**DRAFT – To appear with revisions in CVPR 2004**

# Efficient Graphical Models for Processing Images

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

*Graphical models are powerful tools for processing images. However, the large dimensionality of even local image data poses a difficulty: representing the range of possible graphical model node variables with discrete states leads to an overwhelmingly large number of states for the model, making inference computationally intractable. We propose a representation that allows a small number of discrete states to represent the large number of possible image values at each pixel or local image patch. Each node in the graph represents the best regression function, chosen from a set of candidate functions, for estimating the unobserved image pixels from the observed samples. This permits a small number of discrete states to summarize the range of possible image values at each point in the image. Belief propagation is then used to find the best regressor to use at each point. To demonstrate the usefulness of this technique, we apply it to two problems: super-resolution and color demosaicing. In both cases, we find our method compares well against other techniques for these problems.*

## 1. Introduction

Recent advances in methods for performing approximate inference in graphical models, such as belief propagation [17] and graph cuts [4], have enabled their modelling power to be applied to many vision problems. A graphical model is especially useful for problems which focus on estimating images because a graphical model makes it convenient to express relationships between pixels in both the observed image and the image or scene being estimated.

The chief barrier to using graphical models for estimating images is the huge dimensionality of images. Typically, each pixel or patch of pixels in the image being estimated is treated as a separate variable or node in the graph. A simple discrete representation of images would allocate one discrete state of each variable for each possible gray-level in the image. For a typical gray-scale image, this would require 256 states per node in the graphical model. If each variable corresponds to a patch of $N$ pixels, then $256^N$ states will be required! For algorithms such as belief propagation, the amount of computation required for approximate inference in a graphical model typically varies on the order of $N^2 \times M$ where $M$ is the number of nodes in the graph and $N$ is the number of discrete states per node. The quadratic relationship between the complexity of inference and the number of states limits the sort of problems that discrete-valued graphical models can be applied to.

The ability to express this task in a graphical model with discrete states is important because many of the statistics of natural images are non-Gaussian [19]. This prevents the use of continuous Gaussian models. A simple strategy for keeping inference tractable is to reduce the number of states for each node in the graph. In this paper, we show a convenient representation which allows a graphical model with a low number of states per node to infer high-dimensional image values. A significant benefit of this representation is that the probabilistic model does not have to be designed with this representation in mind. The probabilistic model for the task can be first specified as if a single state were assigned to each gray-level in the image. The model is then easily adapted for use with this representation.

Section 2 describes the representation. To demonstrate its usefulness, we apply it to two problems: super-resolution and color demosaicing. The applications are described in Sections 3, 4, and 5. For both applications, our model takes advantage of the regular statistics of natural images [19]. Our results compare well with other techniques for these problems.

## 2. Using a Low Number of States

The basic task is to estimate some quantity $h_i$, from $N_i(L)$, a neighborhood of pixels in the observed image $L$. For example, $h_i$ could be pixels from an unobserved high-resolution image, while $L$ is the low-resolution observation. Simple linear regression is unlikely to work generally because the relationship between $h_i$ and $L$ is likely not modelled well with a simple Gaussian distribution. However, consider a set of linear regressors, each trained over a subset of the $(h_i, N_i(L))$ pairs. If each regressor in the set were applied to some particular $N_i(L)$, it is likely that at least one of them would predict the corresponding $h_i$ accurately, if the set of linear regressors was large enough.

The problem of estimating $h_i$ can then be rephrased as choosing the linear regressor that best predicts $h_i$ from the observed data. Assume that a set of $N$ regression functions, $M_1(\cdot) \ldots M_N(\cdot)$ has been defined. Let $M_{\widehat{M}_i}(N_i(L))$ be the regressor that best estimates $h_i$. In our representation, the best estimate of $h_i$ is then

$$\hat{h}_i = M_{\widehat{M}_i}(N_i(L)) \tag{1}$$

where $\widehat{M}_i$ is the index of the interpolator chosen at location $i$ that predicts $h_i$ most accurately.

