Spatial Synchronization Using Watermark Key Structure

Eugene T. Lin and Edward J. Delp

Video and Image Processing Laboratory
School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana USA

ABSTRACT

Recently, we proposed a method for constructing a template for efficient temporal synchronization in video watermarking.[1] Our temporal synchronization method uses a state machine key generator for producing the watermark embedded in successive frames of video. A feature extractor allows the watermark key schedule to be content dependent, increasing the difficulty of copy and ownership attacks. It was shown that efficient synchronization can be achieved by adding temporal redundancy into the key schedule.

In this paper, we explore and extend the concepts of our temporal synchronization method to spatial synchronization. The key generator is used to construct the embedded watermark of non-overlapping blocks of the video, creating a tiled structure.[2–4] The autocorrelation of the tiled watermark contains local maxima or peaks with a grid-like structure, where the distance between the peaks indicates the scale of the watermark and the orientation of the peaks indicate the watermark rotation. Experimental results are obtained using digital image watermarks. Scaling and rotation attacks are investigated.

Keywords: watermarking, key structure, autocorrelation, synchronization, tiling

1. INTRODUCTION

Watermark synchronization is a significant challenge in robust blind watermark detection.[5, 6] Synchronization is the process of identifying the correspondence between the coordinates of the watermarked signal and the embedded watermark, or "finding the watermark." If the input signal provided to the watermark detector is watermarked but the detector is unable to establish synchronization, then the embedded watermark will generally not be detected. The fact that synchronization is crucial for successful watermark detection is a well-recognized vulnerability, and attacks have been devised to make synchronization more difficult. The objective of these synchronization or "geometric" attacks[7] is to cause the detector's synchronization process to fail, rendering the watermark undetectable in the watermarked signal. Examples of synchronization attacks against image watermarks include spatial scaling and rotation, as well as more general affine and non-affine coordinate transformations.

A direct approach for synchronization is to perform an *exhaustive* or *naïve search* over the space of coordinate transformations to locate of the watermark. Unfortunately, the space of coordinate transformations is large and this approach is hindered by computational cost and the possibility of false positives.[8] Efficient synchronization for image and video watermark detection has generally been addressed by three methods: embedding the watermark with invariance properties to transformations,[9, 10] designing the watermark to be less sensitive to detector synchronization,[11] or using *templates*. A template is a pattern which describes the coordinates of the embedded watermark. A coordinate transformation applied to the watermarked signal transforms the template in the same manner as the watermark, and thus the location of the watermark after synchronization attack can be obtained by examining the template.[12] Equivalently, a template may be thought of as side-information regarding the structure of the watermarked signal that allows the detector to reduce the search for synchronization.

Recently, we proposed a model for temporal synchronization in video watermarking.[1, 13] In our model, the watermark embedder creates the watermark embedded into each frame of the video using a state machine key

This work was supported by a grant from Digimarc. Address all correspondence to E. J. Delp at ace@ecn.purdue.edu.

generator. The objective of the watermark detector is to determine, for each frame of the input video, which key was used to produce the watermark embedded in that frame. The time-invariant key, time-periodic key, and time-independent key schedules are special cases of the key generator. We described a technique for efficient synchronization by adding temporal redundancy in the watermark key sequence. Only temporal synchronization was addressed in our previous work.

In this paper, we extend and adapt the temporal synchronization model to spatial synchronization of still image and video watermarks. In the case of spatial synchronization, redundancy in the watermark structure can be used to produce a template for efficient synchronization. Specifically, the watermark embedder constructs a watermark with a regular or "tiled" structure. When the autocorrelation of the watermarked image is obtained, peaks (local maxima) occur in a pattern resembling a grid, and this pattern is examined to estimate the rotation and the scale of the watermark. The redundant structure of a tiled watermark reduces security because such a watermark is more easily estimated. However, the redundancy in a tiled watermark may allow detector synchronization even when part of the watermark has been cropped.

Previous works have considered synchronization by using the autocorrelation or constructing a tiled watermark structure. Kutter[14] constructs the watermark by the superposition of an elementary watermark signal and shifted versions of the same signal. The shifted watermarks induce peaks in the autocorrelation, which serves as the basis for obtaining the watermark rotation and scale. Kalker[2] constructs a tiled watermark for efficient synchronization under spatial translation. Alattar[4] constructs a tiled watermark, and then uses a log-polar mapping of the autocorrelation to estimate the watermark rotation and scale. Deguillaume[3] also constructs a tiled watermark, but the affine coordinate transformation is estimated by using the Hough Transform. Unfortunately, the Hough Transform is computationally expensive to obtain, particularly if the rotation and scale must be estimated to a high degree of precision.

Our synchronization method is similar to those proposed by Alattar[4] and Deguillaume[3] in that we use the grid-like pattern of peaks in the autocorrelation to estimate the watermark coordinates. However, we propose an alternate method for watermark scale and rotation estimation than the log-polar mapping and Hough Transform. Our method is limited to uniform scaling and rotation attacks, which is more restrictive than Deguillaume's technique. However, our method does not require the detector to obtain the Hough Transform of the autocorrelation.

2. WATERMARK EMBEDDING

A straightforward approach for extending the temporal synchronization framework to spatial synchronization is to construct the watermark in a block-by-block basis. The watermark embedder is shown in Fig. 1. The original image and the embedding key K_E are provided as inputs. Let $X(n)$ be a block of the original image, where $n = 0, 1, \ldots$ is the block index. The key generator provides the key $K(n)$ to the watermark generator, which uses the key to produce the watermark signal $W(n)$. $W(n)$ is inserted into $X(n)$ to produce the watermarked block $Y(n)$. The block analyzer examines $Y(n)$ to produce a feature vector $F(n)$, which is used by the key generator to produce the key to watermark the next block $X(n+1)$. This process is repeated until all blocks of the image have been watermarked. The redundancy control "resets" the key generator at regular intervals to control the structure of the watermark, which will be described below.

