Chapter 1

Texture modelling for noise removal [DRAFT]

(draft as at 10 February 2004 of a chapter of Paul Harrison's PhD Thesis)

The values of pixels in an image may, for various reasons, be known only imprecisely. If so, the texture of that image may be used to help estimate the true pixel values. Texture is the interrelation of pixels in an image; where there is texture, the value of one pixel may give clues as to the values of other pixels. This chapter describes how to find the most likely pixel values of an image where only upper and lower bounds on these values are known, using a texture model.

Imprecise knowledge of an image usually takes the form of an image with additive noise. This noise might be generated as a result of storing an image in a lossy format such as JPEG or GIF, introduced while converting it from analogue to digital form, or produced by an analogue device. The amount of noise may depend on the position in the image, or even the content of the image. Restoration of images stored in lossy formats is of particular interest, given the large number of these images that now exist (for example, on the World-Wide Web).

The novel feature introduced in this chapter, as compared to other methods

of noise removal, is the use of texture models tailored to individual images. The kind of textural details seen in an un-degraded part of an image may reasonably be expected to occur in degraded parts. Texture that occurs in the form of large variations may also occur in small details occluded by noise. These ideas can be expressed in terms of texture modelling, allowing more accurate restoration of images.

To attempt removal of additive noise from an image, two things must first be stated:

1. The set of possible values for each pixel, and how likely each of these values is (in other words, the amount and nature of noise).
2. The ways in which pixels may be related to one another (in other words, what kinds of texture the image may have).

Given these two statements, the most likely image and image texture may be found. The way the two statements are made is crucial. They must be precise enough that the result is useful, but must also be stated in a way that affords computation of the result in reasonable time. The difficulty lies in finding a useful trade-off between these two constaints. A variety of approaches have been proposed.

Linear filtering is a straightforward approach. It derives from stating that elements in the Fourier transforms of the image (signal) and the uncertainty about the image (noise) will have Gaussian distribution. The elements' variances (the expected "spectral density") are estimated, and from these an optimal "Weiner" filter is derived [Helstrom, 1990]. This filter is always linear. In the case of additive noise, it might be estimated that the image does not contain high frequencies (is smooth), while the noise is white. This produces a low-pass filter.

A low-pass filter will soften any sharp edges in an image, so this method is of limited use for additive noise. The limitations of linear filtering led to investigation of non-linear filters.

Non-linear approaches vary in complexity and theoretical rigour. "Selective Gaussian Blur" is an example of a simple but effective non-linear filter [van

Os, 1999]. It is based on Gaussian blurring, but preserves sharp edges. The input image is blurred, and the blurred image compared to the input. Where the blurred image is close in value (within a specified range) to the input, the blurred pixel value is used, otherwise the input pixel value is used. This is an ad-hoc but useful variation on linear filtering. It is also fast.

A more sophisticated method based on minimizing the "Total Variation" was developed for the US military [Rudin et al., 1992]. The Total Variation method models the texture of the image as tending to have small gradient at each point. The magnitude of the gradient is assumed to have an exponential distribution. Noise is modelled as being Gaussian. Finding the most likely true image involves minimizing a norm similar to L_1 subject to the constaint that the noise removed has an apriori specified variance.

The texture model of the Total Variation method, by using an L_1-like norm, rates smooth gradients and sharp edges as being equally likely. Consequentially, the Total Variation method preserves sharp edges in an image. A texture model that made a more traditional assumption of a Gaussian distribution of gradients would prefer smooth gradients to sharp edges. The Total Variation method is reported to beat the human eye in extracting the features of a noisy image.

Specialized methods have been designed to remove the noise introduced by JPEG compression [Zakhor, 1992, Yang et al., 1993, Llados-Bernaus et al., 1998, Meier, 1999]. JPEG breaks an image into 8×8 pixel blocks and performs a Discrete Cosine Transform on each block. The output of the transform is quantized and stored concisely. When decoding JPEG images, there is uncertainty as to the true value of each element in the block transforms due to quantization. JPEG decoders assume that each element is independant, and for each element minimize the expected error by choosing the centre of the range of possible values. This is a reasonable assumption, as the purpose of the DCT is to decorrelate each 8×8 block. However the DCT ignores correlations between adjacent blocks, and the decoder may produce visible block borders if the quantization level is too high.

The method of Llados-Bernaus et al. [1998] is typical of JPEG restoration

methods. As in the Total Variation method, this method minimizes a norm subject to constraints. The image is taken to be more likely to be smooth than rough, including across block boundaries. Therefore, differences between neighbouring pixels are expected to be small. Penalties are imposed on each difference by the use of a "Huber function", and these penalties are added to give a measure of how unlikely a particular reconstruction is. Huber functions are a compromise between the L_2 and L_1 norms, combining the smoothness of L_2 with the robustness of L_1. They correspond to a Gaussian distribution of values with fat tails. The pixel values are constrained such that the DCT transform elements, when quantized, match the file being decoded.

