

# RECOVERING FROM CROPPING AND TRANSLATION OF WATERMARKED BINARY IMAGES

*David Barbella*

[barbelld@carleton.edu](mailto:barbelld@carleton.edu)

Department of Computer Science, Carleton College, Northfield, MN 55057

## ABSTRACT

We present a method for ensuring that a binary image watermarking scheme is quite resistant to attack by cropping or translation. By locating difficult-to-alter anchor points and referencing everything to a center point (rather than to an easy-to-crop corner point), we are able to recover the shuffling map, which can then be used to recover the initial code. This method is not a watermarking scheme in and of itself, but rather an additional step that can be taken with many other binary watermarking schemes to allow increased resilience to cropping and translation.

*Index Terms*— Watermarking, Binary Image, Cropping, four, five

## 1. INTRODUCTION

Digital watermarking has a variety of uses in security and many other fields. For most image categories, including full-color images and grayscale images, changing the color values of some of the pixels slightly does not result in changes that are visible to the human eye. In a typical image, most any pixel can be changed slightly to encode information, which allows a wide variety of techniques to be used [3-10]. In a binary image, one where each pixel takes on one of only two values, usually black and white, however, the luxury of changing a pixel's value only slightly disappears. The number of pixels that can be changed without producing visible artifacts decreases, so special steps must be taken to ensure that any changes are as nearly invisible as possible.

Many techniques for embedding information into binary images have been suggested. Many of these techniques fall into one of three general categories – those that alter the distance between characters, words or lines in an image of text, those that adjust the properties of the dithering in a dithered image, and those that alter low-level features. The first two categories cannot be deployed on line drawing or signature images, and the third technique is advantageous in that a large quantity of information can be stored in a comparatively small image and that any sort of

binary image can be used. We will be focusing on methods that alter low-level features for these reasons.

One fairly robust way of embedding data into any binary image works by locating pixels that can be flipped with minimal visible disruption and then altering a subset of those pixels in such a way as to give the image certain properties. One way to hide information, used in [1] and elsewhere, is to divide the image into a number subsections that corresponds to the length of the string one wishes to embed, and then to alter pixels in each of those subsections so as to encode one bit of information in each.

Unfortunately, because of the nature of subsection division, watermarked images of this sort can be quite vulnerable to things such as cropping and translation. As noted in [1], due to their natural frailty, we are generally more interested in protecting the image from accidental disruption than from malicious attacks. [1] proposed using a signature line or box as a method of reorienting.

In this paper, however, we suggest an additional step that can be taken during the encoding process that allows us to almost flawlessly recover from such effects without a signature line or box. In addition, the method allows the watermark to be partially recovered even if the cropping cuts into the 'foreground' portion of the image slightly.

## 2. PROPOSED METHOD

There are several classes of binary watermarking schemes that work by altering low-level features. Of these, one category works by changing or 'flipping' the values of selected pixels to meet certain criteria, for example, setting the parity of the number of dark pixels in each subsection of the image. Most of these select pixels according to some rubric that orders them according to how obvious it will be to the human eye that they have been flipped. We will borrow [1]'s terminology and call pixels that can be flipped without producing a visible change 'flippable'. Each pixel receives a "flippability score" based on some rubric, with the pixels that can be most easily changed receiving the highest scores and pixels that cannot be easily changed – such as those in a large field of pixels that they share a color with – receive very low flippability scores. [1] and [3-10]

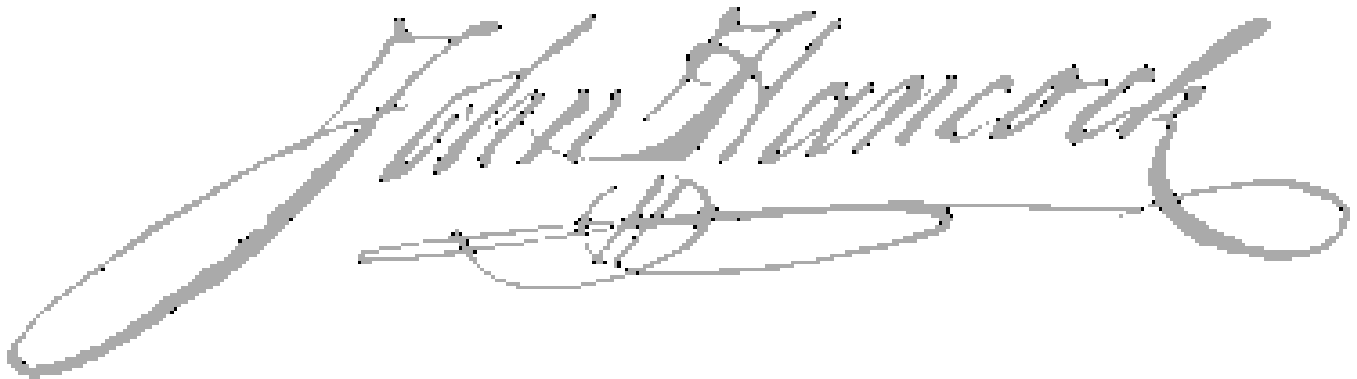
use rubrics based on the colors and connectivity of the adjoining pixels and on the shape of the contour of the object, respectively to order the pixels by flippability. [Picture of very flappable pixels on this page]. Any binary image digital watermarking scheme that orders the pixels by flippability in just about any manner can be adapted to make use of this method. Such methods suffer from a problem when implanting the watermark in that not all areas of an image are equally suitable for embedding; many images have large areas of one color, generally the background color. We refer to the rectangular area which has edges defined by the topmost, leftmost, rightmost, and bottommost pixels of the foreground color as the *content area*. [Picture of content area on this page.] Even if only the content area is used to embed information, it is still likely that there are some areas with no pixels of high flippability score. One solution to this is to shuffle the pixels in the image per [the book with the shuffling information in it]. Unfortunately, this increases vulnerability to cropping greatly if the shuffling map is stored in absolute coordinates and moderately if it is stored in terms of the edges of the content area, if the cropping alters the edges of the content area.

watermarking technique in question. For our example, we will assume a system similar to that found in [Wu paper], where a pixel's flippability is based on the pixels around it, but the general scheme can be extended to a number of other variant schemes.

