Course notes for MIT EECS 6.819/6.869 Spring 2022

Antonio Torralba, Phillip Isola and William Freeman

April 12, 2022

Contents

Chapter 1

Statistical image models

Building statistical models for images is an active area of research in which there has been remarkable progress. Understanding the world of images and finding rules that describe what is likely and what is not to find within an image seems like a very challenging task. What we will see in this chapter is that some of the filters that we have seen in the previous chapters have remarkable properties when applied to images.

First we need to understand what we mean by a *natural image*. An image is a very high dimensional array of pixels $x[n, m]$ and therefore, the space of all possible images is very large. Even if you consider very small thumbnails of 32×32 pixels and 3 color channels, with 8 bits per channel, there are more than 10^{7000} possible images. Of course, most of them will just be random noise.

As natural images are quite complex, research started considering simple visual worlds that retained some of the important properties of natural images while allowing developing analytic tools to characterize them.

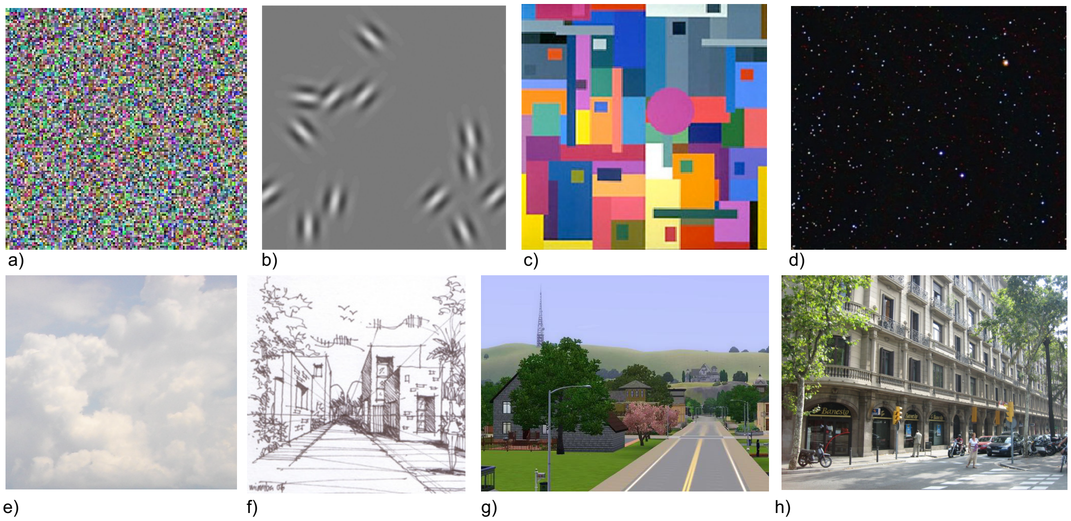


Figure 1.1: Different visual worlds, some real, some synthetic. a) white noise, b) gabor patches, b) mondrian, d) stars, e) clouds, f) line drawings, g) computer graphic imagery (CGI), e) a real street. All these worlds have different visual properties. Even there is something that makes CGI images distinct from pictures of true scenes.

Figure 1.1 shows images that belong to different visual worlds. In this chapter we will talk, in some way or another, about all these visual worlds. The goal is to present a set of models that can be used to describe images with different properties. We will describe models that can be used to capture what makes special each visual world. These models can be used to separate images into different causes, to remove noise from images, to invent image high-frequencies, and to fill-in regions of missing image information.

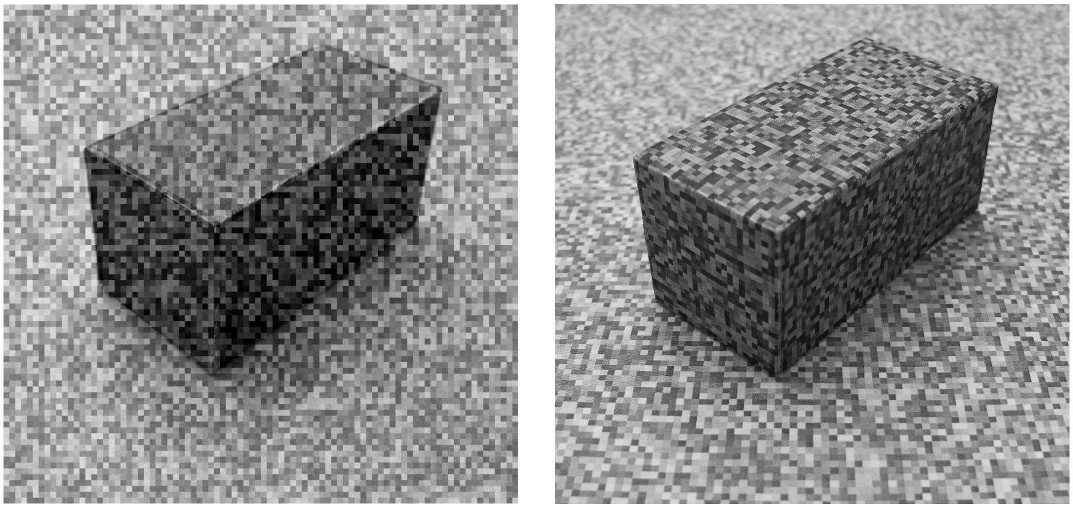


Figure 1.2: Telling noise from texture. Which one is which?

There is a significant interest in building image models that can represent whatever makes special real photographs relative to, let's say, images that just contain noise. One way of building an image model is by having some procedural description, for instance a list of objects with locations, poses and styles and a rendering engine, just as one would do when writing a program to generate CGI. One issue with this way of modeling images is that it will be hard to use it, and will require having an exhaustive list of all the things that we can find in the world. We will see later that the apparent explosion in complexity should not stop us. **[Billf:: what is this in reference to, Antonio?]**

However there is another way of building image models. In the rest of this chapter we will study a very different approach that consists in building statistical models of images that try to capture the properties of small image neighborhoods. Statistical models are very successful in other disciplines such as natural language modeling. Here we will review models of images of increasing complexity starting with the simplest possible neighborhood: *the pixel*.

1.1 Independent Pixels

The simplest image model will consist in assuming that all pixels are independent and their intensity value is drawn from some distribution. We can then write the probability of an image, $p(\mathbf{x})$, as:

$$p(\mathbf{x}) = \prod_{n,m} p(x[n,m]) \tag{1.1}$$

We will see the interest of writing models in this form. One way of getting a sense of how accurate is a statistical image model is to sample from this model and check the types of images that it produces. This will require knowing the distribution $p(x[m,n])$ (we will assume this function is the same in all image locations). What we can do is to take one image from one of the visual worlds that we described in the introduction, we can then estimate the function $p(x[n,m])$ as the histogram of the image. We can then sample new images from Eq. 1.1. Figure 1.3 shows two pairs of images with matched intensity histograms. We can see that this simplistic model is somewhat appropriate for pictures of stars (although star images can have a lot more structure), but it miserably fails in reproducing anything with any similarity to a street picture.

