Watermarking Source Code

version 0.4

Peter Meerwald

Dept. of Scientific Computing, University of Salzburg
Jakob-Haringer-Str. 2, A-5020 Salzburg, Austria
`mailto:pmeerw@cosy.sbg.ac.at`
`http://www.cosy.sbg.ac.at/~pmeerw/Watermarking`

Abstract

This package provides source code for some watermarking algorithms in hopefully portable C code. The programs can be used to study watermarking techniques, perform comparative robustness tests and develop new attacks against embedded watermarks.

However, the provided programs are by no means suitable for real-world application (i.e. copyright protection) and the code solely serves some educational purpose.

Contents

1 Introduction

Academic research in the watermarking field has grown dramatically since approximately 1995. But surprisingly, source code for the proposed watermarking schemes has not been made available. The reason is most likely the security of many watermarking systems lies at least to some extent in the embedding and detection algorithm itself, and not in the keys used – violating the Kerckhoff principle [1].

With the availability of public robustness test for watermarking algorithms, StirMark [4, 3, 5], Unzign[1] and very recently Checkmark[2], the situation begins to improve. Now it is possible to measure the performance of watermarking systems.

In order the compare and evaluate new embedding and detection techniques, it is also necessary to have some reference implementations of the older, now often called classical schemes. In this work, we provide some implementations of watermarking schemes, some of which can be considered 'classical'.

It was the goal to capture the main ideas of the proposed algorithms, as layed out in the respective papers. This is clearly not an easy task as some papers do not disclose all details or state which particular parameters were used to obtain the results outlined in the communications.

I am very interested in hearing your comments, complaints and suggestions regarding this software. Moreover, if you have source code for a watermarking scheme not yet covered or some useful utility I would be happy to include your code in this distribution. Please see the contact information at the top of this document.

If you use the accompanying code, please cite my thesis:

> Peter Meerwald, Digital Image Watermarking in the Wavelet Transform Domain, Master's Thesis, Department of Scientific Computing, University of Salzburg, Austria, January 2001.

2 Software

Most of the software provided herein was written by myself, as part of my Master thesis. Some contributions were made by Vassilis Fotopoulos[2]. The software in

[1] `http://www.altern.org/watermark`
[2] `mailto:vfotop1@physics.upatras.gr`

the archive is organized in the following sub-directories:

Fotopoulos/ contains contributions by Vassilis Fotopoulos

Meerwald/ contains my work

images/ contains the Lena image in PGM format; the default parameters of most algorithms are tuned to work best with that image

linux_bin/ the place where the Linux executables are stored in the binary distribution

win32_bin/ the place where Windows 32-bit executables are distributed; tested with Windows NT 4.0 only

make/ contains the `Makefile` options to build the code on supported platforms

For the purpose of this software package, a watermarking system comprises four parts, namely: signature generation, watermark embedding, watermark extraction and signature comparison or detection – with the exception of Vassilis's code; there are only cast and test programs (corresponds to watermark embedding and detection). Signature is used more less as a synonym for mark and can be thought of as the payload (at least for some schemes :-).

All programs only accept the image in NetPBM format and will also produce only NetPBM-format files (see section 4.1). Unfortunately, most programs have only been tested with 8-bit gray-scale images of size 512×512.

In order to simplify batch testing, the programs allow to read either from a file, e.g.

```
wm_cox_e -s cox.sig image.pgm
```

or from standard input, i.e.

wm_cox_e -s cox.sig $<$ **image.pgm**

The output is usually written to standard output, i.e.

```
wm_cox_e -s cox.sig image.pgm > wm_image.pgm
```

unless redirected to a file, e.g.

```
wm_cox_e -s cox.sig -o wm_image.pgm image.pgm
```

2.1 Featured algorithms

Currently it includes the following watermarking algorithms

- Bruyndonckx [bruyn], refer to

  O. Bruyndonckx, Jean-Jacques Quisquater, and Benoit M. Macq. Spatial method for copyright labeling of digital images. In IEEE Workshop on Nonlinear Signal and Image Processing '95, Thessaloniki, Greece, pages 456 - 459, 1995.

- Corvi, refer to

  Marco Corvi and Gianluca Nicchiotti. Wavelet-based image watermarking for copyright protection. In Scandinavian Conference on Image Analysis SCIA '97, Lappeenranta, Finland, June 1997.

- Cox, refer to

  Ingemar J. Cox, Joe Kilian, Tom Leighton, and Talal G. Shamoon. Secure spread spectrum watermarking for multimedia. In Proceedings of the IEEE ICIP '97, volume 6, pages 1673 - 1687, Santa Barbara, California, USA, 1997.

- Dugad, refer to

  Rakesh Dugad, Krishna Ratakonda, and Narendra Ahuja. A new wavelet-based scheme for watermarking images. In Proceedings of the IEEE International Conference on Image Processing, ICIP '98, Chicago, IL, USA, October 1998.

- Fridrich (2. scheme), refer to

  Jiri Fridrich. Combining low-frequency and spread spectrum watermarking. In Proceedings of the SPIE Symposium on Optical Science, Engineering and Instrumentation, San Diego, USA, July 1998.

