IS&T/SPIE Symposium on
Electronic Imaging: Science
and Technology, February 1995

Fixed-Window Image Descriptors for Image Retrieval

Leonidas J. Guibas, Brian Rogoff, and Carlo Tomasi
Computer Science Department
Stanford University
Stanford, California 95305

ABSTRACT

We work towards a content-based image retrieval system, where queries can be image-like objects. At entry time, each image is processed to yield a large number of indices into its windows. *A window* is a square in a fixed quad-tree decomposition of the image, and an index is a fixed-size vector, called a descriptor, similar to the periodograms used in spectral estimation. The fixed decomposition of images was prompted by the need for fast processing, but leads to windows that often straddle image regions with different textural contents, making indices less effective. In this paper, we investigate different definitions of spectral distance which we plan to use to classify windows according to their texture content.

Keywords: vision, image databases, texture

1 Introduction

In our research, we work towards a content-based image retrieval system, where the queries themselves can be image-like objects.[TG94] Related research directions are overviewed in.[RJ92] At entry time, each image is processed to yield a large number of indices into all its windows. In general, a window is a rectangle of the image, and an index is a vector, called a descriptor, that summarizes various attributes of the image intensity distribution within the window.

Using indices into images avoids searching the entire database at image query time. In fact, a query interface lets the user select descriptors by picking textures, colors and other attributes out of a set of menus, and by navigating in the parameter spaces of these attributes by means of cursors, somewhat in the style of IBM's QBIC system.[FBF+94] Once a set of descriptors has been computed from the user's selection, or from other images or even from sketches done by the user, search can be quickly performed through a system of hash tables.

In order to keep the image entry procedure relatively fast, windows are the elements of a fixed decomposition of each image into a quad-tree. However, a fixed decomposition leads to windows that can straddle image regions with different textural contents. These "mixed texture" windows generate descriptors that do not correspond by themselves to any one texture, with the effect of cluttering the space of all descriptors. In principle, this problem could be addressed by windows of variable shape and size. However, determining these variable windows seems to require the solution to problems, like region segmentation, for which only partial and usually expensive solutions have been proposed. These high computational costs are sometimes justified in image analysis, but not in the image database setting, where large numbers of images must be processed and query response time must be made interactive.

In contrast, we choose to explore the limits and possibilities of the fixed-window approach, and restrict the use of descriptors to image regions that are large enough to contain windows with homogeneous texture content. This general problem raises the following issues:

- What is a descriptor?
- When are the descriptors of two windows similar enough to be considered to contain the same texture?

The last question implies the definition of a distance between window descriptors that is at least approximately invariant with respect to the natural variations within a given type of texture and to the changes in illumination and viewing conditions present in a single image or in perceptually similar images. In other words, we would like the descriptors of two patches to be close to each other if they contain the same texture, and to be distant otherwise. Similar windows can then be summarized by a single descriptor, and windows that contain dissimilar subwindows can be tagged as having mixed textures. Even more importantly, descriptors can be extracted from the user's query, be it a texture selected from a menu or an example from another image.

Another requirement for a good window descriptor is a high degree of detail, since it is not known at image entry time, that is, when image descriptors are computed, just what feature of a texture the user is interested in. In other words, the descriptor should throw away little information. This requirement is in conflict with that of invariance. In fact, in general, the more information the descriptor retains about a given window, the more likely it is that this descriptor will differ from those of windows that a human observer would consider of the same texture, but perhaps at a different scale, rotation, or position. For instance, if a descriptor of grass specifies generally vertical orientation for the blades, grass in a rotated picture would not be found.

Hard as this dilemma may be, finding descriptors that approximate these requirements is an easier problem than texture-based image segmentation. In fact, homogeneous windows can be found by requiring a conservatively high degree of uniformity, and the exact boundaries of different texture regions are not important. All we need is to find windows of fairly uniform texture. It does not matter much that a region with the same texture is broken into more regions, or that a few good windows are discarded for being erroneously considered nonuniform.