In the applications demonstrated in this paper, we will use a graphical model to find $\widehat{M}$ at each spatial location. A convenient feature of this representation based on multiple regressors is that a compatibility function designed for a representation which assigns one state per possible gray level can be easily transformed to create a compatibility function relating two neighboring regressor indices. Given a compatibility function $\psi'(h_i, h_j)$ between two neighboring high-resolution values, the compatibility between the regressors assigned to the two variables $\widehat{M}_i$ and $\widehat{M}_j$ can be defined as

$$\psi(\widehat{M}_i, \widehat{M}_j) = \psi'(M_{\widehat{M}_i}(N_i(L)), M_{\widehat{M}_j}(N_i(L))) \tag{2}$$

The compatibility between two neighboring regressors is found by first using the observed image to calculate the estimates of $h_i$ and $h_j$, using $M_{\widehat{M}_i}$ and $M_{\widehat{M}_j}$ as the regression functions. The original compatibility function is then applied to $h_i$ and $h_j$ to define the compatibility between $\widehat{M}_i$ and $\widehat{M}_j$.

The regressors, $M_1 \ldots M_n$, are found using training data. For the two applications shown here, we trained the regressors using an algorithm similar to the EM algorithm, but with hard assignments.

Each regression function $M$ can be viewed as a scene recipe [20], a simple function that translates the observed image to $h_i$, the quantity being estimated.

## 2.1. Statistical Interpretation

This representation can be described statistically by modelling the joint distribution of $h_i$ and $N_i(L)$ as a mixture of Gaussians. In this interpretation, $M_j$ denotes one component of the mixture:

$$\begin{aligned} P(h_i, N_i(L)) &= \sum_{j=1}^{N} \pi_j M_j(h_i, N_i(L)) \\ &= \sum_{j=1}^{N} \pi_j \mathcal{N}(\{h_i, N_i(L)\}; \mu_j, \Sigma_j) \end{aligned} \tag{3}$$

where $\pi_1 ... \pi_N$ are the mixing coefficients. In this model $\mu_1 \ldots \mu_N$ and $\Sigma_1 \ldots \Sigma_N$ are known and identical for every $h_i$. Only $\pi_1 ... \pi_N$ vary from pixel to pixel.

Even if $\pi_1 ... \pi_N$ are estimated first, maximizing Equation 3 will still require an iterative search for the maximum. This will be infeasible for every pixel in an image, so we approximate the value of $h_i$ that maximizes Equation 3 by choosing the most likely mixture component, $M_{\hat{j}}$, then calculating the most likely value of $\hat{h}_i$, using $M_{\hat{j}}$ as the joint distribution:

$$\hat{h}_i = \arg \max_{h_i} M_{\widehat{M}_i}(h_i, N_i(L)) \tag{4}$$

where $\widehat{M}_i = \arg \max_j \pi_j$.

In the statistical view, our goal is to pick the mixture component that best models the joint distribution of $N_i(L)$ and $h_i$. Standard estimation techniques can then be used to find the best estimate of $h_i$ using $N_i(L)$ and this distribution.

## 2.2. Error Introduced by this Representation

If the set of regression functions is large enough, one of the regressors will typically produce a value very close to the true value. In Figure 1, we show the average mean absolute error per pixel per color channel introduced by using regression functions generate full-color values for the demosaicing problem described in Section 5. This error is calculated from a testing set by using the set to generate high-resolution samples and the corresponding low-resolution observations. Each regressor then estimates the high-resolution samples using the low-resolution observations. The vertical axis in Figure 1 shows the average mean absolute error per color channel between the true high-resolution data and the estimate closest to the high-resolution data. This is a lower-bound on the error. If the model for choosing the right regression function at each point was perfect, the system would still have the amount of error shown in Figure 1 because of the approximation.

Fortunately, the lower bound on the error is quite low. Using 16 regressors, the mean absolute error averages less than one half of a gray level out of 256 possible gray levels per color channel. This low error permits an algorithm designer to focus on perfecting the model. The regression based discretization described in Section 2 can then be applied without introducing significant error.

