Preprocessing media to facilitate later insertion of a watermark

Ingemar J. Cox, Matt L. Miller
NEC Research Institute
4 Independence Way, Princeton, NJ 08540

Abstract

There are several watermarking applications that require the deployment of a very large number of watermark embedders. These consumer applications have severe budgetary constraints that limit the computation resources that are available. Under these circumstances, only simple embedding algorithms can be deployed, which have limited performance. In order to improve performance, we propose preprocessing the original media during content creation. A simple example of this procedure is described and experimental results confirm our assertions.

1 Introduction

There are a number of applications of watermarking in which it is necessary to deploy a very large number of watermark embedders. In such situations, economic constraints are often severe and constrain the computational resources that are available for embedding. Unfortunately, high performance - as measured by effectiveness, fidelity and robustness - watermark embedders can involve perceptual modeling [10], informed coding [7, 3, 4][1] and informed embedding [9], any of which may require great computational resources than is available.

We address this dilemma by proposing a two stage procedure in which a substantial fraction of the computational workload is performed as a preprocessing step on the media prior to its release to the general public. This preprocessing step is designed to permit, at a later time, subsequent watermark embedding based on computationally simple algorithms that are very economic.

Our solution is appropriate in situations where content can be modified before it reaches the watermark embedders. Such situations turn out to be quite common. Section 2 discusses two examples. In Section 3, we describe the basic principles behind preprocessing and a two-step watermarking process. These principles are tested experimentally in Section 4. Finally, a discussion of results and future work are contained in Section 5.

2 Motivation

We are motivated by watermarking applications in which embedding must be very inexpensive. One such example is transactional watermarking (also known as *fingerprinting*) and another is copy generation management.

In late 1996, the DiVX Corporation released an enhanced DVD player based on a pay-per-play business model. In order to allay the piracy concerns of Hollywood studios, DiVX implemented a number of security technologies. One of these was a watermark-based system for transaction tracking. Each DiVX player embedded a unique watermark in the analog NTSC video signal during playback of a movie. These transaction watermarks were intended to be used to track the source of any pirated video that originated from the DiVX customer base. The DiVX DVD player was a consumer level product and, as such, was extremely price sensitive. Accordingly, the computational resources allocated to embedding the transactional watermark had to be small.

[1]Note that the term "preprocessing" as used in [3] differs from our usage here.

Copy generation management is intended to allow a single generation of copies to be made from a master, but no subsequent copies to be made from the first generation copies. In order to reduce the threat of piracy, content owners envisage labeling broadcasted material as `copy_once` and subsequently labeling the material as `copy_no_more` after recording. A number of technical solutions to copy generation management were proposed in the context of DVD recorders [1, 8]. The solution proposed in the Galaxy system uses a fixed watermark to encode the `copy_once` state, and adds a second, *copy_mark*, to encode the `copy_no_more` state [2]. This second watermark would be added during recording, within a consumer DVD recorder. Once again, the computational budget allocated to the secondary watermark embedder was very small.

3 Media preprocessing

One of the main difficulties with inexpensive watermark embedders is that their performance is highly dependent on the cover Works to which they are applied. An embedder might perform well on one Work, successfully embedding a high-fidelity, robust mark, while completely failing to embed in another Work. The idea of preprocessing is to modify all the Works beforehand, moving them to a region for which the inexpensive embedder is known to perform well.

To describe how media can be preprocessed for low-cost watermark embedders, we present three basic systems. We begin, in Section 3.1, by applying the idea of preprocessing to a simple, linear-correlation based watermark. In Section 3.2, we show how the idea can be applied to a more complex system, which employs normalized correlation as its detection metric. Finally, Section 3.3 discusses the application of preprocessing to watermarks that can encode multiple messages.

3.1 Preprocessing for a linear correlation system

In a zero-bit, linear-correlation watermarking system, the detector tests for presence or absence of a watermark by computing the linear correlation between a received Work, $\mathbf{c}$, and a reference pattern, $\mathbf{w_r}$: $z_{\text{lc}} = \mathbf{c} \cdot \mathbf{w_r} = \sum_i \mathbf{c}[i]\mathbf{w_r}[i]$. If z_{lc} is greater than a detection threshold, τ_{lc}, then the detector reports that the watermark is present.

