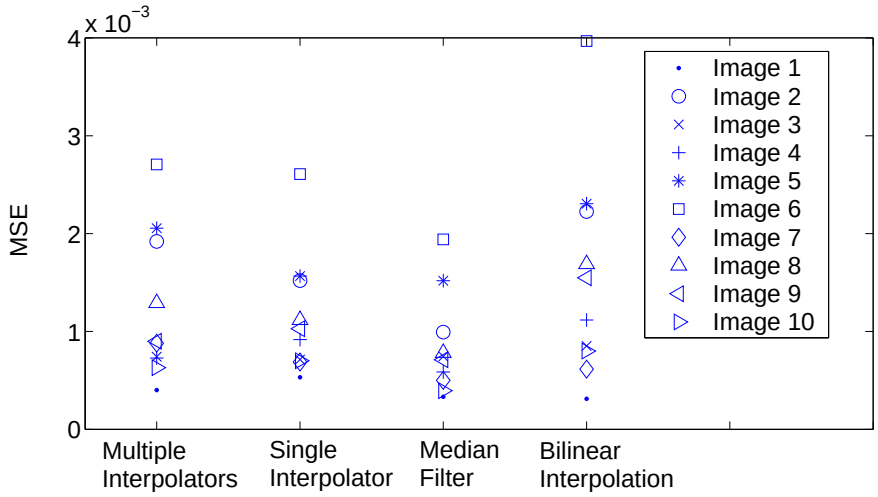


Figure 15: Comparison of the mean squared error of the reconstructions produced by four different algorithms for demosaicing. The images from Figure 6 were used for the test. While the MSE is not a good indicator of perceptual quality for this task, our algorithm performs comparably to others.

(b) Use each of the candidate interpolators to calculate a candidate value for the unobserved data.

(c) Use equation 13 to calculate the constraint function between candidate values of neighboring pixels.

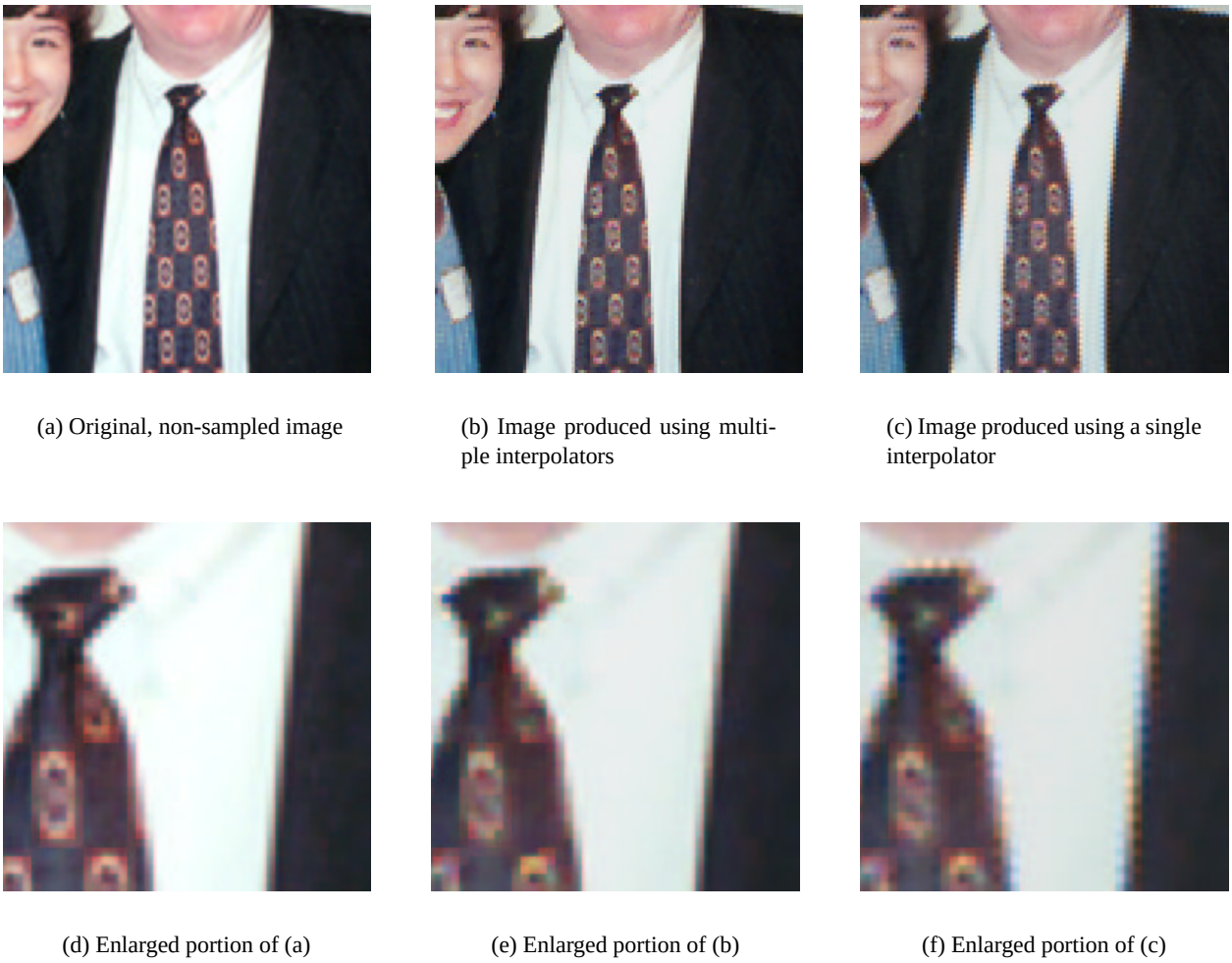
2. Using the max-product Belief Propagation algorithm, find the most likely setting for each of the random variables.

