PROJECTS IN WATERMARKING AND IMAGE AUTHENTICATION

Kelvin Harper

REU 2009 at Utah State University
Email: kelvinharper@gmail.com

ABSTRACT
The purpose of this paper is to discuss further research and refinement in the area of watermarking and tampering detection of digital images. The concept of this watermarking is to embed a watermark that is distributed throughout the image, and that allows for JPEG type compression and still prove authentic. The watermarking scheme is secure and visually undetectable. Experimental results demonstrate a workable solution, with suggestions for improvement, particularly with the embedding and extracting times required to obtain the watermark.

1. INTRODUCTION
The capabilities of currently used image authentication are severely lacking at this time. There is an ever-growing need for legitimately original and unaltered digital images in a variety of fields. Law enforcement and courts need images used as evidence to be verified as un-tampered. Scholarly journals have received tampered images and published them as fact. News media has shown and printed images which were tampered in various ways. Of particular concern are “Photoshoped” images in which cut and paste has been used to add or remove objects to make substantial changes.

Email systems have been hacked, and images altered. As such, there is a need for an authentication system that provides for the legitimate owner/creator of an image to embed a watermark. Then for the recipient to be able to authenticate the image as an un-altered original. My project was an implementation of this.
2. RELATED WORK
My research is based upon a paper by Huajian Liu and Martin Steinebach, “Semi-fragile Watermarking for Image Authentication with High Tampering Localization Capability”. This paper proposes embedding a randomly-permutated watermark which is embedded throughout the image, instead of in small blocks. It also researched a capability to back-trace and find local areas which had been tampered with.

3. PROPOSED APPROACH
From my research during this summer, I propose embedding a watermark as a form of steganography (secret message) which will be invisible to the user, and undetectable unless the same algorithms are run to verify its presence. As the first step for processing a grayscale image, the image is read into Matlab and then processed with Matlab DWT2 built-in function. This breaks the image down into 4 sub bands, chiefly into an original, a horizontal, vertical, and diagonal bands of the original image. For example, if the image started as a 512x512 image, dwt2 will break it into 4 sub bands of 256x256.

Next an array of S is created, which runs through each block of the array reassigning random locations for the watermark to be stored. Within the groups, the locations must be a minimum distance apart in order to maintain the localizing ability of the watermark’s tampering detection. This is one of the less efficient pieces of my current code. Next a key is used to permeate the S’, which is then embedded into the image as the watermark. This is the most time consuming part of the code, where even a simple 160x160 grayscale takes at least 6 minutes on a modern 3 Ghz Intel processor. After the watermark is imbedded, a for loop is used to reconstruct the 4 sub bands of the image. The built-in Matlab function IDWT2 is then used to “rebuild” the image and save it to a file. This gives us a watermarked image, ready to be emailed, transmitted in an unsecure environment, and exposed to tampering.
The recipient of the image loads the image file, and goes about authenticating that it is original and in an unaltered form. This is where the second major function of my code comes into play. The “retrieval” function goes through a similar set of steps, this time to extract the watermarked bit from the image, and be to able to compare to the original watermark. It uses the dwt2 function to scan for break into the 4 sub bands, again using the “haar” mode to do the breakdown. It uses the Matlab function to determine the size of the image, then places S in an array. It uses 3 temp vectors to ensure that the groups are sufficiently spread apart, and saves in an array.

A distance and key is again used to permeate an Sp. This time, instead of embedding the into the image, it is extracted to produce the watermark. Several new functions were created to extract the watermark bit and produce it in a usable format that can be compared to the original watermark. The retrieval function is also rather inefficient, and takes just as long to run as the embedding phase. The Main function has been written to demonstrate these functions, on a 160x160 image for a short processing time.

4. EXPERIMENTAL RESULTS

Workable results came rather late in the implementation of this project. The paper’s heavy use of complex mathematical formulas as well as a steep learning curve on my part of programming in Matlab did slow progress to meet goals. I am proud to say that we have been able to implement the retrieval portion of the project and begin testing effectiveness.

One of the purposes of the project is the ability to use JPEG compression to shrink the file size down to more manageable size, particularly for email purposes. Unfortunately at this point doing JPEG compression within Matlab seems to produce a watermark that is different. In fact we want to detect localized changes in the image. It is capable of detecting those, which is one of the goals.
Other experimental results were the surprisingly long embedding and extracting times needed. Likely the biggest cause of this is due to the creation of arrays that are not fixed-size from the beginning, which requires Matlab to expand then each time a loop runs. This is highly inefficient and needs restructuring to actually resolve. The lengthy run times are a significant issue for useful implementation of this software. Most clients cannot be expected to wait hours for the authentication of one or just a few images, especially as resolutions far above the 512x512 that were our primary test set are in use.

5. CONCLUSIONS
Matlab seems like a reasonable tool to teach elementary things with arrays and image processing, but due to the compiled nature seems poorly suited for efficient software. Not only does it tend to consume enormous memory and processor time, it also has a steep learning curve for those new to it. I have done significant programming in C++ and it seems far better suited for this type of application.

For best efficiency in programming, ensure that arrays have their size pre-defined before they are filled, otherwise they waste time. Also, too many calls between functions tend to waste time.

It would have been nice to test the proposed image tampering localization that the paper described. It seems logical that it would, though coding it in Matlab seems like a nightmare.

6. FUTURE WORK
The randomized permutation technique of embedding the watermark is a nice technique, but I do not think to be the most efficient. Since the watermark is undetectable anyway, it would be possible to use a hidden “key” to determine a start point in the image and embed a simple string watermark that could run sequentially.

This would greatly cut down on the length and complexity of the code. It could still detect tampering schemes in localized areas, since that
part of the watermark would still be distant from the location it “protected”. This may well be a more feasible form of watermarking, at least until computer processors are exponentially faster than today, though it is likely that image resolutions will also increase.

11. REFERENCES

I personally thank Dr Qi for answering many questions as I faced a steep learning curve with image processing and a new tool to me, Matlab. Many thanks also to Xing for coding some of the functions, and helping me understand how to code the others.