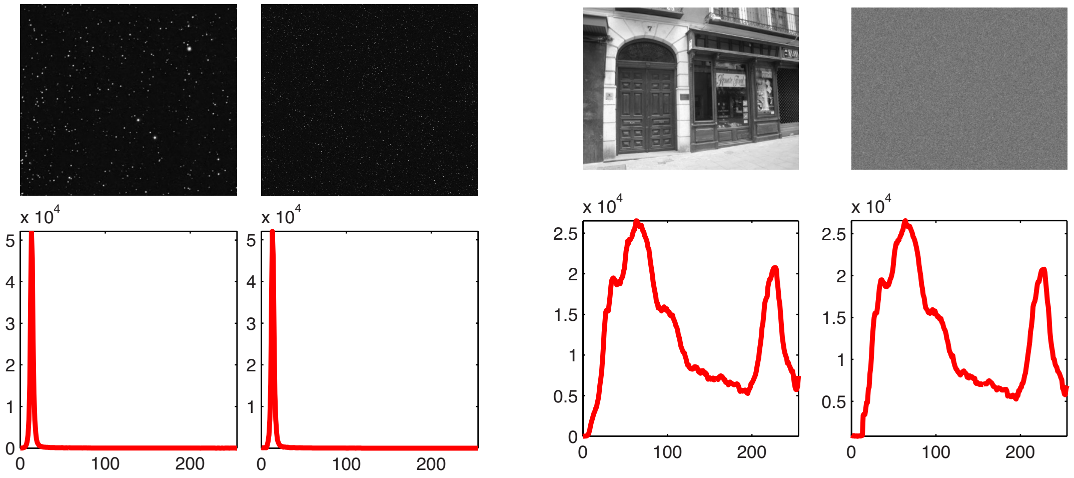


Figure 1.3: Examples of images and white noise with matched histograms. Only the image with stars has some visual similarity with pictures of stars. **[Billf:: need to display cropped close-ups of these images, so the pixels are visible]**

As a generative model of images, this model is very poor. However, this does not mean that image histograms are not important. Manipulating image histograms is a very useful operation. Given two images x_1 and x_2 with histograms h_1 and h_2, we look for a pixelwise transformation $f(x[n,m])$ so that the histogram of the transformed image, $f(x_1)$, matches the histogram of x_2. There are many ways in which histograms can be transformed. One natural choice is a transformation that preserves the ordering of pixels intensities.

Figure 1.4 shows examples of spheres that are reflecting some complex ambient illumination. The two spheres on the left are the two original images. The ones on the left are generated by modifying the image histograms to match the histogram of the opposite image. The figure shows that by simply changing the image histogram we can transform how we perceive the material of each sphere: from shiny to mate.

1.2 Dead Leaves Model

In the quest to find what makes photographs of everyday scenes special, the next simplest image structure is a set of two pixels. Things get a lot more interesting now. If one plots the value of the intensity of one pixel as a function of the intensity value of the pixel nearby, the scatterplot produced by many pixel pairs (fig. 1.5) is concentrated on the identity line. As we look at the correlation between pixels that are farther away, the correlation value slowly decays. The correlation between two pixels is:

$$C(\Delta n, \Delta m) = \rho\left[x[n+\Delta n, m+\Delta m], x[n,m]\right] \tag{1.2}$$

The behavior of this correlation function when computed on natural images is very different from the one we would observe if the correlation was computed over white noise images.

There has been a large number of efforts trying to model what is the minimal set of assumptions that one needs to make in order to reproduce the observed behavior. The *dead leaves model* is a simplified image formation model that tries to approximate some of the properties observed for natural images. This model was introduced in the 60's by Matheron [Matheron1975] and popularized by Ruderman [Ruderman1997]. This model consist in assuming that an image can be modeled as a set of disks (dead leaves) that fall on a flat surface generating a random superposition (e.g., fig. 1.1.c).

This model is very simple and does not produce realistic images, but it is similar to the image model assumed to explain the distribution of albedos in natural images. This model

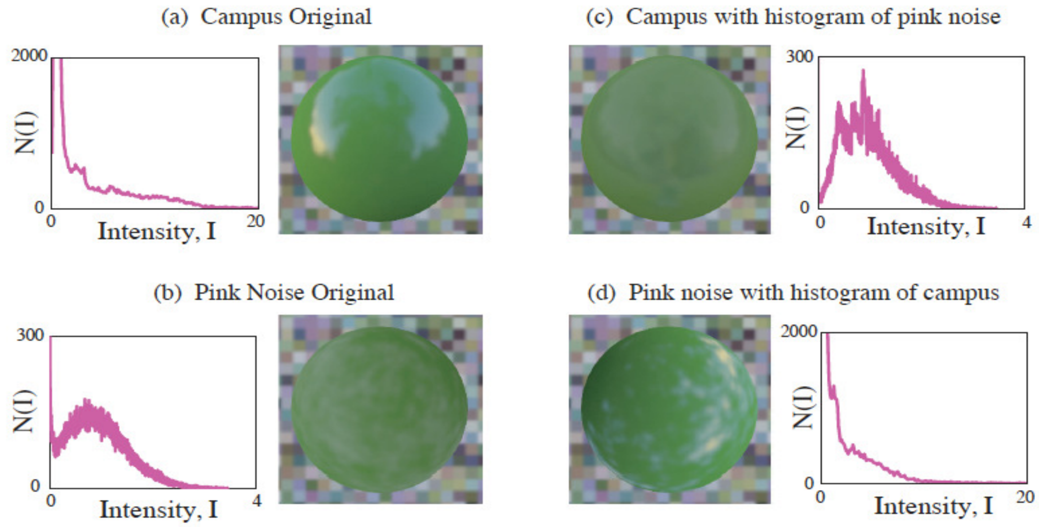


Figure 1.4: This is an example that illustrates the perceptual importance of simple histogram manipulations. Note how by simply modifying the image histograms within the pixels of the sphere, the sphere seems to be made by a different type of material. The sphere was first rendered with two illumination models, (a) from a campus photograph, and (b) from pink noise. In (c) and (d) the sphere histograms were swapped, creating a different sense of gloss in the sphere images, image from [R. W. Fleming].

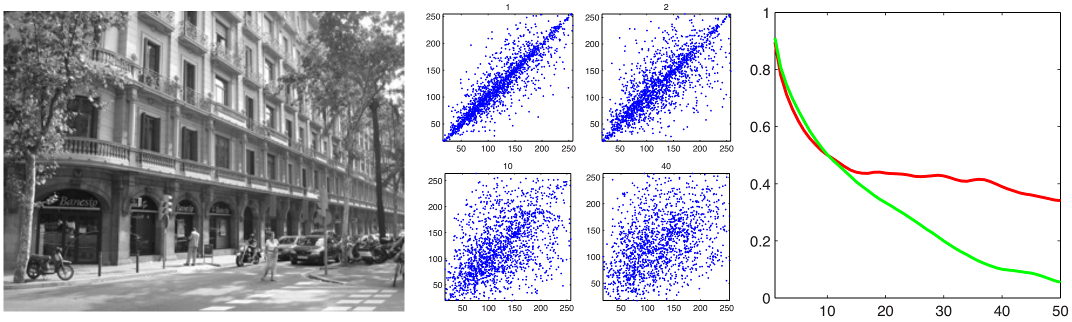


Figure 1.5: Scatter plots of pairs of pixels intensities as a function of distance, and cross-correlation function for vertical and horizontal displacements.

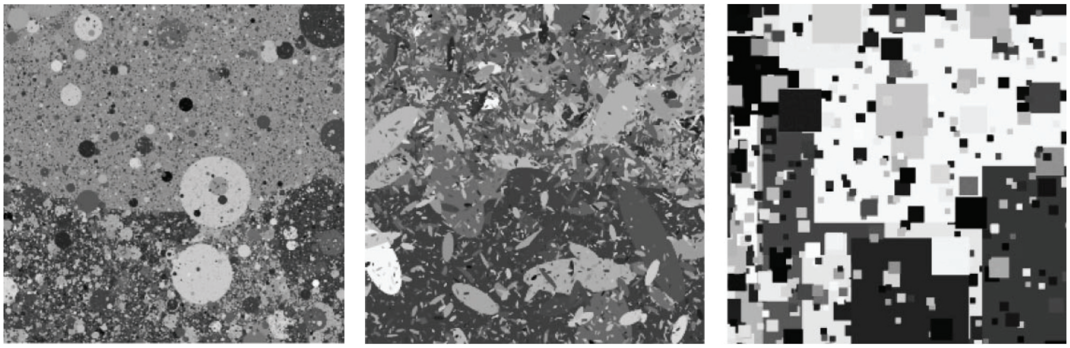


Figure 1.6: Images resulting from the dead leaves model, for different shapes of ”dead leaves”. Left: circles, Middle: ellipses, Right: squares.

is used by the Retinex algorithm [Land1983] to separate and image into two components: illumination and albedo variations.