Blind embedding is computationally trivial. For example, a watermark can be added to a video stream (in baseband) without requiring that the frames be buffered. However, because the embedding effectiveness is less than 100%, such a system might not be acceptable for some applications. Informed embedding can guarantee 100% effectiveness, but it requires that the entire cover Work be examined before the watermark is embedded, so a frame buffer is required. Thus, informed embedding can be substantially more expensive than blind embedding.

Consider a geometric model of the problem, in which cover Works are represented as points in a high dimensional marking space. In blind embedding a fixed vector that is independent of the cover Work is added to each Work, the intention being to move the cover Work into the detection

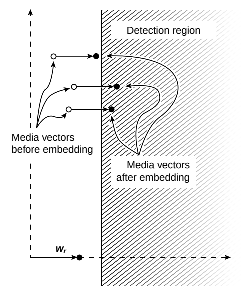

Figure 1: Two-dimensional geometric model of watermarking using a blind embedder.

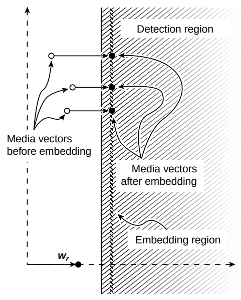

Figure 2: Two-dimensional geometric model of watermarking using an informed embedder

region. A two-dimensional geometric model is illustrated in Figure 1. If a simple correlation detector is used, then this detection region is a half-plane the boundary of which is denoted by the vertical line in Figure 1. Unwatermarked cover Works lie to the left of this boundary and are denoted by the open circles. Notice that some cover Works are closer to the boundary than others[2]. The horizontal arrows represent the watermarking process which moves the cover Work towards the detection region and, hopefully into the detection region. This is also illustrated in Figure 1 where the majority of cover Works have indeed been moved into the detection region but one cover Work has not. The embedder is said to have failed to watermark this particular cover Work, i.e. its effectiveness is less than 100%.

In contrast to blind embedding, informed embedding allows us to automatically vary the strength of the watermark based on the cover Work. Figure 2 illustrates the effect of an informed embedder in which a watermark of different magnitude is added to each cover Work, such that all watermarked Works are guaranteed to be a fixed distance within the detection region. We refer to the region occupied by watermarked Works as the *embedding* region.

Now let us consider a two step process in which informed preprocessing is used to guarantee that subsequent blind embedding will be successful. Figure 3 shows how such a system might work. Here, the preprocessing stage modifies each original cover Work (open circles) so that the processed Works (grey circles) all lie within a narrow region close to, but outside of the detection region. We refer to this narrow region as the *prepping* region. Since the prepping region is outside the detection region, no watermarks are detected in the preprocessed content. However, when a simple blind embedder is subsequently applied to the preprocessed content, it will be 100% effective in embedding the watermark.

[2]In fact, it is also possible for an unwatermarked Work to be to the right of the boundary. This would denote a false positive.

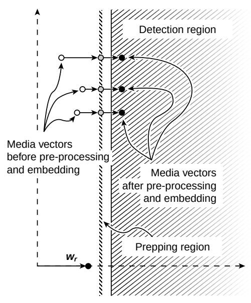
Detection region
Media vectors before pre-processing and embedding
Media vectors after pre-processing and embedding
Prepping region
w_r

Figure 3: Geometry of the preprocessing and embedding.

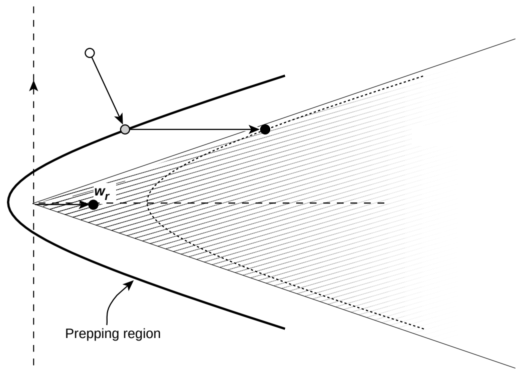

Figure 4: Preprocessing to obtain constant robustness when a blind embedder is applied.

