1 Lecture 10: Markov Random Fields (MRF's)

Tuesday, Feb. 22, 2010 MIT EECS course 6.869, Bill Freeman and Antonio Torralba
Slide numbers refer to the file "lecture10MRF2011"

slide 1 Last time, we talked about issues in perceptual grouping–finding edges, finding segments. We used a variety of different machineries to do the perceptual grouping, ranging from heuristic threshold setting (with the Canny edge detector) to eigenvector approximations to solve an optimization for segmentation.

In this lecture, we'll introduce yet another bit of machinery that can be used for perceptual grouping, and for many other perceptual problems as well: probabilistic graphical models. We introduced them the week before last, and saw how to perform inference in them (marginalize the joint probability down to variables of interest) for cases where the models had a tree or chain-like structure.

Today, we'll talk about those models in their more general form, with loops, potentially many of them. This is a very general hammer, useful for more than just perceptual grouping. We'll show how to apply this model to various problems, discuss solutions, and finally return to segmentation and show how they can be used for that, too.

slide 2 Let's first remind ourselves about graphical models. They depict some class of joint probability functions that share a common structure. We use that depicted structure to do inference efficiently.

Undirected graphical models make conditional independencies explicit. If there isn't an edge between two nodes, then those nodes are statistically independent, if you condition on the other nodes in the graph (indeed, you only need to condition on a "separator set" of nodes, so that there is no edge path through unconditioned nodes between the two nodes).

slide 3, 4

The factorization of the joint probability depicted by the graph depends on its cliques. This slide shows the definition of cliques. The joint probability is a product of functions of the cliques, called clique potentials.

slide 5 We just worked with very simple ones in class earlier, but I think of graphical models as construction toys for joint probabilities. You can model all sorts of things with them, and make explicit the relevant statistical independencies. These independencies you can then use to do efficient inference.

You might object that such structured probabilistic models are the exception, and usually everything depends on everything else and you can't make these independence assumptions to simplify calculations. But these structured probabilities are the rule, not the exception. Imagine how impossible it would be to behave or interact with the world if the appearance of something here depended on things happening outside the room out there. There sometimes are such dependencies, and you want to model those, but independencies happen so often that you want to be able to model them with these graphs.

The nodes can represent all sorts of things that you want to estimate the state of: joint angles, surfaces, depths, etc.

slides 6, 7, 8 Oftentimes we work with MRF's on a regular grid (although we don't need to). The observed nodes might represent pixels and the hidden nodes could represent labels (like, this is grass, or this has an orientation of 45 degrees). Here we adopt the convention that observed variables are shaded. Nodes can also represent patches of observed pixels or of inferred output pixels.

slide 9 For this grid structure, the cliques all involve just pairs of nodes, and so by the Hammersly Clifford theorem, the joint probability is represented by this product of terms (equation in slide). We have a local term, only depending on the local observations, and a pairwise interaction term with neighboring nodes. That pairwise term will let us say, for example, "we'd like for a node to have the same state as its neighbor".

slide 10 We're looking at this as a joint probability. I should note that it is common to take the log of both sides, to find an equation for an energy to be minimized. Of course, the energy retains the same structure, and it is a sum of local and pairwise terms. Many of the calculations in the energy domain are analogous. Typically, you give up the ability to find marginal probabilities, and you're forced to find a single, best estimate, the lowest energy solution. This corresponds to finding the MAP estimate from the joint probability, which it's also possible to do in the probability domain, using a variant of the belief propagation algorithm we showed before.

slide 11 Of course, even with these more general structures for the joint probabilities, beyond chains, we still want to do inference, that is, to compute the marginal probability at a node or a set of nodes, given the observations. We may want to compute the marginal probabilties, or perhaps the maximum probability state (lowest energy configuration).

An additional task we sometimes have is to learn the parameters of the MRF itself–the local evidence functions and the compatibility functions.

slide 12 To start, let's focus on inference in MRF's. That, by itself, is a large, active research area. In this lecture, we'll describe popular methods, and just name two others.

slide 14

The first method we'll discuss is a brute force method that always works, if you wait long enough. It's called Gibbs sampling. It's kind of a divide and conquor algorithm. It may be very difficult to find random draws from a multi-dimensional function, but it's straightforward to make draws from a one-dimensional function. We can condition on the current state of all variables but one, and draw from that conditional distribution. So we randomly cycle through all the variables, freezing the rest and making a conditional draw. This general approach is called Markov chain monte carlo. Eventually, the outputs of the Gibbs sampler will give random draws from the joint probability of the underlying graphical model. We can take an average over independent samples to find the mean of that joint probability.

Some practical issues: because the variables are initialized randomly, not at all corresponding to a random draw from the joint distribution, one typically waits for a "burn-in period" before including outputs from the Gibbs sampler in the average to find the mean. Similarly, sequential outputs from the Gibbs sampler are not independent draws from the joint posterior, since only one variable has changed between the two outputs, so one typically only includes samples after some period that is large relative to the number of nodes in the graph.

slide 15

Here's a way to sample from a 1-d function, that is, to draw samples of x according to the probability density given by the function, $f(x)$. Discretize the domain x into samples x_k and compute the distribution function, $F(x_k) = \sum_{m=1}^{k} f(x_m)$. By construction, the value of the distribution function F is increasing from the value at x_{k-1} in proportion to the probability density, $f(x_k)$. We can select that value in proportion to $f(x_k)$ if we draw α from the uniform distribution, $[0, 1]$ and select x_k if $F(x_{k-1}) < \alpha < F(x_k)$. This leads to the sampling algorithm in the box of slide 15.

slide 16 This figure shows a 2-d depiction of the conditioning and sampling for a Gibbs sampler. By taking an

average of those samples (separated by enough iterations so that the samples become independent draws from the joint probability) will give an estimate of the mean of the desired joint probability.

slides 17-19 Remember that for some problems it makes sense to use the mean of the joint probability as a best estimate (the MMSE estimate), while for other problems, the location of highest probability might be desired (the MAP estimate). The procedure of slides 15 and 16 will give the MMSE estimate.