All of the approaches to noise removal above treat the texture of an image simply. They assert that the image is more likely to be smooth than rough in texture, and to possibly contain edges. No attempt is made to adapt to the texture of individual images. Consequentially, all methods in this paradigm will always reduce the amount of variation in an image. For example, they will all always dull reflective highlights.

The above approaches may be improved apon by tayloring a texture model to each image. This chapter considers the following choice of statements:

1. Each pixel values lies within a specific range, but is equally likely to take any value within that range.
2. The image has texture that may be described in terms of causal prediction.

The next two sections detail and justify this choice.

1.1 Pixel values

Each pixel value is taken to lie within a specific range, but to be equally likely to take any value within that range. This choice does not compete with the texture model to explain small variations, where noise will mean

that almost nothing is known anyway. However it totally rules out consideration of large changes to pixel values, so the texture model need not model large changes, such as edges, with total accuracy.

The pixel ranges may be unbounded on one or both sides. For example, a photograph may have been over-exposed when it was taken. Within regions of over-exposure it is known only that the true pixel values exceed a certain amount. There may also be artifacts such as spots obscuring parts of the image. A pixel within this kind of area may be considered to be equally likely to take any value. This is similar to the technique for removing objects from images in Chapter **??**, but unlike the technique of Chapter **??** it allows extrapolation of texture.

1.2 The texture model

A texture model defines the likelyhood of an image, given that it has a certain texture. Let I be an image with texture model M. The most likely texture and image are sought, $P(M \cap I)$ is to be maximized:

$$P(M \cap I) = P(M)P(I|M) \tag{1.1}$$

If it is assumed each model is equally likely, $P(M)$ is constant and maximizing $P(I|M)$ will maximize $P(I \cap D)$. This is the well known Maximum Likelyhood method.

If an ordering of pixels in the image is chosen (for example, raster scan order), $P(I|M)$ may be written in terms of the conditional probability of each pixel taking a certain value, given the preceding pixel values and the texture model:

$$P(I|M) = \prod P(\text{pixel}_i|M \cap \text{pixel}_0 \cap \text{pixel}_1 \cap \cdots \cap \text{pixel}_{i-1}) \tag{1.2}$$

A way to compute the likelyhood of a pixel value given the preceding pixels

values in an ordering therefore constitutes a texture model.

The predicted distribution of pixel values is taken to be Gaussian with constant variance. With this assumption we need only estimate the mean expected value of each pixel, and the Maximum Likelyhood method becomes equivalent to the method of Least Squares (the derivation of this equivalence is given in Press et al. [1992, pp. 657–659]). Least Squares requires minimization of the sum of squared differences between a set of predicted and actual values. This sum is called a "merit function".

The specific predictor used in this chapter is a weighted sum of a set of values computed from the anti-causal neighbours of the pixel, with causality defined by a raster scan ordering. The values might simply be the values of the pixel's anti-causal neighbours (producing a linear predictor). Other values might result from multiplying several values together, or applying some non-linear function to a value, or combining values in some other way. With sufficient values, any function of the anti-causal neighbours may be approximated to arbitrary accuracy with a weighted sum such as this.

Let $V_1(s) \ldots V_n(s)$ be the set of values used for prediction of the pixel $I(s)$, and $w_1 \ldots w_n$ be the corresponding weights. Then the merit function is:

$$\text{Merit} = \sum_{s \in \Omega_I} (I(s) - w_1 V_1(s) - w_2 V_2(s) \cdots - w_n V_n(s))^2 \qquad (1.3)$$

The weights of the model w_i and the pixel values $I(s)$ (within the specified bounds) are chosen so as to minimize this merit function.

1.3 Optimization algorithm

The merit function may be minimized numerically. An arbitrary initial image is chosen, then a two step process is followed: first an optimum model is found given the current image, then an improved image is found given that model. This process is iterated.

In the first step, the optimum model for the current image may be found precisely. When minimized, the merit function will have zero partial derivative with respect to each model weight. Each partial derivative is a linear function of the model weights. Equating each derivative to zero gives a set of linear equations. These may be solved to find the optimum model.

In the second step the optimum image is approximated numerically, as finding the optimum analytically is difficult. A multigrid method may be used, as described in Press et al. [1992, pp. 871–887]. First a scaled down image of the input is optimized, then successively larger versions. At each scale the optimization is performed via gradient descent.