The key generator is a state machine whose internal state is denoted by $s(n) \in \mathcal{S}$, where $\mathcal{S}$ is the set of states. When block $X(n)$ is watermarked, the key generator supplies key $K(n) = \lambda(s(n))$ to the watermark generator, where $\lambda(\cdot)$ is the *output function.* When the watermarked block $Y(n)$ is produced and the feature vector $F(n)$ is available, the key generator performs a state transition: $s(n+1) = \phi(s(n), K_E, F(n))$, where $\phi(\cdot)$ is the *state transition function.* A cryptographic hash function, where $s(n)$, K_E and $F(n)$ are hashed, is an example state transition function.[1] Prior to watermarking the first block of the image, the state of the key generator is set to an *initial state* $s_0 \in \mathcal{S}$, which may be dependent on K_E[*]. The key used to watermark the first block is thus $K(0) = \lambda(s_0) = K_0$. Also, when the key generator is reset by the redundancy control, its internal state is set to $s(n) = s_0$ and the key used to watermark the block is $K(n) = K_0$.

[*]The key generator may have randomized (non-deterministic) state transitions.[1] However, we assume that the state machine has a unique initial state.

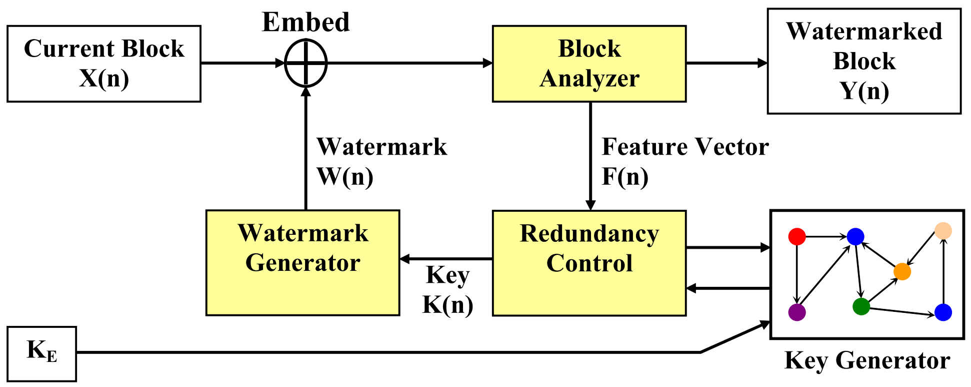


Figure 1. The proposed watermark embedder with state machine key generator.

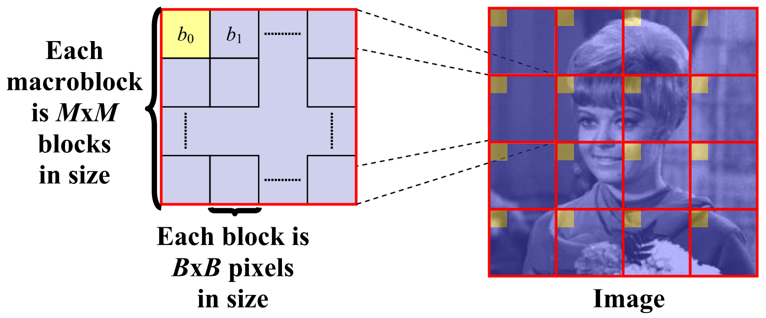


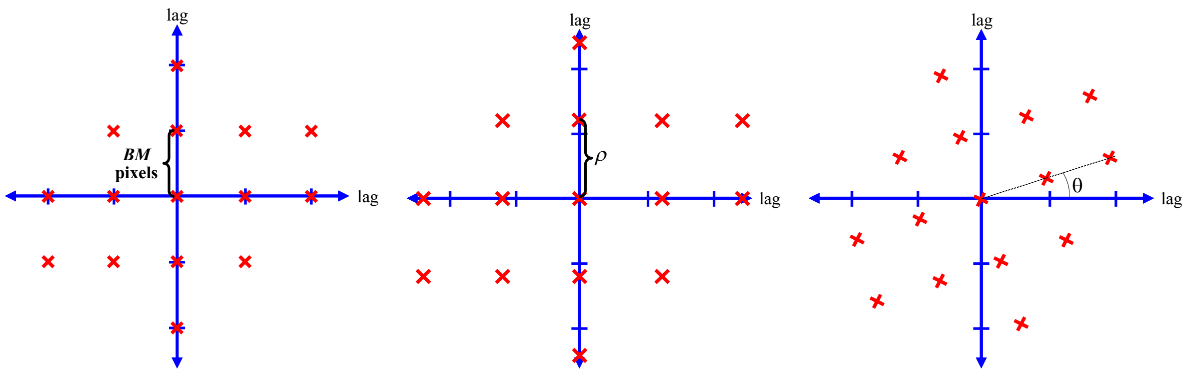
Figure 2. Watermark structure showing macroblocks and blocks. The image shows an example of the structure of a watermark constructed and inserted in the spatial domain. The first block of all macroblocks are identical.

This scheme can produce a watermark with a structure resembling that shown in Fig. 2, where the watermark is partitioned into non-overlapping macroblocks $\Gamma_0, \Gamma_1, \ldots$. Each macroblock is a region of MxM blocks, with blocks $b_0, b_1, \ldots, b_i, \ldots, b_{M^2-1}$. Each block is a region of BxB pixels. For simplicity, we assume that the dimensions of the image are integral multiples of MB. To produce the watermark structure, the key generator is reset prior to watermarking the first block (b_0) of each macroblock. This block is watermarked by using the key $K_0 = \lambda(s_0)$. After this block is watermarked, the key generator performs state transitions to construct the watermark for the remaining blocks $b_1, \ldots, b_{M^2-1}$. After the last block of the macroblock has been watermarked, the key generator is reset by the redundancy control to watermark the first block of the next macroblock. There are other watermark structures that can be produced by this framework (such as by changing the redundancy control or the order which the blocks of the image are watermarked), however only this watermark structure will be considered in this paper.

The first block (b_0) of all the macroblocks are generated by the key K_0. If $W(n)$ is only a function of $K(n)$, then these blocks are identical[†]. These *synchronization blocks* form a regular pattern which shall be the basis for the spatial synchronization technique discussed in Sec. 3. The remaining blocks of the watermark, the *non-synchronization blocks*, are generated by keys that depend on the block analyzer and are generally not identical. Although these blocks do not play a role in synchronization, they are part of the watermark and can be detected by the watermark detector.