3.2 Preprocessing for a normalized correlation system

The same technique can be applied to more complex watermarking systems, such as those that use normalized correlation as a detection metric. Here, the detector computes the normalized correlation between a received Work, **c**, and a reference pattern, $\mathbf{w_r}$, as $z_{\mathrm{nc}} = (\mathbf{c}\cdot\mathbf{w_r})/(\sqrt{(\mathbf{c}\cdot\mathbf{c})(\mathbf{w_r}\cdot\mathbf{w_r})})$. This results in the conical detection region of Figure 4.

Here again, blind embedding can often successfully embed watermarks, but it fails in many cases. It is argued in [6, 9] that a more reliable method of embedding is to seek a fixed estimate of robustness. If we estimate robustness as the amount of white noise that may be added to the watermarked Work before it is likely to fall outside the detection region, then a fixed-robustness embedder will employ a hyperbolic embedding region. Although such an embedder is preferable for many applications, it can be quite costly [5].

To obtain the reliability of a fixed-robustness embedder, while using a simple blind algorithm to embed, we can define a prepping region by shifting the hyperbolic embedding region outside the detection region. The distance that the embedding region must be shifted depends on the embedding strength that will be used by the blind embedder. This is shown in Figure 4. Here, the prepping region is a hyperboloid that lies entirely outside the detection cone. When a blind embedder is applied to a preprocessed Work (grey circle), the Work is moved into the detection region, so that the resulting watermarked Work (black circle) lies on the desired contour of constant robustness (dotted line).

Note that, if the embedding strength that will be used during blind embedding is too low, the shifted embedding region might overlap with the detection region. This would not be satisfactory as a prepping region, since it would lead to false positives. To solve this problem, we can simply remove a portion of the shifted embedding region from consideration during preprocessing. The preprocessor would move each Work to the closest point on the shifted hyperboloid that lies sufficiently far outside the detection region.

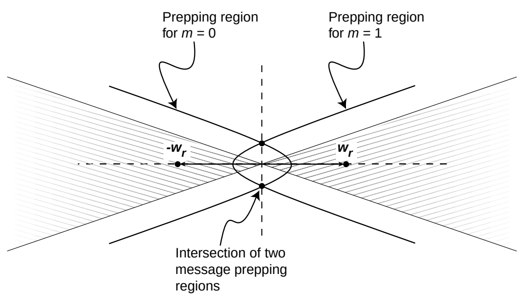
Prepping region for $m = 0$
Prepping region for $m = 1$
$-w_r$
w_r
Intersection of two message prepping regions

Figure 5: Preprocessing for a one-bit, normalized-correlation watermarking system.

3.3 Preprocessing for multiple bit watermarks

The two systems described above apply preprocessing to simple, zero-bit watermarks. That is, the detectors in these systems report whether the watermark is present or absent, but do not distinguish between different watermark messages, so the watermark carries zero-bits of payload information. If we have a system that can embed several different watermark patterns, representing different messages, we must modify our preprocessing method accordingly.

In the simplest case, we might have a system with two possible messages, or 1 bit of payload. For a message of $m = 1$, we might embed a reference mark, $\mathbf{w_r}$. For $m = 0$, we might embed the negation of the reference mark, $-\mathbf{w_r}$. The detector would check for presence of both the positive and negative watermark, reporting the corresponding message if one of them is found. Such a system, then, would define two disjoint detection regions, one for each message.

To ensure that blind embedding will succeed in embedding *any* of the possible messages, the preprocessor must move content to a prepping region that is the *intersection* of appropriate prepping regions for all the messages. For example, consider a 1 bit system using normalized correlation as its detection metric, as illustrated in Figure 5. The two detection regions in this case would be two opposing cones. A fixed-robustness embedder, when embedding $m = 1$, would move each Work to a hyperbolic embedding region within the positive cone. When embedding $m = 0$, it would move each Work to an embedding region within the negative cone. Shifting each of these embedding regions according to the effect of a blind embedder gives us two possible prepping regions – one that ensures the blind embedder can embed message $m = 1$, and one that ensures it can embed message $m = 0$. Only a Work in the intersection of these two regions will allow successful embedding of either message.

