Exploiting the Sparse Derivative Prior for Super-Resolution

by

Bryan Christopher Russell

A.B. Computer Science
Dartmouth College, 2001

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2003

Author . . . . . .
Department of Electrical Engineering and Computer Science
August 8, 2003

Certified by . . . . . .
William T. Freeman
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Exploiting the Sparse Derivative Prior for Super-Resolution

by

Bryan Christopher Russell

Submitted to the Department of Electrical Engineering and Computer Science
on August 18, 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

The distribution of bandpass filters with localized, oriented, and bandpass characteristics applied to natural images are sharply peaked with high kurtosis. These sparse, heavy-tailed distributions have been observed consistently across many classes of natural images, motivating its use as a prior on images. We describe a framework for incorporating this prior into graphical models to infer latent information. Specifically, we use factor graphs and exploit derivative filters to estimate a high resolution image from a single low resolution image. The resulting high resolution images have good image quality with sharp edges and lower reconstruction error than the state-of-the-art techniques to which it was compared. In addition, we describe a novel technique for finding candidate values efficiently in the estimated image, avoiding computational intractability.

Thesis Supervisor: William T. Freeman
Title: Associate Professor of Electrical Engineering and Computer Science

Acknowledgments

I would like to thank my advisor, William Freeman, for his guidance, wisdom, and patience during my first two years at MIT and for his help in editing this thesis.

Many thanks to the members of Bill's group for the many fruitful discussions leading up to this thesis. In particular, I would like to thank my officemates Barun Singh and Marshall Tappen for always answering my random questions relating to computer vision, Linux, etc. with enthusiasm. Also, I would like to thank Kilian Pohl for his friendship and support, as well as introducing me to many interesting German games.

This work was funded by the Nippon Telegraph and Telephone Corporation as part of the NTT/MIT Collaboration Agreement. Also, I would like to gratefully acknowledge the MIT Leventhal fellowship for funding me during my first year at MIT.

Thanks to the many friendships at UniLu, especially to Graham Kelder and the YAGS group. Also, thanks to the members of 302 mid-Mass for their support and the many enjoyable moments.

Thanks are in order for my family, especially to my parents, Frederick and Eva, and to my siblings Lisa and Kevin. Thanks for the many sacrifices that you have made to make all of this possible.

Lastly, words cannot describe my appreciation for the patience, love, and support that I have received from Kathryn Hamm over the past two years.

Contents

List of Figures

Chapter 1

Introduction

Consider enhancing the resolution of natural images, such as the 64×64 low-resolution image shown in Figure 1-1(a). It is desirable to have a solution that not only resembles an image taken by a camera with higher resolution, but also has the perceived quality of a high resolution image. For Figure 1-1(a), we wish to generate an image similar to the 128×128 image in Figure 1-1(b), having sharp edges and minimal perceived artifacts. In this thesis, we examine the task of increasing the resolution of images, which is known as the super resolution problem. In particular, we consider the class of images from natural scenes and try to exploit well-observed statistical regularities of this class to increase edge sharpness and reduce perceived artifacts in the higher resolution.

(a) (b)

Figure 1-1: For the super resolution problem, we are given a low resolution image (a) and would like to generate an image with higher resolution (b). In this example, (a) has a resolution of 64×64 and (b) has a resolution of 128×128.

A good solution would be beneficial to the computer vision and computer graphics communities. Moreover, many commercial applications that use low-resolution images, due to low-quality cameras or data compression, could benefit. This includes, but is not limited to, low-resolution video conversion for display on high definition television, manipulation of images from low-end digital cameras, and displaying images or video streams from mobile phone cameras.

1.1 Problem Description

A system that performs 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]. Here, 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 the system diagram shown in Figure 1-2. We wish to interpolate in a manner that provides a visually plausible approximation to the inverse of the system. Of course, this is an ill-posed problem since there are a large number of high resolution solutions [3, 6, 14, 36, 41].

Two simple approximate solutions are pixel replication[2] and bicubic interpolation. Pixel replication constructs the high resolution image by duplicating each low resolution pixel Z times in each spatial dimension, where Z is the zoom factor[3], and maintaining the same spatial relation of the low resolution duplicates in the high resolution image as in the low resolution image. Figure 1-3(a) shows a one-dimensional signal and figure 1-3(b) shows graphically the super resolution problem for this signal. Figure 1-3(c) shows the pixel replication super resolution solution to the one-dimensional signal. Bicubic interpolation approximates the missing pixels in high resolution image by fitting cubic functions to the observed low resolution pixels.

[1]Some authors refer to extracting a single high resolution image from multiple low resolution frames as super resolution [3]. Here, we deal with only single frames, which is sometimes called "image interpolation".

[2]Pixel replication is also known as zero-order hold [18].

[3]For simplicity, we assume here integer zoom factors.

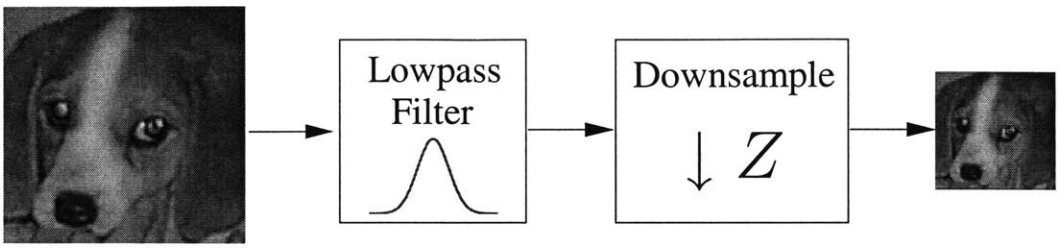


Figure 1-2: A system diagram of a low resolution image L generated from a high resolution image H. We assume that L is generated by first convolving a lowpass filter with H, to reduce aliasing, and then downsampling by a factor of Z to the desired resolution.

More information on cubic interpolation techniques will be discussed in Section 2.1.

In Figure 1-4, we show the output of pixel replication and bicubic interpolation on the low resolution image from Figure 1-1(a). Pixel replication produces jagged edges and "blocky" artifacts. On the other hand, bicubic interpolation produces a blurry image with overly smooth edges. With these two extremes in mind, we wish to produce a more natural looking image with sharp edges and minimal distracting visual artifacts.

To achieve these goals, we will exploit the statistics of the derivatives of natural images. These statistics have been repeatedly observed across large classes of natural images. Moreover, we will show how to incorporate these statistics into a powerful and modularized probabilistic framework to infer the high resolution image. The combination of the derivative statistics and the probabilistic framework allow us to infer a solution that shares a common trait with natural images.

1.2 Overview of System and Thesis Outline

Chapter 2 reviews the previous work relating to super resolution. In particular, Section 2.3 reviews learning-based approaches for super resolution. Here, we introduce a powerful probabilistic representation, which is amenable to inference, known as graphical models. Graphical models form the basis of the work presented in this thesis.

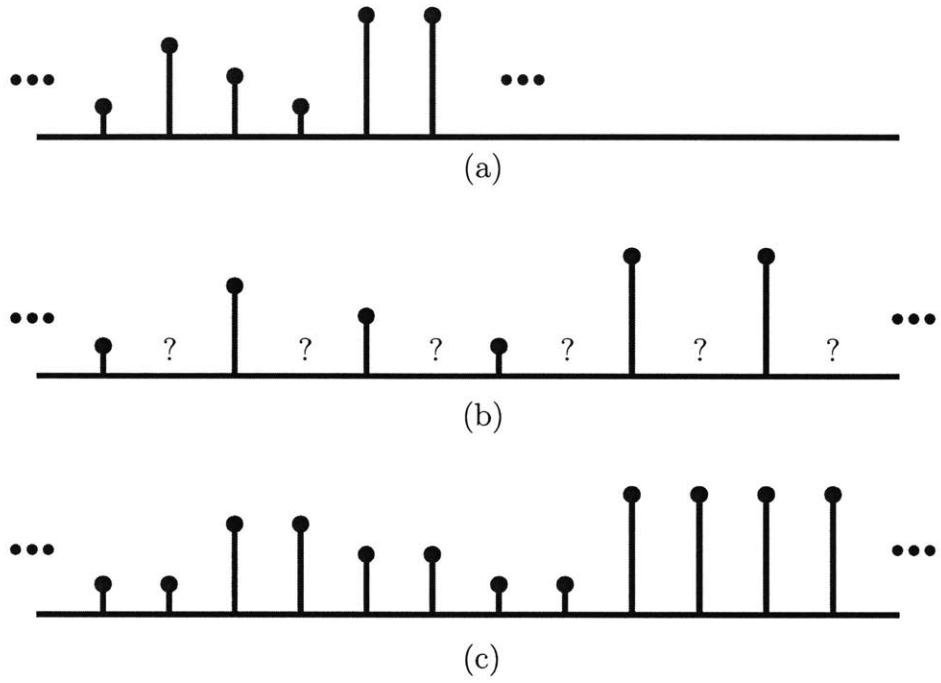


Figure 1-3: Consider a small portion of a one dimensional signal (a). We wish to increase the resolution of the signal by some factor Z. We graphically show this in (b) where $Z = 2$. One simple solution is pixel replication (c), where we duplicate each sample. Notice that the output of pixel replication produces jagged edges.

Figure 1-4: A comparison of two simple techniques for super resolution: (a) the original high resolution image; (b) pixel replication; (c) bicubic interpolation. Notice that pixel replication produces "blocky" artifacts while bicubic interpolation produces an overly smooth result.

Chapter 3 gives an overview of natural image statistics and related work in the field that exploits these statistics.

In Chapter 4, we formulate the posterior distribution for super resolution. Moreover, natural image statistics are used as the prior in the distribution. We show how to find the maximum a posteriori (MAP) solution by using approximate inference methods in the graphical models. We present and explain in-depth the solution for super resolution to a one-dimensional signal, with an extension to the two-dimensional case given in the Appendix.

If we are not careful to control the state dimensionality, the above solution using graphical models can be intractable. Chapter 5 introduces a novel technique for obtaining a tractable number of candidate states for the unknown, latent variables in the posterior distribution.

In Chapter 6, we show the super resolution images that our algorithm produces. Moreover, we compare our algorithm to existing super resolution algorithms.

Chapter 7 reviews the contributions of this paper and outlines areas of possible future research.

Chapter 2

Prior Work

In this chapter, we survey previous work on super resolution. Specifically, we survey functional interpolation, deconvolution, and learning-based methods.

2.1 Functional Interpolation Methods

Suppose we observe the discrete sampling of an unknown function $f(x)$, illustrated in Figure 2-1. We wish to find a smooth approximation $g(x)$ to $f(x)$ using local piecewise-smooth functions:

$$g(x) = \sum_k c_k u_k(x) \tag{2.1}$$

where $u_k(\cdot)$ are the local functions and c_k are parameters dependent on the sampled data. Simple interpolation methods define these local functions under various constraints. This allows us to use Equation 2.1 to estimate the intermediate values of the discrete samples.

Figure 2-2 shows four common one-dimensional kernels: nearest neighbor, linear, cubic[1] [22], and cubic B-spline [15]. Each of these kernels are simple to compute and require a small region of support. However, results using these kernels are blurry or result in heavy aliasing. Other kernels have been proposed to overcome these

[1]In two dimensions, linear and cubic are called bilinear and bicubic respectively.