## 3. Utilizing a Regression-Based Representation

To show the usefulness of this approximation, we show how it can be used for two useful image applications:

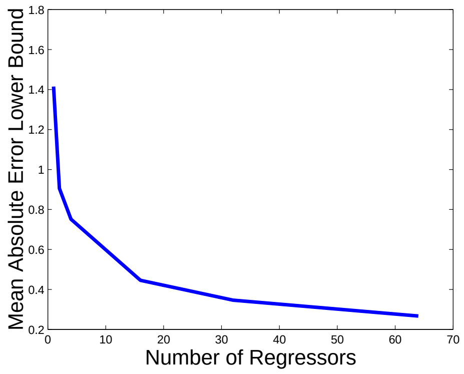


Figure 1: The average mean absolute error per pixel per color channel between the true high-resolution data and the estimate produced by the regressors that is closest to the high-resolution data.

super-resolution and image demosaicing. In both applications, low-resolution observations are used to infer a high-resolution image. These applications also demonstrate a general strategy for designing a system:

1. Design compatibility functions as if there was a single state allocated for each possible pixel value.
2. Use training data to learn a fixed set of linear regression functions $M_1 \dots M_N$.
3. Use Equation 2 to define new compatibility functions.
4. For all locations $i$, use the max-product belief propagation [17] algorithm to choose $\widehat{M}_i$, the index of the best linear regressor for estimating the high resolution information at location $i$ from the observed low resolution image.
5. For all locations $i$, let $\hat{h}_i = M_{\widehat{M}_i}(N_i(L))$.

Details on the belief propagation algorithm can be found [7, 17]

## 4. Super Resolution

The first application we consider is the task of super resolution. The goal of a super resolution resolution algorithm is to produce a high-resolution image from a single low-resolution image. This task is also often referred to as image interpolation.

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 [9]. 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 [8, 15]. These methods result in sharper images, but may contain haloing artifacts. An alternate solution involves deconvolving the blurring filter [3, 21]. While the results are quite good, deconvolution methods, as well as image sharpening methods, only enhance features that are present in the low resolution image. Learning-based methods use prior information to enhance the solution space [1, 7, 6]. These methods add realistic image details, but can also add artifacts.

### 4.1. Model

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 and then down-sampling.

To take advantage of the local spatial regularities of images, $H$ is modelled as a collection of non-overlapping $4 \times 4$ patches. Each $x_i$ represents the value of the $4 \times 4$ patch at location $i$. By choosing this patch size, we are increasing the resolution of the input image by a factor of 4.

To recover $H$ from $L$, we model the joint probability distribution of $H$ and $L$ as a graphical model with two types of compatibility functions [7]. The first compatibility function, $\phi'(\cdot)$, constrains $H$ to match $L$ when $H$ is down-sampled. If $y_i$ is a low-resolution pixel in $L$, let $p_i$ be the pixel in $H$ such that $N_{p_i}(H) * w = y_i$, where $w$ is the anti-aliasing filter used before down-sampling the image, $N_{p_i}(H)$ is a neighborhood of pixels in $H$ surrounding $p_i$, and $*$ is the convolution operator evaluated at $p_i$. The pixel $p_i$ can be thought of as the center of the group of high-resolution pixels used to generate $y_i$. With these definitions, the compatibility function for a low-resolution observation $y_i$ and a set of patches is

$$\phi'(\{x\}_{p_i}, y_i) = \exp\left(-\frac{1}{2}\left(\frac{N_{p_i}(H) * w - y_i}{\sigma_R}\right)^2\right) \quad (5)$$

where $\{x\}_{p_i}$ denotes the smallest set of patches necessary to cover $N_{p_i}(H)$, assuming $N_{p_i}(H)$ is the same size as the filter $w$. This compatibility function can be interpreted as forcing the inferred high-resolution image to match the low-resolution observation when it is scaled down.