The watermark structure is characterized by the parameters M and B. For $M = 1$, the watermark is constructed by regularly repeating, or tiling, an elementary watermark signal of size BxB pixels. All the blocks are synchronization blocks and there are no non-synchronization blocks. For $M > 1$, the watermark contains

[†]In particular, we assume that the watermark generator does not change or adapt the watermark structure based on the original image block, except for possibly amplitude scaling (perceptual shaping).[15]



(a) No Attack (b) Uniform Scaling (c) Rotation

Figure 3. Autocorrelation of watermark showing location of local maxima or peaks. Peaks are indicated by 'X'. Not all the peaks in the autocorrelation are shown.

non-synchronization blocks which are not necessarily repeated. As M increases, the fraction of synchronization blocks to the total number of watermark blocks decreases.

3. WATERMARK SYNCHRONIZATION AND DETECTION

When the watermarked image is provided to the watermark detector, the detector establishes synchronization and then detects the watermark. In this section, we shall focus on the synchronization process when the watermarked image may be attacked by uniform scaling and rotation.

The objective of the synchronizer is to estimate the rotation and scale of the embedded watermark. Once the synchronizer estimates the rotation and scale, these transformations are "reversed" to obtain a normalized watermark. Detecting the normalized watermark is a straightforward adaptation of the watermark detection protocol[1, 13] from frame-by-frame detection to block-by-block detection, and will not be described here. Also, this synchronization process does not estimate the translation (spatial shifting) of the watermark. The shift be estimated in a manner similar to Kalker[2] or by using an auxiliary template.

For a watermark with the structure described in Sec. 2, the synchronization blocks are repeated at regular intervals. When the autocorrelation function of the watermark is obtained, the repeated blocks induce local maxima, or peaks, with a grid-like arrangement similar to Fig. 3(a). The distance between two adjacent peaks of the grid is the *neighbor distance* ρ. If the watermark has not been attacked, $\rho = BM$ pixels. Uniformly scaling the watermark causes the distance of the peaks to change to $\rho = BMf$ pixels, where f is the scaling factor. Fig. 3(b) illustrates the effect of uniform re-scaling on the autocorrelation peaks of the watermark. Rotating the watermark does not change the neighbor distance, but causes the peaks to be rotated by the same rotation angle θ as shown in Fig. 3(c). Thus, if the watermark has been re-scaled by unknown scaling factor f and rotated by unknown angle θ, the synchronizer can estimate the watermark scale and rotation by estimating ρ and θ.

Several issues complicate the synchronization process. First, the signal power of the embedded watermark is much lower than that of the original image. The autocorrelation of the watermarked image may not show the expected grid of peaks unless the interference arising from the original image is reduced. One method to reduce the interference is to apply a spatial filter which reduces the correlation of neighboring pixels. We use the de-correlating filter used by Kalker[2] where the input image of the watermark detector is convolved with the FIR filter

$$h = \frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \qquad (1)$$

prior to obtaining the autocorrelation, although other methods for reducing host signal interference can be used. Another issue is that the autocorrelation of a watermarked image may have additional peaks, missing peaks, and peaks that are slightly moved (or "perturbed") from their ideal positions.

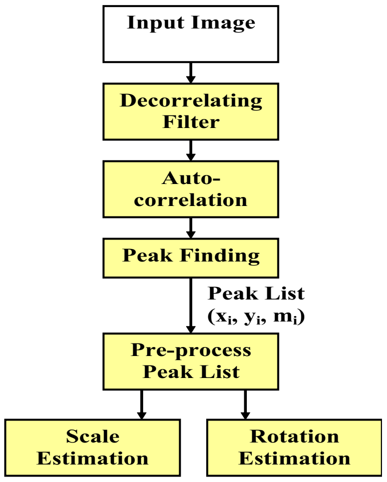


Figure 4. Watermark synchronization

The synchronization process is shown in Fig. 4. The decorrelating filter (1) is convolved with the input image and the autocorrelation of the filtered image is obtained. Then, the location and magnitude of all local maxima of the autocorrelation is obtained. The method for peak finding is simple: A window is moved across the autocorrelation and the local maximum inside the window is obtained. If the center value is the maximum, then it is a peak. After peak finding, a list of peaks $p_i = (x_i, y_i, m_i)$ is obtained, where x_i and y_i are the coordinates (position) of peak p_i in the autocorrelation and m_i is the magnitude of p_i.

Since the autocorrelation function of an image is symmetric about the origin, half the peaks are redundant and are removed from the peak list. Then, the peak list is truncated to retain only the P peaks of greatest magnitude. Retaining too few peaks does not allow the grid structure of Fig. 3 to be detected, and retaining too many peaks causes an excessive number of "noise" peaks (or peaks that are not part of the grid structure). We used $P = 50$ in our experiments, which provided a reasonable balance between detecting too many noisy peaks and too few peaks.

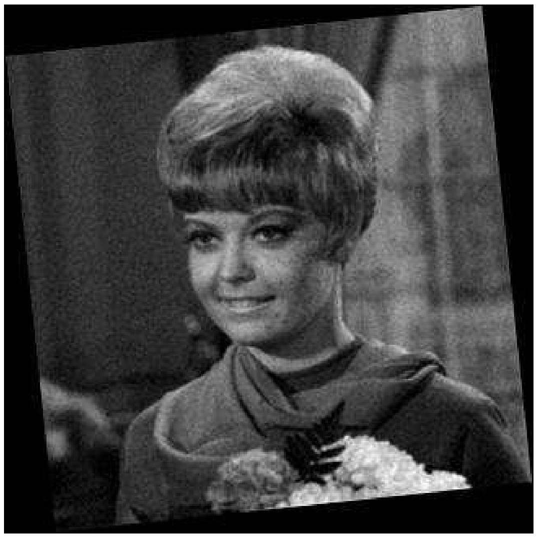
The image shown in Fig. 5 will be used to illustrate the scale and rotation estimation. This image has been watermarked in the spatial domain ($M = 3$, $B = 16$) and attacked by scaling the image using $f = 1.15$ followed by six degrees rotation. Fig. 5(a) shows the watermarked and attacked image. The positions of the peaks in $\mathcal{P}$ are shown in Fig. 5(b). The grid pattern is present, but also accompanied by a number of additional "noise" peaks that arise from the interference of the original image. The neighbor distance is $\rho = MBf = 55.2$ pixels.