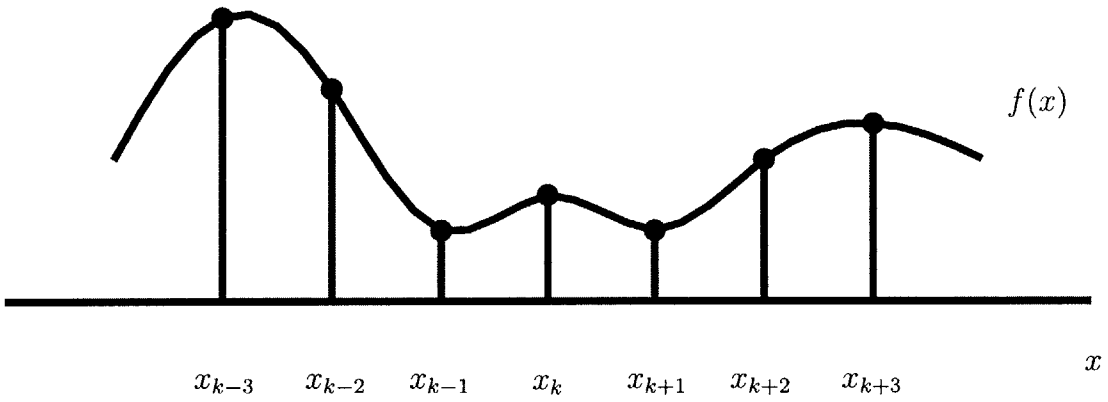


Figure 2-1: A discrete sampling of an unknown function $f(x)$. Functional interpolation methods approximate $f(x)$ by a set of local functions.

limitations, such as the sharpened Gaussian kernel [35].

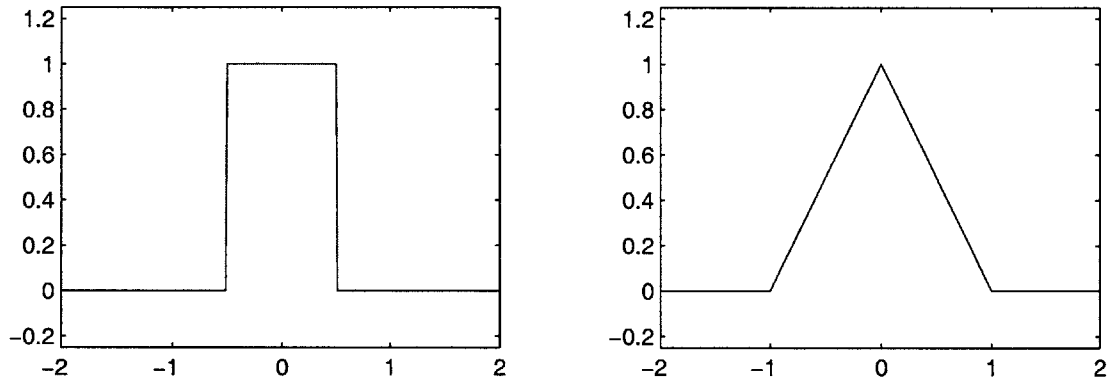
Several methods attempt to sharpen the images produced by functional interpolation methods. Greenspan et al. use a nonlinear enhancement algorithm to sharpen the image [13]. This algorithm learns a scaling factor and clipping constant to scale, clip, and then bandpass the original image. The result looks sharp, but suffers from ringing artifacts. Another method uses a differential equation to smooth jagged isophotes from functionally interpolated images [28]. The results are pleasing, resulting in smooth contours. However the sharpness depends heavily on the initial interpolation. Another approach classifies local pixel neighborhoods into constant, oriented, and irregular categories and then enhances the oriented neighborhoods [43].

2.2 Deconvolution Methods

To improve upon the blurry results yielded by functional interpolation, researchers have proposed methods to deconvolve the blurring filter. The problem is posed as a linear observation with Gaussian noise:

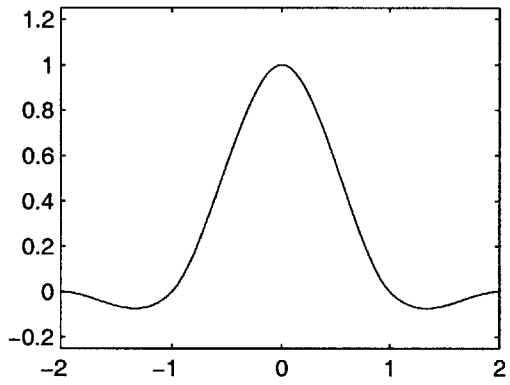
$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{2.2}$$

where $\mathbf{y}$ is the observed, blurry image, $\mathbf{x}$ is the desired unblurred image, $\mathbf{H}$ contains the spatially-invariant blurring convolutions, and $\mathbf{n}$ is i.i.d. Gaussian noise. Note that

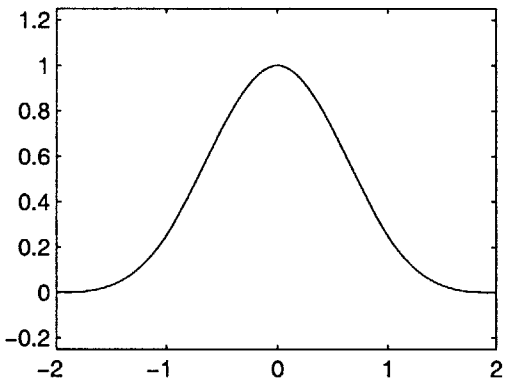


(a) Nearest neighbor

(b) Linear



(c) Cubic



(d) Cubic B-spline

Figure 2-2: Simple kernels used for functional interpolation.

$\mathbf{x}$ and $\mathbf{y}$ are lexicographically ordered vectors of the images.

Solutions using the wavelet transform have been developed. Here, we assume that $\mathbf{x} = \mathbf{W}^{\mathrm{T}}\theta$ where $\mathbf{W}$ is an orthogonal matrix representing the wavelet transform and θ are the wavelet coefficients. The goal is to infer θ, often using the sparse prior on wavelet coefficients, to be discussed in Chapter 3. However, in general $\mathbf{H}$ is not orthogonal, which raises complexities. Work has been done to overcome these difficulties, but require significant numerical calculations [4, 26, 46]. More efficient results have been obtained through the expectation maximization (EM) algorithm [6, 9]. Also, an algorithm combining Fourier and wavelet methods has been propsed [30]. While these methods produce nice results, in general it is difficult to directly represent and solve the system assuming various constraints on the solution space.

2.3 Learning-based Methods

Learning-based methods attempt to incoporate a priori information to enhance the solution space. These methods are posed using well-developed frameworks with efficient inference algorithms.

An important subset of the learning-based methods for super resolution uses graphical models, which is a powerful visual framework with efficient inference algorithms. Graphical models have had an immense impact on the computer vision community [12, 42]. Here, we consider graphical models and describe the algorithms to perform inference.

2.3.1 Introduction to Graphical Models

Let x_i denote a random variable on the sample space Ω_i. Here, we assume that the sample space is discrete and finite, although the ideas presented in this section can be generalized to continuous spaces. For a set of random variables $x_1 \ldots x_N$, let the joint probability be given by $\Pr(x_1, \ldots, x_N)$. Let us consider computing the marginal probability of x_1:

$$\Pr(x_1) = \sum_{x_2,\ldots,x_N} \Pr(x_1,\ldots,x_N). \tag{2.3}$$

Notice that if each random variable has S states, then the marginal probability computation has time-complexity of $\mathcal{O}(S^N)$. However, if the joint probability can be factorized into many functions each using fewer arguments, then we can reduce this time-complexity. For example, suppose the following factorization of the joint probability on the random variables $x_1,\ldots,x_5$:

$$\Pr(x_1,x_2,x_3,x_4,x_5) = \frac{1}{C}\psi_1(x_1,x_3)\psi_2(x_2,x_3,x_4)\psi_3(x_4,x_5) \tag{2.4}$$

where C is a normalization constant. Notice that the factors $\psi(\cdot)$ may not correspond to an actual probability distribution. We can compute the marginal probability for x_1 as follows:

$$\begin{aligned}
\Pr(x_1) &= \sum_{x_2,x_3,x_4,x_5} \Pr(x_1,x_2,x_3,x_4,x_5) && (2.5)\\
&= \frac{1}{C}\sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_3)\psi_2(x_2,x_3,x_4)\psi_3(x_4,x_5) && (2.6)\\
&= \frac{1}{C}\sum_{x_2,x_3} \psi_1(x_1,x_3)\sum_{x_4}\psi_2(x_2,x_3,x_4)\sum_{x_5}\psi_3(x_4,x_5). && (2.7)
\end{aligned}$$

Here, the time-complexity is $\mathcal{O}(S^2)$ instead of $\mathcal{O}(S^5)$. Also, notice that these factorizations express conditional independence relations. For example, in Equation 2.4, if we are given the value of x_3, then x_1 is independent from all of the other random variables.

A graphical model explicitly represents a class of joint probability distributions, with each distribution sharing the same conditional independence relations. Graphical models mainly come in three varieties—Bayesian networks [19, 32], Markov random fields [12, 32], and factor graphs [23]—each having different representational capabilities. Here, we will consider the factor graph representation, examples of which appear in Figure 2-3.

A factor graph is a bipartite graph with variable and constraint nodes, indicated by the transparent circle and the solid square nodes respectively. Each variable node has a one-to-one correspondence to a single random variable. The constraint nodes correspond to the set of functions, known herein as constraint functions, in the factorization. Variable nodes connected by edges to a given constraint node comprise the arguments to the corresponding constraint function. Figure 2-3(a) shows the factor graph for the factorized joint distribution in Equation 2.4. In general, for a given factor graph the joint probability distribution is proportional to the product of the constraint functions:

$$\Pr(x_1, x_2, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{M} \psi_i(\cdot). \tag{2.8}$$

where M are the number of constraint nodes and $(\cdot)$ indicates the appropriate arguments to the function.

To see how the conditional independence relations are expressed in factor graphs, let us consider the example above where we observe the value of x_3, illustrated in Figure 2-4 where the observed node is shaded. Two sets of random variables are independent if the only existing paths connecting two nodes in the different sets crosses the observed nodes. For example, in Figure 2-4 there are no paths connecting x_1 and the other nodes without having the path contain x_3.

Suppose we wish to compute the maximum a posteriori (MAP) estimate of a given distribution. Let X_O and X_H be respectively the set of observed and hidden random variables. Using the rules for conditional independence in factor graphs above, observe that the conditional probability depends only on constraint nodes with at least one neighboring hidden variable node. In other words,

$$\Pr(\{X_H\} \mid \{X_O\}) = \frac{1}{C} \prod_{c_i \in N(X_H)} \psi_i(\cdot) \tag{2.9}$$

where $N(X_H)$ is the set of all neighboring constraint nodes of X_H. With this, we compute the MAP estimate $\hat{X}_H$ as follows:

$$\hat{X}_H = \underset{\{X_H\}}{\operatorname{argmax}} \Pr(\{X_H\} \mid \{X_O\}). \tag{2.10}$$

To compute the MAP estimate, an efficient message-passing protocol, known as max-product belief propagation (BP), has been developed to compute the exact maximum posterior probability at each variable node in factor graphs without loops [23, 32]. Messages are passed along every edge in the factor graph, and after a finite number of message-passing iterations the algorithm converges to the correct maximum posterior probability.

For factor graphs, there are two types of messages—those propagating from variable to constraint nodes and vice versa. Both of these message types are illustrated in Figure 2-5. If there are S states for each random variable, then each message will be a vector of length S.

