Hiding Behind Corners: Using Edges in Images for Better Steganography

Kathryn Hempstalk
University of Waikato
Department of Computer Science
Hamilton, New Zealand
kah18@ waikato.ac.nz

ABSTRACT
Digital steganography involves taking an electronic file and hiding it inside another electronic file. Current digital techniques do not tend to take the cover (what the message is hidden in) into account, and thus leave telltale marks on the stego-object (what the cover becomes after hiding information). Since these marks will cause people other than the intended recipient to suspect a hidden message, it is important to make them as inconspicuous as possible. This paper investigates using the cover's original information to avoid making marks on the stego-object, by hiding raw electronic files inside digital colour images. Steganalysis and machine learning is then used to evaluate the hiding process in order to ensure the information is hidden in the best possible way.

Categories and Subject Descriptors
I.4.9 [**Image Processing and Computer Vision**]: Applications – *Steganography.*

General Terms
Algorithms, Design, Security.

Keywords
Steganography, steganalysis, LSB embedding, image filtering.

1. INTRODUCTION
Hiding information by embedding secret data into an innocuous medium is often referred to as steganography. Steganography can be applied electronically by taking a message (a binary file) and some sort of cover (often a sound or image file) and combining both to obtain a "stego-object". The stego-object is essentially the cover with its redundant information replaced with the message. Unfortunately, replacing the redundant information with structured message data often results in the stego-object being an augmented version of the cover "marked" by the hidden data – and this makes it easy to suspect the presence of steganography. Most of these marks can be attributed to the hiding algorithm's lack of concern for the cover's content. If the cover's original content were taken into account then the message could be concealed in areas of the cover where it would be less likely to leave marks.

Previous attempts at achieving this goal have required the user to provide the original cover as well as the stego-object. The best areas to hide are first identified in the original cover, then these areas are mapped across to the stego-object and the hidden information is retrieved. The original cover must be provided because the information overwritten in the message hiding process may have been used to identify the best hiding areas. However, to provide the original object is not secure, because taking the differences between the two objects would be enough to suspect the existence of (and in some cases, recover) the hidden information.

This paper investigates an approach that eliminates the need for providing the original object. We use images as a cover medium and introduce two new algorithms based on using image filters to determine effective hiding places. The next section describes these algorithms. Section 3 briefly describes some steganalysis methods used to test the effectiveness of the new algorithms. Section 4 describes the experimentation performed using the algorithms and the steganalysis techniques and Section 5 describes the results of this experimentation. Finally, the paper is concluded in Section 6.

2. ALGORITHMS
The simplest way to hide binary data on an image is to use a lossless image format (such as a Bitmap) and replace the *x* least significant bits of each pixel in scan lines across the image with the binary data. This is not secure as an attacker can simply repeat the process to quickly recover the hidden information. This technique, known here as "BlindHide" because of the way it *blind*ly *hide*s information, is also not good at hiding – the initial portion of the image is left degraded while the rest remains untouched.

A tool known as "Hide and Seek for Windows 95" [1] attempts to get around the security issues in BlindHide by randomly distributing the hidden information across the image. A more modern version of this algorithm, dubbed "HideSeek", is used here. HideSeek uses a random seed (provided by hashing a password) to pick the order in which it will write to the pixels. HideSeek is much more secure than BlindHide, but does not necessarily leave the image in a better condition. The noise introduced by HideSeek is randomly placed and often causes the resulting stego-image to look speckled.

The noise left behind by both HideSeek and BlindHide is much more noticeable to the naked eye in large blocks of colour – where a single modified pixel stands out amongst its uniform neighbours. This is expressed explicitly by the Laplace formula [2]. The Laplace formula simply measures the difference between a pixel and its four touching neighbours. The magnitude of the formula increases with the colour variation and this can be used to detect steganography by counting the number of pixels at a given magnitude. Untouched images are more likely to contain a large number of pixels with zero magnitudes since there is no reason for small random variations to occur in large blocks of colour. Stego-images often contain small variations, and can be detected easily

by examining Laplace magnitude counts. Therefore it seems reasonable to suggest if a hiding algorithm were able to use the Laplace formula during embedding, it would be able to hide the information in a less noticeable way.

To do this, we introduce the FilterFirst algorithm. FilterFirst uses an edge-detecting filter, such as the Laplace formula, to find the areas of the image where there are pixels that are the least like their neighbours. It hides in the highest values of the *filter first*. Since we are only changing the x least significant bits for steganography, we can use the y most significant bits for the filter. Here x and y are integers where $1 <= x <= 7$ and $y = 8 - x$. We can guarantee that FilterFirst will be able to retrieve the information from the same pixels it hides in because the bits used in filtering are not changed by the hiding process. FilterFirst eliminates the need to provide any extra information, such as the original image, yet ensures the same pixels are used for hiding and retrieval.

However, FilterFirst is similar to BlindHide in that it is not a secure algorithm. An attacker can repeat the filtering process and retrieve the hidden information with very little effort. Whilst it should be more difficult to identify steganographic images using FilterFirst, it is much easier to retrieve the hidden information than with HideSeek.

To create an algorithm that both hides effectively and securely we combine HideSeek and FilterFirst to create BattleSteg. BattleSteg stands for *Battle*ships *Steg*anography and is based on playing an augmented game of Battleships to determine the best places to hide. In this algorithm, the h% of highest filter values is designated as 'ships'. 'Shots' are randomly picked as pixel positions on the cover image, until a ship is found (known as a 'hit'). When a hit occurs, the next series of shots are clustered around that hit in the hope there are more ships in that area. After i initial shots we return to shooting randomly to prevent huge expanding clusters of shots – which may cause noticeable visual degradation on the stego-image. BattleSteg is overall more likely to avoid pixels in large blocks of colour than HideSeek, yet has a similar amount of security as without the random seed it is impossible to know where to place the shots.