Note that the two points in the prepping region shown in Figure 5 actually correspond to a high-dimensional hypersphere in media space. Thus, although the figure appears to define a prepping region of only two points, the actual prepping region is a high-dimensional surface, and, with appropriate watermark extraction techniques, it is possible to implement a preprocessor that does not introduce too much distortion (see Section 4).

4 Experimental results

To demonstate the preprocessing technique, we implemented a preprocessor for the E_BLK_BLIND/D_BLK_CC image watermarking system described in [5]. This is a one-bit, normalized-correlation system which operates in a linear projection of image space.

E_BLK_BLIND is a simple blind embedder. Although its description and implementation in [5] are a bit more complicated (to allow easy modifications into more sophisticated embedders), it essentially just adds or subtracts a scaled, tiled watermark pattern to the image. It takes as input an image, $\mathbf{c}$, to be watermarked, a message of either $m = 1$ or $m = 0$, an embedding strength, α, and an 8×8 reference mark, $\mathbf{w_r}$. If $m = 1$, the embedder addes $\alpha\mathbf{w_r}$ to each 8×8 block in the image. If $m = 0$, it subtracts $\alpha\mathbf{w_r}$ from each block.

The D_BLK_CC detection algorithm consists of two steps. In the first step, a mark vector, $\mathbf{v}$, is extracted from an image, $\mathbf{c}$, by averaging together 8×8 blocks to form one array of 64 values.

In the second step, the correlation coefficient[3], z_{cc}, is computed between the averaged 8×8 block, $\mathbf{v}$, and the reference mark, $\mathbf{w_r}$. That is, $z_{\rm cc} = (\tilde{\mathbf{v}} \cdot \tilde{\mathbf{w_r}})/(\sqrt{(\tilde{\mathbf{v}} \cdot \tilde{\mathbf{v}})(\tilde{\mathbf{w_r}} \cdot \tilde{\mathbf{w_r}})})$, where $\tilde{\mathbf{v}} = (\mathbf{v} - \bar{\mathbf{v}})$, $\tilde{\mathbf{w_r}} = (\mathbf{w_r} - \bar{\mathbf{w_r}})$ and $\bar{\mathbf{v}}$ and $\bar{\mathbf{w_r}}$ are the means of $\mathbf{v}$ and $\mathbf{w_r}$. It compares $z_{\rm cc}$ against a detection threshold, $\tau_{\rm cc}$. If $z_{\rm cc} > \tau_{\rm cc}$, it reports that message $m = 1$ has been embedded. If $z_{\rm cc} < -\tau_{\rm cc}$, it reports that message $m = 0$ has been embedded. Otherwise, it reports that there is no watermark present.

We implemented a preprocessor for this system according to the principles described in Section 3.3 and illustrated in Figure 5. The preprocessor performs the following steps

1. Extract a mark vector, $\mathbf{v_o}$ from the unwatermarked Work, in the same manner as the detector.

2. Identify a 2-dimensional plane that contains $\mathbf{v_o}$ and the reference mark, $\mathbf{w_r}$. The plane is described by two, orthogonal, unit vectors $\mathbf{X}$ and $\mathbf{Y}$, obtained by Gram-Schmidt orthonormalization.

3. Project $\mathbf{v_o}$ into the $\mathbf{X}$,$\mathbf{Y}$ plane to obtain $x_{\mathbf{v_o}} = \mathbf{v_o} \cdot \mathbf{X}$ and $y_{\mathbf{v_o}} = \mathbf{v_o} \cdot \mathbf{Y}$.

4. Find the point in the prepping region, $x_{\mathbf{v_p}}$,$y_{\mathbf{v_p}}$, that is closest to $x_{\mathbf{v_o}}$,$y_{\mathbf{v_o}}$. As shown in Figure 5, the prepping region in this 2-dimensional plane comprises only two points. Since $y_{\mathbf{v_o}}$ is guaranteed to be positive, the upper of these two points will always be the closest to $x_{\mathbf{v_o}}, y_{\mathbf{v_o}}$. Thus, $x_{\mathbf{v_p}} = 0$, and $y_{\mathbf{v_p}}$ is a positive value chosen to ensure that blind embedding will yield the desired level of robustness. This is computed as

$$y_{\mathbf{v_p}} = \sqrt{\frac{\alpha^2(1 - \tau_{\rm cc}^2)}{\tau_{\rm cc}^2} - R} \qquad (1)$$

where α is the embedding strength that will be used for embedding, and R is the desired robustness.