For messages propagating from variable to constraint nodes at iteration t, the update rule is:

$$\mu_{ij}^{(t)}(x_i) \leftarrow \prod_{c_k \in N(x_i) \backslash c_j} \mu_{ki}^{(t)}(x_i) \tag{2.11}$$

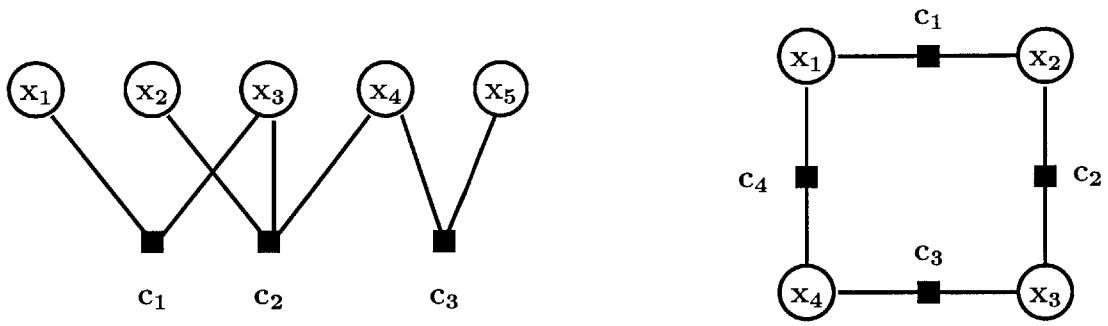
where $N(x_i) \backslash c_j$ are all of the neighbors of x_i except c_j, and $\mu_{ij}(x_i)$ is the message that node x_i sends to c_j for a particular setting of x_i. For messages sent from constraint to variable nodes at iteration t:

$$\mu_{ij}^{(t+1)}(x_j) \leftarrow \max_{\{N(c_i) \backslash x_j\}} \psi_i(\{N(c_j)\}) \prod_{x_k \in N(c_i) \backslash x_j} \mu_{ki}^{(t)}(x_k). \tag{2.12}$$

For each iteration, we first compute all of the variable to constraint node messages and then use the results to compute the constraint to variable node messages. The messages are normally initialized to one, but the algorithm converges for any nonzero initialization.

After convergence, we compute the belief for each node as follows:

$$b_i(x_i) \leftarrow \prod_{c_k \in N(x_i)} \mu_{ki}(x_i). \tag{2.13}$$



(a) (b)

Figure 2-3: Two simple examples of factor graphs without loops (a) and with loops (b). The transparent circle and solid square nodes correspond to variable and constraint nodes respectively. Each variable node x_i represents a random variable while each constraint node c_i represents the constraint function $\psi_i(\cdot)$. In this case, the factor graph in (a) represents the factorization in Equation 2.4.

It can be shown that the belief evaluated at a setting of x_i is proportional to the maximum of the posterior probability:

$$b_i(x_i) \propto \max_{\{X_H \setminus x_i\}} \Pr(\{X_H\} \mid \{X_O\}). \tag{2.14}$$

From Equation 2.10, we get the MAP estimate for x_i:

$$\hat{x_i} = \operatorname*{argmax}_{x_i} b_i(x_i). \tag{2.15}$$

While BP has been shown to converge to an exact solution after a finite number of iterations for graphs without loops, this convergence property does not hold true in general if BP is applied to loopy graphs, an example of which appears in Figure 2-3(b). In fact, exact inference for graphs with loops has been shown to be NP-hard [37]. However, for many loopy graphs we can converge to an approximate solution after a finite number of iterations by applying BP to it anyways. Much work has been done to study the convergence properties of loopy-BP [29, 47, 45].

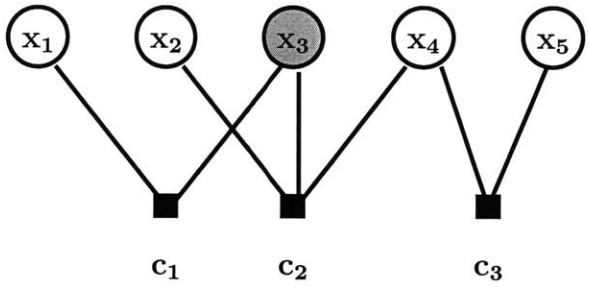
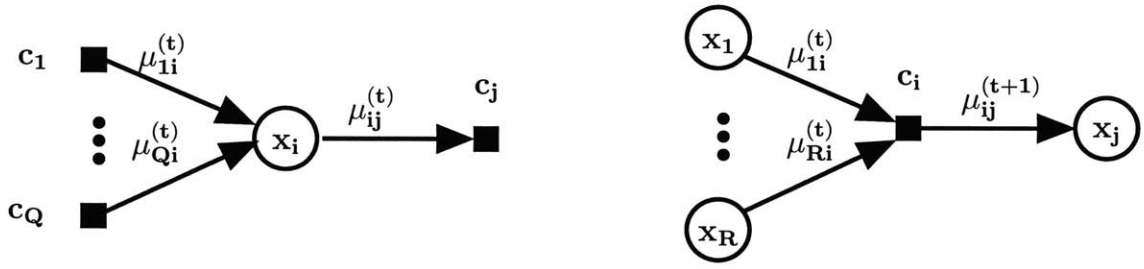


Figure 2-4: The same factor graph as in Figure 2-3(a), except that we observe the random variable x_3, illustrated by the shaded node. Two sets of nodes A and B are conditionally independent given a third set C if there exist only paths connecting $a \in A$ and $b \in B$ passing through some node in C. Here, x_1 is conditionally independent of x_2, x_4, and x_5 given x_3.



(a) (b)

Figure 2-5: When belief propagation (BP) is applied to factor graphs, there are two types messages: variable to constraint node (a) and vice versa (b). Here, we have indicated incoming and outgoing messages by arrows and have labelled the messages. See Equations 2.11 and 2.12 for the message update equations.

2.3.2 Algorithms Using Graphical Models

Schultz et al. formulated the super resolution problem using graphical models, assuming the Huber function [17] as a prior on the derivatives [36]. Similar approaches, but specialized only to images with faces, learn a prior on faces and achieve high quality results [2, 25]. An unsupervised approach using a dynamic-tree graphical model has been proposed [40].

An example-based method learns a priori the relationship between low and high frequencies by storing in a database corresponding patch pairs from a set of training images [10, 11]. Instead of representing each pixel as a random variable, patches are represented for more efficient computation. The algorithm then searches the database for appropriate patches at each spatial position based on the observed low resolution image and constraint to enforce spatial consistency among the patches. Work producing nice results along similar lines uses primal sketch priors [41].

Chapter 3

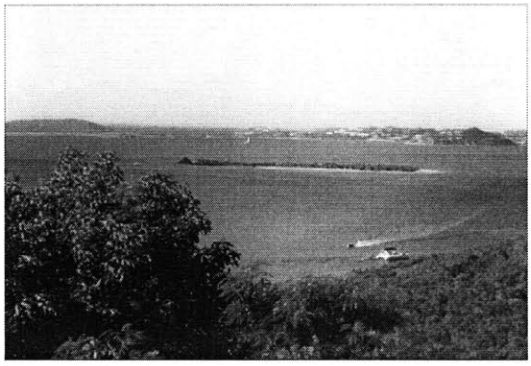
Natural Image Statistics

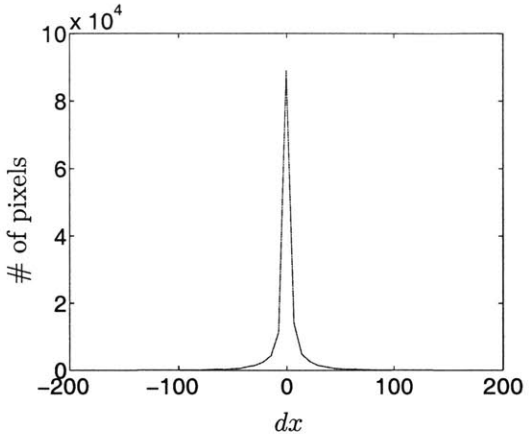
In the previous chapters, we introduced the super resolution problem and surveyed prior work. In this chapter, we introduce statistics that are repeatedly observed across many different classes of natural images. In addition, we survey how these statistics have been applied to various tasks in computer vision, computer graphics, and image processing. In subsequent chapters we will show how to exploit these statistics as a prior on latent variables, which will lead us to a method for super resolution.

3.1 Overview

To understand how the human vision system has evolved, many researchers have looked at the properties of natural scenes, whose images are projected onto the retina. Drawing from the receptive fields of the striate cortex [16], characterized as being localized, oriented, and bandpass, the statistics of filters[1] with these properties applied to natural images have been studied. It is well-documented that the histogram of these filters applied to large classes of natural images, similar to the one in Figure 3-1(a), are exponential with high fourth-order statistics (kurtosis) [8, 21, 27, 31, 34, 38]. These sparse, "heavy-tailed" distributions resemble the histogram shown in Figure 3-1(b) which, compared with Gaussian distributions of the same variance, have lower entropy.

[1]Localized, oriented, bandpass filters are also known as wavelets.





(a) (b)

Figure 3-1: (a) An example of a natural image. Filters with localized, oriented, and bandpass characteristics have high fourth-order statistics (kurtosis). We see these characteristics in the histogram of the horizontal derivatives (b) applied to the natural image in (a).

The distribution in Figure 3-1(b) can be described by the generalized Laplacian distribution:

$$p(x) \propto \exp\left(-\frac{1}{2}\left(\frac{|x|}{\sigma}\right)^{\alpha}\right) \tag{3.1}$$

where σ is the standard deviation and α is the exponent parameter. For natural images, empirically $0 < \alpha < 1$. Notice that if $\alpha = 2$, then Equation 3.1 becomes a Gaussian distribution. In Section 4.1.1, we consider the importance of the α parameter and its effect on the solution space.

Often, Equation 3.1 when $0 < \alpha < 1$ is known as the *natural image prior* because of its omnipotence across large classes of natural images. To gain an intuition of the natural image prior, let us consider the case of the derivative filter. The distribution says that large derivatives are sparse in a given natural image. Loosely speaking, this means an image consists 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 edge.

3.2 Applications Using Natural Image Statistics

The strength and repeatability of the statistics of natural images make it invaluable as a statistical prior. Simoncelli successfully used this prior as the basis of a Bayesian noise-removal algorithm [39]. Simoncelli and Buccigrossi used the regular statistical structure of images for compression [5]. Levin et al. showed how this prior can enable an image to be decomposed into transparent layers [24]. Portilla and Simoncelli used joint natural image statistics to synthesize new textures [33]. As mentioned in Chapter 2, several authors developed deconvolution methods exploiting the sparse statistics of wavelet coefficients [6, 9]. Farid and Lyu used this prior to help detect hidden messages in high resolution images [7].

Chapter 4

Applying the Natural Image Prior

In Chapter 3, we introduced the statistics of derivatives for natural images and concluded that they should be a useful prior. In this chapter, we incorporate this prior into the graphical model framework, introduced in Section 2.3, to perform super resolution. We will first look at a generic incorporation of this prior, which is applicable to a wide range of image processing applications. Then, we will apply the framework to the super resolution problem.

4.1 Combining Graphical Models with the Natural Image Prior

In Section 2.3, we introduced graphical models and, in particular, the factor graph representation. Recall that a factor graph is a bipartite graph consisting of random variable and system constraint nodes. The joint probability is given in Equation 2.8, which is a product of the constraints, represented as functions $\psi(\cdot)$ of the random variables.