The particular form of the correlation function for natural images is made more explicit when studied on the Fourier domain. One remarkable property of most natural images is that if you take the 2D Fourier transform of an image, its magnitude falls off as:

$$|\mathbf{X}[w]| \simeq \frac{1}{|w|^{\alpha}} \tag{1.3}$$

where w denotes spatial frequency ($w = \sqrt{u+v}$), $\mathbf{X}$ is the Fourier transform of the image $\mathbf{x}$, and $\alpha \simeq 1$. And this is true for all orientations (you can think of this as taking the Fourier transform and looking at the profile of a slice that passes by the origin).

1.3 The Gaussian Model

We want to write a statistical image model that takes into account the correlation statistics of natural images (see [Simoncelli2005] for a review). If the only constraint we have is the correlation function among image pixels, then, among all the possible distributions the one that has the maximum entropy (maximum entropy principle) is the Gaussian distribution:

$$p(\mathbf{x}) \propto \exp\left(-\frac{1}{2}\mathbf{x}^T\mathbf{C}^{-1}\mathbf{x}\right) \tag{1.4}$$

where $\mathbf{C}$ is the covariance matrix of all the image pixels. Note that now this model does not assume independency across pixels any more. This model takes into account that intensity values for different pixels are correlated as discussed in the previous sections. This model is related also to studies that use principal component analysis to study what are the typical components of natural images. For stationary signals, the covariance matrix $\mathbf{C}$ has a circulant structure (assuming a tiling of the image) and can be diagonalized using the Fourier transform. The eigenvalues of the diagonal matrix correspond to the power (squared magnitude) of the frequency components of the Fourier transform. This is, the matrix $\mathbf{C}$ can be written as $\mathbf{C} = \mathbf{F}\mathbf{D}\mathbf{F}^T$, where $\mathbf{F}$ is the matrix that contains the Fourier basis (as seen in Chapter Fourier Analysis *****) and the diagonal matrix $\mathbf{D}$ has the values $\frac{1}{|v|^{\alpha}}$ for all the frequencies v.

Sampling images

Once a distribution is defined, we can use it to sample new images. First we need to specify the parameters of the distribution. In this case, we need to specify the covariance matrix $\mathbf{C}$. We can estimate this as the average covariance matrix of a set of images. Figure 1.7 shows an example of an image of clouds and the magnitude of its Fourier transform. A sample of the distribution can be obtained by filtering white Gaussian noise using the image magnitude as a filter.

Image denoising under the Gaussian model: Wiener filter

As an example of how to use image priors for vision tasks, we will study how to do image denoising using the prior on the structure of the correlation of natural images. In this problem, we observe a noisy image $\mathbf{x}_g$ corrupted with white Gaussian noise:

$$x_g[n,m] = x[n,m] + g[n,m] \tag{1.5}$$

and the goal is to recover the uncorrupted image $x[n,m]$. The noise $g[m,n]$ is white Gaussian noise with variance σ_g^2. The denoising problem can be formulated as finding $x[n,m]$ that maximizes the maximum a posteriori (MAP estimate):

$$\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{x}_g) \tag{1.6}$$

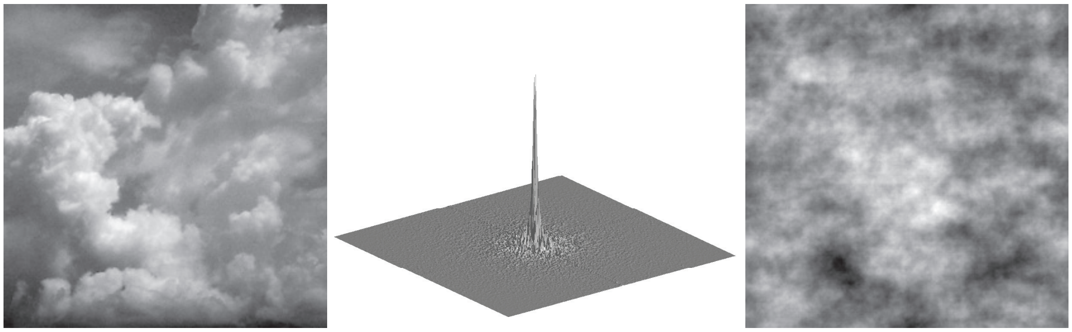
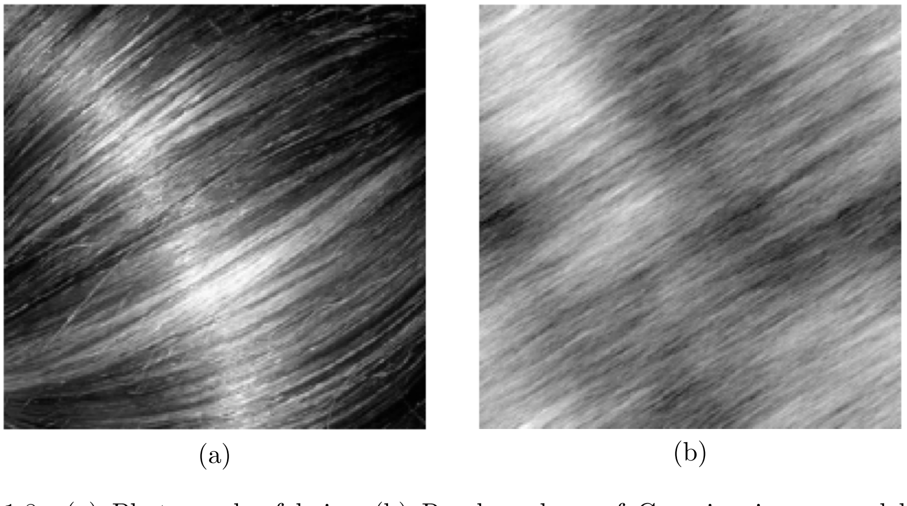


Figure 1.7: Randomizing the phase of picture of clouds. Left) clouds, Center) magnitude of its Fourier transform, Right) Another random draw from the same Gaussian image model, obtained by randomizing the phase, while keeping the magnitude of the Fourier transform the same.



(a)

(b)

Figure 1.8: (a) Photograph of hair. (b) Random draw of Gaussian image model using covariance matrix fit from (a). For this unusual image, the model works well, but this is the exception for the Gaussian image model.

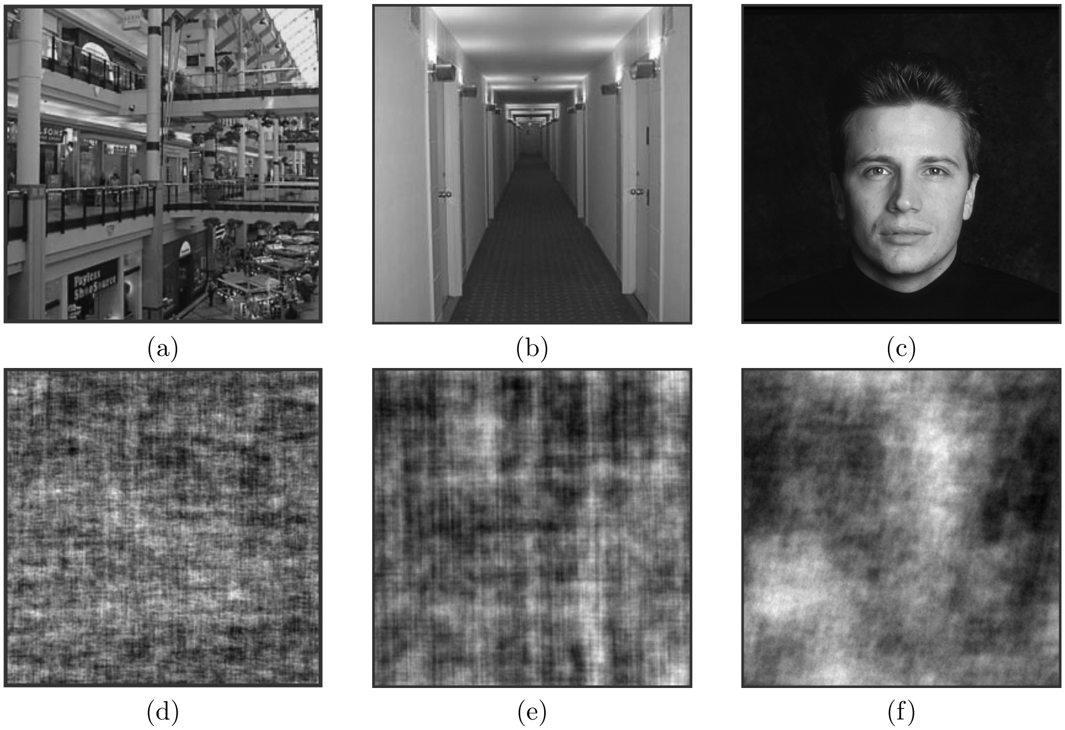


Figure 1.9: Top row, (a), (b), (c): three images. Bottom row (d), (e), (f): Images with identical covariance structure as the image above it. These two images are equally probable in the Gaussian image model for each image of the top row.