5. Obtain a preprocessed mark vector, $\mathbf{v_p}$, by projecting $x_{\mathbf{v_p}}$,$y_{\mathbf{v_p}}$ back into 64-dimensional space: $\mathbf{v_p} = x_{\mathbf{v_p}}\mathbf{X} + y_{\mathbf{v_p}}\mathbf{Y}$

6. Perform the inverse of the original extraction operation on $\mathbf{v_p}$ to obtain the preprocessed cover Work, $\mathbf{c_p}$.

To test these procedures, we first tested the watermarking system on un-preprocessed images, using a weak embedding strength of $\alpha = 0.5$. Watermarks of $m = 1$ and $m = 0$ were embedded in each of 2000 images from the Corel image database. Figure 6 shows the resulting detection values. The dotted line is a histogram of detection values for unwatermarked images, and each of the solid lines shows detection values for one of the embedded messages. With a detection threshold of $\tau_{\rm cc} = 0.55$, this succeeded in embedding watermarks in just over 45% of the trials.

[3]As pointed out in [5], the correlation coefficient between two vectors is just their normalized correlation after projection into a space with one fewer dimension. Thus, the detector computes the normalized correlation in a 63-dimensional space.

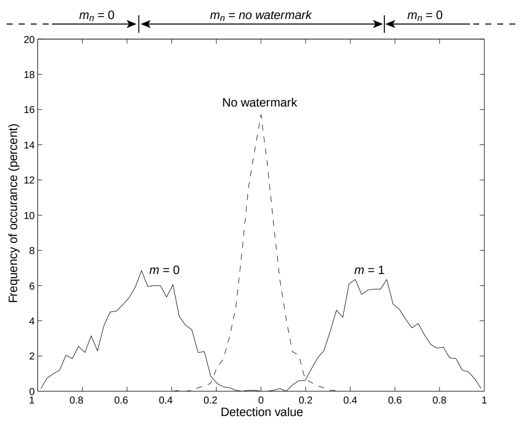

Figure 6: Results of the watermarking system with no preprocessing and $\alpha = 0.5$.

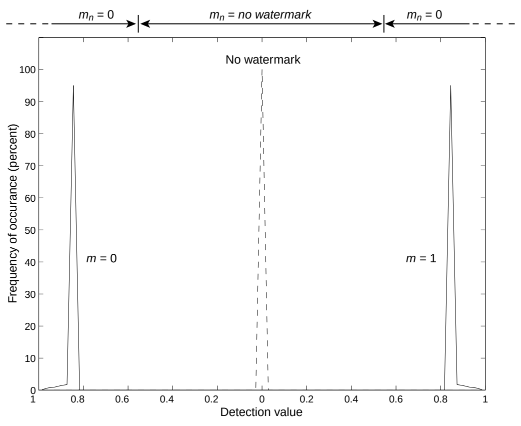

Figure 7: Results of the watermarking system applied to preprocessed images.

Next, we applied the preprocessor to each of the 2000 images, with $\tau_{cc} = 0.55$, $\alpha = 0.5$ and $R = 30$, and ran the same test again. The results are shown in Figure 7. As expected, applying the blind embedder to preprocessed images succeeded in embedding watermarks in 100% of the trials. In addition, the detection values obtained from preprocessed images before embedding a watermark are very narrowly distributed around 0. This indicates that they are less likely to yield false positives than are unpreprocessed images. In some applications, if we can guarantee that the detector will never be run on unpreprocessed images, we could take advantage of this to lower the detection threshold, thereby obtaining even better robustness.