Let the set of constraint functions be given as:

$$\Psi = \{\psi_i(\cdot) | 1 \leq i \leq M\} \tag{4.1}$$

where M is the number of constraints in the system. To incorporate the derivative

statistics of natural images, we need to ensure that a subset of Ψ constrains derivatives of neighboring pixels to have the Laplacian distribution, given in Equation 3.1. Specifically,

$$\Psi_N = \left\{ \psi_i(\hat{x}_i) \;\middle|\; \psi_i(\hat{x}_i) = \exp\left(-\frac{1}{2} \left(\frac{|D * \hat{x}_i|}{\sigma_N} \right)^{\alpha} \right) \right\} \subseteq \Psi \tag{4.2}$$

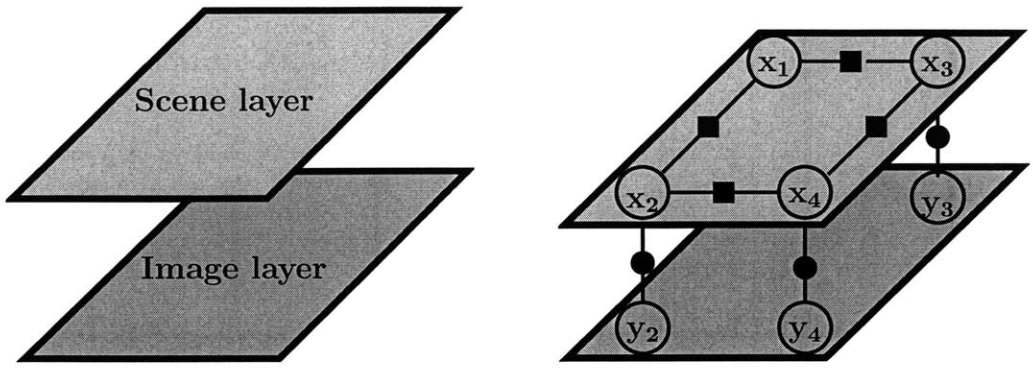
where $\hat{x}_i$ is a patch of spatially adjacent pixels, D is a derivative kernel, α is the exponent parameter, set between 0 and 1, and σ_N is the standard deviation parameter. This set of constraint functions needs to cover all of the random variables representing pixels in the latent image.

To illustrate the usefulness of the prior on image derivatives for the task of super resolution, let us look at a common strategy used in computer vision for inferring missing information, shown in Figure 4-1(a). Two layers are used, one representing observed "image" data and the other representing the hidden "scene" data [12, 42]. To infer the scene data, we place constraints between the image and scene data, as well as within the scene data itself. This is illustrated in a simple instantiation in Figure 4-1(b) where the top circle nodes represent pixels in the scene, the bottom circle nodes represent pixels in the observed image, the solid circles represent constraints between image and scene data, and the solid squares are the constraints within the scene data, given by the natural image prior.

4.1.1 Example Interpolation Problem

To illustrate the power of combining graphical models with the statistics of derivatives, let us consider the simple one-dimensional interpolation problem in Figure 4-2. Our goal is to interpolate y_1 from all of the other observed values in the step signal, where $y_i = 2$ for $i \leq 0$ and $y_i = 1$ for $i \geq 2$.

We wish to use the techniques from Section 4.1 to derive a factor graph with derivative statistics constraints. Let us assume a simple derivative kernel of $[-1, +1]$. We derive the factor graph in Figure 4-3 to find solutions to the interpolation problem, where the circles represent the samples of the signal and the solid squares represent



(a) (b)

Figure 4-1: (a) A common strategy used in computer vision to infer missing information is to represent the observed and hidden data as two different layers. The hidden data is known as the "scene" layer and the observed data is known as the "image" layer. (b) To infer the scene layer, we place constraints between the image and scene layers, as well as within the scene layer itself. Here, we demonstrate a simple factor graph instantiation.

the derivative statistics constraint:

$$\psi(y_i, y_{i+1}) = \exp\left(-\frac{1}{2}\left(\frac{|y_{i+1} - y_i|}{\sigma_N}\right)^{\alpha}\right) \tag{4.3}$$

for $-2 \leq i \leq 3$. Therefore, the joint probability distribution is:

$$\Pr(y_{-2}, y_{-1}, y_0, y_1, y_2, y_3, y_4) \propto \prod_{i=-2}^{3} \psi(y_i, y_{i+1}). \tag{4.4}$$

If we observe all of the random variables except y_1, then:

$$\begin{aligned}
\Pr(y_1|y_{-2}, y_{-1}, y_0, y_2, y_3, y_4) &= \Pr(y_1|y_0, y_2) && (4.5)\\
&\propto \psi(y_0, y_1)\psi(y_1, y_2) && (4.6)\\
&\propto \exp\left(-\frac{1}{2}\left(\left(\frac{|y_1 - y_0|}{\sigma_N}\right)^{\alpha} + \left(\frac{|y_2 - y_1|}{\sigma_N}\right)^{\alpha}\right)\right) && (4.7)
\end{aligned}$$

where we have taken into account from the factor graph that y_1 is independent of all

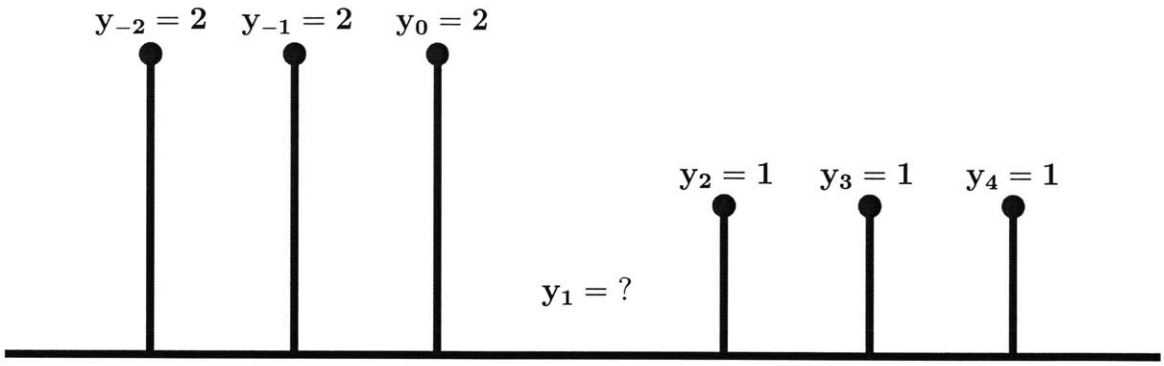


Figure 4-2: A simple interpolation problem where y_1 must be interpolated from y_0 and y_2. Realistically, y_1 will lie somewhere between 1 and 2. Using Equation 3.1, we will show how the setting of α influences the value of y_1.

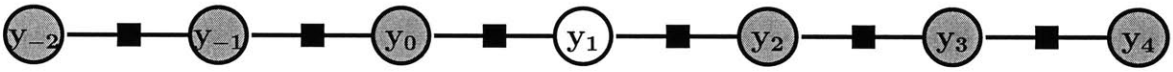


Figure 4-3: Factor graph for the interpolation problem in Figure 4-2. The circle nodes represent the random variables and the solid squares represent the derivative statistics constraint. Here, the shaded nodes indicate observed values.

the other nodes given y_0 and y_2[1].

Examining Equation 4.7, shown in Figure 4-4 for different settings of α, reveals that an extremum of $\Pr(y_1|y_0, y_2)$ will occur at $y_1 = 1.5$, regardless of the value of α. For $\alpha < 1$, the extremum will be a local minima, causing the most probable value of y_1 to be either 1 or 2. For $\alpha > 1$, $\Pr(y_1 = 1.5|y_0, y_2)$ will be the global 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.

[1]Note that in general spatially adjacent bandpass filters are actually not independent [5]. Here, we make a variational approximation [20] to the true probability distribution.

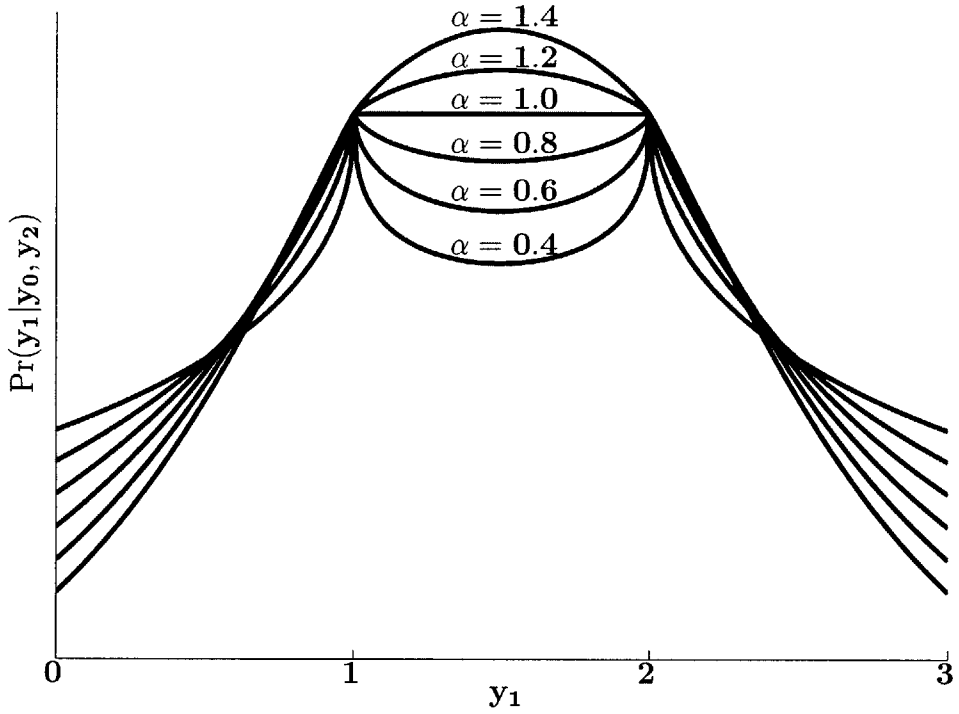


Figure 4-4: Each curve represents the probability of y_1 for the corresponding setting of α. For $\alpha > 1$, the most likely values of y_1 is 1.5, leading to a softer image edge. For $\alpha < 1$, y_1 is most likely 1 or 2, either of which gives a sharper edge transition.

4.2 Formulating the Super Resolution Probability Distribution Function

To do super resolution, we again consider the techniques from Section 4.1 to incorporate the natural image prior and local constraints in the graphical model framework. Here, we define the local constraint based on the system diagram in Figure 1-2. We want the error between the low resolution image and the decimated high resolution image, which we call the reconstruction error, to be minimized. For this, we assume a Gaussian distribution on the error. So, the set of local constraints on the reconstruction error is given by:

$$\Psi_R = \left\{ \psi_i(\hat{x}_i, y_i) \;\middle|\; \psi_i(\hat{x}_i, y_i) = \exp\left(-\frac{1}{2} \left(\frac{W * \hat{x}_i - y_i}{\sigma_R} \right)^2 \right) \right\} \tag{4.8}$$

where y_i is a low resolution pixel, $\hat{x}_i$ is a patch of spatially adjacent high resolution pixels that correspondingly subsamples to y_i, W is a lowpass filter, and σ_R is the

standard deviation parameter.

With the set of local and natural image prior constraints in hand, we get the overall posterior distribution:

$$\Pr\left(\{x\} \mid \{y\}\right) = \frac{1}{C} \prod_{\psi_i(\cdot)\in\Psi_N} \psi_i(\hat{x}_i) \prod_{\psi_i(\cdot)\in\Psi_R} \psi_i(\hat{x}_i, y_i) \tag{4.9}$$

where $\{x\}$ are the latent high resolution pixels, $\{y\}$ are the observed low resolution pixels, and C is a normalization constant. Ψ_N and Ψ_R spans over the entire low and high resolution images.