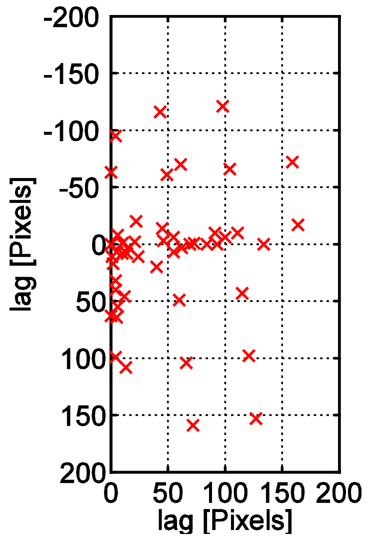
3.1. Watermark scale estimation

We assume that the peaks of $\mathcal{P}$ are spatially arranged in a grid with unknown neighbor distance ρ and oriented with unknown angle θ. The objective of the scale estimation is to estimate ρ. The scale estimation technique considers the distances between all pairs of peaks, for two reasons. First, the distance between any two peaks of the grid is independent of the orientation of the grid. Second, the arrangement of the peaks implies that certain distances should occur frequently, which becomes the basis by which the watermark scale can be estimated. Thus, the first step is to obtain the distance

$$d_{ij} = d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2}$$

between every pair of peaks $p_i, p_j \in \mathcal{P}$, $i \neq j$. There are P peaks in $\mathcal{P}$, so there are $\binom{P}{2}$ pairs of peaks. The pairwise peak distances are used to construct a *distance histogram* $\mathcal{D}$. The value of each bin in the histogram is the number of d_{ij}'s in the interval $\left[(n - \frac{1}{2})\Delta_s, (n + \frac{1}{2})\Delta_s\right)$ for $n = 0, 1, 2, \ldots$ and histogram bin size $\Delta_s > 0$. The bins with high values are important because these bins correspond to d_{ij}'s which occur most frequently. "Noise" peaks are not likely to form a regular pattern that gives rise to high bin values. The distance histogram of *Girl*


(a) Attacked image



(b) Peak positions

Figure 5. Example of watermark scale and rotation estimation using image *Girl*. PSNR of watermarked image is 33.8 dB, $M = 3$, $B = 16$. Attack is re-scaling by factor $f = 1.15$ followed by rotation of 6°.

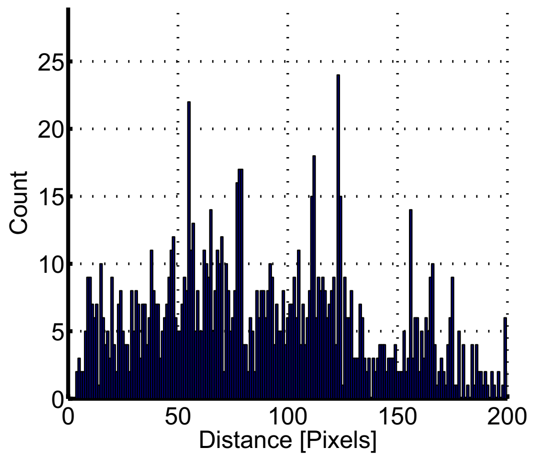
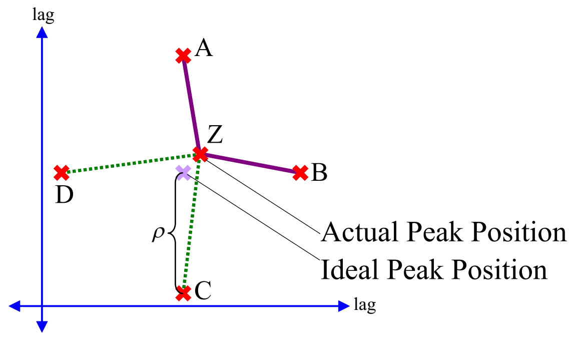


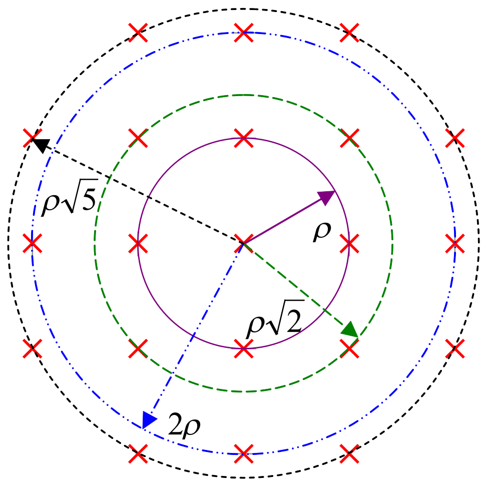
Figure 6. Distance histogram $\mathcal{D}$ of *Girl* (histogram bin size $\Delta_s = 1$). Large values occur at the bins corresponding to the distances of 55 pixels, 78 pixels, 110 pixels, and 123 pixels.

is shown in Fig. 6. The bin whose corresponding to the correct the neighbor distance of 55.2 pixels has a high value, however it is not the bin with the maximum value.

The construction of the distance histogram from the pairwise peak distances does not consider the fact that the peaks may be slightly shifted or "perturbed" from their ideal positions. Small shifts in the peak positions may arise from the interpolation of the image data when the watermarked image is attacked. Fig. 7(a) illustrates the effect when a peak is slightly shifted from its ideal position. In the ideal case where there is no perturbation in the positions of the peaks, each of the distances between the non-center peaks (A,B,C,D) to the center peak (Z) is ρ. When $\mathcal{D}$ is obtained, there is a high value at the the bin whose distance interval contains ρ. However, when the peaks are not in their ideal locations then some pairwise distances become slightly longer and other distances become slightly shorter. These distances may map onto different bins in $\mathcal{D}$. When this occurs, the value of the bin whose distance interval contains ρ is reduced and ρ cannot be estimated from $\mathcal{D}$.