We can modify the shape of the probability density to favor the probability peaks, guiding the Gibbs sampler to make draws from only the peak of the joint probability, letting us find the MAP estimate. A common way to do that is introduce a shaping parameter, T, to scale the arguments of the exponents in the joint probability. This accentuates the highest values of the joint probability. As $T \rightarrow 0$, the highest value of $P(x, T)$ dominates all other values, and the Gibbs sampler will only produce samples from near the peak, at x_{MAP}. This procedure is called "simulated annealing", by analogy with the cooling of a physical system to find its ground state energy.

A problem with simulated annealing is that it is very slow: Gibbs sampling is an inner-loop step in the simulated annealing procedure. If the "cooling parameter", T is lowered too quickly, then the random samples won't be able to diffuse toward the most probable areas quickly enough and the system can get stuck in some locally maximal area of the density function.

slides 20-22

An early method to find the most probable state of an MRF, called "Iterated conditional modes" (ICM), is closely related conceptually to the Gibbs sampling procedure. As with Gibbs sampling, we freeze the value of every node except one. Then with that one, in ICM we find the mode of the conditional distribution, instead of drawing a sample from it, as with Gibbs sampling.

There are some distributions for which ICM will quickly find the MAP estimate, but for most distributions of interest, it doesn't perform as well as other methods we're discussing here.

slides 23, 24

Another method to solve Markov Random Fields is to run the belief propagation algorithm, even though it wasn't designed for networks with loops. This is sometimes called "Loopy belief propagation".

Because you may not have finished your homework for Wednesday yet (which has a BP problem in it), let's review belief propagation. We have a graphical model, with observed (shaded) and hidden nodes. We want to find the marginal probability at

slides 25- 30

This is a review of material covered in Weds, Feb. 23, 2011 lecture.

slide 31

I should note, there is a variant of belief propagation. The algorithm we just covered, which lets you compute the marginal probability at each node, is called the sum-product algorithm, because of the summation over message products. From the marginal probability at a node, you can compute the MMSE estimate, the mean value at the node.

A related algorithm lets you compute the MAP estimate efficiently. We can write the MAP estimate in terms of "max" operations, instead of the "sum" operations of the MMSE computation. All the arguments we went through to motivate the sum-product algorithm carry through, with "max" operations passing through factors in the joint posterior, instead of "sum" operations. The result is the "max-product" algorithm for the message updates, shown

here. Taking the maximum over the corresponding "max-marginals" at a node, leads to the MAP estimate at each node (there are some additional details if there are ties in the maximum value over the joint posterior).

slides 32-35 This is all good for graphs without loops. BP gives the correct Bayesian calculation for the desired marginal probabilities, or for the max-marginals, for the MAP version. Since it's an efficient way to compute the marginal probabilities, you can imagine that it has been re-discovered many different times in different contexts, and it has. In the context of Gaussian random variables, observed over time, it is the Kalman filter. For discrete variables, in the MAP version, it is the Viterbi algorithm.

But what to do about general graphs with loops? We note that the update and marginal probability rules themselves are all local rules, and it would certainly be feasible to run the BP updates in a graph with loops, inititalizing the messages to random, or else uniformative, values. As you might expect, researchers have tried this, and, to the surprise of some, have found some remarkable success in doing so. The world's best error correcting codes are decoded using BP run in a network with loops. (That decoding algorithm was a re-discovery of a decoding algorithm proposed by Bob Gallagher (here at MIT) in the 1960's). "Loopy BP" has been applied with success to many computer vision problems. It, or improvements to it for graphs with loops ("tree-reweighted BP"), performs well in competition with other algorithms for solving MRF's with very many loops (see the Szeliski et al 2008 comparison paper).

After some of the experimental work came theoretical justification for the good performance of loopy BP. For Gaussian graphical models, you can show that, after BP converges, the means will be correct, although the covariances will be overly optimistic. (You can see how that might be true, since in the BP updates, you are combining messages which may depend on each other as if they were statistically independent. So rumors get amplified.) The max-product algorithm (which computes MAP solutions) finds the local maximum over a very large region of the space of solutions. There is a strong connection with the physics literature. You can show that, after the algorithm has converged to a solution, it corresponds to a local minimum energy solution to a particular approximation used in statistical physics called the Bethe Free Energy. Wainwright, and later Boykov, have subsequently proposed a variant of the BP update rule to be used in loopy graphs which improves the performance of the max-product algorithm.

testMRF.m is a matlab script that runs ICM, and the sum-product and max-product BP algorithms on a small, binary MRF designed for denoising, and compares the results.

slides 36, 37 Now let's take a digression from the algorithm itself, and show how it might be used within the context of a low-level computer vision algorithm. Consider the task of single-image super-resolution. You are given a single low-resolution image as the input, and you seek your best estimate of what the high-resolution version of that image would be. You might imagine using such an algorithm to estimate an HDTV resolution video frame from an NTSC resolution version frame, or to enlarge photos taken from a low-resolution cell-phone camera.

slides 38-76

The following text, excerpted from a chapter in an upcoming book on Markov Random Fields, and co-authored with Ce Liu, describes the algorthm presented in slides 38-76. A description of slides following 76 begins directly after the bibiliography of the document below.