Let us consider the simple case of zooming by a factor of two. We start by converting Equation 4.9 into a factor graph (we describe the factor graph for the one-dimensional case in Section 4.3 and the two-dimensional case in the Appendix) and specifying the message propagation equations. To solve the factor graph, we use the max-product belief propagation (BP) algorithm [32] to compute the maximum a posteriori (MAP) estimate of the random variables. Considering all possible pixel intensities is intractable. To overcome this, we assume for each low resolution pixel a small set of candidate 2×2 high resolution patches. The sets of patches for each spatial loacation are the possible states of the latent variables, thereby making the problem tractable. We discuss the tractability issue and how to obtain candidate patches in Chapter 5.

4.3 One-Dimensional Super Resolution Example

To illustrate how to represent Equation 4.9 as a factor graph, let us consider the one-dimensional case. Figure 4-5(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 assume that each latent variable represents a patch of two pixels, as illustrated in Figure 4-5(e).

We notate the two pixels a and b of patch candidate j of latent node x_i in our model as x_{ij}^a and x_{ij}^b respectively, shown in Figure 4-5(f).

To derive the message-passing equations for the factor graph, we need to consider three cases: messages passed from latent node to constraint node, derivative constraint node to latent node, and reconstruction constraint node to latent node, illustrated in Figure 4-5(b)-(d). Let μ_i be an S-tuple representing messages passed between 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 messages at that setting of x_i except from the target constraint node. Using Figure 4-5(b), we write this explicitly as follows:

$$\mu_4^{(t)}(x_i) \leftarrow \mu_1^{(t)}(x_i)\mu_2^{(t)}(x_i)\mu_3^{(t)}(x_i). \tag{4.10}$$

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 4.1. Using Figure 4-5(c), the message is computed as follows:

$$\mu_{i+1}^{(t+1)}(x_{i+1}) \leftarrow \max_{x_i} \mu_i^{(t)}(x_i) \prod_{p\in\{a,b\}} \exp\left(-\frac{1}{2}\left(\frac{|x_{i+1}^p - x_i^p|}{\sigma_N}\right)^\alpha\right). \tag{4.11}$$

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 4.2. Using Figure 4-5(d), the message is computed as follows:

$$\mu_{i+1}^{(t+1)} \leftarrow \max_{x_i} \mu_i^{(t)}(x_i) \exp\left(-\frac{1}{2}\left(\frac{w^T x' - y_i}{\sigma_R}\right)^2\right) \tag{4.12}$$

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.

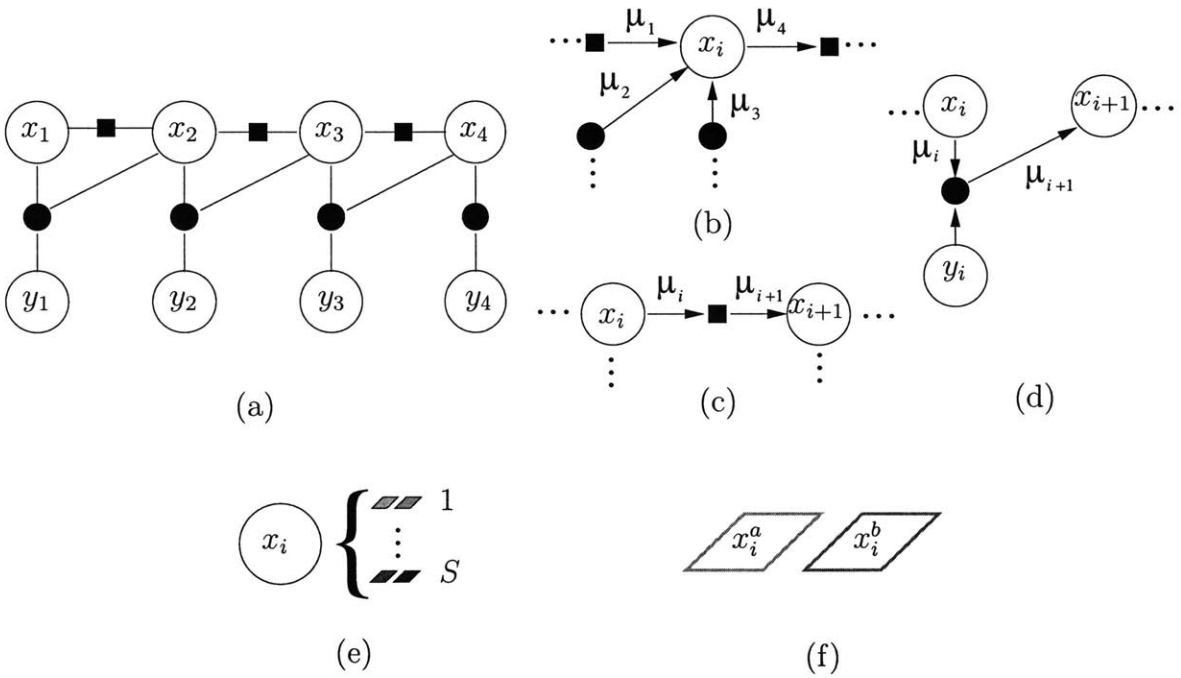


Figure 4-5: (a) Factor graph for 1D super resolution example. The random variables are represented by the transparent nodes where the x_i are the latent variables and the y_i are the observed variables, the solid squares represent the derivative constraint, and the solid circles represent the reconstruction 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 4.10, 4.11, and 4.12 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.

Chapter 5

Obtaining Candidates

In Chapter 4, we showed how to combine the natural image prior with the graphical model framework to do super resolution. While representing the posterior distribution in a graphical model and deriving the message propagation equations, we were extremely careful in the description of the latent variables. For the case of zooming by a factor of two, we said that each latent variable has S states and that each state corresponds to a 2×2 patch of pixels. In this chapter, we motivate the use of patches for the latent variables and show how to obtain a small set of S candidate patches for each latent variable.

5.1 Motivation

The graphical models that we use require a discrete representation of the latent variables. For the posterior distribution given in Equation 4.9, we could have easily defined each latent variable to correspond to a single pixel in the high resolution image. If we use the common discretization of pixel intensities, then each latent variable would have 256 states. With this, a problem in complexity arises.

Consider directly representing the reconstruction constraint in Equation 4.9 as a factor graph, where each latent variable corresponds to a single pixel in the high resolution image. For the two-dimensional case using a 3×3 Gaussian kernel W, this is illustrated in Figure 5-1(a) with message update equation:

$$\mu_9(x_9) \leftarrow \max_{x_1} \cdots \max_{x_8} \mu_1(x_1) \cdots \mu_9(x_9) \exp\left(-\frac{1}{2}\left(\frac{W * \hat{x} - y}{\sigma_R}\right)^2\right) \tag{5.1}$$

where:

$$\hat{x} = \begin{pmatrix} x_1 & x_4 & x_7 \\ x_2 & x_5 & x_8 \\ x_3 & x_6 & x_9 \end{pmatrix}. \tag{5.2}$$

The time-complexity of this update equation is $\mathcal{O}(S^9)$, where $S = 256$ in this case. The complexity gets much worse if we consider lowpass kernels requiring larger spatial support.

If we use 2×2 paches of pixels, then we get the factor graph shown in Figure 5-1(b) with message update equation:

$$\mu_4(x_4) \leftarrow \max_{x_1} \max_{x_2} \max_{x_3} \mu_1(x_1)\mu_2(x_2)\mu_3(x_3) \exp\left(-\frac{1}{2}\left(\frac{W * x' - y}{\sigma_R}\right)^2\right) \tag{5.3}$$

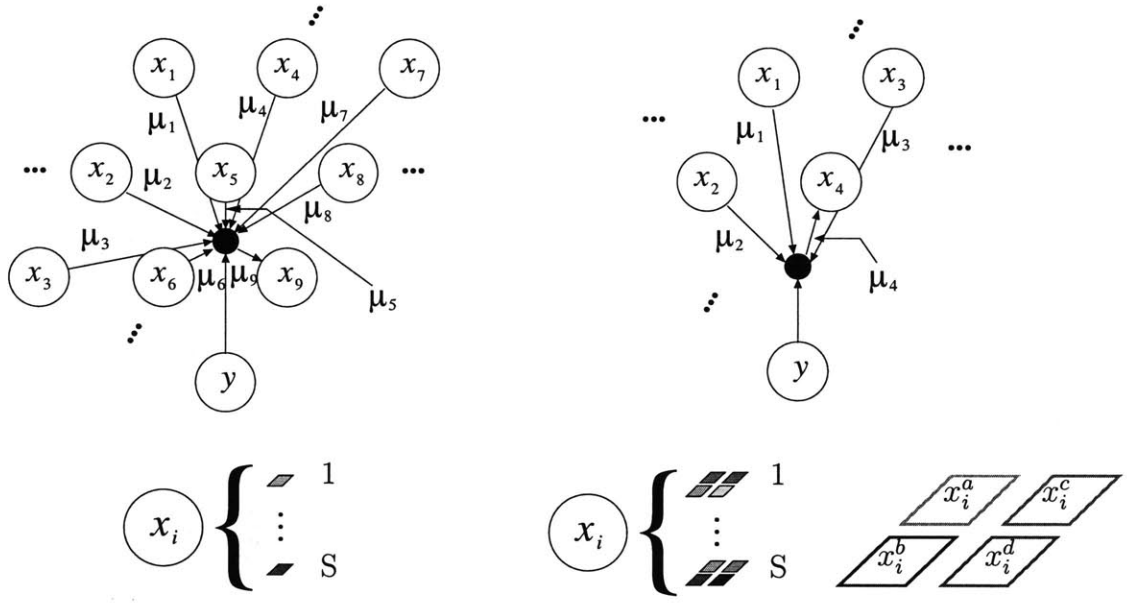
where:

$$x' = \begin{pmatrix} x_1^a & x_1^c & x_3^a \\ x_1^b & x_1^d & x_3^b \\ x_2^a & x_2^c & x_4^a \end{pmatrix}. \tag{5.4}$$

The time-complexity of this update equation is $\mathcal{O}(S^4)$. However, by grouping pixels into 2×2 patches we get 256^4 possible patches for each latent variable. It is desirable to somehow significantly reduce the number of states per latent variable while still obtaining a good solution.

5.2 Learning Interpolators

Freeman et al. reduce the number of states by choosing a small number of plausible patches for each latent variable [11]. Local image information is used to select a set of



(a) pixel representation

(b) patch representation

Figure 5-1: (a) Factor graph for the reconstruction constraint of Equation 4.9, assuming that each latent variable corresponds to a single pixel. If the common discretization of pixel intensities is used, then there are $S = 256$ states for each latent variable. The time-complexity for message passing in this graph is $\mathcal{O}(S^9)$ (see Equation 5.1). (b) Factor graph for the reconstruction constraint of Equation 4.9, assuming that each latent variable corresponds to a 2×2 patch of pixels. The time-complexity for message passing in this graph is $\mathcal{O}(S^4)$ (see Equation 5.3). However, there are now $S = 256^4$ states. This motivates us to find a small set of candidate patches for each latent variable.

candidate patches for each location in the image. A database storing the local image information and the corresponding hidden image data is built a priori from a set of training images. To obtain the set of candidate patches, the database is searched and the top S candidates matching the local image information is used.

Two issues arise in obtaining candidates using a database. First, we must store the database of local and hidden image information pairs. To have enough variability in the solution, the database must be large and somewhat representative of the solution space. Second, we must search this database in an efficient manner to draw the top S candidates.