In Section 2 we propose a definition of window descriptor that is in a sense intermediate between the output of a bank of filters run on the window and the window itself. Our choice is a slight variation of the classical periodogram,[OS75] that is, a smoothed and normalized version of the magnitude of the Fourier transform. With some impropriety of language, we call this the *spectrum* of the window[1]. Taking the magnitude achieves invariance to position, while smoothing has two effects: one is a certain degree of invariance to small geometric distortions, and the other is a greater efficiency and uniformity of the representation. In fact, once suitably smoothed, the spectrum can be sampled down to a small and possibly fixed size.

The key issue of spectral distance is then addressed in Section 3, where we compare three different definitions of distance. The first is based on a Hausdorff-like[HK92] metric on the most significant components of the spectrum. Our definition penalizes variations in the locations of the most significant spectral components on the frequency plane, but is permissive with regard to changes in the amplitudes of the spectral components. The second and third definition have an opposite behavior: they are both tolerant, but in different degrees, of variations in the position of the spectral components, but penalize amplitude differences. The latter two measures are different variations of the Sum-of-Squared-Differences distance[Ana89] used for feature tracking. We give efficient algorithms for each distance definition in Section 3. Then, in Section 4, we show some preliminary experiments that compare these three distance definitions on real images. We conclude in Section 5 with limitations of our approach and plans for future work.

[1]The spectrum is the square of the magnitude.[OS75]

2 Spectral Descriptors

In this section, we define our particular choice of spectral description of an image window, and relate it to texture characterization. Spectral estimation is an old science (see for instance[OS75] for the basic theory), so our definition is nothing new. We include it in order to motivate our choice, and to point out a few small differences with respect to the standard definitions.

The output of a set of filters, possibly followed by some nonlinear operation, is often used in the literature on texture discrimination (see, for instance,[Tur86MP90] and many others). This description, although biologically plausible, is not sufficiently detailed, unless a very large number of filters is used (for instance, in,[MP90] 192 separate channels are used). On the other extreme, the intensities of the image window are as complete a descriptor as possible of the window itself. However, this descriptor also varies wildly with viewing transformations, lighting, and noise. The continuum between these two extremes is easily visualized in the frequency domain. In fact, while at one extreme the discrete Fourier transform of a window contains exactly the same information as the window itself, a filter, on the other end, yields as its output a weighted average of some samples of the Fourier transform. Our choice of a smoothed spectrum is in a sense intermediate between the two, since smoothing amounts to a weighted average of samples in every neighborhood of the spectrum, and the effect of smoothing vanishes as the smoothing kernel is narrowed to an impulse.

More precisely, given a window W of intensities $I(\mathbf{x})$, we describe it by the most significant components of a smoothed, normalized, square-root spectrum with its dc component removed. That is, we first let

$$S_W(\mathbf{f}) = \left| \sum \sum I(\mathbf{x}) g(\mathbf{x}) e^{j 2\pi \mathbf{f}^T \mathbf{x}} \right| \tag{1}$$

where $g(\mathbf{x})$ is a wide Gaussian window. This windowing, as will be illustrated below, is equivalent to smoothing the Fourier transform of $I(\mathbf{x})$, but is computed more efficiently. We then normalize $S_W(\mathbf{f})$ and restrict it to its p-th percentile most significant components, that is, to those frequency samples that that are greater than p percent of all the samples in the spectrum, and we normalize the result. Let $\alpha_W(p)$ be the p-th percentile of the values in $S_W(\mathbf{f})$, that is, let p percent of the spectral components be below $\alpha_W(p)$. Then, we define a p-th percentile *significance mask* as follows:

$$m_W(\mathbf{f}) = \begin{cases} 1 & \text{if } S_W(\mathbf{f}) > \alpha_W(p) \text{ and } \mathbf{f} \neq 0 \\ 0 & \text{otherwise} \end{cases} . \tag{2}$$

Notice that this mask zeros the dc component of the spectrum. The window descriptor is then

$$s_W(\mathbf{f}) = \frac{m_W(\mathbf{f}) S_W(\mathbf{f})}{\sum \sum m_W(\mathbf{f}) S_W(\mathbf{f})} . \tag{3}$$

Equations (1) through (2) together comprise our definition of spectral descriptor. We now analyze this definition in some detail.