Pixels suitable for anchor pixels have the following properties:

First, a pixel selected as an anchor pixel should be one that is unlikely to be cropped away. For this reason, we will not choose any pixel within  $j$  pixels of the edge of the content area, where  $j$  is a fairly small positive number, and we will not use any pixels outside of the content area as anchors.

Second, an anchor pixel should be highly unlikely to be altered by watermarking. A threshold  $t$  is selected. Pixels are then selected to be anchors if their flippability scores are less than or equal to  $t$  and all other pixels relevant for determining their flippability scores are also less than or equal to  $t$ . [Pictures of highly unflappable pixels on this page]. In the [Wu Paper] method, this is primarily the pixels that are 8-connected to the pixel in question. In rare cases with certain classes of images this might produce very few



The method works as a nearly autonomous separate step inserted into the encoding process; it can also be performed at any time on the original image, before or after watermarking, to get the anchor point information necessary to recover from a translation or cropping. It works by using anchor points to locate a center focus point. Once the center focus point is found, the image can be 'uncropped' or 'detranslated' automatically.

## 2.1. Anchor Selection

By selecting anchor points that are unlikely to be changed or have been changed by watermarking, we can acquire a valid set of anchor points regardless of which form of the picture as available – watermarked or unwatermarked. The precise method for doing this will depend on the

anchor points; the  $t$  requirement can simply be raised for the pixels that surround the pixel in question.

Once an anchor pixel is found, it is necessary to preserve two pieces of information about it – its position relative to the focus pixel, and its color. By scanning the entire image – either before or after watermarking – we can select any number of anchor points we like. Most binary images that take the form of a line drawing, text, or a signature will generate a very large number, as they tend to have large empty areas – that is, areas that are filled with the background color and which contain a large number of pixels of the very lowest possible flippability score. If it is necessary to save storage space by storing only a small number of anchors, it is best to store primarily anchors of whichever color is less represented (often the foreground color) and to store anchors from all areas of the content area. Generally only a modest number of anchors are

necessary, and storing a large number of redundant anchors actually considerably slows down recovery.

The focus pixel can be any pixel in the picture, but the focus auditioning process, discussed below, is simplest if the focus pixel is one that is impossible to crop away. The pixel at the center of the content area is a good choice for the majority of image classes. When the shuffle map is created, the pixel locations should be recorded in relation to the focus pixel, not in relation to the edge of the image.

## 2.2. Image recovery from anchors

We have stored a number of anchor pixels in terms of their location relative to a selected focus pixel. To recover the image from the crop attempt, we begin a process of 'auditioning' focus pixels. The cropped image is scanned over, and for each point, the following process occurs:

For each anchor point, check to see if that anchor point's color would match if the candidate point was the focus point. If it would match, that point's score increases. If it is outside of the frame, the score is not affected. If it would be a mismatch, the score is either reduced or left alone. (In practice, it does not matter which.) After each candidate pixel is examined in this way, whichever had the highest score is used as the focus pixel. The image is realigned by using the found point as the focus point for purposes of deshuffling.

## 3. RESULTS

In all experiments conducted, the system was able to recover from cropping and translation perfectly. Crops that cut into the content area by more than a few pixels destroy the hidden information, but can still be recovered from. It is never necessary with this system to include other artifacts in order to recover from a crop or translation.

## 4. FURTHER EXPERIMENTS

One great weakness of systems described in this paper is that they are very vulnerable to crops that cut into the content area itself. A system that was even marginally robust against such decay would be a large improvement.

## 5. REFERENCES

[1] M. Wu and B. Liu: "Data Hiding in Binary Image for Authentication and Annotation", *IEEE Trans. on Multimedia*, vol. 6, no. 4, pp.528-538, August 2004.

[2] K. R. Castleman, *Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[3] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—a survey," *Proc. IEEE*, vol. 87, pp. 1062–1078, July 1999.

[4] F. Hartung and M.Kutter, "Multimedia watermarking techniques," *Proc. IEEE*, vol. 87, pp. 1079–1107, July 1999.

[5] I. J. Cox, M. L. Miller, and J. A. Bloom, *Digital Watermarking*. San Mateo, CA: Morgan Kaufmann, 2001.

[6] I. Cox, J. Kilian, T. Leighton, and T. Shamoan, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. Image Processing*, vol. 6, pp. 1673–1687, Dec. 1997.

[7] C. Podilchuk and W. Zeng, "Image adaptive watermarking using visual models," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 525–538, May 1998.

[8] M. Wu, H. Yu, and B. Liu, "Data hiding in images and videos: Part II—Designs and applications," *IEEE Trans. Image Processing*, vol. 12, pp. 696–705, June 2003.

[9] M. Wu and B. Liu, "Watermarking for image authentication," in *Proc. IEEE Int. Conf. Image Processing (ICIP'98)*, vol. 2, Chicago, IL, 1998, pp. 437–441.

[10] M. M. Yeung and F. Mintzer, "An invisible watermarking technique for image verification," in *Proc. IEEE ICIP'97*, vol. 2, Santa Barbara, CA, 1997, pp. 680–683.