Instead of drawing high resolution patch candidates from a large database, we propose generating them directly from local information in the low resolution image. A set of learned functions generate high resolution patch candidates from a small patch of low resolution pixels[1]. Thus, given the low resolution image information y around some point, the high resolution candidate $f(y)$ at that point is modeled as a linear function of y:

$$f(y) = \mathbf{T}y \tag{5.5}$$

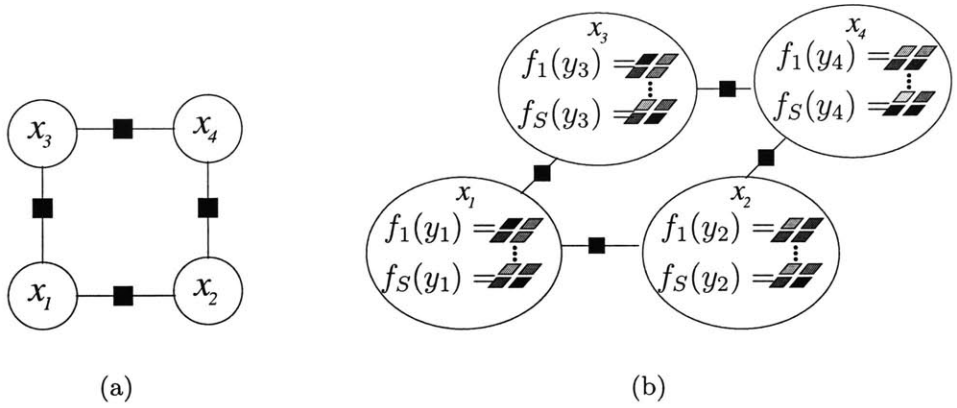
where $\mathbf{T}$ is a matrix relating the low-resolution image information to a candidate high-resolution patch[2]. To generate multiple candidates, we find a set of matrices, $\mathbf{T}_1 \ldots \mathbf{T}_S$, each of which interpolates the same observed information to a different candidate for the latent variable. This is illustrated in Figure 5-2(b).

Given a training set of low resolution patches $Y = \{y_1, \ldots, y_N\}$ and the corresponding high resolution patches $X = \{x_1, \ldots, x_N\}$, the interpolators are found via a simple EM algorithm:

1. Use k-means clustering to initially assign each training example pair (x_i, y_i) to one of S clusters.

[1]These local functions can be seen as mapping the low resolution image data to the high resolution "scene data" [44].

[2]Although not used to produce the results in this thesis, non-linear functions can be computed similarly by expanding y to include non-linear functions of the image data, such as polynomials.



(a) (b)

Figure 5-2: (a) An example of a simple factor graph. (b) The x_i nodes in the graph represent latent variables having groups of pixels, not single pixels, as states. In this graph, each candidate state of a node represents a patch of four pixels. These candidate patches are computed from local image data by a set of different linear regressions, $f_1(\cdot) \ldots f_S(\cdot)$.

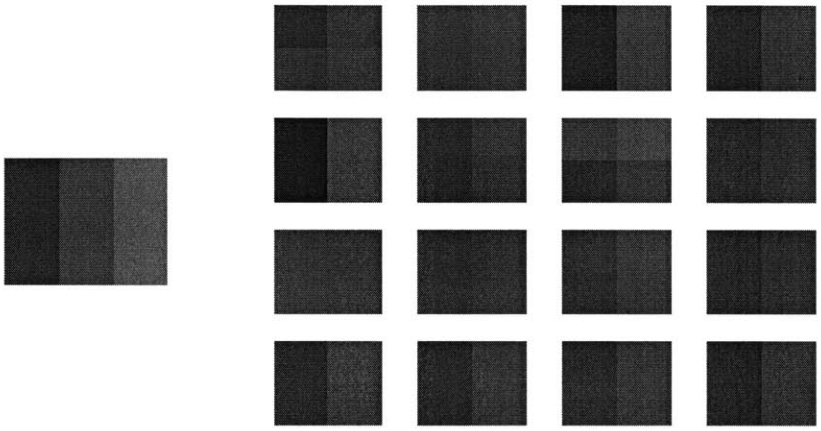
2. For each cluster $j = 1 \ldots S$, set $\mathbf{T}_j$ to be the least-squares solution to $\mathbf{X}_j = \mathbf{T}_j \mathbf{Y}_j$, where $\mathbf{X}_j$ and $\mathbf{Y}_j$ are each matrices stacked with all of the training examples assigned to cluster j. Note that if there are k examples in cluster j, then $\mathbf{X}_j$ and $\mathbf{Y}_j$ will each have k columns.

3. Assign each training example pair (x_i, y_i) to the cluster where the corresponding $\mathbf{T}_j y_i$ best predicts x_i.

4. Repeat steps 2 and 3 until the reconstruction error $\sum_{j=1}^{S} ||\mathbf{X}_j - \mathbf{T}_j \mathbf{Y}_j||$ reaches the desired threshold.

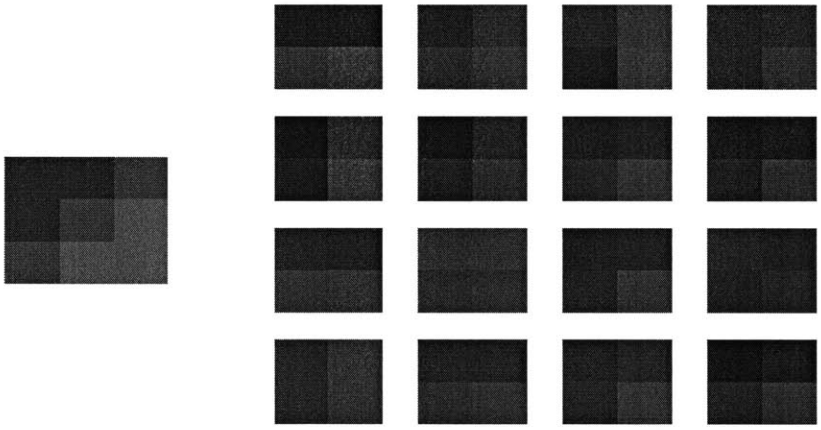
By training S interpolators, we no longer need to store a large database of patches or search the database to obtain S candidates. We greatly reduce the storage requirements and the time to obtain candidates.

For super resolution, we empirically found that learning 16 interpolators using 3×3 low resolution patches will suffice. Figure 5-3 shows the outputs of the trained interpolators on a small low resolution patch. We see that the set of interpolators can generate a variety of high resolution patches.



Low resolution patch High resolution patch candidates

(a)



Low resolution patch High resolution patch candidates

(b)

Figure 5-3: Two 3×3 low resolution patches and their corresponding 2×2 high resolution patch candidates. Notice that for each low resolution patch (the local information), a range of possible high resolution candidates are generated.

Chapter 6

Performing Super Resolution

In this chapter, we outline the algorithm by using the techniques developed in Chapters 4 and 5. We then demonstrate the algorithm and compare the results against other algorithms.

6.1 Outline of Algorithm

With the message-passing equations in hand, we can now describe the algorithm for super resolution. We follow the procedure as outlined in Section 5.2 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:

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 $\mathbf{T}_1 \ldots \mathbf{T}_S$ and l, linearly interpolate to obtain a set of high resolution candidate patches $h_1 \ldots h_S$.

2. With the candidate high resolution patches and observed low resolution image in hand, run BP using the two-dimensional message-passing equations in the Appendix.

Figure 6-1: 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.

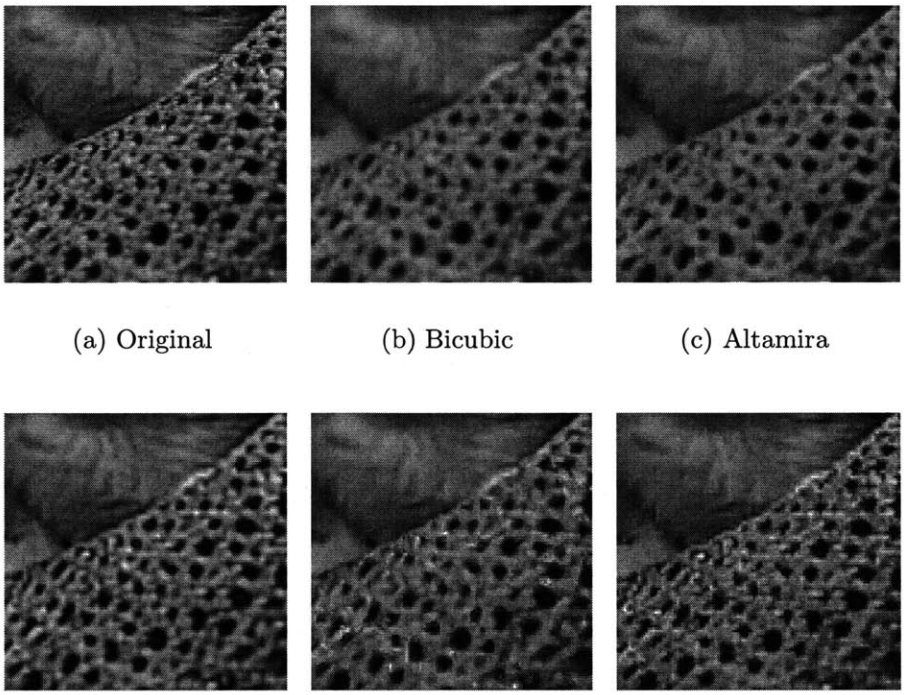
3. For each node, insert into the corresponding position the high resolution patch with the highest belief.

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 5.2.

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.

6.2 Results

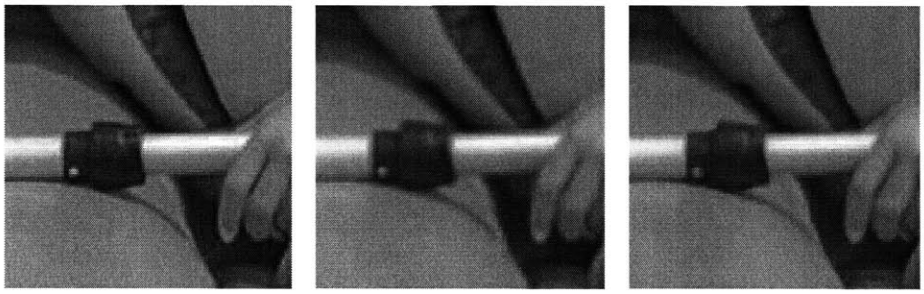
To evaluate our super resolution algorithm, we (1) decimated a test image by filtering with a 3×3 Gaussian kernel and subsampled as described above and (2) super resolved back to the original dimensions. We compared the natural image prior algorithm against the original image, bicubic interpolation, Freeman et al. fast example-based super resolution algorithm [10], Photoshop Altamira plug-in [1], and Greenspan et al. nonlinear enhancement algorithm (using band-pass filtering, $c = 0.4$, and $s = 5$) [13].



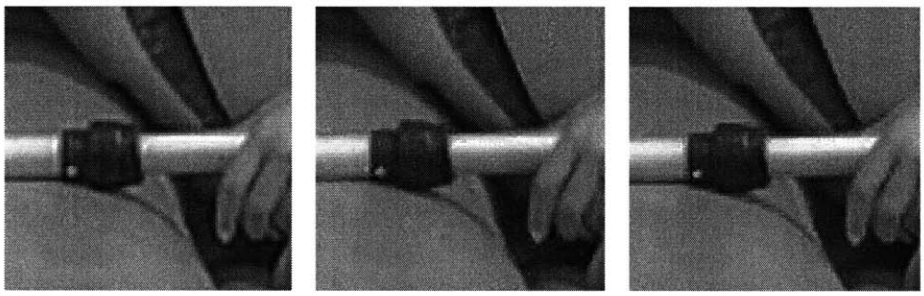
(a) Original (b) Bicubic (c) Altamira

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

Figure 6-2: 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 gives a sharper image than 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.