The second compatibility function in our system relates the patches in $H$ to each other. The constraint is based on the observation that the distribution of an image derivative, $\Delta h$, in a natural image is described well as

$$p(\Delta h) \propto e^{-\frac{|\Delta h|^\alpha}{s}} \quad (6)$$

where $\alpha$ typically ranges between 0.5 and 1.2 [14]. We use this distribution to produce $\psi'(\cdot)$, which is the compatibility between neighboring patches in the high-resolution image:

$$\psi'(x_i, x_j) = \prod_{(p_i, p_j)} e^{-\frac{|p_i - p_j|^\alpha}{s}} \quad (7)$$

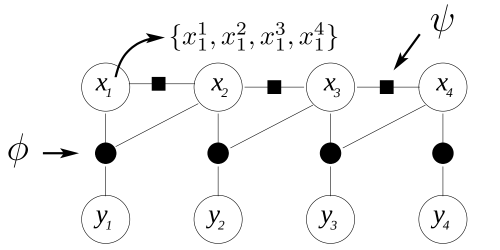


Figure 2: Factor graph for 1D super resolution example. The random variables are represented by the transparent nodes where the $x_i$ are the latent variables and the $y_i$ are the observed variables, the solid squares represent $\psi(\cdot)$, the constraint between neighboring patches, and the solid circles represent $\phi(\cdot)$, the compatibility function between the high-resolution signal and low-resolution observations. The nodes labelled $x_i$ represent patches, so the upsampling is not visible from this graph.

where $(p_i, p_j)$ are neighboring pixels with $p_i$ contained in the patch $x_i$ and $p_j$ in $x_j$. Effectively, the product is calculated along the border of the patches $x_i$ and $x_j$. For our super-resolution experiments, we set $\alpha = 0.7$ and $s = 0.01$.

Given $L$, we estimate $H$ by using the steps described in Section 3. For each patch $x_i$, we estimate $\widehat{M_i}$, which is the index of the regression function which best predicts the value of $x_i$ from the low-resolution image $L$. Once $\widehat{M_i}$ is found, $x_i$ can be estimated using $M_{\widehat{M_i}}(\cdot)$. The distribution of the indices $\widehat{M}$ is modelled as a graphical model:

$$P(\{\widehat{M}\}|L) \propto \prod_i \phi(\{\widehat{M}\}_{y_i}, y_i) \prod_{(x_j,x_k)} \psi(x_j, x_k) \qquad (8)$$

where $\{\widehat{M}\}_{y_i}$ denotes the indices of the smallest set of patches which affect the low-resolution pixel at the same location as $y_i$ when $H$ is down-sampled. The compatibility functions $\psi(\cdot)$ and $\phi(\cdot)$ are found using $\psi'(\cdot)$ and $\phi'(\cdot)$ using the same method as in Equation 2. Once the compatibility functions have been defined, $\widehat{M}$ can be found for each patch by using the max-product Belief Propagation algorithm.

Figure 2 shows a 1-D example of the model defined by Equation 8, visualized as a factor graph [11]. Each node labelled $x_i$ represents four high-resolution values. Each node labelled $y_i$ represents a low-resolution observation. The solid squares represent $\psi(\cdot)$, the constraint between neighboring patches. The solid circles represent $\phi(\cdot)$, the compatibility function between the high-resolution signal and low-resolution observations.

### 4.2. Results

The linear interpolators are trained using a set of natural images. We use a $7 \times 7$ Gaussian kernel and subsample to get a low/high resolution pair. For each low resolution pixel, the corresponding $3 \times 3$ low resolution local evidence patch is extracted along with the $4 \times 4$ high resolution patch. For the experiments in this paper, we set $\alpha = 0.7$, $s = 1$, and $\sigma_R = 0.01$ and ran BP for 10 iterations. The training set consisted of nine $432 \times 576$ pixel gray-scale natural images, which generated roughly 500,000 low/high resolution patch pairs that were used to train 64 linear interpolators.