Markov Random Fields for Super-resolution and Texture Synthesis

Bill Freeman and Ce Liu

February, 2010

2 Introduction

Suppose we want to digitally enlarge a photograph. The input is a single, low-resolution image, and the desired output is an estimate of the high-resolution version of that image. This problem can be phrased as one of "image interpolation": we seek to interpolate the pixel values between our observed samples. Image interpolation is sometimes called super-resolution, since we are estimating data at a resolution beyond that of the image samples. In contrast with multi-image super-resolution methods, where a high-resolution image is inferred from a video sequence, we are interested in estimating high-resolution images from a single low-resolution example [5].

There are many analytic methods for image interpolation, including pixel replication, linear and cubic spline interpolation [15], and sharpened Gaussian interpolation [16]. When we interpolate in resolution by a large amount, such as a factor of four or more in each dimension, these analytic methods typically suffer from a blurred appearance. Following a simple rule, they tend to make conservative, smooth guesses for image appearance.

We can address this problem with two techniques. The first is to use an example-based representation to handle the many special cases we expect. We describe the pre-processing and representaton issues for our example-based representation below. Second, we use a graphical model framework to reason about global structure. The super-resolution problem has a structure similar to other low-level vision tasks: we accumulate local evidence (which may be ambiguous) and propagating it across space. A Markov random field is an appropriate structure for this: local evidence terms can be modeled by unary potentials $\psi_i(x_i)$ at a node i with states x_i. Spatial propagation occurs through pairwise potentials, $\phi_{ij}(x_i, x_j)$, between nodes i and j, or through higher order potentials. The joint probability then has the factorized form,

$$P_{\vec{x}}(\vec{x}) = \frac{1}{Z} \prod_i \psi_i(x_i) \prod_{(ij) \in E} \phi_{ij}(x_i, x_j), \tag{1}$$

where E is the set of edges in the MRF denoted by the neighboring nodes, i and j, and Z is a normalization constant such that the probabilities sum to one [11]. The local statistical relationships allow information to propagate long distances over an image.

2.1 Image pre-filtering

To develop the super-resolution algorithm, we first specify the desired model of subsampling and image degradation that we seek to undo. For the examples in this paper, we assume we low-pass filter the desired high-resolution image, then subsample by a factor of four in each dimension, to obtain the observed low-resolution image. The low-pass filter is a 7x7 pixel Gaussian filter, normalized to have unit sum, of standard deviation 1 pixel. We start from a high-resolution image, and blur it and subsample to generate the corresponding low-resolution image. We apply this model to a set of training images, to generate some number of paired examples of high-resolution and low-resolution image patch pairs.

It is convenient to handle the high- and low-resolution images at the same sampling rate–the same number of pixels. After creating the low-resolution image, we perform an initial interpolation up to the sampling rate of the full-resolution image. Usually this is done with cubic spline interpolation, to create what we will call the "upsampled low-resolution image".

We want to exploit whatever invariances we can to let the training data generalize beyond the training examples. We use two heuristics to try to extend the reach of the examples. First, we don't believe that all spatial frequencies of the low-resolution are needed to predict the missing high-frequency image components, and we don't want

to have to store a different example patch for each possible value of the low-frequency components of the low-resolution patch. So we apply a low-pass filter to the upsampled low-resolution image in order to divide it into two spatial frequency bands. We call the output of the low-pass filter the "low-band", L; the upsampled low-resolution image minus the low-band image gives what we'll call the "mid-band", M. The difference between the upsampled low-resolution image and the original image is the "high-band", H.

A second operation to increase the scope of the examples is contrast normalization. We assume that the relationship of the mid-band, M, to high-band, H, data is independent of the local contrast level. So we normalize the contrast of the mid- and high-band images in the following way:

$$[\hat{M}, \hat{H}] = \frac{[M, H]}{\mathrm{std}(M) + \delta} \tag{2}$$

where $\mathrm{std}(\cdot)$ is standard deviation operator, and δ is a small value which sets the local contrast level below which we do not adjust the contrast. Typically, $\delta = 0.0001$ for images that range over zero to one.

2.2 Representation of the unknown state

We have a choice about what we estimate at the nodes of the MRF. If the variable to be estimated at each node is a single pixel, then the dimensionality of the unknown state at a node is low, which is good. However, it may not be feasible to draw valid conclusions about single pixel states from only performing computations between pairs of pixels. That may place undo burden on the MRF inference. We could remove that burden if a large patch of estimated pixels is assigned to one node, but then the state dimensionality at a node may be unmanagably high.

To address this, we work with entire image patches at each node, to provide sufficient local evidence, but use other means to constrain the state dimensionality at a node. First, we restrict the solution patch to be one of some number of exemplars, typically image examples from some training set. In addition, we take advantage of local image evidence to further constrain the choice of exemplars to be from some smaller set of candidates from the training set. The result is an unknown state dimension of 20 to 40 states per node.