(a) Effect of shifted or "perturbed" peak on pairwise peak distances



(b) Geometric structure of grid peaks

Figure 7. Improving the scale estimation

The effect of perturbed peaks may be reduced by smoothing the distance histogram. Specifically, the distance histogram $\mathcal{D}$ is treated like a discrete one-dimensional signal $\mathcal{D}(n)$ where the bin values are the values of the signal. Smoothing can be accomplished by convolving $\mathcal{D}$ with a *smoothing filter* h_s to generate the processed distance histogram $\mathcal{D}^* = \mathcal{D} * h_s$. The distance intervals of the bins of $\mathcal{D}^*$ correspond to the same intervals as the bins of $\mathcal{D}$.

While the distance histogram provides some information regarding ρ, the estimation of ρ can be improved by using additional geometric properties of the grid. Fig. 7(b) shows a regular grid of peaks with neighbor distance ρ. The center peak represents any peak on the grid. The four nearest neighboring peaks lie at the distance ρ from the center peak. Estimating ρ directly from $\mathcal{D}$ or $\mathcal{D}^*$ uses only these four peaks to infer a grid structure. However, a regular grid structure also has four peaks at a distance of $\rho\sqrt{2}$ from the center peak, four additional peaks at a distance 2ρ, and eight peaks at the distance of $\rho\sqrt{5}$ from the center peak. Observing all these distances suggests the presence of a grid more than merely observing the distance ρ. The distance histogram of *Girl* (Fig. 6, with $\rho = 55.2$ pixels) shows large values at bins whose distance intervals include the distances 55.2 pixels, $55.2\sqrt{2} = 78.0$ pixels, 110.4 pixels, and $55.2\sqrt{5} = 123.4$ pixels.

A score function is used to estimate the neighbor distance. The score function S_s is a discrete function with values at $k\Delta_s$ for integral k:

$$S_s(k\Delta_s) = q_s(k\Delta_s) + q_s(\sqrt{2}k\Delta_s) + q_s(2k\Delta_s) + q_s(\sqrt{5}k\Delta_s) \quad \text{for} \quad k = 1, 2, \ldots \tag{3}$$

where $q_s(z)$ is the value of the bin in $\mathcal{D}^*$ whose distance interval contains z. For fixed k, the score $S_s(k\Delta_s)$ is a quantity that reflects the likelihood (based on the pairwise distances) that the autocorrelation peaks form a grid with neighbor distance $\hat{\rho} = k\Delta_s$. Having obtained $S_s(\cdot)$, the watermark detector detects the watermark using the neighbor distance which has the largest score ($\hat{\rho} = \arg\max S_s(k\Delta_s)$). If the watermark is not detected, then the watermark detector attempts to detect the watermark using the neighbor distance with the second highest score. This process is repeated in decreasing order of score until the watermark is detected or the watermark detector "gives up."

The score function obtained for the *Girl* image is shown in Fig. 8. The maximum score occurs at the distance of $\hat{\rho} = 55$ pixels so the detector successfully estimates the scale of the watermark on the first attempt. If the

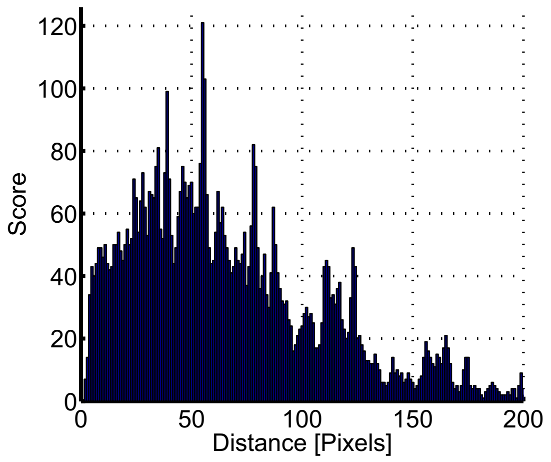


Figure 8. Score function $S_s(k\Delta_s)$ of *Girl* ($\Delta_s = 1$).

first estimate was not the correct neighbor distance, the detector would then try $\hat{\rho}_2 = 56$ pixels, then $\hat{\rho}_3 = 39$ pixels, $\hat{\rho}_4 = 78$ pixels, and so on.

3.2. Watermark rotation estimation

The estimation of the watermark orientation is challenging because of the symmetry of the template. Rotating the watermark by angles of θ, $\theta + \frac{\pi}{2}$, $\theta + \pi$, and $\theta + \frac{3\pi}{2}$ produce indistinguishable patterns of peaks. Horizontal and vertical flipping also produce indistinguishable patterns. As mentioned by Deguillaume,[3] all eight ambiguities arising from combinations of rotation and flipping must be searched by the detector if other means for distinguishing these transformations are not used, such as another template.

The objective of watermark rotation estimation is to estimate the orientation of the grid of peaks θ, where $0 \le \theta < \frac{\pi}{2}$. We shall consider rotations by angles $\theta + k\left(\frac{\pi}{2}\right), \quad k = \ldots, -1, 0, 1, \ldots, \quad 0 \le \theta < \frac{\pi}{2}$ to be congruent. The technique for rotation estimation is conceptually similar to the scale estimation. The pairwise angles

$$\psi_{ij} = \psi(p_i, p_j) = \arctan\left(\frac{y_i - y_j}{x_i - x_j}\right) \tag{4}$$

are used to construct the *angle histogram* $\mathcal{A}$. The bins of $\mathcal{A}$ partition the interval $\left[0, \frac{\pi}{2}\right)$ to intervals of the histogram bin size $\Delta_r > 0$. For example, $\Delta_r = 1^\circ = \frac{\pi}{180}$ uses 90 bins in $\mathcal{A}$.