In these equations we write the image as a column vector $\mathbf{x}$. This posterior density can be written as:

$$\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{x}_g) = \max_{\mathbf{x}} p(\mathbf{x}_g|\mathbf{x})p(\mathbf{x}) \tag{1.7}$$

where the likelihood and prior functions are:

$$\begin{aligned} p(\mathbf{x}_g|\mathbf{x}) &\propto \exp(-\left|\mathbf{x}_g - \mathbf{x}\right|^2 / \sigma_g^2) && (1.8) \\ p(\mathbf{x}) &\propto \exp\left(-\frac{1}{2}\mathbf{x}^T \mathbf{C}^{-1} \mathbf{x}\right) && (1.9) \end{aligned}$$

The solution to this problem can be obtained in closed form:

$$\mathbf{x} = \mathbf{C}\left(\mathbf{C} + \sigma_g^2 \mathbf{I}\right)^{-1} \mathbf{x}_g \tag{1.10}$$

This is just a linear operation. It can also be written in the Fourier domain as:

$$\mathbf{X}(v) = \frac{A/|v|^{2\alpha}}{A/|v|^{2\alpha} + \sigma_g^2} \mathbf{X}_g(v) \tag{1.11}$$

1.4 The wavelet marginal model

Despite the popularity of the Gaussian model, it fails in capturing the structure of natural images. As we showed before, sampling from a Gaussian prior model generates images of clouds. Although the Gaussian prior does not make any image to have zero probability (i.e., by sampling for a very long time it is not impossible to get a picture of the Mona Lisa), the most typical images under this prior are clouds.

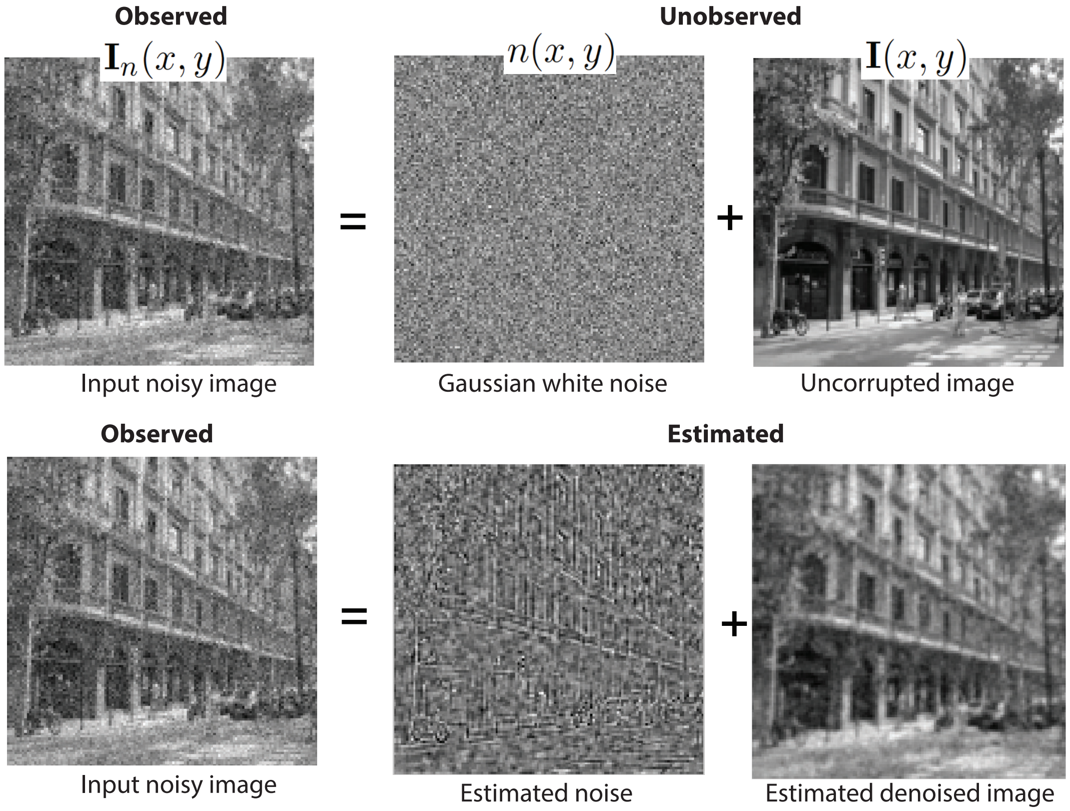


Figure 1.10: Top row: ground truth decomposition of image into Gaussian white noise and the image, uncorrupted by noise. Bottom row: Gaussian image model denoising results. Note that the estimated noise image shows spatial structure from the original image.

The Gaussian model captures the fact that pixel values are correlated and that such correlation decreases with distance. However, it fails in capturing other very important properties of image such that images contain flat regions, edges and lines. Here, we extend the Gaussian model in two ways. (1) We will use localized filters, instead of sinusoids that span the image, as in the power spectrum model. This helps to capture local structure such as lines and edges. (2) Instead of characterizing the outputs of those filters by a single number (eg, the mean power), we measure the histogram shape of all the responses.

In the 80s, a number of papers noticed a remarkable property of images: when filtering images with band-pass filters (i.e, image derivatives, Gabor filters, etc.) the output had a non-Gaussian distribution. Fig. 1.11 shows the histogram of the input image and the histogram of the output of applying the filters $[-1, 1]$ and $[-1, 1]^T$.

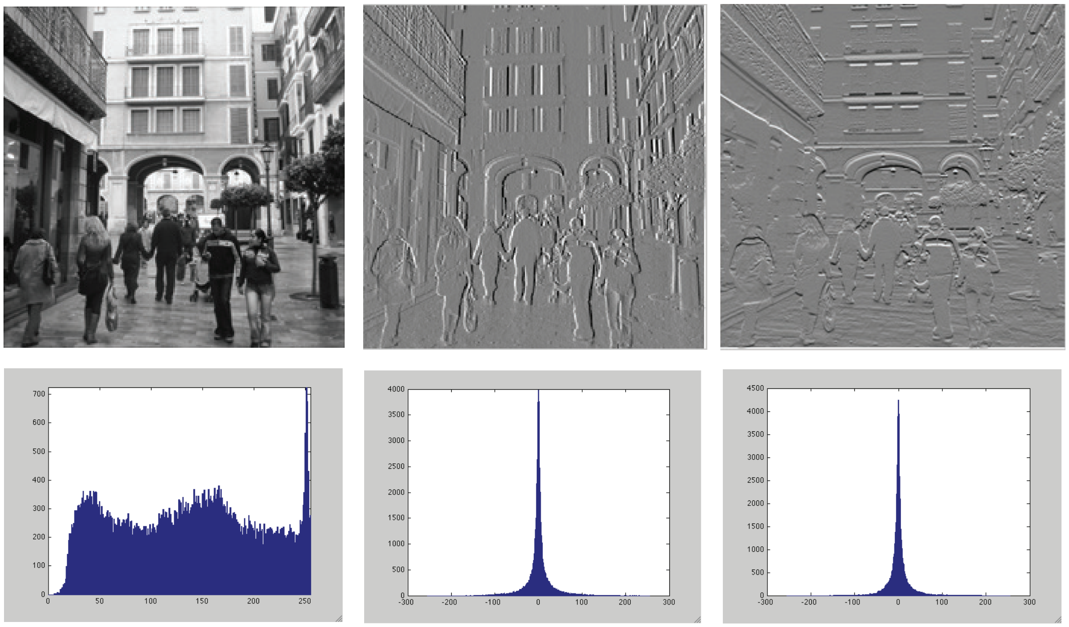


Figure 1.11: Top row: image, and horizontal and vertical derivatives of the image. Bottom row: histogram of each image in top row. Note that the histograms of the two bandpass filtered versions of the original image have characteristic non-Gaussian distributions.