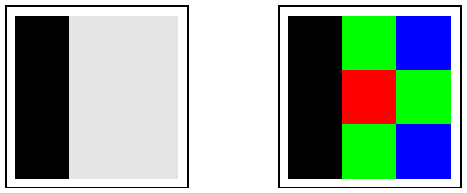
To evaluate our super resolution algorithm, we first decimated a test image by filtering it with a $7 \times 7$ Gaussian kernel then sub-sampled it. Next we super resolved back to the original dimensions.For comparison, we show our results against two other super-resolution algorithms. We compared the our algorithm against bicubic interpolation and a nonlinear image sharpening algorithm [8]. For the sharpening algorithm, we used band-pass filtering with the algorithm-specific parameters set to $c = 0.2$, and $s = 7$.

A comparison of the outputs for the different super resolution algorithms are shown in Figure 4 and Figure 3. In both cases, using bicubic interpolation results in overly smooth results. In Figure 3, our method produces noticeably sharper images without significant artifacts, such as the halos around the specularities in Figure 3. Our method introduces some artifacts in Figure 4(d), but still produces the sharpest results.

Our method also outperforms the others in terms of the mean absolute error between each algorithm's results and the true high-resolution image. The mean absolute error is listed in the captions of Figures 4 and 3.

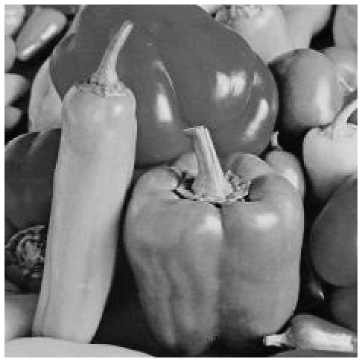
## 5. Demosaicing CCD Output

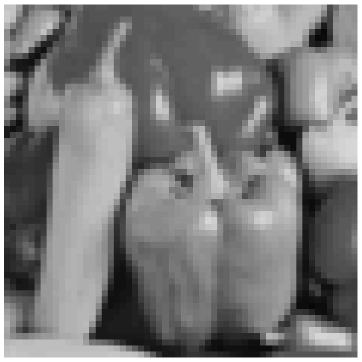
A similar problem to super-resolution is that of estimating a full-color image from samples of only one color band. Typical CCD's are only able to sample one color band at each pixel in the sensor. This is known as poly-chromatic sampling because the samples at neighboring pixels represent the intensity of different color bands. Figure 5(b) shows

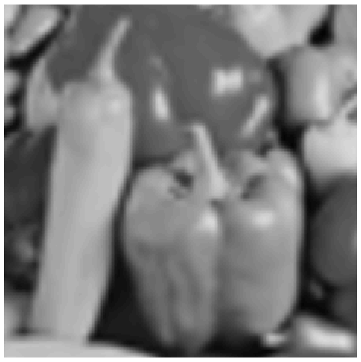


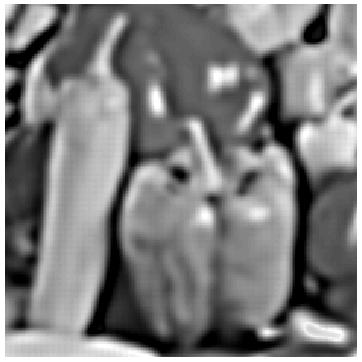
(a) Input image (b) Sampled input image

Figure 5: An example of poly-chromatic sampling. (a) The original input image. (b) The polychromatically sampled version of the input image.