Once the angle histogram is obtained, it is smoothed by convolving with the smoothing filter h_r to produce the smoothed histogram $\mathcal{A}^* = h_r \circledast \mathcal{A}$. Unlike the smoothing for the distance histogram, angles are congruent in $\left[0, \frac{\pi}{2}\right)$ and the convolution is circular. After histogram smoothing, the rotation score is obtained

$$S_r(k\Delta_r) = q_r(k\Delta_r) + q_r\left(k\Delta_r + \frac{\pi}{6}\right) + q_r\left(k\Delta_r + \frac{\pi}{4}\right) + q_r\left(k\Delta_r + \frac{\pi}{3}\right) \quad \text{for} \quad 0 \le k\Delta_r < \frac{\pi}{2} \tag{5}$$

where $q_r(z)$ is the value of the bin of $\mathcal{A}^*$ whose interval contains the angle congruent to z in $0 \le \theta < \frac{\pi}{2}$. Similarly to estimating the neighbor distance, the watermark detector first estimates the rotation to be $\hat{\theta} = \arg\max S_r(k\Delta_r)$. If the watermark is not detected using $\hat{\theta}$, then the detector attempts the angle with the second highest score for $\hat{\theta}_2$, and so on until either the the watermark is detected or the detector gives up.

4. EXPERIMENTAL RESULTS

To evaluate the synchronization technique, we implemented an additive watermark in the spatial domain. The watermark embedder embeds the watermark in the luminance of the original image block $X(n)$ to produce the watermarked image block $Y(n) = X(n) + \sigma W(n)$, where $W(n)$ is a zero mean, unit variance Gaussian signal produced by a pseudo random number generator seeded with $K(n)$. The block analyzer produces the feature $F(n)$ by generating a random number (and ignoring the watermarked image block). The key generator's state transition function hashes the current state, embedding key, and features to produce the next state, using SHA-1[16] as the hash function: $s(n+1) = \text{hash}(s(n), K_E, F(n))$. Using the hash function and the random feature values effectively create random keys $K(n)$ for all non-synchronization blocks and the corresponding $W(n)$ to be uncorrelated. The original images were *Girl*, *Fruit*, *Crowd*, and *Peppers*, with each image 256x256 pixels in size. The embedding amplitude was $\sigma = 5.0$, which produced watermarked images with PSNR of 33.8 dB compared with the original image. Embedding parameters were $M = 1, 2, 3$ blocks and $B = 8, 16, 32$ pixels. Attacks were performed using bicubic interpolation.

The watermark synchronization uses a peak list of $P = 50$ peaks. For scale estimation, $\Delta_s = 1$ pixel, which allows ρ to be estimated to the nearest pixel. For histogram smoothing, the FIR filter kernel h_s is $[0.15 \;\; 0.50 \;\; 1.00 \;\; 0.50 \;\; 0.15]$ where 1.0 is the value of $h_s(0)$. The design of this filter was ad hoc. A more rigorous design of h_s would assume a probability model for the peak perturbations and then use the probability model to determine the filter, which may be dependent on Δ_s. For rotation estimation, $\Delta_r = 1°$, which allows θ to be estimated to the nearest degree. Histogram smoothing was not performed for rotation estimation so $\mathcal{A}^* = \mathcal{A}$.

4.1. Scale and rotation rank

The synchronization performance is measured in terms of the number of attempts that the watermark detector requires before obtaining the correct scale and orientation. In Section 3.1, it was described that the watermark detector estimates the neighbor distance of the grid (which estimates of the watermark scale) by choosing the distance d with the highest score $S_s(d)$. If this estimated scale is not correct, the watermark detector estimates the scale using the distance with the second highest score. Consecutive attempts continue in decreasing order of $S_s(\cdot)$ until the watermark detector correctly estimates the watermark scale. We define the *scale rank* as the total number of attempts that the watermark detector requires to obtain the correct watermark scale. For example, if the detector correctly estimates the scale of the watermark on the first attempt, the scale rank is 1, which is the best rank. A large scale rank indicates a failure in the scale estimation technique. The *rotation rank* is similarly defined as the total number of attempts that the watermark detector requires to obtain the correct watermark orientation.

4.2. Discussion

Synchronization performance was examined under scaling attack, without rotation. The scaling factor was varied from $f = 0.3$ to $f = 2.1$. For $M = 1$ watermarks with block sizes of $B = 16$ and $B = 32$, the scale rank is less than four and the rotation rank is less than three for $f \geq 0.5$. The scale estimation performance degrades when the neighbor distance of the grid is small, which occurs for small macroblocks ($M = 1, B = 8$) or when f is small. There are several reasons for this. First, when the distances between peaks become smaller, perturbations of the peak positions introduce larger relative error in the distances. Second, the "blurring" of the distance histogram caused by the histogram smoothing increases the difficulty of precisely estimating small neighbor distances. Third, constructing the distance histogram effectively quantizes the peak distances according to the bin intervals, which may be problematic in estimating the neighbor distance when Δ_s is not small compared to ρ. For $M = 2$, the scale rank is less than five for the *Girl*, *Fruit*, and *Peppers* images when $f \geq 0.5$ and $B = 8$ or $B = 16$. There is a sharp increase in the scale rank when $f < 0.8$ for *Crowd*. The decreased performance in *Crowd* occurs from "noise" peaks appearing in the autocorrelation. These noise peaks are from the original image. Scale estimation generally fails for any f when $M = 2$ and $B = 32$, and for $M = 3$ and $B = 16$ or $B = 32$. In these cases, the magnitudes of the peaks induced by the synchronization blocks are weak, which causes many grid peaks to be perturbed or missing and a large number of "noise" peaks to appear in the autocorrelation. When $M = 3, B = 8$, there is some success in scale estimation when $f \geq 1.0$. For $M = 2$ and $M = 3$, the rotation estimation successfully estimates $\theta = 0°$. The rotation ranks are in the range 1–3, even when the scale estimation fails.

We examined the synchronzation performance under rotation attack. The watermarked image was rotated by various angles from $\theta = 0°$ to $89°$, without re-scaling. For $M = 1$, the scale and rotation rank is less than four for $B = 16$ and $B = 32$ and any θ. The scale estimation fails for the $B = 8$ watermark. For $M = 2$, the scale and rotation ranks are small for $B = 8$ and $B = 16$, but both the scale and rotation estimation fail when $B = 32$. For $M = 3$, scale and rotation estimation is successful only for $B = 8$. Generally, the watermarks constructed using M and B values that have good performance under the scaling-only attack also have good performance under the rotation-only attack. The scale and rotation rank are not significantly affected by θ, except for the $M = 2, B = 16$ and $M = 3, B = 8$ watermarks. In these watermarks, there is an increase in the rotation rank when θ is near $0°$ or near $90°$. It is believed that the combination of peak perturbations and "noise peaks" confuse the rotation estimation.