Removing phase information discards potentially useful information. For instance, figure 1 shows a texture from Brodatz's album[Bro66] on the left and the corresponding texture without phase information on the right. The texture without phase looks more random than the original, and consequently perceptually different: some potentially important information has been discarded. This problem can be addressed by preserving phase information after removing the best-fit plane to the phase, for position invariance, and smoothing, as for the magnitude information. However, we do not pursue this direction in this paper.

Periodograms are usually defined[OS75] in terms of the *squared* magnitude of the Fourier transform of the signal, which is the Fourier transform of the signal's autocorrelation function. This square is important when power considerations must be made, but is irrelevant for our purposes.

To reduce the variance of random fluctuations in the spectral components, periodograms are usually computed as an average spectrum over many subwindows of the given window W. However, in our application, windows are

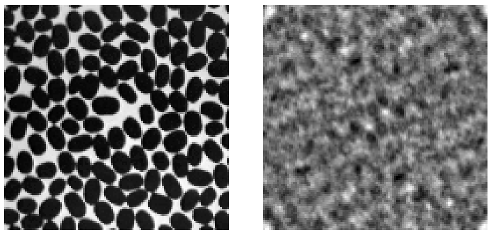

Figure 1: Texture (left) created by beans and the corresponding texture without phase information (right).

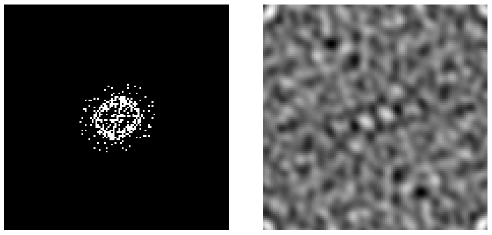

Figure 2: The support (left) of the 2 percent of most significant frequency components from the texture in the right part of figure 1 and the texture (right) obtained by retaining only these components.

often small, while this averaging is of any significance only for much larger signal regions. Our spectrum is also normalized in order to reduce the effect of brightness variations. Furthermore, the dc (zero frequency) component of the spectrum is ignored (see equation (2)). This component is much larger than the others for nonnegative signals (like images are), so it would swamp the rest of the spectrum in any comparison. Its removal is equivalent to subtracting the average image intensity within the window, which further reduces the effects off illumination changes.

Only the most significant components of the spectrum are preserved. Figure 2 illustrates the point that most of the texture information is carried by a very small collection of frequency samples. The left part of figure 2 shows the fill pattern of the matrix that contains the frequency components above the 98-th percentile for the right picture in figure 1, that is, the locations of these components on the frequency plane (the origin is at the center). The right part of figure 2 shows the texture that is obtained by retaining only these components. Although some sharpness has been lost, a very substantial part of the original texture information is carried by only 2 percent of the frequency samples. A different, but related, compression of spectra was proposed in,[PL94] where the spectrum is first decomposed in to its periodic, directional, and random components. Figure 3 indicates that the shape of the significance mask of equation (2) can be used as an approximate signature of a particular texture, a fact that will be used in the first spectral distance definition of section 3.

Finally, the spectrum is smoothed. This is equivalent to windowing the image intensities (the Fourier dual of convolution is windowing). Windowing is common practice in spectral estimation, and the particular choice of window shape is of minor significance. In an efficient implementation, intensity windowing is preferable to spectral smoothing.