4.3 Results

For the first evaluation of our demosaicing algorithm, we compared it against using a single linear interpolator to find the two unobserved color values at each point. While [23] suggests performing non-linear interpolation by augmenting the observations with quadratic functions of the data, we found that did not improve the results on our training set. Both the single global interpolator used for comparison and the set of interpolators used by our algorithm were trained on a set of 18 natural images. The images were a combination of scanned photographs and high-resolution digital pictures. Each of the images was down-sampled to reduce the effects of any demosaicing that occurred when the images were captured. The candidate values for each pixel were created by a set of twenty different interpolators. The images were sampled according to the pattern shown in Figure 11(b). In this pattern there are actually four different types of local neighborhoods. Therefore, we learn four different sets of twenty interpolators. The choice of which interpolator set is used at a pixel depends entirely on the type of pixel being interpolated.

To evaluate the performance of the two algorithms, we use L_2 norm of the difference between each pixel of the demosaiced image and each pixel of the true image. Over the whole training set, we found that average L_2 error of the pixels produced by our method was 86% of those produced by using a single linear interpolator. Note that [23] showed that using a single interpolator produced a significant improvement over simply interpolating each color band separately.

We also used the test set shown in Figure 6 to compare our algorithm to other demosaicing solutions. Figure 15 shows the mean squared error of all color bands for four different algorithms. We compared our method to using a single interpolator, bilinearly interpolating the color planes independently, and an algorithm utilizing the median filter [9]. The median filter algorithm has been found to perform well experimentally in [20] and [26]. Our method outperforms the others, except for the median filter algorithm, in terms of MSE. The images in the test set are affected by chromaticity artifact caused by image compression. These likely adversely affected the results of our algorithm.