Performance under both scaling and rotation attacks was also investigated. In the attack implementation, the watermarked image is re-scaled first, and then rotated. Bicubic interpolation was performed after both the scaling and rotation processes. In Fig. 9, the attack scaling factor f is varied from 0.3 to 2.1 and the rotation is fixed at $\theta = 3°$. In Fig. 10, the scaling is fixed to $f = 1.2$ and the rotation is varied. The synchronization performance is generally good for $M = 1$ watermarks with sufficient block size, with only problems when the block size is too small ($B = 8$) or when the scaling factor is small. The scale rank for the *Fruit* image has a dramatic rise at $f = 1.2$ and $\theta = 40°$. The reason for this will be described below.

When $M = 2$, the technique fails for large block sizes ($B = 32$). When the scaling is varied with fixed rotation, the behavior is similar to the scaling-only attack except that the estimation begins to fail at higher scale factors ($f < 0.8$). The *Crowd* image is particularly difficult, as there are a considerable number of "noise peaks" in the autocorrelation. When the rotation is varied, we observe difficulties in estimating the scale for rotations near $\theta = 45°$. The difficulties arise because the grid pattern is not present in the autocorrelation. For these rotation angles, a significant area of the attacked image consists of the border regions created by padding when the image dimensions are expanded to contain the rotated image. When the autocorrelation is obtained, these border regions reduce the magnitude of the peaks that form the grid while at the same time the edges between the border region and the image cause very strong peaks to occur along the edge. These border regions also cause the sharp rise in scale rank for *Fruit*, $M = 1, B = 32$ described above, although in this case the border regions do not completely destroy the peak grid. One way to address the border region issue is to crop the watermarked image to only the center-most area prior to obtaining the autocorrelation. Cropping removes the large areas near the image borders and allows the autocorrelation grid to be obtained.

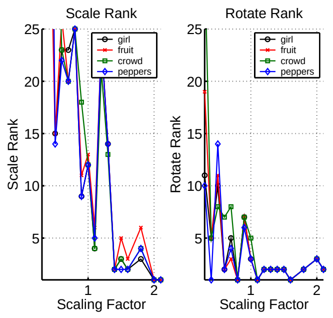
5. CONCLUSIONS AND FUTURE WORK

In summary, the experiments show that the proposed synchronization technique is successful under uniform scaling and rotation attack for $M = 1$ watermarks of sufficiently large block size. When $M = 2$, synchronization is successful for $B = 8$ and $B = 16$. When $M = 3$, synchronization is successful only when $B = 8$. Synchronization performance decreases rapidly with increasing M, and performance is better for smaller B when $M > 1$. Scale estimation is often successful for $f \geq 0.5$, however host signal interference causes the scale estimation to fail at larger scale factors for some images (*Crowd*). Scale estimation also has some difficulty under scaling and rotation attack, which is caused by the padding of the rotated image. More investigation is needed to determine how padding affects the autocorrelation. In particular, the effects of padding were observed when the attack consists of both scaling and rotation, but were not observed for only rotation attack. The robustness of the autocorrelation peaks may be improved by using the characteristics of the image.[17]

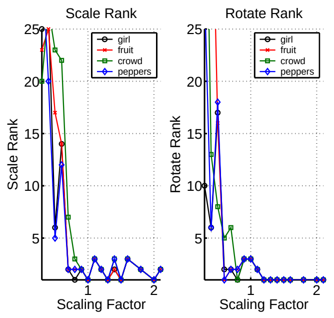
We are also interested in investigating how to improve the precision of the scale and rotation estimation. We noted that peaks in the the autocorrelation may be perturbed from their ideal locations. Some of the perturbations of the peak positions occur because peak locations are discretized to pixel locations. This "quantization" effect on the peak locations limits the precision by which the rotation and scale can be estimated.[18]

REFERENCES

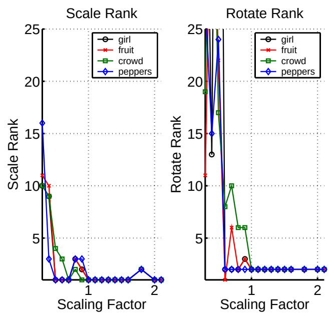
1. E. T. Lin and E. J. Delp, "Temporal synchronization in video watermarking–further studies," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents V*, vol. 5020, Santa Clara, CA, Jan. 21–24, 2003, pp. 493–504.
2. T. Kalker, G. Depovere, J. Haitsma, and M. Maes, "A video watermarking system for broadcast monitoring," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, vol. 3657, San Jose, CA, Jan. 25–27, 1999, pp. 103–112.
3. F. Deguillaume, S. Voloshynovskiy, and T. Pun, "A method for the estimation and recovering from general affine transforms in digital watermarking applications," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents IV*, vol. 4675, San Jose, CA, Jan. 21–24, 2002, pp. 313–322.
4. A. M. Alattar and J. Meyer, "Watermark re-synchronization using log-polar mapping of image autocorrelation," *Proceedings of the International Symposium on Circuits and Systems 2003*, vol. 2, May 25–28, 2003, pp. 928–931.
5. E. T. Lin, A. M. Eskicioglu, R. L. Lagendijk, and E. J. Delp, "Advances in digital video content protection," *Proceedings of the IEEE*, 2004, submitted.
6. M. Barni, C. I. Podilchuk, F. Bartolini, and E. J. Delp, "Watermark embedding: Hiding a signal within a cover image," *IEEE Communications Magazine*, vol. 39, no. 8, pp. 102–108, Aug. 2001.
7. S. Voloshynovskiy, S. Pereira, T. Pun, J. J. Eggers, and J. K. Su, "Attacks on digital watermarks: Classification, estimation-based attacks, and benchmarks," *IEEE Communications Magazine*, vol. 39, no. 8, pp. 118–126, Aug. 2001.
8. J. Lichtenauer, I. Setyawan, T. Kalker, and R. Lagendijk, "Exhaustive geometrical search and the false positive watermark detection probability," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents V*, vol. 5020, Santa Clara, CA, Jan. 21–24, 2003, pp. 203–214.
9. C.-Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, M. L. Miller, and Y. M. Lui, "Rotation, scale, and translation resilient watermarking for images," *IEEE Transactions on Image Processing*, vol. 10, no. 5, pp. 767–782, May 2001.
10. J. J. O'Ruanaidh and T. Pun, "Rotation, scale, and translation invariant spread spectrum digital image watermarking," *Signal Processing*, vol. 66, no. 3, pp. 303–317, May 1998.
11. I. Setyawan, G. Kakes, and R. L. Lagendijk, "Synchronization-insensitive video watermarking using structured noise pattern," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents IV*, San Jose, CA, Jan. 21–24, 2002, pp. 520–530.
12. S. Pereira and T. Pun, "Robust template matching for affine resistant image watermarks," *IEEE Transactions on Image Processing*, vol. 9, no. 6, pp. 1123–1129, June 2000.
13. E. T. Lin and E. J. Delp, "Temporal synchronization in video watermarking," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents IV*, vol. 4675, San Jose, CA, Jan. 21–24, 2002, pp. 478–490.
14. M. Kutter, "Watermark resisting to translation, rotation, and scaling," *Proceedings of the SPIE: Multimedia Systems and Applications*, A. G. Tescher, B. Vasudev, J. V. Michael Bove, and B. Derryberry, Eds., vol. 3528, Jan. 1999, pp. 423–431.
15. R. B. Wolfgang, C. I. Podilchuk, and E. J. Delp, "Perceptual watermarking for images and video," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1108–1126, July 1999.
16. *Secure Hash Standard*, National Institute of Standards and Technology Std. 180-1, Apr.17, 1995.
17. C.-H. Lee, H.-K. Lee, and Y. Suh, "Autocorrelation function based watermarking with side information," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents V*, vol. 5020, Santa Clara, CA, Jan. 21–24, 2003, pp. 349–358.
18. M. Álvarez Rodríguez and F. Pérez-González, "Analysis of pilot-based synchronization algorithms for watermarking of still images," *Signal Processing: Image Communication*, vol. 17, no. 8, pp. 611–633, Sept. 2002.