3 Spectral Distance

Even without phase information and after normalization, smoothing, and dc-component suppression, the spectrum is not an invariant window descriptor. In fact, viewing deformations like those inherent in perspective views of a slanted surface can deform the spectrum substantially. How can two spectra be compared to each other

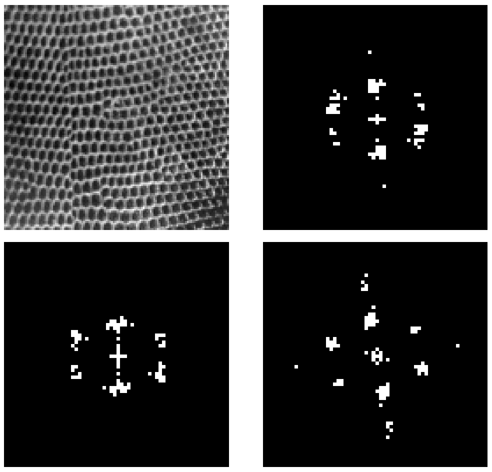
Figure 3: A reptile texture (top left) from[Bro66] and the 98-percentile spectral supports of three of its subwindows. Notice that the bottom right support is slightly rotated with respect to the other two.

in a way that is forgiving of these deformations, but at the same time preserves the distinctions between spectra that cannot be related by a simple deformation?

In this section we address this question by comparing three different metrics for spectral comparison. The first is based on a variation of the Hausdorff distance[HK92] on the high-percentile spectral supports like those in figure 3, and the other two are based on the Sum of Squared Differences (SSD) distance (see for instance[Ana89]), with variations that allow for some flexibility in the position of a spectrum's components on the frequency plane. We then consider combinations of these metrics, and look at one way to set thresholds between "small" and "large" distances.

3.1 A Hausdorff-Like Distance

Our first attempt at defining spectral distance is based only on the binary masks $m_W(\mathbf{f})$ mentioned in Section 2. As already mentioned there, a very substantial part of the texture information seems to be carried by a very small percentage of dominant frequencies. In this section we choose to ignore the coefficients of these dominant spectral components and match solely on the basis of the *location* of these components. We report a distance between two texture windows reflecting the average distance in the spectral plane of the dominant components of the two texture spectra.

Suppose we are given two such binary masks μ_1 and μ_2 whose underlying windows are congruent—in other words which are of the same size. We will think of each mask as defining a "shape" in the plane, consisting of all the "on" (white in our Figures 2, 3) pixels in the mask. We wish to define a distance between the two shapes μ_1 and μ_2. Note that, if the percentage p used in obtaining these masks was the same, then (to within the precision of our floating-point computation), these two masks should have the same number of on pixels. However, in matching these masks as shapes, we do not require a one-to-one matching between their pixels. Instead, we use a variant of the popular Hausdorff distance.[HK92] The Hausdorff distance is defined in terms of the directed Hausdorff distance $\delta(A, B)$ from shape A to shape B, which is the maximum, over all points x of A, of the distance from x to the nearest point of B. The Hausdorff distance is then just $\max\{\delta(A, B), \delta(B, A)\}$. Note that this distance may match many points of A to the same point of B, and vice-versa. Note also that in our application it is quite fortunate that we can get by by setting the percentage p quite high, so that each of the two masks μ_1 and μ_2 has only a small fraction of on pixels. Had we, for example, chosen $p = 50\%$, then each mask would have had half its pixels on and, on the average, an on pixel of μ_1 would have been within a distance one of an on pixel of μ_2, rendering a Hausdorff-like distance useless.

The Hausdorff distance has been used extensively in tracking, OCR, and other vision applications to compare directly two images. In these applications, because of image noise, one usually selects not the maximum distance, as in the definition given above, but instead a high percentile of the distance, thus rejecting outliers. Since we are applying th Hausdorff concept not to images directly but to their already thresholded spectra, we felt it more appropriate to measure distance by defining $\delta(A, B)$ to be the *average* over all points x of A, of the distance from x to the nearest point of B. Also, for computational efficiency, we compute this distance from x to the nearest point of B only approximately. Let us explain this more precisely. Suppose we want to compute $\delta(\mu_1, \mu_2)$. We construct a sequence of masks $\mu_2^{(0)} = \mu_2, \mu_2^{(1)}, \mu_2^{(2)}, \ldots$, as follows. The mask $\mu_2^{(1)}$ is obtained from $\mu_2^{(0)}$ by tiling $\mu_2^{(0)}$ into 2×2 pixel blocks and marking all pixels in a block as on if any one of the four of them was on in $\mu_2^{(0)}$. For $\mu_2^{(2)}$ we consider blocks of size 4×4, and so on. If x is an on pixel of μ_1 that for the fist time becomes on in $\mu_2^{(i)}$ as $i = 0, 1, 2, \ldots$, we define its distance to μ_2 to be $2^i - 1$. It is not hard to check that this distance is within a factor of 2 of the actual distance, and therefore our final average directed distance is also correct to within a factor of two.