Figure 2 illustrates this representation. The top row shows an input patch from the (bandpassed, contrast normalized) low-resolution input image. The next two rows show the 30 nearest-neighbor examples from a database of 658,788 image patches, extracted from 41 images. The low-res patches are of dimension 25×25, and the high-res patches are of dimension 9×9. The bottom two rows of Fig. 2 show the corresponding high-resolution image patches for each of those 30 nearest neighbors. Note that the mid-band images look approximately the same as each other and as the input patch, while the high-resolution patches look considerably different from each other. This tells us that the local information from the patch by itself is not sufficient to determine the missing high resolution information, and we must use some other source of information to resolve the ambiguity. The state representation is then an index into a collection of exemplars, telling which of the unknown high resolution image patches is the correct one, illustrated in Fig. 3. The resulting MRF is shown in Fig. 1.

2.3 MRF parameterization

We can define a local evidence term and pairwise potentials of the Markov random field if we make assumptions about the probability of encountering a training set exemplar in the test image. We assume any of our image exemplars can appear in the input image with equal probability. We account for differences between the input and training set patches as independent, identically distributed Gaussian noise added to every pixel. Then the local evidence for a node being in sample state x_i depends on the amount of noise needed to translate from the low-resolution patch corresponding to state x_i to the observed mid-band image patch, $\vec{p}$. If we denote the band-passed, contrast normalized mid-band training patch associated with state x_i as $\vec{M}(x_i)$ then

$$\psi_i(x_i) = \exp |\vec{p} - \vec{M}(x_i)|^2 / (2\sigma^2) \tag{3}$$

where we write 2-d image patches as rasterized vectors.

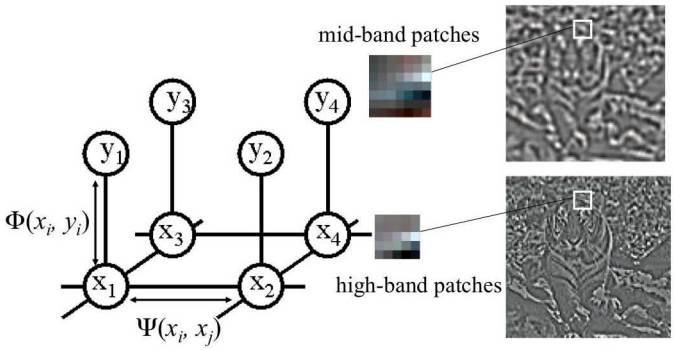


Figure 1: Patch-based MRF for low-level vision. The observations y_i are patches from the mid-band image data. The states to be estimated are indices into a dataset of high-band patches.

To construct the compatibility term, $\phi_{ij}(x_i, x_j)$, we assume we have overlapping high-band patches that should agree with their neighbors in their regions of overlap, see Fig. 4. Any disagreements is again attributed to a Gaussian noise process. If we denote the band-passed, contrast normalized high-band training patch associated with state x_i as $\vec{H}(x_i)$, and introduce an operator O_{ij} that extracts as a rasterized vector the pixels of the overlap region between patches i and j (with the ordering compatible for neighboring patches), then we have

$$\phi_{ij}(x_i, x_j) = \exp |O_{ij}(H(x_i)) - O_{ji}(H(x_j))|^2/(2\sigma^2), \tag{4}$$

In the examples we show below, we used a mid-band and high-band patch size of 9x9 pixels, and used a patch overlap region of size 3 pixels.

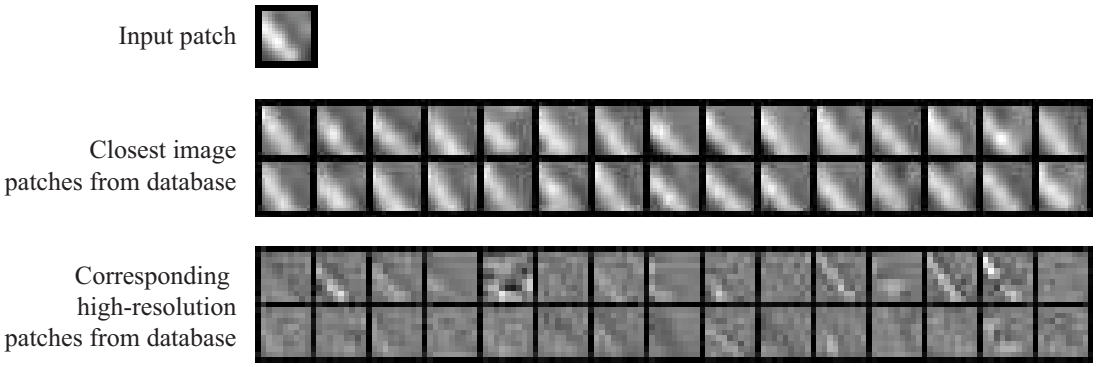


Figure 2: top: input patch (mid-band bandpass filtered, contrast normalized). We seek to find the high-resolution patch associated with this. Middle: Nearest neighbors from database to the input patch. The found patches match this reasonably well. Bottom: The corresponding high-resolution patches associated with each of the retrieved mid-band bandpass patches. These show more variability than the mid-band patches, indicating that more information than simply the local image matches is needed to select the proper high-resolution image estimate. Since the resolution requirements for the color components are lower than for luminance, we use an example-based approach for the luminance, and interpolate the color information by a conventional cubic spline interpolation.