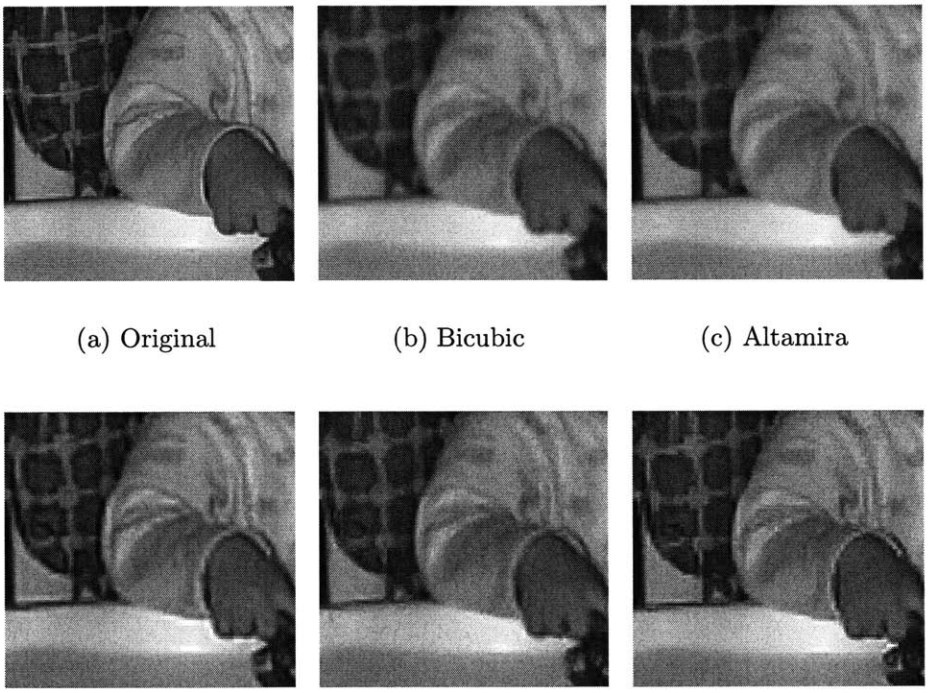


(a) Original (b) Bicubic (c) Altamira



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

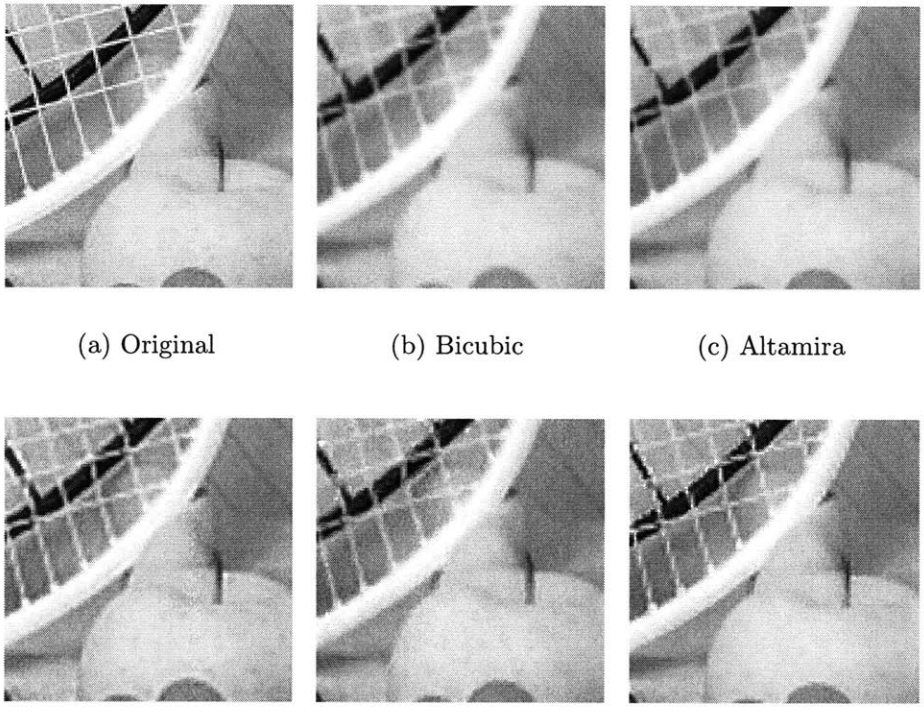
Figure 6-3: 128 × 128 bar region cropped from image 1, decimated to 64 × 64 and then super resolved. (a) True high resolution; (b) Bicubic interpolation; (c) Altamira; (d) Greenspan et al. nonlinear enhancement; (e) Freeman et al. example-based; (f) our natural image prior based algorithm. As in Figure 6-2, 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

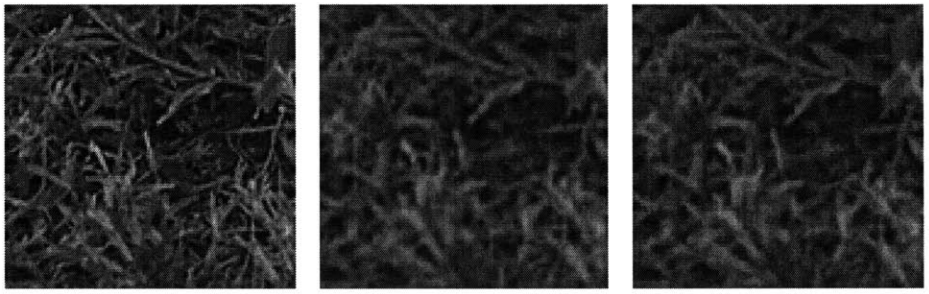
Figure 6-4: 128 × 128 region cropped from image 3, decimated to 64 × 64 and then super resolved. (a) True high resolution; (b) Bicubic interpolation; (c) Altamira; (d) Greenspan et al. nonlinear enhancement; (e) Freeman et al. example-based; (f) our natural image prior based algorithm.



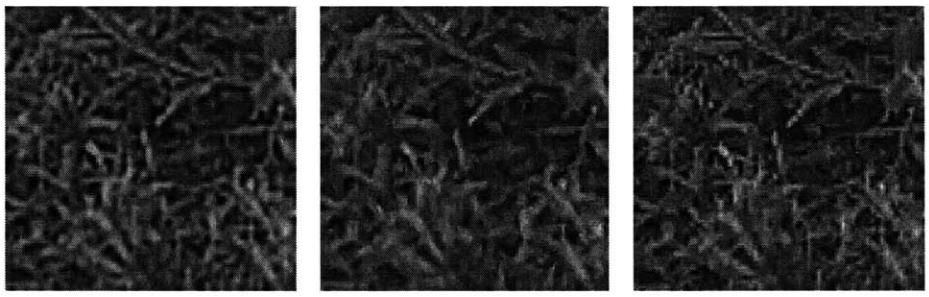
(a) Original (b) Bicubic (c) Altamira

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

Figure 6-5: 128×128 region cropped from image 4, 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.



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



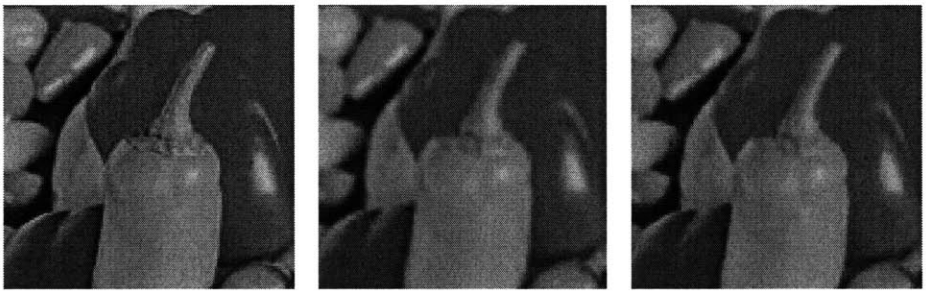
(d) Nonlinear enhancement (e) Example-based (f) Natural image prior

Figure 6-6: Image 5, 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.

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

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

Figure 6-7: 128 × 128 region cropped from image 6, 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.



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

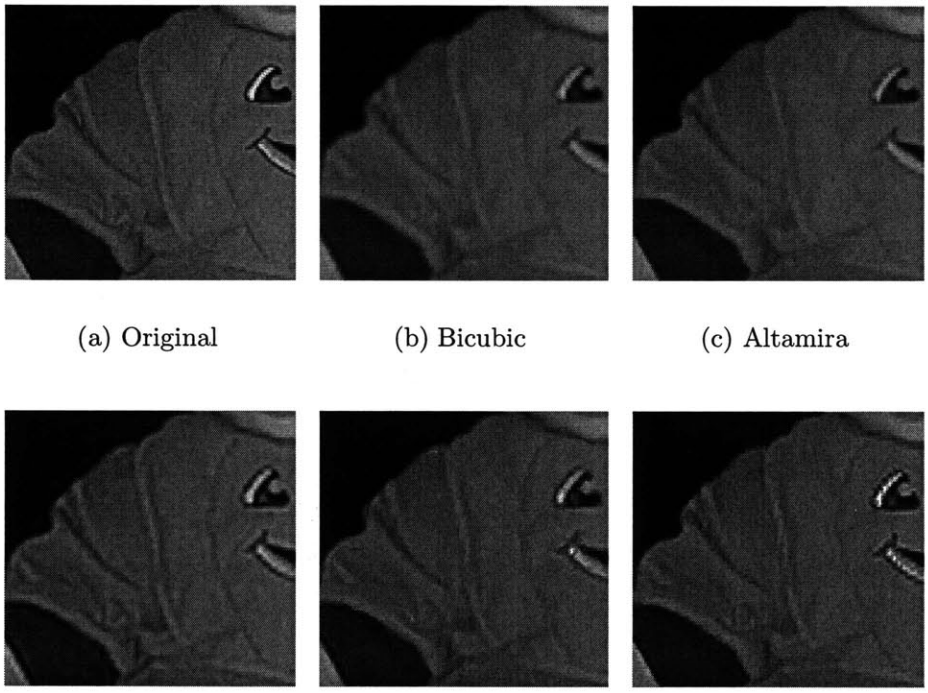
(d) Nonlinear enhancement (e) Example-based (f) Natural image prior

Figure 6-8: 128 × 128 region cropped from image 7, 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.