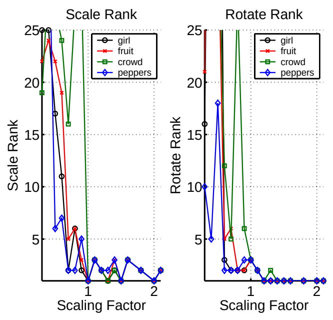
(a) $M = 1,\ B = 8$



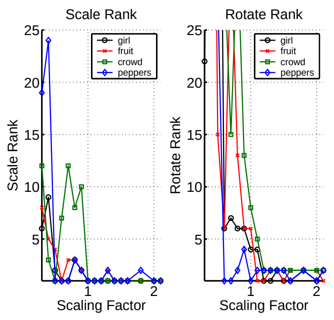
(b) $M = 1,\ B = 16$



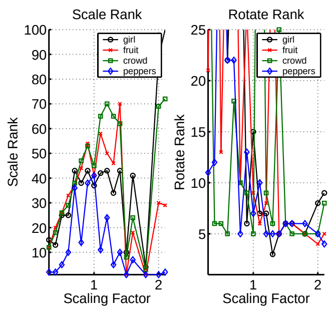
(c) $M = 1,\ B = 32$



(d) $M = 2,\ B = 8$

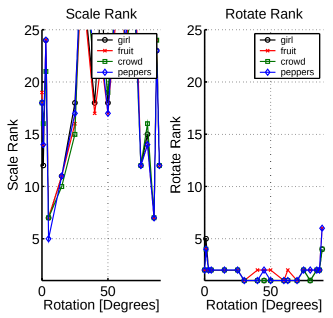


(e) $M = 2,\ B = 16$

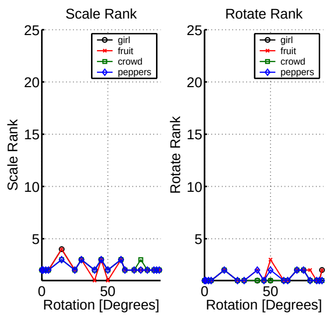


(f) $M = 2,\ B = 32$

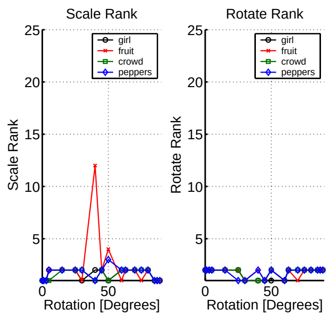
Figure 9. Performance of synchronizer after scaling (variable) and rotation (fixed at $\theta = 3°$) attack



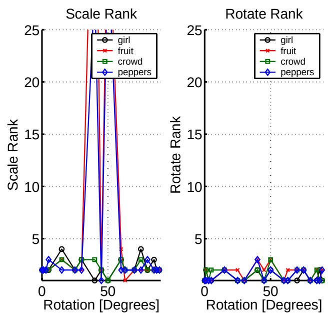
(a) $M = 1,\ B = 8$



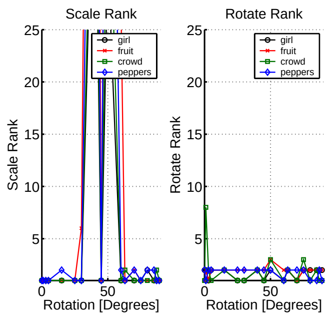
(b) $M = 1,\ B = 16$



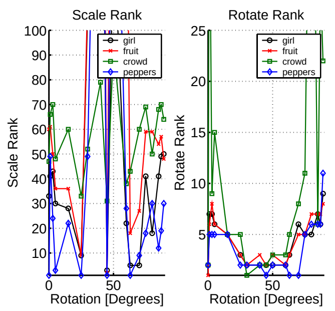
(c) $M = 1,\ B = 32$



(d) $M = 2,\ B = 8$



(e) $M = 2,\ B = 16$



(f) $M = 2,\ B = 32$

Figure 10. Performance of synchronizer after scaling (fixed at $f = 1.2$) and rotation (variable) attack