First, the histogram of the image pixels is already non-Gaussian (which already goes against the Gaussian model). However, the image histogram does not have any interesting structure and different images have different histograms. Something very different happens to the outputs of the filters $[-1, 1]$ and $[-1, 1]^T$. The histograms of the filter outputs are very clean (note that they are computed over the same number of pixels), have a unique maximum around zero, seem to be symmetric.

Note that this distribution of the filter outputs goes against the hypothesis of the Gaussian model. If the Gaussian image model was correct, then any filtered image should also be Gaussian as a filtering is just a linear operation over the image. Fig. 1.12 makes this point, showing histograms of bandpass filterings of Gaussian noise, and two other natural images. Note that the image subband histograms are significantly different than Gaussian.

The non-Gaussian distributions of image subbands, $\mathbf{w} = \mathbf{w} \circ \mathbf{h}$, can be parametrized by a simple function:

$$p(\mathbf{w}) = \frac{\exp\left(-\left|\mathbf{w}/s\right|^r\right)}{2s/r\Gamma(1/r)} \tag{1.12}$$

This function is called the Generalized exponential distribution. It has two parameters: the exponent r which changes the shape of the distribution and s which affects the variance.

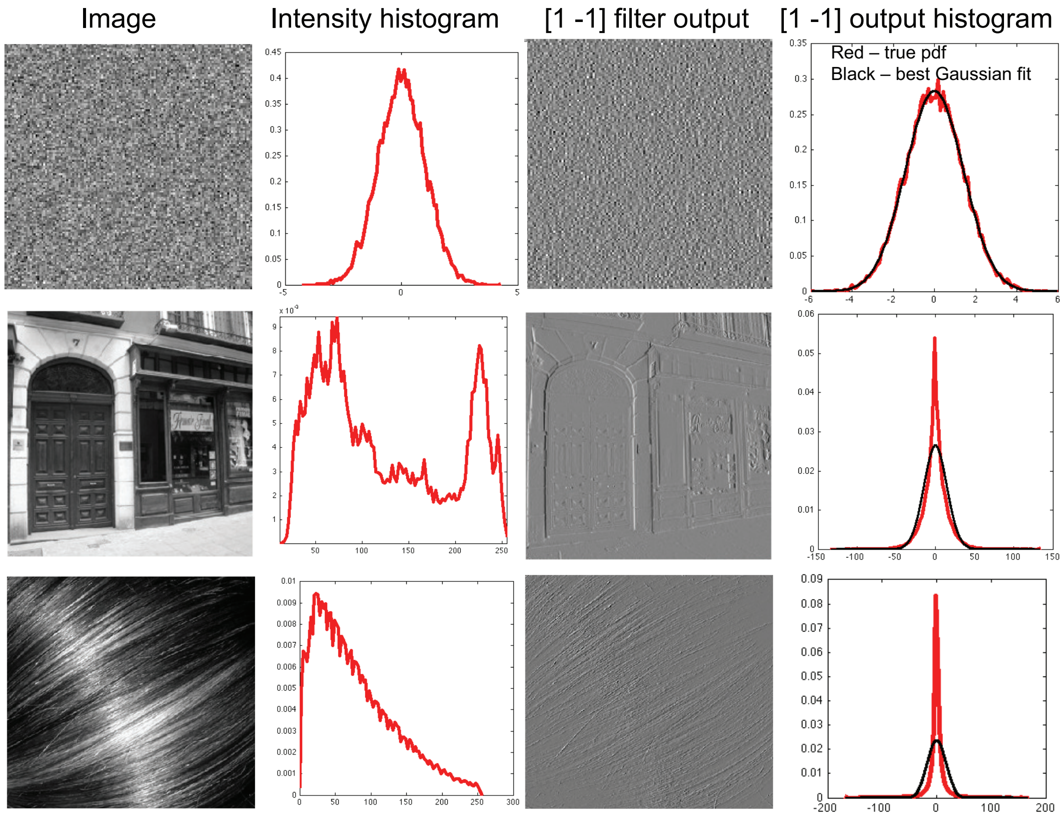


Figure 1.12: Comparison of histograms of images from different visual worlds, and with bandpass filtered versions of those images. Top row: Gaussian noise. Bandpass filtered Gaussian noise is still Gaussian distributed. Middle and bottom rows: Two very different looking images, when bandpass filtered, have similar looking non-Gaussian, narrowly peaked histogram distributions

This distribution fits remarkably well the outputs of bandpass filters of natural images. In natural images, typical values of r and in the range $[0.4, 0.8]$.

Fig. 1.13 shows the shape of the distribution when changing the parameter r. $r = 2$ gives the Gaussian distribution. $r = 1$ is called the laplacian distribution. When $r \to \infty$ the distribution converges to a uniform distribution in the range $[-s, s]$. And when $r \to 0$, the distribution converges to a delta function.

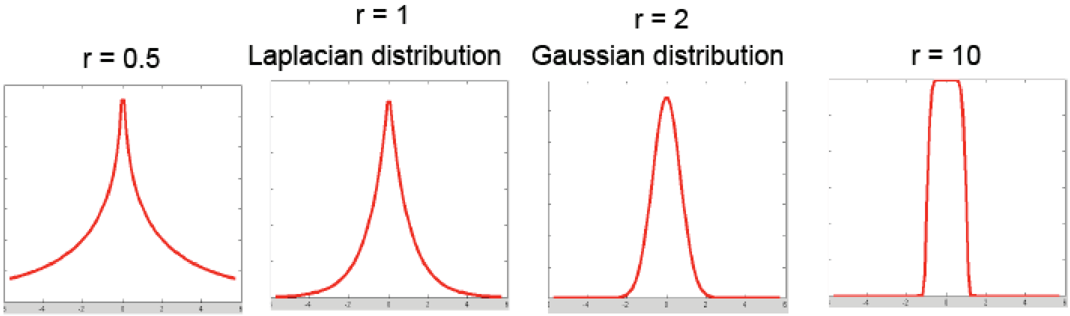


Figure 1.13: Changing the exponent r changes the shape of the distribution generating some special cases (Gaussian, uniform, laplacian).

Now that we have seen that image derivatives have a special distribution for natural images, let's define a prior model that captures the distribution of image derivatives. We can use this prior model in synthesis and noise cleaning applications.

What is the most likely image under the model eq. 1.13? It is a constant image.

$$p(\mathbf{x}) = \prod_k \prod_{n,m} p(h_k(\mathbf{x}, n, m)) \tag{1.13}$$

This model assumes that all the output values of all the filters are independent.

1.4.1 Toy model of image impainting

To build an intuition on what types of image structures are more likely under the wavelet marginal model, let's consider a simple 1D image of length $N = 11$ that has a missing value in the middle:

$$\mathbf{x} = [1,\ 1,\ 1,\ 1,\ 1,\ ?,\ 0,\ 0,\ 0,\ 0,\ 0] \tag{1.14}$$

The middle value is missing because of image corruption or occlusion and we want to guess the most likely pixel intensity at that location. This task is called **image impainting**. In order to guess the value of the missing pixel we will use the image prior models we have presented in this chapter.

We will denote the missing value with the variable a. Different values of a will produce different types of steps. For instance, setting $a = 1$ or $a = 0$ will produce a sharp step edge. Setting $a = 0.5$ will produce a smooth step signal. What is the value of a that maximizes the probability of this 1D image under the wavelet marginal model?