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

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

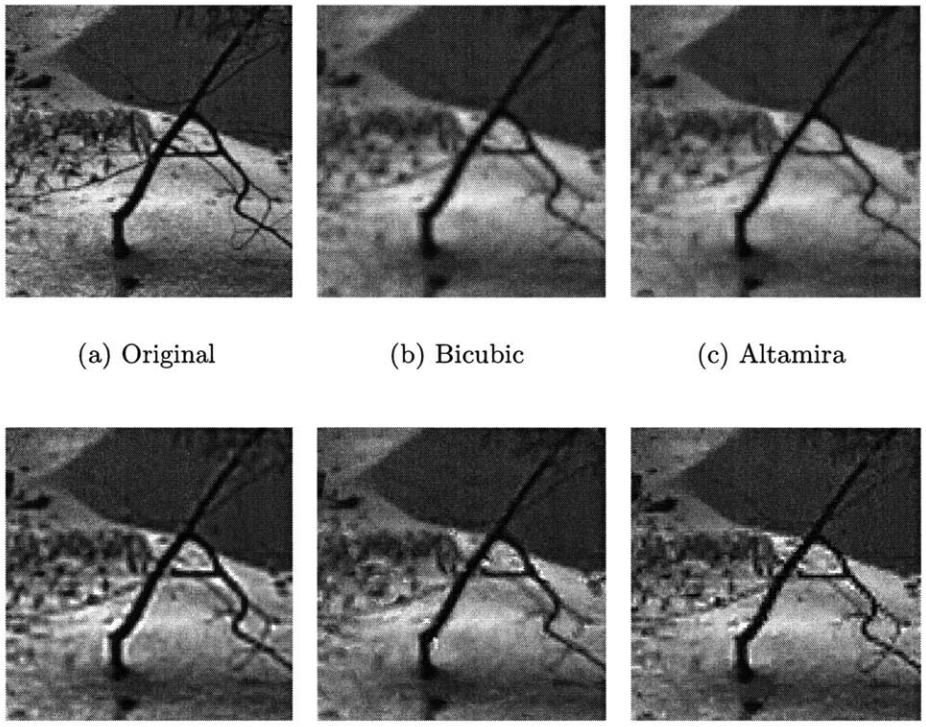
Figure 6-9: 128×128 region cropped from image 8, decimated to 64×64 and then super resolved. (a) True high resolution; (b) Bicubic interpolation; (c) Altamira; (d) Greenspan et al. nonlinear enhancement; (e) Freeman et al. example-based; (f) our natural image prior based algorithm.



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

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

Figure 6-10: 128×128 region cropped from image 9, 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.



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

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

Figure 6-11: 128×128 region cropped from image 10, 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.

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

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

Figure 6-12: 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 6-2 and 6-3, 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.

We tested our algorithm on the set of images shown in Figure 6-1, none of which were used for training.

A comparison of the outputs for the different super resolution algorithms are shown in Figures 6-2 through 6-12. Here, we show cropped sections of two natural images from the test gallery, in addition to a synthetic image of fonts. In Figure 6-13, 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 algorithm.

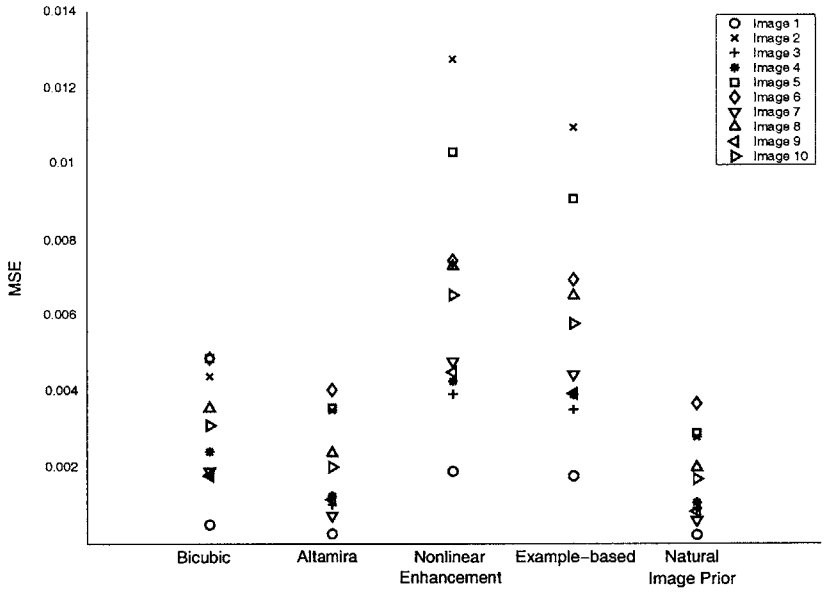


Figure 6-13: Plot of mean-squared error (MSE) for super resolution. Notice that our natural image prior based algorithm has the lowest MSE in comparison to the other methods shown here. While MSE is not always a good measure of image quality, for this problem we feel the MSE correlates reasonably well with the image quality for the different methods, as shown in Figures 6-2, 6-3, and 6-12.

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 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 6-12(d).

Chapter 7

Conclusion

In this chapter, we review the contributions of this thesis and suggest several avenues for future work.

7.1 Contributions

In this thesis, we outlined a general framework for incorporating the statistics of derivatives into graphical models that is used in the field [24], and then applied this framework to the problem of super resolution. For tractability, we formulated a novel approach for generating latent candidate values.

7.1.1 Application of Graphical Models Using Derivative Statistics to the Super Resolution problem

We showed that localized, oriented, bandpass filters applied to natural images have well-observed and robust statistics. We then showed how to incorporate the statistics of the derivative filter into factor graphs to constrain latent pixel values. The power of this incorporation was demonstrated on a simple interpolation problem, which showed how the framework biased the solutions. We then applied this framework to the super resolution problem by introducing, in addition to the derivative statistic constraints, constraints that cause the inferred high resolution image to decimate to the observed

low resolution image. For tractability reasons, we grouped pixels into patches and formulated the factor graph to infer the best patch out of a small set of candidate patches for each latent variable. We demonstrated the resulting super resolved images, which have sharp edges and few visual artifacts, againsts several existing algorithms. We find that our algorithm competes well overall in visual quality and with respect to reconstruction error.

7.1.2 Learned Interpolators to Generate Candidates

To generate a small, well-representative set of candidate patches, we formulated a novel technique that uses a set of interpolators to interpolate locally-observed information to get latent information. We train these interpolators via an EM algorithm on a set of locally-observed, latent information pairs. The interpolation technique is much more efficient than existing search-based methods, while still providing quality candidate sets.

7.2 Perceptual Quantification

While the images produced by the algorithm presented in this thesis have the lowest overall error compared to the other competing algorithms presented, this measurement may not translate to perceptual superiority. Psychophysical experiments, which compare various competing algorithms, need to be done. The results of these experiments may provide insights into improving the overall algorithm.

Appendix A

Two-Dimensional Message-Passing Equations

Here, we give the two-dimensional message-passing equations for the super resolution problem. The natural image prior and reconstruction constraints are shown graphically in Figure A-1. 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 A-1(a)-(d). The propagation equation is given by:

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

The reconstruction constraint is shown graphically in Figure A-1(e) and is given by the propagation equation:

$$\mu_4^{(t+1)}(x_4) \leftarrow \max_{x_1} \max_{x_2} \max_{x_3} \mu_1^{(t)}(x_1)\mu_2^{(t)}(x_2)\mu_3^{(t)}(x_3) \exp\left(-\frac{1}{2}\left(\frac{W * x' - y_1}{\sigma_R}\right)^2\right) \tag{A.2}$$

where W is a 3×3 Gaussian kernel and:

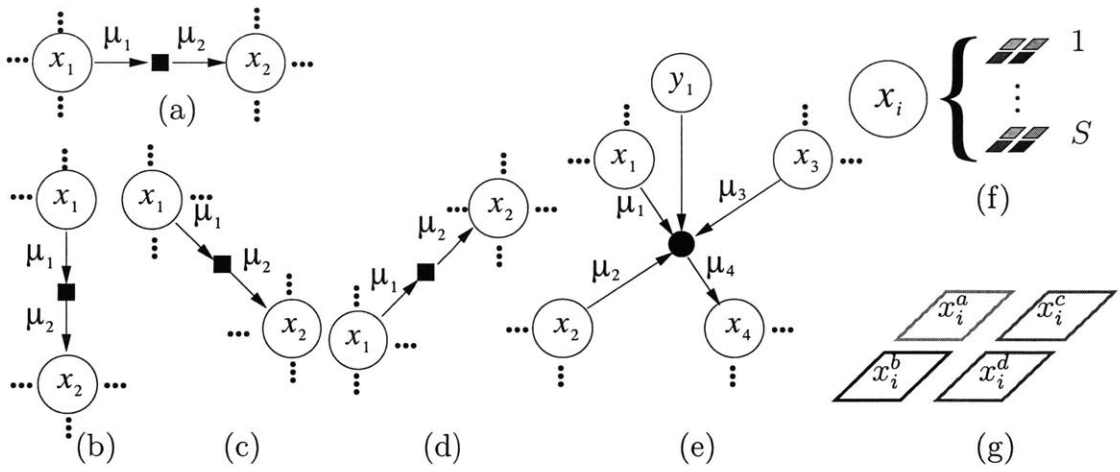


Figure A-1: (a)-(d) Factor graph segment for the directional derivative constraint (horizontal, vertical, and two diagonal directions respectively). (e) The graph segment for the reconstruction constraint. In each of the segments, it is important to remember that the latent nodes x_i represent patches, not individual pixels. In (f), we pictorally 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.

$$x' = \begin{pmatrix} x_1^a & x_1^c & x_3^a \\ x_1^b & x_1^d & x_3^b \\ x_2^a & x_2^c & x_4^a \end{pmatrix}. \tag{A.3}$$

Bibliography

[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] 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.

[5] R. W. Buccigrossi and E. P. Simoncelli. Image compression via joint statistical characterization in the wavelet domain. *IEEE Transactions on Image Processing*, 8(12):1688–1701, 1999.

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

[7] H. Farid and S. Lyu. Higher-order wavelet statistics and their application to digital forensics. *IEEE Workshop on Statistical Analysis in Computer Vision (in conjunction with CVPR), Madison, Wisconsin*, 2003.

[8] D. J. Field. What is the goal of sensory coding? *Neural Computation*, 6:559–601, 1994.

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

[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] S. Geman and D. Geman. Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

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

[14] J. Hadamard. *Lectures on the Cauchy Problem in Linear Partial Differential Equations.* Yale University Press, New Haven, CT, 1923.

[15] 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.

[16] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195:215–244, 1968.

[17] P. J. Huber. *Robust Statistics.* Wiley, New York, 1981.

[18] A. K. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[19] F. Jensen. *An Introduction to Bayesian Networks.* Springer, 1996.

[20] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

[21] D. Kersten. Predictability and redundancy of natural images. *J. Opt. Soc. of Am. A*, 4(12):2395–2400, 1987.

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

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

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

[25] 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.

[26] J. Liu and P. Moulin. Complexity-regularized image resotration. *Proc. IEEE Int. Conf. on Image Proc. (ICIP'98)*, 1:555–559, 1998.

[27] 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.

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

[29] K. Murphy, Y. Weiss, and M. Jordan. Loopy-belief propagation for approximate inference: an empirical study. *Uncertainty in AI*, 1999.

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

[31] 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.

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

[33] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *Int'l Journal of Computer Vision*, 40(1):49–71, 2000.

[34] D. L. Ruderman and W. Bialek. Statistics of natural images: Scaling in the woods. *Physical Review Letters*, 73(6):814–817, 1994.

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

[36] 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.

[37] S. E. Shimony. Finding the maps for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.

[38] 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.

[39] E. P. Simoncelli. Bayesian denoising of visual images in the wavelet domain. In P Muller and B Vidakovic, editors, *Bayesian Inference in Wavelet Based Models*, volume 141 of *Lecture Notes in Statistics*, pages 291–308. Springer-Verlag, New York, 1999.

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

[41] 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.

[42] R. Szeliski. Bayesian modeling of uncertainty in low-level vision. *International Journal of Computer Vision*, 5(3):271–301, 1990.

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

[44] 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.

[45] M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *Uncertainty in Artificial Intelligence*, 2002.

[46] 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.

[47] 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.