(a) Original High Resolution Input


(b) Pixel Replication


(c) Bicubic Interpolation


(d) Nonlinear enhancement

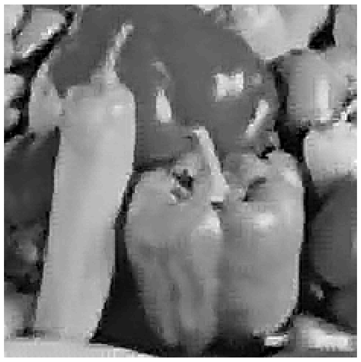
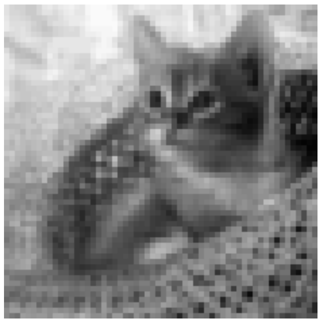
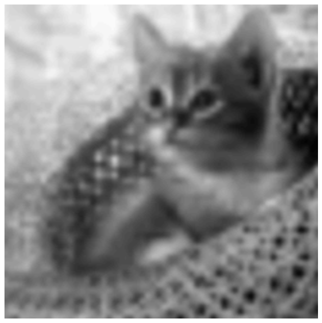
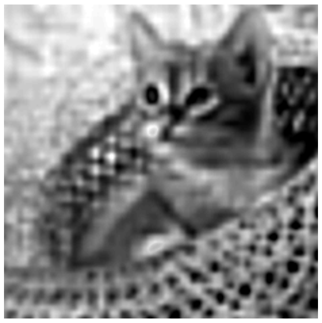

(e) Multiple Regressors

Figure 3: Results on interpolating a $64 \times 64$ image of peppers. The mean absolute error between each result and the true high-resolution values in shown in brackets. (a) The original high-resolution image (b) Interpolated using pixel replication. This represents the low-resolution input. (c) Using bicubic interpolation [12.5] (d) Using the Greenspan et al. nonlinear enhancement algorithm [34.8] (e) Using multiple regressors [7.9]. Our method produces a noticeably sharper image without the "haloing" artifacts seen in (d).


(a) Pixel Replication


(b) Bicubic Interpolation


(c) Nonlinear enhancement

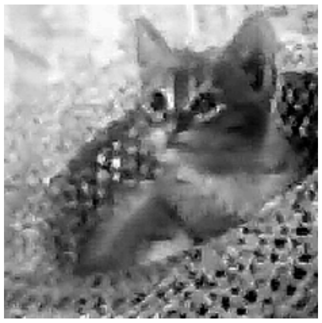
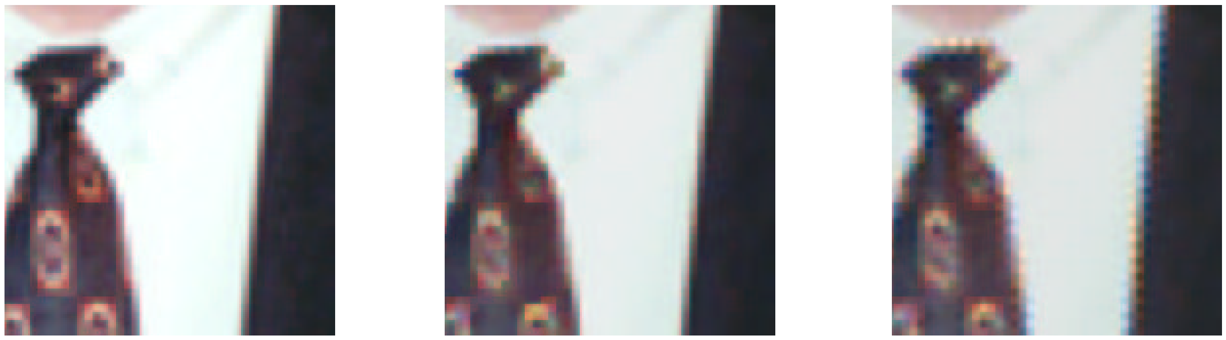

(d) Multiple Regressors

Figure 4: Results on interpolating a $64 \times 64$ image of a cat. The mean absolute error between the result and the true high-resolution image is shown in brackets. (a) Interpolated using pixel replication (b) Using bicubic interpolation [19.5] (c) Using the Greenspan et al. nonlinear enhancement algorithm [37.2] (d) Using multiple regressors [16.7]. In this case, our method produces some artifacts, but still produces the sharpest results.