First we need to specify the prior distribution. For this example we will use a model specified by a single filter with the form $[-1, 1]$, and we will use the distribution given by eq. 1.12. In this case, we can write the wavelet marginal model in close form. Applying the filter $[-1, 1]$ to the 1D signal and using mirror boundary conditions, results in:

$$\mathbf{y} = h(\mathbf{x}) = [0,\ 0,\ 0,\ 0,\ 0,\ 1-a,\ a,\ 0,\ 0,\ 0,\ 0] \tag{1.15}$$

Now, we can use Eq. 1.13 and obtain the probability of $\mathbf{x}$ as a function of a:

$$p(\mathbf{x}) = \prod_n p(h(\mathbf{x},n)) = \prod_n p(y[n]) \tag{1.16}$$

$$= \frac{\exp\left(-\left|(1-a)/s\right|^r\right)\exp\left(-\left|a/s\right|^r\right)}{(2s/r\Gamma(1/r))^N} \tag{1.17}$$

$$= \frac{1}{(2s/r\Gamma(1/r))^N}\exp - \frac{\left|1-a\right|^r + \left|a\right|^r}{s^r} \tag{1.18}$$

the first factor only depends on r. The second factor depends on a and r. We are interested in seeing the values of a that maximize $p(\mathbf{x})$ for each value of r. Therefore, only the second factor is important for this analysis. The next figure plots the value of the second factor as we vary a and r. The red line represent the places where the function reaches a maximum as a function of a for each value of r.

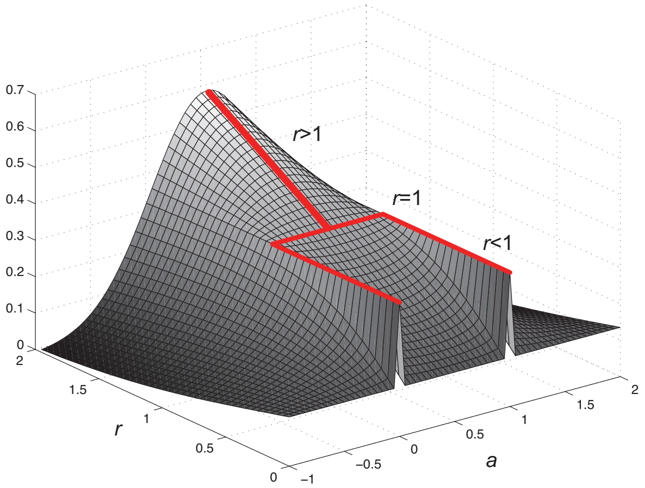
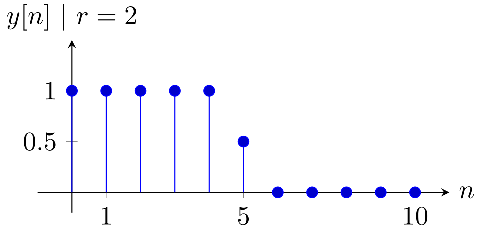
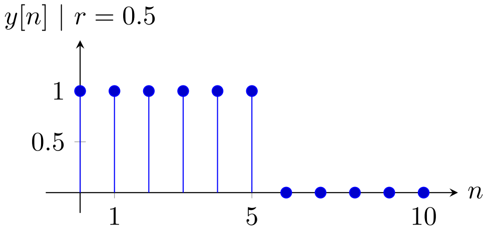


Figure 1.14: Best values of a that maximize the probability of the 1D image $[1,\ 1,\ 1,\ 1,\ 1,\ a,\ 0,\ 0,\ 0,\ 0,\ 0]$ under the wavelet prior model.

For $r = 2$ the best value of a is $a = 0.5$, and this solution is the best for any value of $r > 1$. For $r = 1$, any value $a \in [0, 1]$ is equally good, and $p(\mathbf{x})$ decreases for values of a outside that range. And for $r < 1$, the best solution is for $a = 0$ or $a = 1$. Therefore, values of $r < 1$ prefer sharp step edges. Note that $a = 0$ and $a = 1$ are both the same step edge translated by one pixel. The following plots show the final result using two different values of r:





When using $r = 2$, the result is a smooth signal. For $r = 0.5$, the result is a sharp step edge.

1.4.2 Image synthesis

The statistical characteristics of image subbands allow for powerful image synthesis and also denoising algorithms. Figure 1.15 shows a random draw from the kurtotic wavelet statistical model of images. The long tails of the Laplacian subband distributions results in a small number of high-amplitude wavelet basis functions. For the subband representation used here, the basis functions are x-y separable, giving the large amplitude, large-scale wavelets visible in the image.

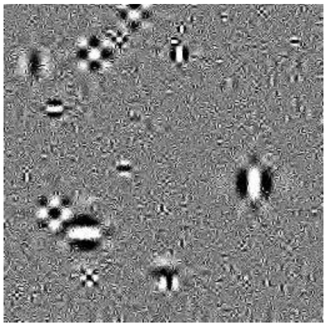


Figure 1.15: Random draw from wavelet model, for an x-y separable model.

Heeger-Bergen Texture Synthesis

An influential wavelet-based image synthesis algorithm is the Heeger-Bergen texture synthesis method, and it's origin story is this [Heeger1995]: David Heeger heard Jim Bergen give a talk at a human vision conference about representations for image texture. Bergen noted the usefulness, for characterizing texture, of measuring the mean squared responses of filters applied to images. He proposed that if that were good, then measuring the probability distribution of each filter's responses would be even better. He proposed that if two images gave the same distribution of filter responses, for all filters, then the textures should look the same to a human observer. Heeger thought that was not true, and sought to prove it to Bergen by implementing Bergen's proposal, using a steerable pyramid [Simoncelli and Freeman1995]. To Heeger's surprise, the first example he tried worked very well, and this work led to Heeger and Bergen's influential joint paper on texture synthesis, [Heeger and Bergen1995].

Figures 1.16 and 1.17 show the main idea. First, as shown in Fig. 1.16, the image is transformed into a representation as the outputs of many oriented filters over different spatial scales. A steerable pyramid [Simoncelli and Freeman1995] can be used. Fig. 1.16 shows the multi-scale representation of [Peyre2010]. The transform should represent all spatial frequencies of the image, so that all aspects of a texture can be synthesized, and the subbands should not be aliased, to avoid artifacts in the synthesized results.

A source texture image is required, and the iterative image synthesis procedure begins with a random noise image. The crux of the algorithm is to alternate matching the intensity domain pixel histogram of the image, then transforming the resulting image into the transform domain and enforcing a match, subband by subband, of the image histograms there. After several iterations of the histogram matching steps in both domains, the algorithm appears to converge and the result is the synthesized texture. The results are often quite good. We note, however, that correlations of filter responses across scale is not captured in this procedure, and long or large-scale structures are often depicted poorly. But for images like the small rocks of 1.17, the algorithm generally works well.

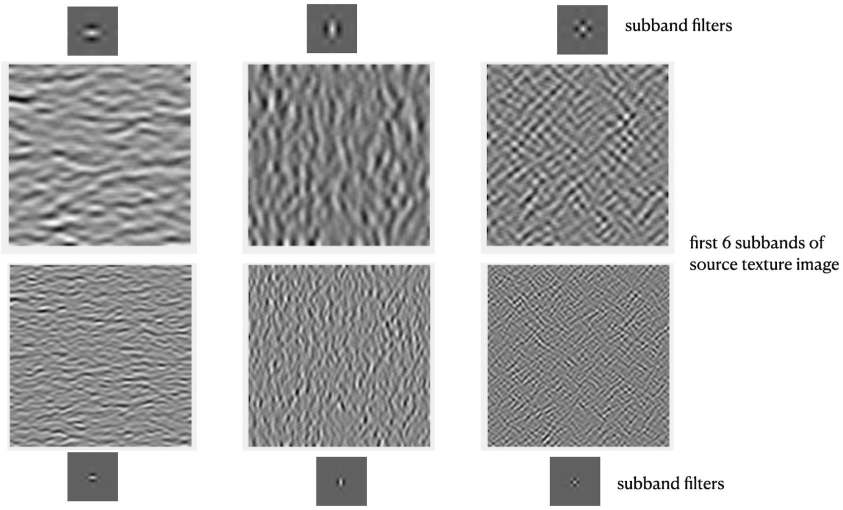


Figure 1.16: The image subbands, and subband filters, for two spatial resolution levels of the source texture image of Fig. 1.17 (bottom left).