2.4 Loopy belief propagation

We have set-up the Markov Random Field such that each possible selection of states at each node corresponds to a high-resolution image interpretation of the input low-resolution image. The MRF probability, the product of all the local evidence and pairwise potentials in the MRF, assigns a probability to each possible selection of states

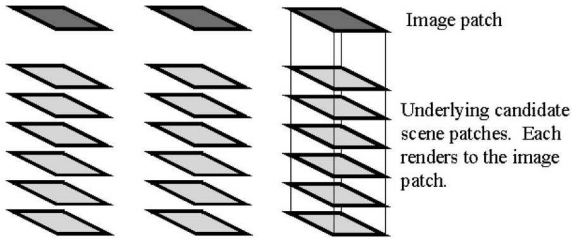


Figure 3: The state to be estimated at each node. Using the local evidence, at each node, we have a small collection of image candidates, selected from our database. We use the belief propagation to select between the candidates, based on compatibility information.

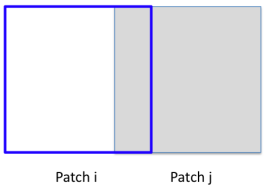


Figure 4: The patch-patch compatibility function is computed from the sum of squared pixel differences in the overlap region.

according to Eq. (1). Each configuration of states specifies an estimated high-band image, and we seek the high-band image that is most favored by the MRF we have specified. This is the task of finding a point estimate from a posterior probability distribution.

In Bayesian decision theory [2] the optimal point estimate depends on the loss function used–the penalty for guessing wrong. With a penalty proportional to the square of the error, the best estimate is the mean of the posterior. However, if all deviations from the true value are equally penalized, then the best estimate is the maximum of the posterior. Using Belief Propagation [13], both estimates can be calculated exactly for an MRF that is a tree.

We consider first the case of the posterior mean, which requires marginalizing the posterior over the states of all other nodes. For a network without loops, the sums over node states for the marginalization can be distributed efficiently over the network in a message-passing algorithm. We define a set of messages, $m_{ij}(x_j)$ along each direction of each edge; the messages can be initialized to random values between zero and one. The messages are functions of the states of the node receiving the message. A message from node i to node j is updated according to,

$$m_{ij}(x_j) \leftarrow \sum_{x_j} \phi(x_i, x_j)\phi_j(x_j) \prod_{k \in \eta(j)_1} m_{kj}(x_j) \tag{5}$$

For the case of a tree network, these updates occur until the messages no longer change. Then the marginal probability at each node is the product of all the incoming messages and the local potential:

$$p_{x_i}(x_i) = \phi_i(x_i) \prod_{j \in \eta(i)} m_{ji}(x_i) \tag{6}$$

When the Markov network forms a tree, belief propagation is simply an efficient redistribution of the sums involved in marginalization, and iterations of Eq. (5) yield the exact marginals by Eq. (6).

Interestingly, for a network with loops, it is often still useful to apply the same update and marginal probability equations, although in that case, the marginal probabilities are only an approximation. The message updates are run until convergence, or for a fixed number of iterations (here, we used 30 iterations). Fixed points of these iterative update rules correspond to stationary points of a well-known approximation used in statistical physics, the

Bethe approximation [18]. Good empirical results have been obtained with that approximation [7, 5], and we use it here.

For our approximation to the MMSE estimate, we take the mean (weighted by the marginals from Eq. (6)) of the candidate patches at a node. It is also possible to approximate the MAP estimate by substituting the summation operator of Eq. (5) with "max", then selecting the patch maximizing the resulting "max-marginal" given in Eq. (6). These solutions are often sharper, but with more artifacts, than the MMSE estimate.

To piece together the final image, we undo the contrast normalization of each patch, average neighboring patches in regions where they overlap, add-in the low and mid-band images, and add-in the analytically interpolated chrominance information. Figure 5 summarizes the steps in the algorithm, and Fig. 6 shows other results. The perceived sharpness is significantly improved, and the belief propagation iterations significantly reduce the artifacts that would result from estimating the high-resolution image based on local image information alone. (Figure 7 provides enlargments of cropped regions from those two figures.) The code used to generate the images in Sect. 2.4 is available for download at http://people.csail.mit.edu/billf/.