We will denote our Hausdorff-like distance between the spectra thresholded by p of two texture windows W_1, W_2 of the same size by $\vartheta(W_1, W_2, p)$.

3.2 The Best-Neighbor SSD

The technique of the previous section fails to distinguish the bean texture of Figure 1 from dissimilar looking textures (such as the pebble texture in Figure 4), because at the p where the spectra were thresholded, the corresponding masks look very similar. In the actual spectra, however, we can see that the beans have a very high intensity ring around the origin, reflecting the rather uniform bean size and distribution. This example shows how the Hausdorff-like distance can fail because, after thresholding, it completely ignores spectral intensity information.

As already mentioned, a method for computing spectral distance based entirely on intensity information is the Sum of Squared Differences (SSD). It is not clear *a priori* how to weigh intensity difference vs. location difference in comparing two spectra — the units are completely different. In this section we report on a method that reports an SSD-like (intensity-based) distance, but is also sensitive to location information. Suppose we wish to compute the distance between two spectra A and B of the same size. Let us fix a certain neighborhood shape N, which we can think of simply as a description of a certain set of location displacements we are willing to allow. Now, for each pixel a in A we find its *best-neighbor* b in B. This is that pixel in B that is within the neighborhood N centered at a, and which minimizes the value of $|I(A[a]) - I(B[b])|$, where I denotes intensity. Our best-neighbor SSD distance is defined as the sum of the squares of all these best-neighbor distances, summed over all the pixels in A. Note that, for example, this method will report a zero distance (up to boundary effects) for two spectra that are simply shifts of each other within some vector in N. We will denote the best-neighbor SSD distance between the spectra of two texture windows W_1, W_2 of the same size by $\sigma(W_1, W_2)$.

In order to speed up this computation, we again approximate this best-neighbor distance, this time by random sampling. Instead of letting N be some fixed-size square window centered at the origin, we instead define N by random sampling a number of pixels around the origin using a Gaussian probability density function and then save those displacements.

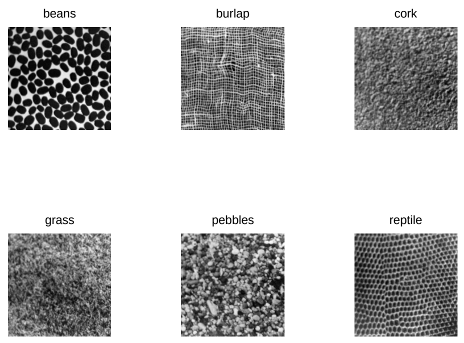

Figure 4: Six texture samples from[Bro66].

3.3 The Warped SSD

The best-neighbor SSD allows some flexibility in the positions of spectral components by searching for the best match to each component in a small neighborhood. Thus, the allowable deformation between two spectra is bounded by the size of this search neighborhood. Given, however, that the main family of deformations we are interested in allowing is associated to the viewing geometry, it seems to make sense to model these deformations and incorporate the model into the definition of the SSD distance. If the geometric deformations of perspective are approximated by affine transformations, the corresponding spectral deformations are affine as well. Consequently, we can define a *warped SSD* distance between two spectra as the residual SSD distance after one spectrum has been deformed as well as possible into the other by an affine transformation. We compute the optimal warping by means of a Newton-Raphson style iterative search for the best A, which we devised in past research on feature tracking.[ST94]

Intuitively, this new definition of SSD would seem to work well. In fact, if two spectra are indeed at least approximately related by an affine transformation, the optimization stage will find this transformation, and the residual SSD distance will be smaller than without it. If on the other hand the two spectra are unrelated, the optimization stage will not be able to find a good transformation, and the residual SSD distance will remain large. However, the experiments show that this is not the case, unless the deformations due to the viewing geometry are very large. For small deformations, in fact, the decrease in distance between unrelated spectra is usually greater than that between related spectra, thereby making performance in fact worse. This is illustrated next.