(a) Original, non-sampled image

(b) Image produced using multiple interpolators

(c) Image produced using a single interpolator

(d) Enlarged portion of (a)

(e) Enlarged portion of (b)

(f) Enlarged portion of (c)

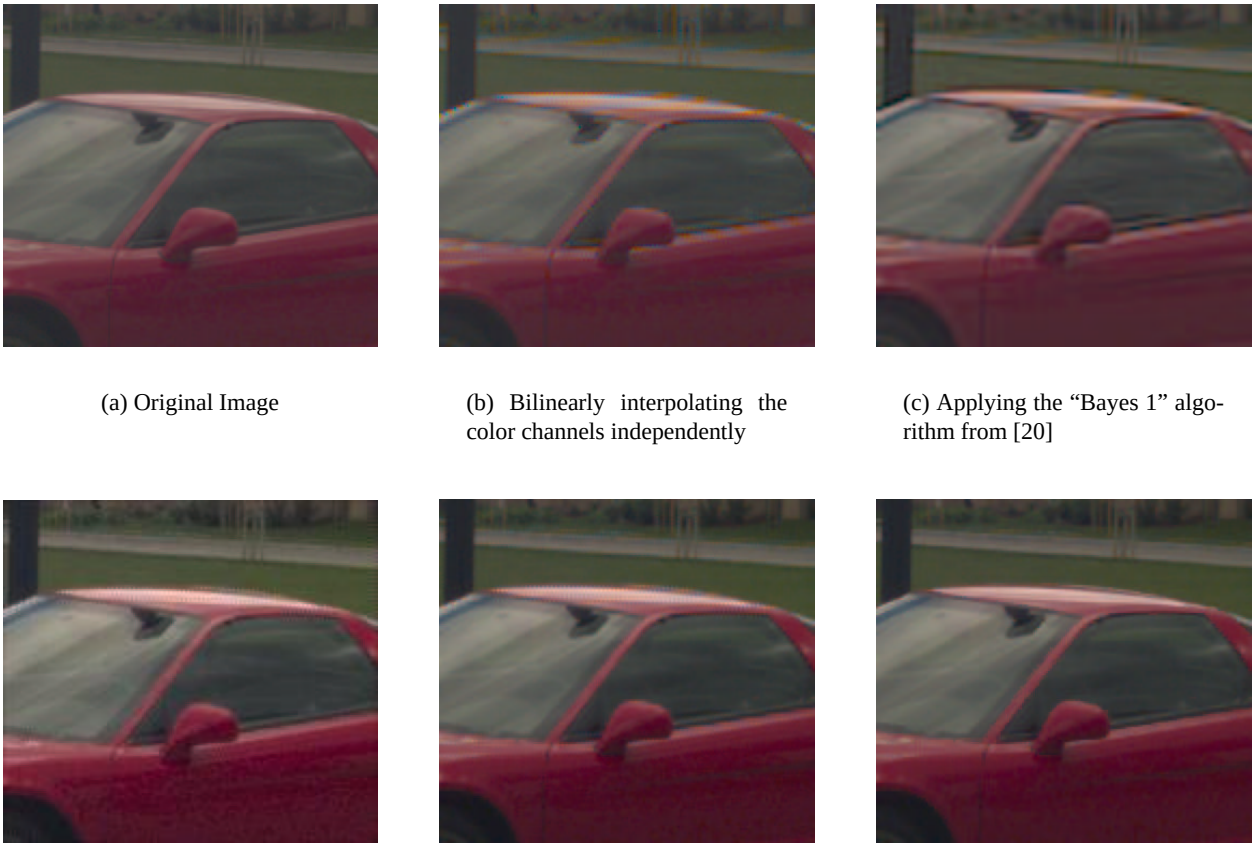
Figure 16: A comparison of the results produced by using multiple interpolators versus a single interpolator. Notice that the image produced with multiple interpolators, shown in (c), does not have the color fringes along the coat and tie.