The question arises of whether we could obtain equally good results, with the same fidelity, by just increasing the embedding strength used during blind embedding. Blind embedding with no preprocessing, yields an average mean-squared-error between marked and unmarked images of exactly α (because of the way we scaled $\mathbf{w_r}$). Preprocessing, however, introduces additional fidelity degradation. The average mean-squared-error between original images and images that have been both preprocessed and watermarked was just under 1.04. If, instead of applying preprocessing, we simply increased α to 1.04, we would obtain the same fidelity impact as preprocessing plus embedding, but we would have substantially stronger watermarks than with $\alpha = 0.5$. In fact, blind embedding with $\alpha = 1.04$ yields an effectiveness of 94% which is significantly better, but still not as good as with prepping.

Of course, since we can assume that we have substantial computing power available during preprocessing, we can improve on the fidelity impact of preprocessing by applying more sophisticated algorithms, such as perceptual modeling. Such improvements would increase the disparity between watermarking with and without preprocessing.

5 Conclusion

There are several watermarking applications in which a potentially very large number of embedders must be deployed under severe computational constraints that limit performance. In order to attain the performance of sophisticated embedding algorithms, and yet maintain a simple, inexpensive embedder, we propose preprocessing media before it is released. Most of the computational cost is shifted to the preprocesing stage where it is assumed that significant resources are available.

Geometrically, before the preprocessing, unwatermarked Works can be thought of as being randomly distributed in a high dimensional vector space. Within this space lies a detection region – Works falling within this region are said to be watermarked. Unwatermarked Works are seldom if ever found in the detection region. Traditional embedding algorithms seek to add a watermark pattern to a Work in order to move the Work into the detection subspace, subject to fidelity and robustness constraints. During the preprocessing stage suggested here, a signal is added to a Work such that the preprocessed Work lies on a predetermined surface near, but outside of the detection region. That is, the unwatermarked, but preprocessed Works are no longer *randomly* distributed in the high dimensional space but lie in a well-defined region.

The advantage of this preprocessing step is two-fold. First, since preprocessed Works lie on a well-defined surface, near yet outside of the detection region, simple embedding techniques are sufficient to watermark the Works with good fidelity and robustness. The second advantage is that the computational cost associated with the preprocessing step is not borne by the consumer electronic devices. Instead, content creators bare this cost, the preprocessing being performed by dedicated devices located with content creators. The performance of the overall system need no longer be constrained by the computational budget allocated to the embedder in the consumer device.

References

[1] J. A. Bloom, I. J. Cox, T. Kalker, J-P Linnartz, M. L. Miller, and B. Traw. Copy protection for DVD video. *Proc. IEEE*, 87(7):1267–1276, 1999.

[2] J. A. Bloom, M. L. Miller, and I. J. Cox. Method for data preparation and watermark insertion. *United States Patent*, 6,332,194 B1, 2001.

[3] B. Chen and G. W. Wornell. Preprocessed and postprocessed quantization index modulation methods for digital watermarking. In *Security and Watermarking of Multimedia Contents II*, volume SPIE-3971, pages 48–59, 2000.

[4] Jim Chou, S. Sandeep Pradhan, and Kannan Ramchandran. On the duality between distributed source coding and data hiding. *Thirty-third Asilomar conference on signals, systems, and computers*, 2:1503–1507, 1999.

[5] I. J. Cox, M. L. Miller, and J. A. Bloom. *Digital Watermarking.* Morgan Kauffman, 2001.

[6] I. J. Cox, M. L. Miller, and A. McKellips. Watermarking as communications with side information. *Proc. IEEE*, 87(7):1127–1141, 1999.

[7] J. J. Eggers, J. K. Su, and B. Girod. A blind watermarking scheme based on structured codebooks. In *IEE Seminar on Secure Images and Image Authetication*, pages 4/1–4/21, 2000.

[8] T. Kalker. System issues in digital image and video watermarking for copy protection. In *IEEE Int. Conf. on Multimedia Computing and Systems*, volume 1, pages 562 –567, 1999.

[9] M. L. Miller, I. J. Cox, and J. A. Bloom. Informed embedding: Exploiting image and detector information during watermark insertion. In *IEEE International Conference on Image Processing*, September 2000.

[10] Raymond B. Wolfgang, C. I. Podilchuk, and E. J. Delp. Perceptual watermarks for digital images and video. *Proc. of the IEEE*, 87(7):1108–1126, 1999.