The multigrid method is necessary so that large features can be optimized within a reasonable time. For example, gradients that have been converted to a series of steps by palettization (as in Figure 1.3) would take a long time to optimize at maximum scale.

Three images undergo scaling: an image of the minimum possible values, an image of the maximum possible values, and the current best estimate of the optimum image.

The best-estimate image is is scaled up by means of bi-linear interpolation, and scaled down by its adjoint operator, as in Press et al. [1992, pp. 875–876].

The minimum and maximum images only ever need to be scaled down, as they are never modified. When scaling down the minimum image, each pixel in the scaled down image takes the maximum of the nine pixels it overlaps in the original image (as illustrated in Figure 1.1). Similarly each pixel in the scaled down maximum image takes the minimum of the nine pixels it overlaps in the original. Where the value of a scaled down minimum pixel exceeds the value of its corresponding scaled down maximum, both are replaced by the value of the corresponding scaled down best-estimate pixel.

This method of scaling ensures that pixels in the best-estimate image almost never scale up to a value outside of the bounds posed by the minimum and maximum images. While this method is ad-hoc it should not affect the outcome, as the final step is to optimize the image at its original scale.

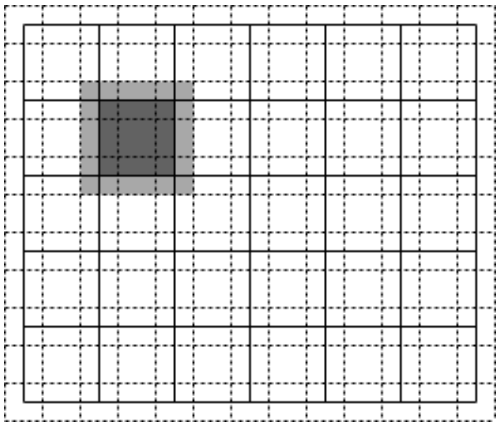

Figure 1.1: Grid positions when scaling an image. A pixel in the coarser grid has been highlighted in dark gray, and the pixels it overlaps in the finer grid highlighted in light gray.

Reflection is used at the edges of the image. If the image is tilable, tiling may be used instead.

1.4 Results

The standard "Lena" image [Sjööblom, 1972] was used to test the method. Four different versions of Lena were used:

A. Lena encoded with a palette of only 16 gray levels. The levels were evenly spaced and the image was not dithered.

B. Lena over-compressed using JPEG, with only 0.385 bits per pixel.

C. Lena with simulated over-exposure. Pixel values exceeding three-quarters of the maximum brightness were clipped.

D. The unmodified Lena image, which contains a small amount of noise. This may be from the grain of the original photograph, the printing process, the scanning process, or some combination thereof.

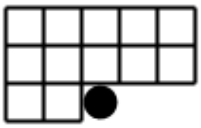

Figure 1.2: Neighbours of a pixel used by the predictor.

The texture model used was based on the twelve neighbours shown in Figure 1.2. The values used by the predictor were: a constant, the value of each neighbour, and the result of multiplying the values of each possible pairing of the neighbours. This set of values allows the predictor to take the form of any quadratic function of the neighbours' values. The predictor had a total of 91 terms.

In case A, the true pixel values could vary from those of the image by up to $\frac{1}{32}$ of its total dynamic range. The input and result are shown in Figure 1.3. As can be seen, the distinct levels of the input image have been smoothed into continuous gradients. With the exception of some fine texture in the hat, the detail of the original image has been well preserved.

In case B, the distortion is especially visible near the edges of the 8×8 Discrete Cosine Transform blocks used by JPEG. The range of variation was therefore set higher in the edge pixels of each block than the other pixels, and higher still in the corner pixels. Pixels in the corners were allowed to vary by up to 12% of the image's total dynamic range, pixels on the edges were allowed to vary by 8%, and the remaining pixels 4%. The result is shown in Figure 1.4. In the result, the "blockiness" of the input has been largely removed, without degrading the image significantly.

In case C, the uncertainty is in the value of the pixels in the over-exposed regions. In these regions the pixels could originally have had any value brighter than their current value. Results are shown in Figure 1.5. The reconstruction is less than perfect. As large areas of the picture were obliterated by the simulated over-exposure, this is not surprising. The shoulder has been reconstructed approximately, but with some strange diagonal banding that appears to mimic the hat's texture. The over-exposed sections of the hat have not been reconstructed convincingly,

	Input image	Result
A	34.29	34.93
B	33.73	33.74
C	34.41	38.05
D	∞	39.54

Table 1.1: Peak-Signal to Noise Ratio in decibels, as compared to the original Lena image.

but are some improvement on the original. The reconstruction of a fine light edge about the bottom and rear of the hat is a notable success (see the detail image), as is the reconstruction of Lena's left eye.