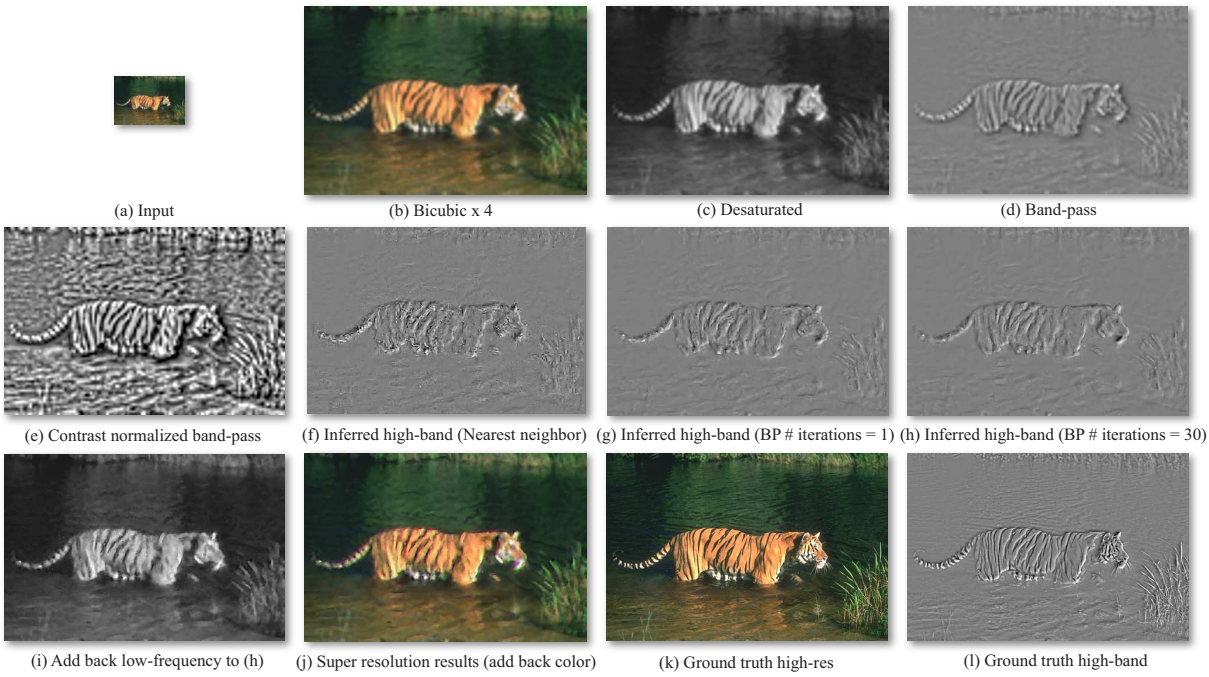
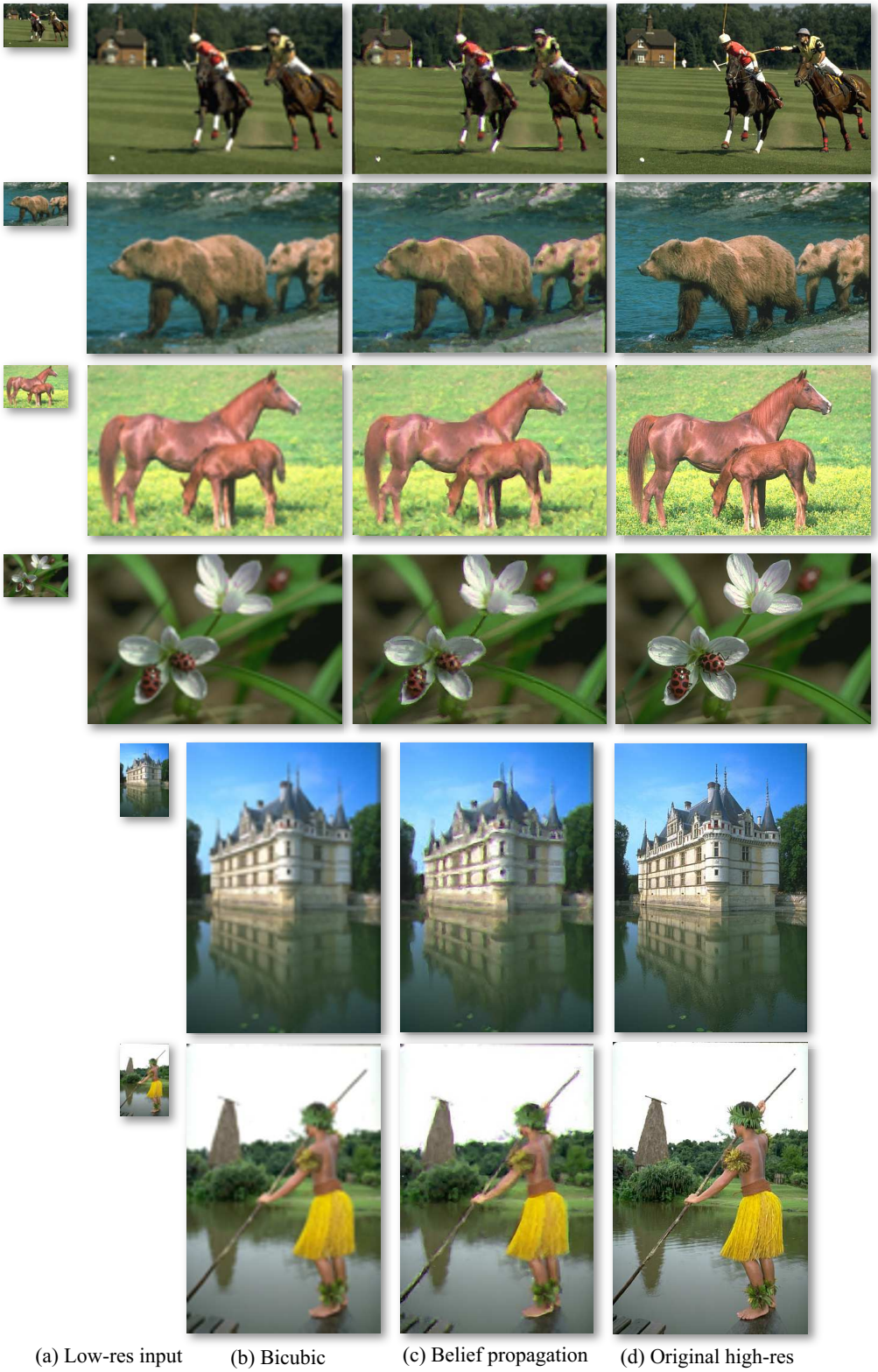