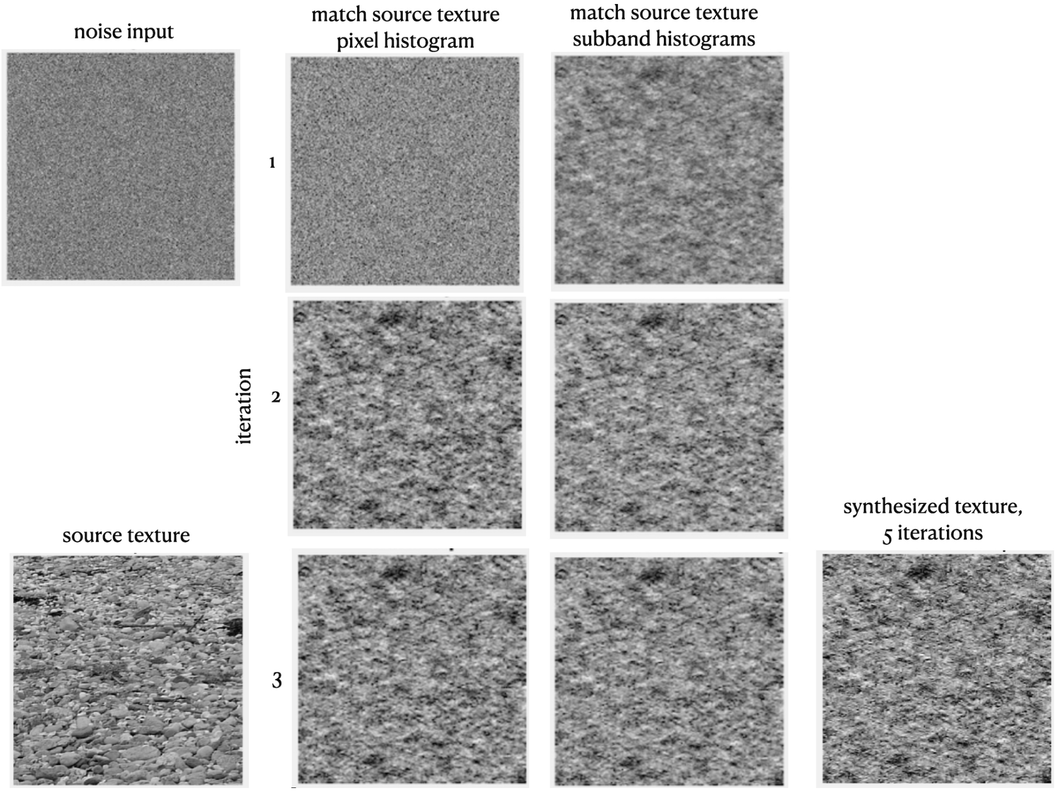


Figure 1.17: The steps of the Heeger-Bergen texture synthesis algorithm.

1.4.3 Image denoising (Bayesian image estimation)

The highly kurtotic distribution of bandpass filter responses in images provides a regularity that is very useful for image denoising. The plots of Fig. 1.18 show how this works. The horizontal axis for each plot is the value at a pixel of an image subband coefficient. The red vertical line shows the observed noisy value, and the red Gaussian curve, labeled "likelihood", is proportional to the probability that the true value had any other coefficient value. If we assume zero-mean additive Gaussian noise of variance σ^2, observed subband coefficient value y, then the likelihood of any coefficient value, w, is

$$P(y|w) = \exp\left(-\frac{(w-y)^2}{2\sigma^2}\right). \tag{1.19}$$

The prior probability of any coefficient value is the Laplacian distribution of the subbands,

$$P(w) = \exp\left(-\frac{|w|}{\sigma}\right). \tag{1.20}$$

By Bayes rule, the posterior probability is the product of those two functions,

$$P(w|y) \propto \exp\left(-\frac{|w|}{\sigma}\right) \exp\left(-\frac{(w-y)^2}{2\sigma^2}\right) \tag{1.21}$$

The black curve shows the posterior for several different coefficient observation values.

Figure 1.19 shows the resulting look-up table to modify the noisy observation to the MAP estimate for the denoised coefficient value. Note that this implements a “coring” function: if a coefficient value is observed near zero, it is set to zero, based on the very high prior probability of zero coefficient values.

1.5 Non-parametric Markov Random Field image models

The Gaussian and kurtotic wavelet image priors both involve modeling the image as a sum of statistically independent basis functions, Fourier basis functions, in the case of the Gaussian model, or wavelets, in the case of the wavelet image prior.

A powerful class of image models are known as Markov Random Field (MRF) models. They have a precise mathematical form, which we will describe later, in Chapter yyy. For now, we present the intuition for these models: If the values of enough pixels surrounding a given pixel are specified, then the probability distribution of the given pixel is independent of the other pixels in the image. To fit and sample from the MRF structure requires the mathematical tools of graphical models, to be introduced in Chapter yyy, but here we present an algorithm, introduced by Efros and Leung, [Efros and Leung1999], that exploits intuitions about the Markov structure of images. While it imposes the Markov structure only approximately, the results of this image generation algorithm are visually compelling. This can be partly attributed to its non-parametric nature, allowing the algorithm to synthesize a rich set of visual details.

1.5.1 Image synthesis (Efros-Leung texture synthesis model)

The Efros-Leung algorithm is recursive. In the recursion step, we have synthesized pixels in the neighborhood of the pixel to be synthesized, see Fig. 1.20. We look through the input image for examples of similar configurations of pixel values, with a sum of squared intensity differences lower than some threshold, ϵ. Randomly select one of the matching instances, and copy the value of the pixel p into the pixel to be synthesized. Repeat for all pixels to be synthesized. It can be natural to synthesize the pixels in a raster-scan ordering. Image

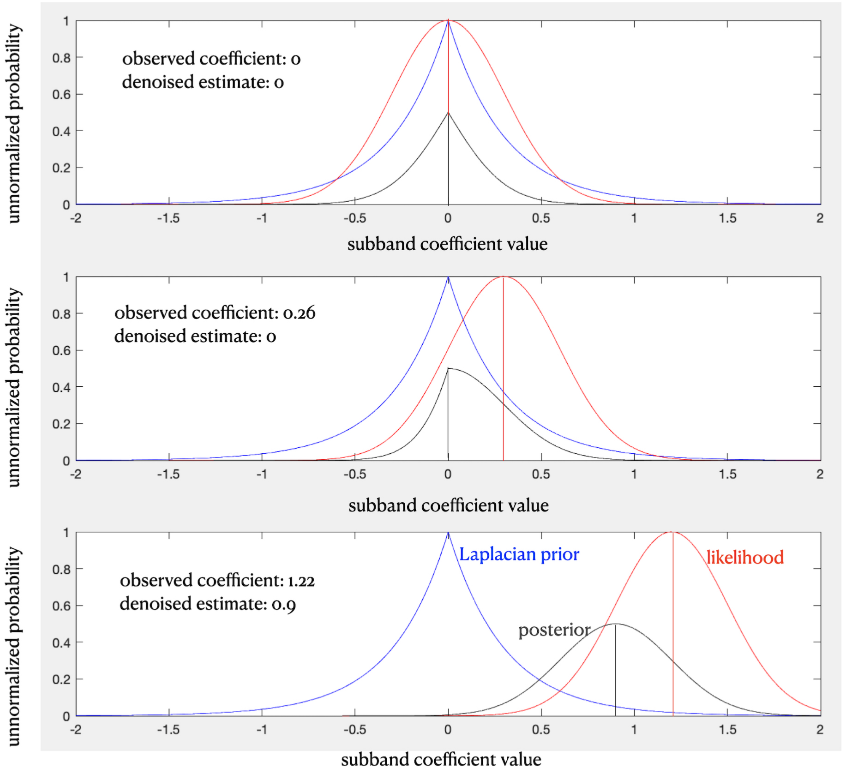


Figure 1.18: Showing the likelihood, prior, and posterior terms for the estimation of a subband coefficient from several noisy observations.