In case D, the error was assumed to be small and constant ($\frac{1}{30}$ of the image's dynamic range). The result (Figure 1.6) has an air-brushed, "cover-girl" quality. Much of the noise has been removed. The fine detail in the hat has been degraded, but the image is otherwise sharp.

The Peak-Signal to Noise Ratio for each case is listed in Table 1.1. Though only an approximation of the perceived image degredation [Bhaskaran and Konstantinides, 1995, pp. 9], the PSNR is a simple and objective measure. The PSNR improved slightly for image A, and substantially for image C. Image B was not improved significantly, possibly due to systematic errors in the DC coefficients of the compression blocks; the method can not reconstruct these errors, but they are also unimportant perceptually.

1.5 Discussion

Previous methods of noise removal have used simple models of texture, preventing extrapolation of features such as highlights and even reducing the existing features in restored images. The method given here has been shown to not have this limitation, with features such as highlights recreated. Sharp edges were also preserved. Increasing the accuracy of the texture model should allow restoration of other feature types.

While more detailed than previous models, the texture model used here is not fully general. For full generality, the texture model would need to be able to predict arbitrary probability distributions. Specification and optimization of a model this general is difficult. Lesser generalizations are possible, including use of more terms in the predictor, prediction of the standard deviation of each pixel in addition to the expected value, and use of robust merit functions based on L_1 or Huber functions (as in Rudin et al. [1992] and Llados-Bernaus et al. [1998]).

The most likely texure model and image were found, but this may not represent well the set of likely images and models. For example, the image produced is almost always smoother than average. It may be better to choose a random sample from the possibility space, or average over all possibilities (weighted by likelyhood). To this end, a set of random sample images might be produced by the Metropolis-Hastings algorithm or one of its variants (see Robert and Casella [1999]).

An image restoration technique that can invent details is slightly dangerous. In places where restoration must be reliable, such as medicine or the military, the method should only be used with care. However, a technique that tailors itself to an image to better restore its specifics may provide enhancements otherwise impossible, which could be crucial in these places.

If compression artifacts can be removed as they were in this chapter, is lossy compression easy? The standard assumption has been that lossy compression requires a detailed model of human perception, taking into account phenomena such as masking. But what if compression artifacts arise because the decoder produces implausible images that lack internal textural self consistency, and that human perception is simply sensitive to this? Were decoders to produce more plausible images given limited information, detailed modelling of perception might become unimportant. The lossy aspect of lossy compression might reduce to a simple form of quantization, and the well-studied techniques of lossless compression could be applied. As further evidence, see the author's lossy audio compression software [Harrison, 2001], which contains 33kB of source code only.

The method described in this chapter represents a generic method for filling in gaps where a set of data has missing elements or is known only

imprecisely. It might, for example, be transferred to the audio domain to reconstruct noisy or clipped recordings, or applied to other time series data. Combined with Metropolis-Hastings sampling, it may be a useful statistical technique for filling in missing data or for prediction.

Figure 1.3: Input and result for test case A.

Figure 1.4: Input and result for test case B.

Figure 1.5: Input and result for test case C.

Figure 1.6: Input and result for test case D.

Bibliography

V. Bhaskaran and K. Konstantinides. *Image and Video Compression Standards*. Kluwer Academic Publishers, 1995.

P. F. Harrison. Bonk audio compression software. WWW: `http://www.logarithmic.net/pfh/Bonk`, 2001.

C. W. Helstrom. Image restoration by the method of least squares. In *Selected Papers on Digital Image Processing*, pages 619–625. SPIE Optical Engineering Press, 1990.

R. Llados-Bernaus, M. A. Robertson, and R. L. Stevenson. A stochasitic technique for the removal of artifacts in compressed images and video. In *Signal Recovery Techniques for Image and Video Compression and Transmission*, pages 35–68. Kluwer Academic Publishers, 1998.

T. Meier. Reduction of blocking artifacts in image and video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(3): 490–500, 1999.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipies in C, 2nd edition*. Cambridge University Press, 1992.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, 1999.

L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

L. Sjööblom. Swedish accent. *Playboy*, 11:135–141, 1972.

BIBLIOGRAPHY

T. van Os. Selective Gaussian blur. WWW: `http://thom.best.vwh.net/gimp/`, 1999.

Y. Y. Yang, N. P. Galatsnos, and A. K. Katsaggelos. Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images. *IEEE Transactions on Circuits and Systems for Video Technology*, 3(6):421–432, 1993.

A. Zakhor. Iterative procedure for reduction of blocking effects in transform image coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 2(1):91–95, 1992.