3.4 Comparison of the Methods

One way to compare the distances given above is to test their discriminating power on a set of textures. Figure 4 shows six texture samples from.[Bro66] Each texture sample (128×128 pixels in size) is divided into four subwindows. Figure 5 is a gray-level representation of the 24×24 symmetric matrix of the distances between the resulting 24 subwindows (four from each of the six texture samples). Rows and columns correspond to the textures in alphabetical order, and to subwindows in the following order: top left, top right, bottom left, bottom right. A perfect distance matrix would then have its six 4×4 diagonal blocks equal to zero, and all its off-diagonal entries very large. The actual matrix obtained with the best-neighbor SSD distance measure is shown in figure 5. Analogous matrices relative to other SSD distances are not shown. In fact, they are numerically different, but not enough to make a big difference on the gray values of this display. The numerical values are compared below.

The matrix in figure 5 has smaller entries in the diagonal blocks, as expected, and in particular the beans texture is very distant from all the others, as evidenced by the bright entries in the first four rows and four

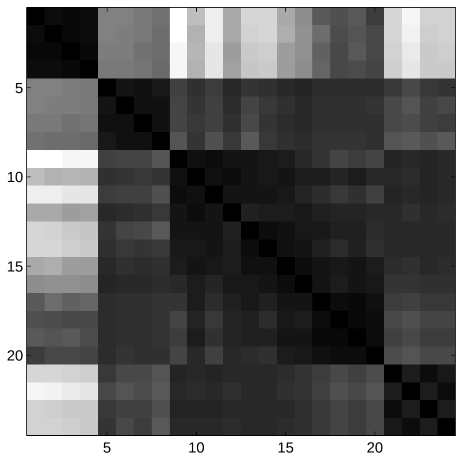

Figure 5: Distance matrix between subwindows of the six textures in figure 4. Four subwindows are extracted from each of the six texture samples.

SSD with no warp or jitter	0.1697
warped SSD	0.3358
best-neighbor SSD	0.1357

Table 1: Confusion parameters for three variations of SSD distances.

columns. However, differences between cork, grass, and pebbles are somewhat blurred (although still visible). Cork and grass, in particular, are high-frequency textures with no clear low-frequency distinctions. Their spectra are rather flat, making them hard to distinguish. The beans and reptiles texture, on the other hand, have both a very strong geometric component, which leads to the ring visible in the spectrum of figure 2 and to the hexagonal structure of the spectra in figure 3. These would probably be picked up as strong periodic components in the Wold decomposition of.[PL94]

An indication of the relative quality of the SSD measures presented above can be obtained by computing a *confusion parameter* as the ratio between the average within-texture distance and the average across-texture distance. A ratio of 0 is ideal; a ratio of 1 is total confusion (no texture discrimination) and a ratio greater than one is a distance definition that ought to be replaced with its reciprocal.

Table 1 shows the confusion parameters for three different types of SSD distance. While best-neighbor jittering improves performance, warping makes it substantially worse. The degradation due to warping occurs because the Brodatz texture samples are taken from nearly fronto-parallel surfaces, with little geometric distortion. Consequently, the affine warping has little or no effect on the within-texture distances, while it lowers the across-texture distances. Of course warping would improve the confusion parameter in the presence of large geometric distortions, but the fact that across-texture distances are made worse remains, casting serious doubts on the usefulness of warping. Affine distortions applied to a spectrum, if unbounded, can modify the shape of the spectrum too much, thereby dangerously lowering the distinctions between unrelated textures. If any technique like this is to be used, some constraint on the amount or type of warping should probably be imposed. The better performance of the best-neighbor SSD corroborates this point, since in this experiment the neighborhood was constrained to about three frequency samples.