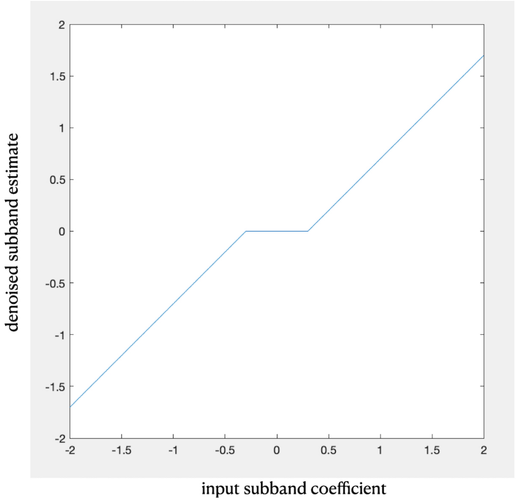


Figure 1.19: Showing the likelihood, prior, and posterior terms for the estimation of a subband coefficient from several noisy observations.

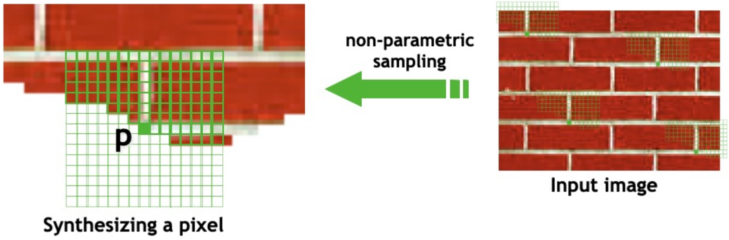


Figure 1.20: Pixel synthesis in the Efros-Leung algorithm

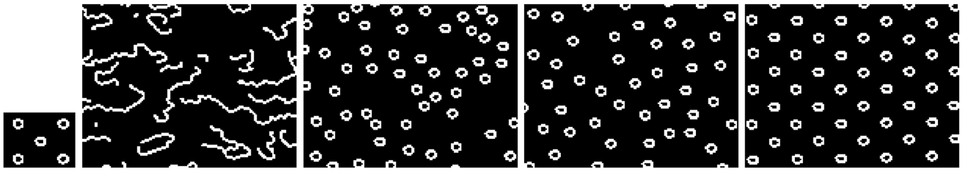


Figure 1.21: Synthesized texture from the Efros-Leung algorithm as a function of patch size.

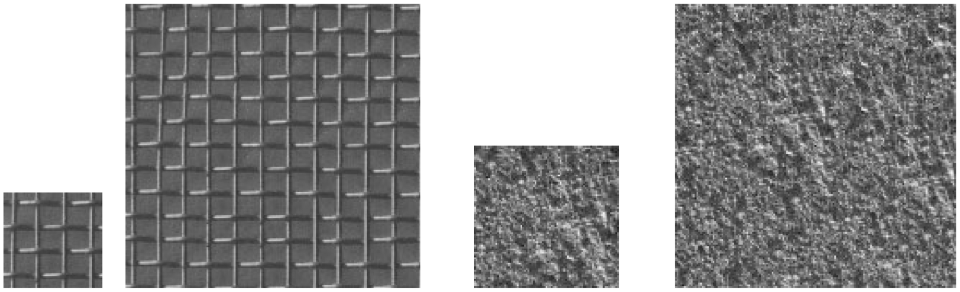


Figure 1.22: Image synthesis results from the Efros-Leung algorithm.

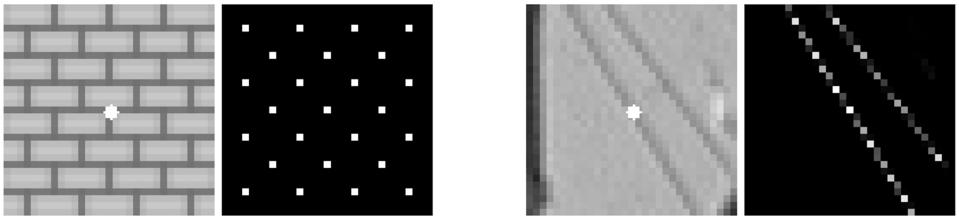


Figure 1.23: Regions of support for the non-local means denoising algorithm

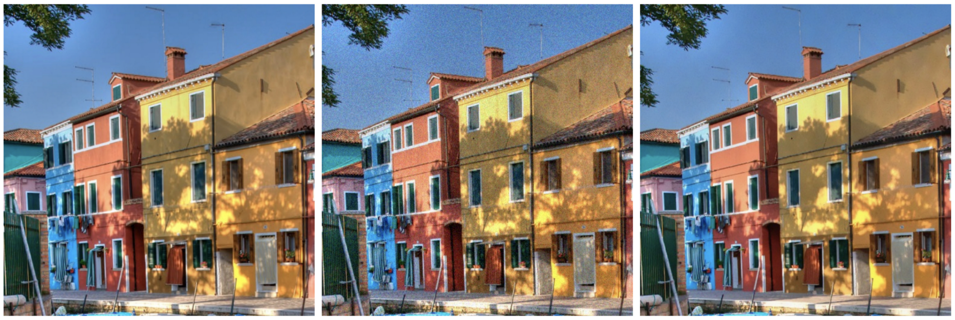


Figure 1.24: Non-local means denoising algorithm results.

pixels without sufficient neighboring pixels for context can be randomly sampled from the input texture.

It is remarkable that such a simple model works so well. Embellishments have been developed which account for structures over scale [DeBonet and Viola1998] or which are more efficient by operating on patches ([Efros and Freeman2001, Barnes et al.2009].

1.5.2 Denoising model (Non-local Means)

The image generation method of efros and leung has a natural denoising counterpart (although it is usually not presented in that way).

1.6 Conclusion

The models presented here do not try to understand the content of the pictures, and the rules about the structure of images is rather limited. However, this models are very powerful when used to solve challenging image processing tasks.

Bibliography

[Barnes et al.2009] Barnes, C., E. Shechtman, A. Finkelstein, and D. B. Goldman. 2009. Patchmatch: A randomized correspondence algorithm for structural image editing. In *Proceedings of acm siggraph.*

[DeBonet and Viola1998] DeBonet, J. S., and P. Viola. 1998. Texture recognition using a non-parametric multi-scale statistical model. In *Proc. ieee computer vision and pattern recognition.*

[Efros and Freeman2001] Efros, A. A., and W. T. Freeman. 2001. Image quilting for texture synthesis and transfer. In *Acm siggraph.* In *Computer Graphics* Proceedings, Annual Conference Series.

[Efros and Leung1999] Efros, A. A., and T. K. Leung. 1999. Texture synthesis by non-parametric sampling. In *Intl. conf. on comp. vision.*

[Heeger1995] Heeger, D. 1995. personal communication.

[Heeger and Bergen1995] Heeger, David J., and James R. Bergen. 1995. Pyramid-based texture analysis/synthesis. In *Computer graphics proceedings*, 229–238.

[Land1983] Land, E. H. 1983. Recent advances in retinex theory and some implications for cortical computations: color vision and the natural image. *Proc. Nat. Acad. Sci. USA* 80: 5163–5169.

[Matheron1975] Matheron, G. 1975. *Random sets and integral geometry.* Wiley.

[Peyre2010] Peyre, G. 2010. Texture Synthesis Using Wavelets. http://www.numerical-tours.com/matlab/graphics$_{2s}ynthesis_{w}avelets/9$.

[R. W. Fleming] R. W. Fleming R. O. Dror, E. H. Adelson. Surface reflectance estimation under unknown natural illumination.

[Ruderman1997] Ruderman, D. L. 1997. Origins of scaling in natural images. *Vision Research* 37 (23): 3385–3398.

[Simoncelli2005] Simoncelli, E. P. 2005. Statistical modeling of photographic images. In *Handbook of image and video processing*, 431–441. Academic Press. Chap. 4.7.

[Simoncelli and Freeman1995] Simoncelli, E. P., and W. T. Freeman. 1995. The steerable pyramid: a flexible architecture for multi-scale derivative computation. In *2nd annual intl. conf. on image processing.* Washington, DC. IEEE. IEEE.