However, the mean squared error does not capture the important perceptual differences. The important difference in performance lies along the edges in the image, where color fringing can occur. The number of pixels along edges is relatively small compared to the total number of pixels in the image, so differences in performance along the edges will not much effect on the overall MSE. Examining the images qualitatively shows a much greater improvement by using multiple interpolators, especially along edges in the image. The most noticeable artifacts of demosaicing algorithms are colored fringes along edges. Figure 16 shows the

difference in fringing caused by using one interpolator versus multiple interpolators. Using one interpolator causes the fringes along the suit coat shown in Figure 16(c). These are caused when the correlation between color bands implied in the interpolator is incorrect. For example, if the interpolator believes that red and green are correlated, a red edge will have a greenish fringe when it is demosaiced. By using multiple interpolators and belief propagation, our algorithm significantly reduces the amount of color fringing in Figure 16(b).

In Figure 17 we compare the results of our algorithm, in Figure 17(f), to two other methods for demosaicing. Figure 17(b) shows the results from independently bilinearly interpolating the three color bands. Figure 17(c) shows the results from applying the "Bayes 1" algorithm from [20]. Again the results from using multiple interpolators have significantly less color fringing.

The importance of setting the exponent α to be less than one, which was discussed in Section 2.1, is illustrated in Figure 19. Setting α greater than 1 leads to multiple small derivatives being favored over a single large derivative. This leads to the artifacts in Figure 19(b). When α is less than one, sharp edges are preferred, resulting in Figure 19(b).



(a) Original Image

(b) Bilinearly interpolating the color channels independently

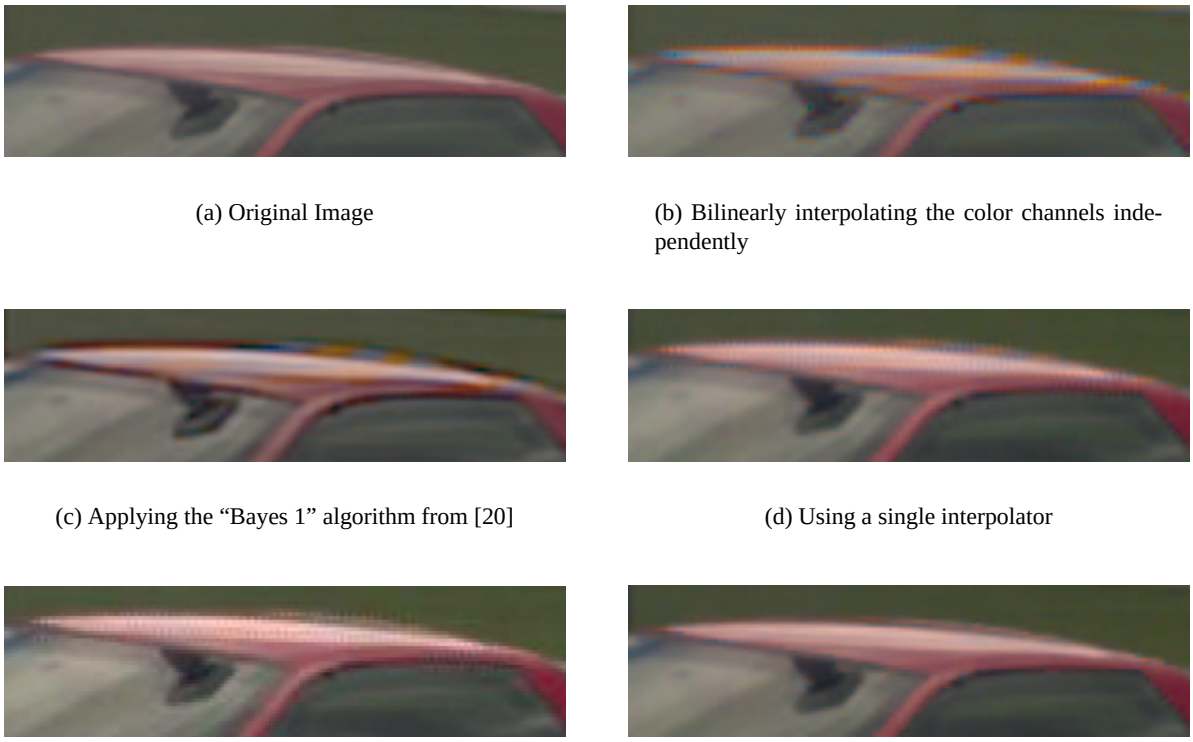
(c) Applying the "Bayes 1" algorithm from [20]