An analogous experiment using the Hausdorff-like distance gave discrimination results that were qualitatively similar to the best-neighbor SSD, but in general worse. In particular, the beans and pebble textures of Figure

4 were not well discriminated against each other. This, plus the fact that the Hausdorff-like computation is significantly slower than the best-neighbor SSD, leads us to conclude the best-neighbor SSD is the preferred spectral matching method for textures, at least among the one we have examined.

3.5 Combination Distances

Several other distance definitions are possible, but we have not experimented with them. For example, we could imagine using combinations of the Hausdorff-like and best-neighbor SSD distances by an expression of the form

$$\sqrt{\alpha\vartheta(W_1, W_2, p) + \beta\sigma(W_1, W_2)}\ ,$$

with $\alpha + \beta = 1$, though it is not clear how to determine the best choice of weights α and β.

A related possibility is to measure spectral distance using the so-called *earth-movers* distance between two images of the same size that have been normalized to have the same average intensity. The idea here is that we view each image as a height field reflecting the distribution of sand (or earth) in a sandbox. The earth-movers distance between two such sandboxes is the minimum amount of work needed to move the sand in the first box and give the height distribution in the second box. The work unit is here is moving one unit of sand (intensity distance) by one unit of horizontal displacement (location distance). The nice aspect of this definition is that its unit of work reflects both types of earlier distances together, in one common unit. We have not yet considered algorithms for efficiently computing this distance. An approximation can be obtained by a variant of the best-neighbor SSD calculation, where we weigh the best match between $a \in A$ and $b \in B$ by $|I(A[a]) - I(B[b])|(\kappa d(a, b) + 1)|$, where $d(a, b)$ is the distance between pixels a and b, and κ is a parameter.

4 Segmentation Experiments

We have used some of the above distance algorithms to group tiles of a particular level in a quad-tree decomposition of an image into similar groups, thus achieving a primitive kind of segmentation. We note here that for our image searching application, it will be enough if we can identify large regions of uniform texture in an image — we do not need to accomplish a full segmentation in the traditional sense.

For our experiments we compared each tile of an image with its right and bottom neighbors. Ideally, there should be some distance threshold so that if we mark all tile boundaries across which the tiles have textures with spectral distance above that threshold, then we should obtain an image segmentation to with the resolution of the tiling we have. An experiment using the best-neighbor SSD distance and two natural scene images is shown in Figure 6.

The New York skyline and the roller coaster have a strong textural component and have been well segmented from their texturally different surroundings. But an assortment of other somewhat random boundaries has also been introduced. When running this and related experiments on real images, we ran into serious problems related to the window size. In fact, windows that are too small with respect to the characteristic periods of a texture result in poor descriptors of that texture. Windows that are too large are likely to contain multiple textures. Given the irregular shapes of textural regions in real images, it is not too likely that one can find a rectangular window of just the right size for a given region. Unfortunately, the computation of Fourier transforms adapts poorly to regions of irregular shapes. One solution to this dilemma may be to stick with a quad-tree decomposition of images, but to tile textural regions with combinations of elements of the appropriate size from the quad-tree. This in turn requires methods to combine spectra from adjacent windows of possibly different sizes into a single spectral descriptor of an irregularly shaped region.