(a) True Image (b) Multiple Regression functions (c) Single Interpolator

Figure 6: 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.

the poly-chromatically sampled version of Figure 5(a), using the Bayer Mosaic as the sampling pattern[2]. 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 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, $l$.

In [13], Brainard shows how to find the linear interpolation coefficients using Bayesian methods. Researchers have used learning methods to find the coefficients from both test patterns [22] and real images [10, 16]. In [16], the system also performs non-linear interpolation by expanding $l$ to include its squared terms.

In terms of the model presented in Section 2, each of these algorithms assumes that a single linear regressor can model every pixel in the image. However, the 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 a cyan edge is applied to a white edge, then the results will be incorrect. On the other hand, if the interpolator best matched to a white edge is used then the results will be excellent. Intuitively, allowing the correlation between color bands to vary from pixel to pixel will greatly improve performance. Using our model effectively allows the system to choose from a set of interpolators at each point. Each interpolator, or regressor, is trained for a different correlation between color channels.

## 5.1. Model

As with the super-resolution system described in Section 4, our system for demosaicing is defined by two compatibility functions. The first type, $\phi'(\cdot)$, expresses the compatibility between a neighborhood of observed color samples, $N_L(y_i)$, and a candidate value of $\hat{M}_i$, the index of the regressor produces the best estimate of the two unobserved color samples at pixel $i$. In our system, this is modelled as a multivariate Gaussian:

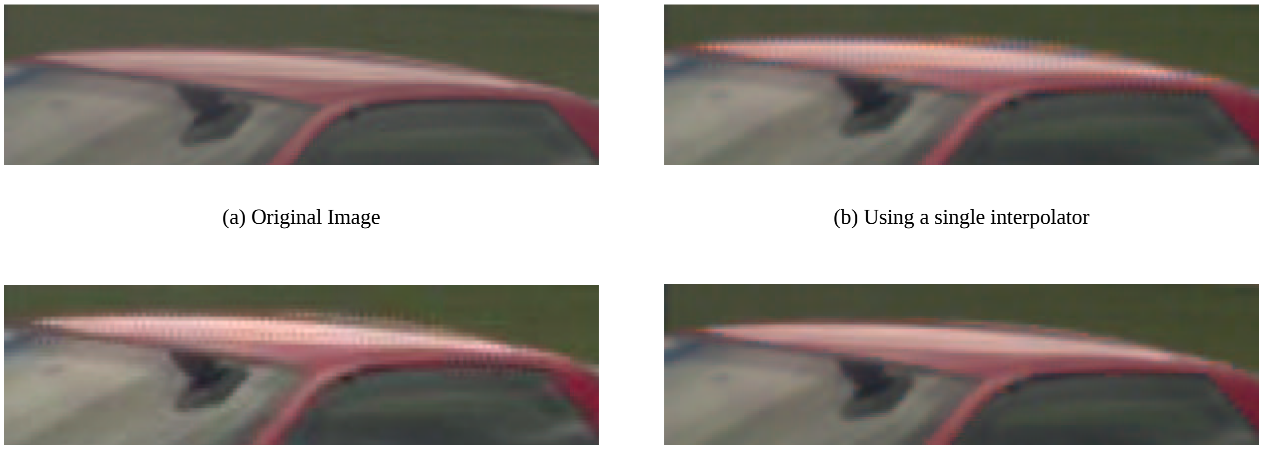
$$\phi'(N_L(y_i), \widehat{M_i}) = \mathcal{N}(N_L(y_i); \mu_{\widehat{M}_i}, \Sigma_{\widehat{M}_i}) \tag{9}$$

The second compatibility function, $\psi'(\cdot)$, relates neighboring pixels. If $x_i$ and $x_j$ are the unobserved color samples for two neighboring pixels, let $p_i$ and $p_j$ be RGB triplets created from combining $x_i$ and $x_j$ with the observed samples $y_i$ and $y_j$. Note that $y_i$ and $y_j$ are observations of different color bands. The compatibility between $x_i$ and $x_j$ is defined as

$$\psi(x_i, x_j) = \exp\left(\frac{-|p_i^r - p_j^r|^\alpha - |p_i^g - p_j^g|^\alpha - |p_i^b - p_j^b|^\alpha}{s}\right) \tag{10}$$

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

With $\phi(\cdot)$ and $\psi(\cdot)$ defined, the full-color image can be recovered using the steps described in Section 3 and Section 5.