(d) Median Filter

(e) Using a single interpolator

(f) Using multiple interpolators

Figure 17: Comparision of our algorithm with two methods of demosaicing discussed in [20]. Figures 17(a), 17(b), and 17(c) came from [19]. Enlarged portions of these figures are shown in Figure 18.



(a) Original Image

(b) Bilinearly interpolating the color channels independently

(c) Applying the “Bayes 1” algorithm from [20]

(d) Using a single interpolator

(e) Median Filter

(f) Using multiple interpolators

Figure 18: Enlarged portions of the figures from Figure 18. Using multiple interpolators, shown in (f), increases the quality of the reconstruction, especially along the roof of the car.

5 Conclusion

Solving both the problem of super-resolution and CCD demosaicing requires inferring an unobserved, full-resolution image from a sampled version of the image. In this paper, we have outlined an approach which uses factor graphs to accomplish this. The unobserved patches of pixels in the full-resolution image are modeled as nodes in the graph and the characteristic distribution of the derivatives of real-world images are used to define the constraint functions between nodes in the factor graph. The general nature of this approach makes it applicable to many problems besides super-resolution and CCD demosaicing.

To make inference computationally feasible, we use discrete-valued nodes in the factor graph. Instead of using the states of each variable to represent actual image values, each state represents a linear mapping that relates the observed image data to the unobserved, full-resolution image. This permits a small number of discrete states to model the range of possible image values at each point in the image. This computational efficiency makes this approach valuable for tasks where a full-resolution image must be found from a set of samples.

(a) Sample results for $\alpha = 0.7$

(b) Sample results for $\alpha = 2.0$

Figure 19: The effect of the exponent α on the results. The results are sharper when $\alpha = 0.7$ because the statistical prior favors fewer, large derivatives.

Acknowledgments

Erik Sudderth provided valuable suggestions, particularly with regards to factor graphs and the graphical models in Figure 5(a). This work was funded by the Nippon Telegraph and Telephone Corporation as part of the NTT/MIT Collaboration Agreement, in addition to the Texas Instruments DSPS Leadership University Program. MFT was funded by an NDSEG Fellowship from the Department of Defense.

Appendix

Here, we give the two-dimensional propagation equations for the super resolution problem. The natural image prior and reconstruction constraints are shown graphically in Figure 20. For the natural image constraint, we use the derivative kernel $[-1, 0, 1]$ and apply it in four directions (horizontal, vertical, and two diagonal directions) as shown in Figure 20(a)-(d). The propagation equation is given by:

$$\mu_2(x_2) \leftarrow \max_{x_1} \mu_1(x_1) \prod_{p \in \{a,b,c,d\}} \exp\left(-\frac{1}{2}\left(\frac{|x_2^p - x_1^p|}{\sigma_N}\right)^\alpha\right). \tag{15}$$

The reconstruction constraint is shown graphically in Figure 20(e) and is given by the propagation equation:

$$\mu_4(x_4) \leftarrow \max_{x_1} \max_{x_2} \max_{x_3} \mu_1(x_1)\mu_2(x_2)\mu_3(x_3) \exp\left(-\frac{1}{2}\left(\frac{W * x' - y_1}{\sigma_R}\right)^2\right) \tag{16}$$

where W is a 3×3 Gaussian kernel and:

$$x' = \begin{pmatrix} x_1^a & x_1^c & x_3^a \\ x_1^b & x_1^d & x_3^b \\ x_2^a & x_2^c & x_4^a \end{pmatrix}. \tag{17}$$