- Kim, refer to

  Jong Ryul Kim and Young Shik Moon. A robust wavelet-based digital watermark using level-adaptive thresholding. In Proceedings of the 6th IEEE International Conference on Image Processing ICIP '99, page 202, Kobe, Japan, October 1999.

- Koch, refer to

  Eckhard Koch and Jian Zhao. Towards robust and hidden image copyright labeling. In Proceedings of the IEEE International Workshop on Nonlinear Signal and Image Processing, pages 452 - 455, Halkidiki, Marmaras, Greece, June 1995.

- Wang, refer to

  Houng-Jyh Wang, Po-Chyi Su, and C.-C. Jay Kuo. Wavelet-based digital image watermarking. Optics Express, volume 3, pp. 497, December 1998.

- Xia, refer to

  Xiang-Gen Xia, Charles G. Boncelet, and Gonzalo R. Arce. Wavelet transform based watermark for digital images. Optics Express, volume 3, pp. 497, December 1998.

- Xie, refer to

  Liehua Xie and Gonzalo R. Arce. Joint wavelet compression and authentication watermarking. In Proceedings of the IEEE International Conference on Image Processing, ICIP '98, Chicago, IL, USA, 1998.

- Zhu, refer to

  Wenwu Zhu, Zixiang Xiong, and Ya-Qin Zhang. Multiresolution watermarking for images and video: a unified approach. In Proceedings of the IEEE International Conference on Image Processing, ICIP '98, Chicago, IL, USA, October 1998.

- Piva/Fotopoulos [cast|test-pv,hart,sub], contribution by Vassilis Fotopoulos, refer to

  M.Barni, F. Bartolini, V. Cappellini, A. Piva. A DCT-Domain System for Robust Image Watermarking, Signal Processing, vol. 66, pp 357 - 372, 1998.

  and

  V. Fotopoulos, A. N. Skodras, A Subband DCT approach to image watermarking, X European Signal Processing Conference, September 4 - 8, 2000, Tampere, Finland.

More algorithms will be added over time, I have implemented about 13 watermarking algorithms in the spatial-, DCT-, and wavelet domain so far.

2.2 Utility programs

A good way to check the effect of a watermarking algorithm is computing the difference image, i.e. subtracting the original image from the watermarked image. Alternatively, one can also have a look at the modified coefficients in the transform domain. The following programs facilitate these tasks:

cmp_pgm compute difference image, PSNR, ...

cmp_dct compute full-frame DCT domain difference image cmp_dct8x8 compute 8x8 block-based DCT difference image

cmp_dwt compute DWT domain difference image

For example, to produce the difference image of two PGM files and compute the PSNR along with some other measures, the following command can be used:

```
cmp_pgm -p -i original.pgm -o diff.pgm watermarked.pgm
```

3 Usage

Note, almost all programs will output usage information if called with the `-h` argument.

3.1 Generating a mark

First, you have to generate an appropriate signature file for the corresponding embedding/detection algorithm; e.g. if you are going to use Cox' scheme, then you would run

```
gen_cox_sig
```

The programs outputs some parameters and a sequence of Gaussian distributed random numbers (which is the watermark sequence). You want to save that into a signature file, so you run

```
gen_cox_sig > cox.sig or
gen_cox_sig -o cox.sig
```

You can influence e.g. the embedding strength that will be used in the embedding step by running

```
gen_cox_sig -a 0.5 > too_strong_cox.sig
```

Usually, the programs for generating a signature will supply reasonable default values for marking a 8-bit gray-scale image of size 512×512.

3.2 Watermark embedding

Watermark embedding is performed with the following command (for our example, we are using Cox' scheme):

```
wm_cox_e -s cox.sig -o cox_lena.pgm lena.pgm
```

The signature file is parsed to obtain the particular watermark sequence and the embedding strength. The watermarked image is written to the file `cox_lena.pgm`. Now it the time to check the perceptual quality of the produced image and also have a look at the difference image (see section 2.2).

3.3 Watermark extraction

To extract the embedded signature, we execute the command

```
wm_cox_d -s cox.sig -i lena.pgm -o cox.wm cox_lena.pgm
```

Since Cox' algorithm is not blind, the original image is needed as a reference to extract the embedded mark. The embedded mark will be stored in `cox.wm`. The original signature, `cox.sig`, is used to get the auxiliary embedding parameter correct (e.g. embedding strength).

3.4 Comparing the mark

The final step is comparing the original signature against the extracted signature. The result here is usually a correlation factor. Values around 0 indicate that the mark has not been found, values around 1

In most programs a analytical detection threshold for some detection probability is not used. Hence, one has to observe the output of the detector for many different keys (around 1000 I'd suggest) to establish a reasonable threshold for detection. A good value to go with initially might be 0.2 which means we claim the watermark detected if the correlation factor is > 0.2.