(a) Original Image

(b) Using a single interpolator

(c) Median Filter Method [5]

(d) Using multiple regressors

Figure 7: A comparison of our results against using a single interpolator and the median filter method [5]. Using a mixture representation increases the quality of the reconstruction, especially along the roof of the car. Figure 7(a) came from [12].

## 5.2. 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 [16] 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 regressors 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. Our model used a set of twenty regressors. In this pattern there are actually four different types of local neighborhoods. Therefore, we learn four different sets of twenty regressors. The choice of which set of regressors 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 [16] showed that using a single interpolator produced a significant improvement over simply interpolating each color band separately.

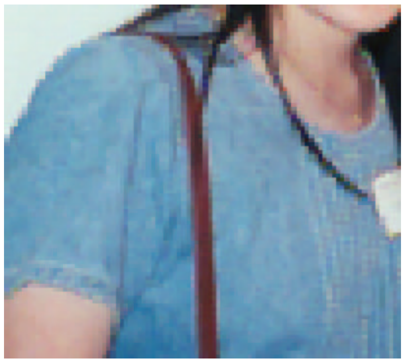
However, the mean squared error does not capture the important perceptual differences. The important difference in performance lies along the edges in the image, where color fringing can occur. Examining the images qualitatively shows a great improvement by using the regressor-based representation, especially along edges in the image. The most noticeable artifacts of demosaicing algorithms are colored fringes along edges. Figure 6 shows the difference in fringing caused by using one interpolator versus using a set of regression functions. Using one interpolator causes the fringes along the suit coat shown in Figure 6(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 regressors and belief propagation, our algorithm significantly reduces the amount of color fringing in Figure 6(b).

In Figure 7 we compare the results of our algorithm, in Figure 7(d), to a second method that utilizes the median filter [5]. The median filter algorithm has been found to perform well experimentally in [13, 18], so we use it as a representative of competing approaches. Again the output using multiple regression functions has the least amount of artifacts.

The importance of setting the exponent $\alpha$ to be less than one, is illustrated in Figure 8. Setting $\alpha$ greater than 1 leads to multiple small derivatives being favored over a single large derivative. This leads to the artifacts in Figure 8(b). When $\alpha$ is less than one, sharp edges are preferred, resulting in Figure 8(a).

## 6. Conclusion

In this paper, we have designed graphical models for two image processing tasks: super-resolution and image demosaicing. For both applications, the flexibility of the graphical model allowed us to incorporate the statistics of natural


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

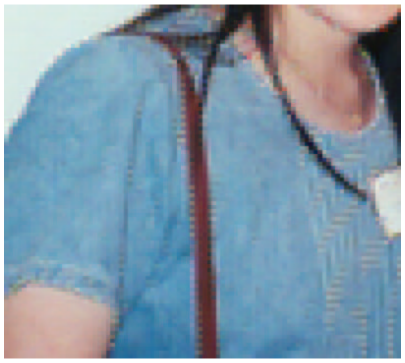

(b) Sample results for $\alpha = 2.0$

Figure 8: The effect of the exponent $\alpha$ on the results. The results are sharper when $\alpha = 0.7$ because the statistical prior favors fewer, large derivatives.

images to define compatibility functions between the high-resolution pixels being estimated.

Performing inference in the graphs is tractable because of an efficient discretization introduced in this paper. The state of each variable corresponds to a single linear regressor that estimates the high-resolution pixel values from the low-resolution observations. 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 allows effective use of graphical models in image processing problems.

## References

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

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

[3] 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.

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, November 2001.

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

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

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

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

[9] 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.

[10] 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.

[11] 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.

[12] 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*.

[13] 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.

[14] 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.

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

[16] 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.

[17] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.