Figure 5: Images showing the example-based super-resolution processing. (a) input image, of resolution 120×80. (b) Cubic spline interpolation up to a factor of four higher resolution in each dimension. (c) We extract the luminance component for example-based processing (and use cubic spline interpolation for the chrominance components). (d) A high-pass filtering of this image gives us the mid-band output, shown here. (e) Display of the contrast normalized mid-band. The contrast normalization extends the utility of the training database samples beyond the contrast value of each particular training example. (f) the high frequencies corresponding to the nearest neighbor of each local low-frequency patch. (g) After 1 iteration of belief propagation, much of the choppy high frequency details of (f) are removed. (h) converged high resolution estimates. (i) Image (c) added to image (h)–the estimated high frequencies added back to the mid and low-frequencies. (j) Color components added back in. (k) comparison with ground truth. (l) true high frequency components.



(a) Low-res input (b) Bicubic (c) Belief propagation (d) Original high-res

Figure 6: Other example-based super-resolution outputs. (a) Input low-res images. (b) Bicubic interpolation (x4 resolution increase). (c) Belief propagation output. (d) The true high-resolution images.

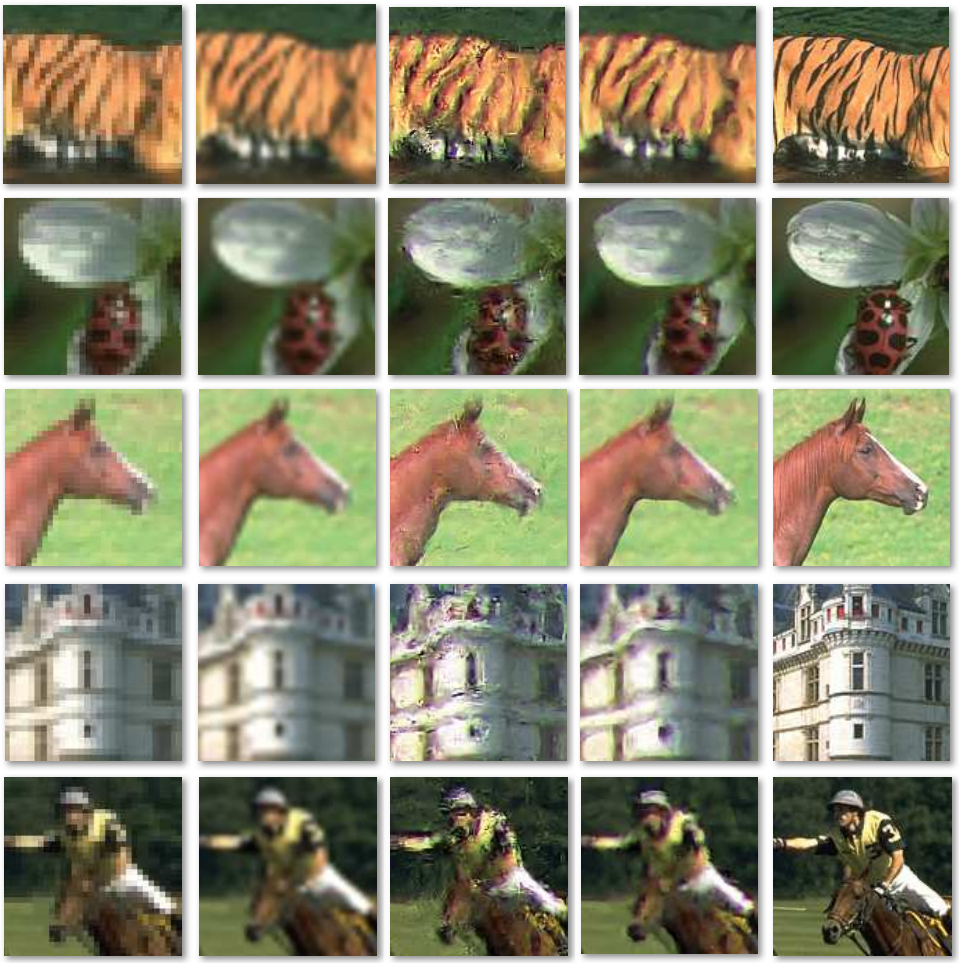


Figure 7: The closeups of Figure 5 and 6. (a) Input low-res images. (b) Bicubic interpolation (x4 resolution increase). (c) Nearest neighbor output. (d) Belief propagation output. (c) The true high-resolution images.

3 Selected related applications by others

Markov random fields have been used extensively in image processing and computer vision. Geman and Geman brought Markov random fields to the attention of the vision community, and showed how to use MRF's as image priors in restoration applications, [8]. Poggio, Gamble and Little used MRF's in a framework unifying different computer vision modules, [14].

The example-based approach has been built on by others. This method has been used in combination with a resolution enhancement model specific to faces [1] to achieve excellent results in hallucinating details of faces [12]. Huang and Ma have proposed finding a linear combination of the candidate patches to fit the input data, then applying the same regression to the output patches, simulating a better fit to the input [17]. (A related approach was also used in [6]).

Optimal seams for image transitions were found in a 2-d framework, using graph cuts in Kwatra et al [10]. Example-based image priors were used for image-based rendering in the work of Fitzgibbon, Wexler, and Zisserman, [4]. Fattal used edge models for image upsampling [3]. Glasner et al also used an example-based approach for super-resolution, relying on self-similarity within a single image [9].