The appropriate command for comparing the mark is

```
cmp_cox_sig -s cox.sig cox.wm
```

3.5 Batch testing - benchmarking

If you want to run many test you can pipe the images to be do be watermarked (and tested) through the embedder and detector. The programs then act like a filter. Try something like the following in a Unix shell script:

```
gen_cox_sig > cox.sig
for i in *.pgm
do
  wm_cox_e -s cox.sig $i | \
  wm_cox_d -s cox.sig -i $i | \
  cmp_cox_sig -s cox.sig
done
```

4 Recompiling

Note, that most watermark embedding/extraction programs use the built-in random number generator of the C library, i.e. `srandom()` and `random()`. Therefore, if you recompile, chances are that you won't be able to use your images watermarked with the previous version.

The Makefile options for compiling on the different platforms can be found in the `make/` sub-directory of the archive.

4.1 Prerequisites

4.1.1 NetPBM

NetPBM is responsible for image file I/O and provides a definition of a simple image file format along with many image file format filters that allow to convert images to and from NetPBM format.

You need to get and install the NetPBM library at `http://wuarchive.wustl.edu/graphics/graphics/packages/NetPBM/` or `http://netpbm.sourceforge.net`. The library provides `pgm.h` and the appropriate implementation.

4.2 Unix/Linux platform

All programs were developed using Linux and GNU C. The programs should compile and work with all recent versions of Linux and GNU C.

4.3 Win32 platform

The programs were ported to the Windows platform using the Cygwin[3]and Mingw[4] environment. Most notable, the file mode for standard input and standard output has to be set to binary mode. This is accomplished with the `setmode()` or `_fsetmode()` commands.

5 FAQ

Q: How can I report problems?

A: See the contact information at the beginning of this document.

Q: The compiler complains about `pgm.h`?

A: You need to get and install the NetPBM library, see section 4.1.

Q: What is the best algorithm?

A: Depends on your application.

Q: What is the most robust algorithm?

A: Depends on the attack. See some results on `http://www.cosy.sbg.ac.at/~pmeerw/Watermarking`.

Q: I need code for a full-frame DCT?

A: See the files Meerwald/dct.* in the archive.

Q: I need code for a 8x8 block DCT?

A: See the files `Meerwald/dct.*` in the archive.

Q: I need code for the wavelet transform (DWT)?

[3] `http://www.cygwin.com`
[4] `http://www.mingw.org`

A: See the files `Meerwald/wavelet.*` in the archive.

Q: I get the message 'unable to open filter.dat' - what to do?

A: Make sure the file filter.dat is in the current directory or accessible via path/filename specified in the signature file. Use the signature generation command to specify an absolute path if necessary.

Q: I can't compile the code using some Microsoft product?

A: Make your life easier, install GNU software! See section 4.

6 Revision history

version 0.4 (June 21, 2001)

- bug fixes
  - wm_xia_{e|d}.c variable level uninitialized
  - wm_zhu_{e|d}.c variable level uninitialized
  - issue with random() vs. rand() and RAND_MAX in frid2_common.c
- added option to bruyn algorithm to disable block skipping
- added algorithm kim

version 0.3 (June 18, 2001)

- created a nice (?) manual/documentation
- added algorithms by Dugad, Wang, Zhu, Fridrich
- added Makefiles for Win32 platform (mingw32)

version 0.2 (February 22, 2001)

- added contribution by Vassilis Fotopoulos (Piva's algorithm,
- DCT, Hartley and subband domain) - see Fotopoulos/ subdirectory
- stuff moved to Meerwald/ subdirectory
- added Bruyndonckx, Corvi, Koch, Xia, Xie algorithms

version 0.1 (February 18, 2001)

- initial release

7 Legal statement

My license is called "I-don't-care" license: (1) You can do with the accompanying software whatever you want, but don't blame me if it doesn't work or it causes damage. (2) If you think my work is useful, tell me and tell others, but you are not obliged to do so. I suggest not to remove information contained in this other documentation file.

References

[1] Auguste Kerckhoff. La cryptographie militaire. *Journal des sciences militaires*, 9:5 – 38, January 1883.

[2] Shelby Pereira, Sviatoslav Voloshynovskiy, M. Madueno, and Thierry Pun. Second generation benchmarking. In *Proceedings of the 4th Information Hiding Workshop '01*, Portland, OR, USA, April 2001.

[3] Fabien A. P. Petitcolas and Ross J. Anderson. Weaknesses of copyright marking systems. In *Multimedia and Security Workshop at the 6th ACM International Multimedia Conference*, pages 55 – 61, Bristol, England, 1998.

[4] Fabien A. P. Petitcolas and Ross J. Anderson. Evaluation of copyright marking systems. In *Proceedings of IEEE International Conference on Multimedia Computing and Systems '99*, volume 1, pages 574 – 579, Florence, Italy, June 1999.

[5] Fabien A. P. Petitcolas, Ross J. Anderson, and Markus G. Kuhn. Attacks on copyright marking systems. In David Aucsmith, editor, *Information hiding: second international workshop*, volume 1525 of *Lecture notes in computer science*, Portland, OR, USA, April 1998. Springer Verlag, Berlin, Germany.