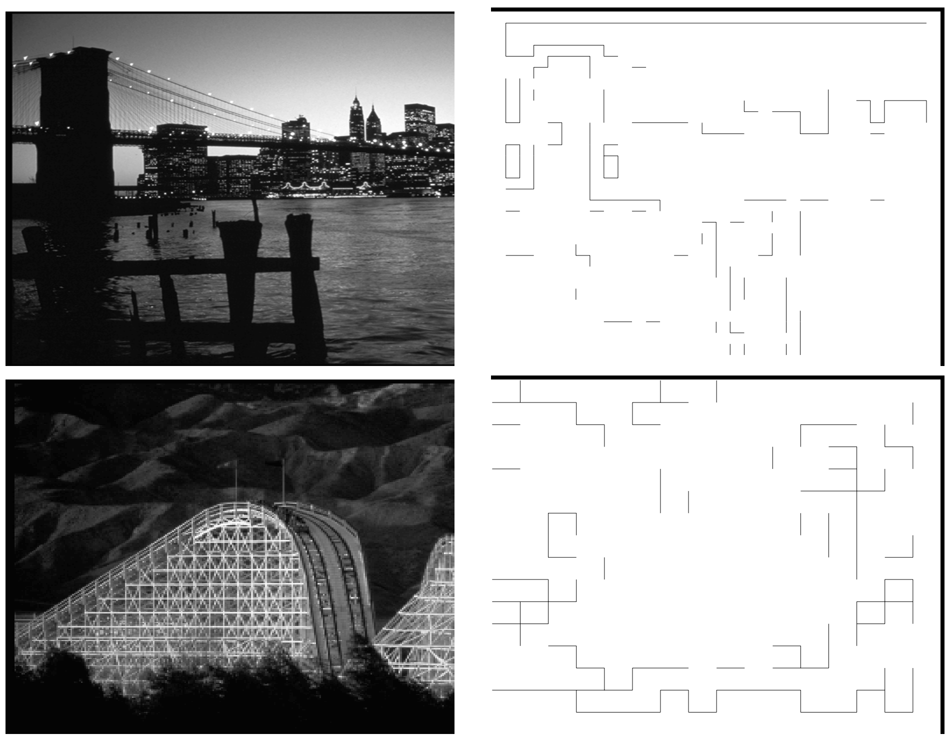

Figure 6: Natural scenes (left) and significant texture tile boundaries (right) at a given quad-tree level.

5 Conclusions

In this paper, we have explored some aspects of texture metrics within the rather popular framework of Fourier-style descriptors and fixed-size image windows. We have defined two different types of distances, the Hausdorff-like distances and the SSD-like distances which in a sense lie at two opposite ends of a continuum. The former penalize horizontal distance in the Fourier domain, but are lenient towards vertical differences between values of the spectra. The SSD distances we defined do the opposite: through best-neighbor match or affine warping they are tolerant of horizontal deformations of the spectra, but penalize vertical discrepancies. We have suggested variations of the earth-movers distance as a definition that combines aspects of these extremes.

In a sense, the Hausdorff-like distances can be considered as lower bound to the SSD-like distances. Among the SSD-like distances, best-neighbor matching improves texture discrimination power, while affine warping makes things worse. This negative result surprised us when we first saw it, but is in retrospect obvious. In fact, as explained above, the affine warping is more successful at decreasing across-texture discrepancies than at decreasing within-texture deformations.

Acknowledgement: The authors acknowledge the support of ARPA grant DAAH04-94-G-0284 under which this research was carried out.

6 REFERENCES

[Ana89] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2(3):283–310, January 1989.

[Bro66] P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover, New York, NY, 1966.

[FBF$^+$94] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3:231–262, 1994.

[HK92] D. Huttenlocher and K. Kedem. *Distance metrics for comparing shapes in the plane*, pages 201–219. Academic Press, 1992.

[MP90] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, 7(5):923–932, May 1990.

[OS75] A. V. Oppenheim and R. W. Shafer. *Digital Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1975.

[PL94] R. W. Picard and F. Liu. A new Wold ordering for image similarity. In *IEEE Conference on ASSP*, April 1994.

[RJ92] Editor Ramesh Jain. *NSF Workshop on Visual Information Management Systems.* Redwood, CA, 1992.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, June 1994.

[TG94] C. Tomasi and L. Guibas. Image descriptions for browsing and retrieval. In *Proceedings of the ARPA Image Understanding Workshop*, November 1994.

[Tur86] M. Turner. Texture discrimination by Gabor functions. *Biological Cybernetics*, 55:71–82, 1986.