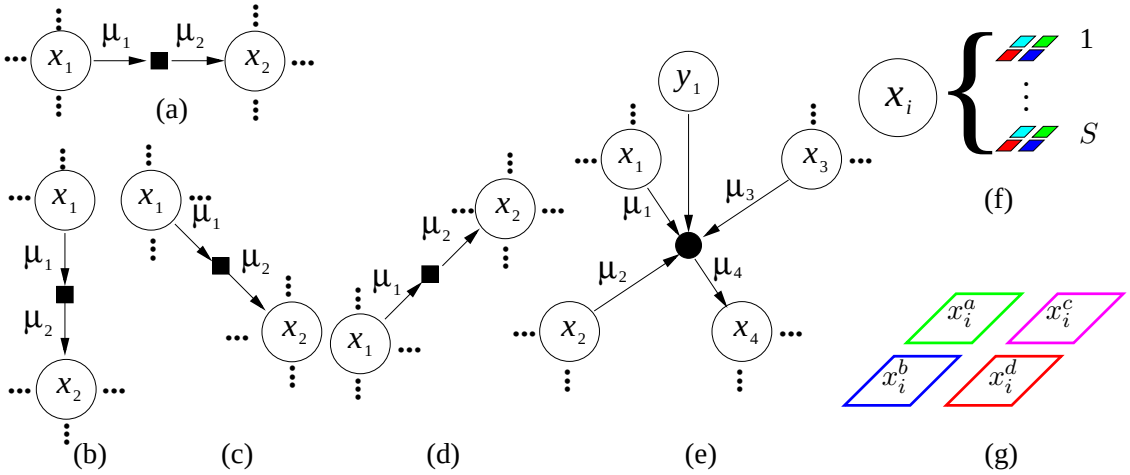


Figure 20: (a)-(d) Factor graph segment for the directional derivative constraint (horizontal, vertical, and two diagonal directions respectively). (e) The graph segment for the reconstruction constraint. In each of the segments, it is important to remember that the latent nodes x_i represent patches, not individual pixels. In (f), we pictorially show that for a given latent node, there are S candidate patches. In (g), we show in detail a given patch candidate for x_i.

References

[1] Altamira Genuine Fractals 2.5, www.lizardtech.com.

[2] S. Baker and T. Kanade. Hallucinating faces. *Fourth International Conference on Automatic Face and Gesture Recognition*, 2000.

[3] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *IEEE Conf. Computer Vision and Pattern Recognition*, 2000.

[4] B. E. Bayer. Color imaging array. US Patent No. 3,971,065, 1975.

[5] M. Belge, M. Kilmer, and E. Miller. Wavelet domain image restoration with adaptive edge-preserving regularity. *IEEE Trans. Image Processing*, 9(4):597–608, 2000.

[6] J. Dias. Fast GEM wavelet-based image deconvolution algorithm. *IEEE International Conference on Image Processing-ICIP'03*, 2003.

[7] M. Figueiredo and R. Nowak. Image restoration using the EM algorithm and wavelet-based complexity regularization. *IEEE Transactions on Image Processing*, 2002.

[8] W. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[9] W. T. Freeman. Median filter for reconstructing missing color samples. U.S. Patent No. 5,373,322, 1988.

[10] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super resolution. *IEEE Computer Graphics and Applications*, 2002.

[11] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[12] H. Greenspan, C. Anderson, and S. Akber. Image enhancement by nonlinear extrapolation in frequency space. *IEEE Trans. on Image Processing*, 9(6), 2000.

[13] H. H. Hou and H. C. Andrews. Cubic splines for image interpolation and digital filtering. *IEEE Trans. Acoust. Speech Signal Processing*, ASSP-26(6):508–517, 1978.

[14] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE Trans. Acoustics, Speech, Signal Processing*, 29(6):1153–1160, 1981.