References

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

[2] J. O. Berger. *Statistical decision theory and Bayesian analysis*. Springer, 1985.

[3] R. Fattal. Upsampling via imposed edges statistics. In *ACM SIGGRAPH*, 2007. In *Computer Graphics* Proceedings, Annual Conference Series.

[4] A.W. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. In *Proc. International Conference on Computer Vision*, 2003.

[5] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Intl. J. Computer Vision*, 40(1):25–47, 2000. See http://www.merl.com/reports/TR2000-05/.

[6] W. T. Freeman, J. B. Tenenbaum, and E. C. Pasztor. Learning style translation for the lines of a drawing. *ACM Transactions on Graphics (TOG)*, 22:33 – 46, 2003.

[7] B. J. Frey, R. Koetter, and N. Petrovic. Very loopy belief propagation for unwrapping phase images. In *Adv. in Neural Info. Proc. Systems*, volume 13. MIT Press, 2001. http://www.psi.toronto.edu/pubs/2001/sppunips01.pdf.

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

[9] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *Proc. International Conference on Computer Vision*, 2009.

[10] V. Kwatra, A. Schdl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. In *ACM SIGGRAPH*, 2003. In *Computer Graphics* Proceedings, Annual Conference Series.

[11] S. Z. Li. *Markov random field modeling in image analysis*. Springer, 2009.

[12] C. Liu, H. Y. Shum, and W. T. Freeman. Face hallucination: theory and practice. *International Journal of Computer Vision*, 75(1):115–134, 2007.

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

[14] T Poggio, E B Gamble, and J J Little. Parallel integration of vision modules. *Science*, 242:436–440, 1989.

[15] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge Univ. Press, 1992.

[16] W. F. Schreiber and D. E. Troxel. Transformation between continuous and discrete representations of images: A perceptual approach. *PAMI*, 7(2):176–178, 1985.

[17] J. Yang, J. Wright, T. Huang, and Y. Ma. image super-resolution as sparse representation of raw image patches. In *Proc. IEEE Computer Vision and Pattern Recognition*, 2008.

[18] J. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *International Joint Conference on Artificial Intelligence (IJCAI 2001)*, 2001. Distinguished Papers Track.

slides 77

We can also apply this method–BP, using this patch-based data-driven method–to other low-level vision problems. Here, we show its application to motion.

The input is pairs of frames of image data (in this case, vector quantized into discrete states). The output is a set of motion states (again vector quantized from example motion vectors).

slides 78-80 What behavior do we expect from this algorithm? From the local evidence only, we will be confronted with the "aperture problem". From the local, straight-edge of a contour, we can only discern the component of the contour's motion perpendicular to the contour. The motion ambiguity can be resolved by aggregating motion evidence from other orientations of the contour. We may expect to see that happen.

Another local ambiguity we'll need to resolve is the figure/ground problem (your problem set due Weds also addresses this problem).

slides 82-84

Those amiguities, and their resolutions, can be seen in the first six iterations of loopy belief propagation. Remember that messages are passed to neighboring nodes across scale, as well as in space.

Note the limitations of this algorithm: states are quantized rather coarsely, to save in storage and testing speed. The algorithm (loopy belief propagation) is not guaranteed to converge.

slides 86-91

Finally, to connect with the previous class, let's look at how you might do segmentation with a Markov Random Field model.

A conventional MRF is called a generative model, because you could generate samples of image data from it, using Gibbs sampling to produce images that are probable under the joint density described by the MRF.

But there is another way to use Markov Random Fields. We can define a special class of Markov Random Field, called a Conditional Random Field (CRF), where the potential functions between hidden variables depend on the image data. These graphical models depend on the image data, and so do not represent prior probabilities for the images.

These conditional random fields are commonly used in stereo depth reconstructions. The pairwise potentials are then set to encourage neighboring depth estimates to agree, but in the presence of a strong image gradient, that constraint is relaxed.

ObjCut is a relatively recent paper that enforces an object-based prior on a CRF to encourage segmentation of image data, enforcing a prior preference that the segments conform to the shape of a learned object class, in this case, a cow.

slides 92-100

Finally, let's talk about how we might learn the pairwise potential functions of an MRF. It is straightforward to show that in a maximum likelihood estimate of model parameters, the model's prediction for the marginal probability at any clique will be equal to the observed marginal probabilities for that clique of nodes. We can measure those marginal probabilities simply by counting frequencies of occurence of states in a labeled training set.

If we measure the marginal probabilities, what does it tell us about the joint probabilities? There is a procedure, called Iterative Proportional Fitting (IPF), that adjusts an estimate of a joint probability based on observations of the marginals. The result is a maximum entropy estimate of the joint probability. The update rule is to scale the joint probability by the ratio of the observed marginal probability over the current model's predicted marginal probability.

$$P(\vec{x})^{t+1} = P(\vec{x})^{t} \frac{P(x_c)^{\text{observed}}}{P(x_c)^{t}} \tag{7}$$

If we substitute in the product of the clique potentials for $P(\vec{x})^{t+1}$, and only modify the clique potential corresponding to the marginal we measure, then we have this update equation:

$$\phi_c(x_c)^{t+1} = \phi_c(x_c)^{t} \frac{P(x_c)^{\text{observed}}}{P(x_c)^{t}} \tag{8}$$