[15] K. Knopf and R. Morf. A new class of mosaic color encoding patterns for single-chip cameras. *IEEE Transactions on Electron Devices*, ED-32(8), August 1985.

[16] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 42(2):498–519, 2001.

[17] A. Levin, A. Zomet, and Y. Weiss. Learning to perceive transparency from the statistics of natural images. *Neural Information Processing Systems*, 2002.

[18] C. Liu, H. Shum, and C. Zhang. A two-step approach to hallucinating faces: global parametric model and local nonparametric model. *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2001.

[19] P. Longere, P. B. Delahunt, X. Zhang, and D. H. Brainard. Supplementary material for perceptual assesment of demosaicing algorithm performance. *http://color.psych.upenn.edu/depreference/index.html*.

[20] P. Longere, P. B. Delahunt, X. Zhang, and D. H. Brainard. Perceptual assessment of demosaicing algorithm performance. *Proceedings of the IEEE*, 90:123–132, 2002.

[21] S. G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 11(7):674–694, July 1989.

[22] B. Morse and D. Schwartzwald. Image magnification using level set reconstruction. *Proc. International Conf. Computer Vision (ICCV)*, pages 333–341, 2001.

[23] S. K. Nayar and S. G. Narasimhan. Assorted pixels: Multi-sampled imaging with structural models. In *ECCV (4)*, volume 2353 of *Lecture Notes in Computer Science*, pages 636–652. Springer, 2002.

[24] R. Neelamani, H. Choi, and R. Baraniuk. Wavelet-based deconvolution for ill-conditioned systems. *IEEE Trans. on Image Processing*, 2001.

[25] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[26] R. Ramanath, W. E. Snyder, G. L. Bilbro, and W. A. S. III. Demosaicking methods for bayer color arrays. *Journal of Electronic Imaging*, 11(3):306–315, July 2002.

[27] W. F. Schreiber. *Fundamentals of Electronic Imaging Systems*. Springer-Verlag, New York, 1986.

[28] R. R. Schultz and R. L. Stevenson. A Bayesian approach to image expansion for improved definition. *IEEE Trans. Image Processing*, 3(3):233–242, 1994.

[29] E. P. Simoncelli. Statistical models for images: Compression, restoration and synthesis. In *31st Asilomar Conference on Signals Systems, and Computers*, pages 673–678, Pacific Grove, CA., 1997.

[30] E. P. Simoncelli. Bayesian denoising of visual images in the wavelet domain. In P. Muller and B. Vi-

dakovic, editors, *Bayesian Inference in Wavelet Based Models*, volume 141 of *Lecture Notes in Statistics*, pages 291–308. Springer-Verlag, New York, 1999.

[31] E. P. Simoncelli and R. W. Buccigrossi. Embedded wavelet image compression based on a joint probability model. In *4th IEEE International Conference on Image Processing*, 1997.

[32] A. Storkey. Dynamic structure super-resolution. *Neural Information Processing Systems*, 2002.

[33] J. Sun, N. Zheng, H. Tao, and H. Shum. Image hallucination with primal sketch priors. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[34] S. Thurnhofer and S. Mitra. Edge-enhanced image zooming. *Optical Engineering*, 35(7):1862–1870, 1996.

[35] A. Torralba and W. T. Freeman. Properties and applications of shape recipes. In *IEEE Conf. Computer Vision and Pattern Recognition*, volume 2, pages 383–390, 2003.

[36] Y. Wan and R. Nowak. A wavelet-based approach to joint image restoration and edge detection. *SPIE Conference on Wavelet Applications in Signal and Image Processing VII*, 1999.

[37] Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. Information Theory, Special Issue on Codes on Graphs and Iterative Algorithms*, 47(2):723–735, 2001.

[38] M. A. Wober and R. Soini. Method and apparatus for recovering image data through the use of a color test pattern. U.S. Patent 5,475,769, December 1995.

[39] S. Zhu and D. Mumford. Prior learning and Gibbs reaction-diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11), 1997.