3. STEGANALYSIS

Just as clever techniques have been devised for hiding information, an equal number of clever techniques have been designed to detect the hidden information [3]. These techniques are collectively known as 'steganalysis'. As introduced earlier, the Laplace formula is one such steganalytic method. Two other popular techniques are RS Analysis [4] and Sample Pairs Analysis [5].

RS Analysis makes small modifications to the least significant bit plane in an image then uses these modifications and a discrimination function to classify groups of pixels. The counts of the groups based on the modifications allow the calculation of an estimated embedding rate. Images that do not contain steganography often have a natural embedding rate of up to 3%, whereas images containing hidden information usually have estimated embedding rates which accurately reflects the amount of hidden information.

RS Analysis is a special case of Sample Pairs Analysis, which also uses least significant bit modifications to help calculate an estimated embedding rate. Sample Pairs Analysis utilises finite state machines to classify groups of pixels modified by a given pattern. Both steganalysis techniques are very accurate at predicting the embedding rate on stego-images using least significant bit embedding. Since the two proposed techniques, FilterFirst and BattleSteg, both use least significant bit embedding, we can use RS Analysis and Sample Pairs Analysis to compare them against more traditional techniques such as BlindHide and HideSeek.

4. EXPERIMENTAL DESIGN

Rather than evaluating a set of images by calculating the steganalysis information and checking the values by hand, machine learning is used. The idea behind this is simple – if a machine learner has trouble accurately predicting whether steganography is present, then the steganographic method is more effective for that picture than a method where the machine learner can correctly classify the image. Across many images, the accuracy of all predictions should indicate the effectiveness of one algorithm over another.

For testing in this paper, 100 images were combined with 2 messages. Both messages were of set length, s, one containing random data and the other the text of Moby Dick. Each algorithm was set to write to only the least significant bit of each colour (red, green, blue). 200 stego-images were then obtained by embedding each message in each image using a given steganography technique.

The 200 stego-images and 100 original images then had all their steganalysis values calculated and added to a file. This file contained the estimated embedding rates from RS Analysis and Sample Pairs Analysis, as well as counts of occurrences of each Laplace formula value encountered in every image. The file was then used to train SMO [6][7], a support vector machine learning algorithm from the WEKA Machine Learning Workbench [8].

SMO begins by identifying features that distinguish untouched images from stego-images. The machine learner runs through 10 times 10-fold cross-validation, giving the accuracy values that indicate the effectiveness of the algorithm being tested. More effective algorithms will have lower accuracy rates through cross-validation, as SMO will be unable to determine features that distinguish whether an image contains steganography. This process was repeated for each algorithm and for different values of s.

5. EXPERIMENTAL RESULTS

Using embedding rates varying from 5% to 55% (by which time all algorithms were scoring ~100% accuracy), the four algorithms were evaluated using WEKA's SMO. BattleSteg and FilterFirst were run with two variations – once using the Laplace formula for filtering and once using the Sobel filter [9] (a gradient based edge detection algorithm). As the results in Figure 1 show, the machine learner could do no better predict steganography (the majority for the image set) for FilterFirst for embedding rates up to 10%. The three other algorithms, BlindHide, BattleSteg and HideSeek, all had accuracy rates up at 90% before they even got to a 10% embedding rate.

Figure 1 also shows that FilterFirst is much more effective at hiding than the two traditional algorithms, HideSeek and BlindHide. Unfortunately, BattleSteg did not perform as well as expected, beating only HideSeek. BlindHide appears to sit between BattleSteg and FilterFirst, but should be avoided as a steganographic technique due to its lack of security and ability to often be detectable by the naked eye.

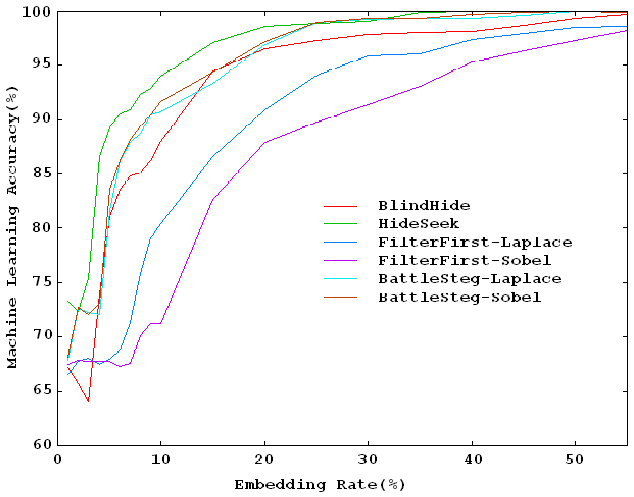

Figure 1: Graph of Embedding Rate versus Machine Learning Accuracy

These results seem to suggest that FilterFirst should be the algorithm of choice for effective steganography. By simply shuffling the order in which the highest filter bits are written to, it becomes a secure and effective hiding technique. A possible alternative to shuffling is to use a fixed proportion of the highest-filter valued pixels and then select the remaining pixels randomly. BattleSteg does attempt to do this, but it cannot guarantee the proportion of high filter valued pixels it writes to. It is possible for BattleSteg to never have a 'hit' whilst using this algorithm.

6. CONCLUSION

This paper has introduced two new techniques for image steganography, FilterFirst and BattleSteg. These two techniques attempt to improve on the effectiveness of the hiding by using edge detection filters to produce better steganography. These techniques have been tested against more traditional image steganography techniques, BlindHide and HideSeek, by using steganalysis methods.

Using a machine learner to predict the use of steganography on images, we have shown that the proposed FilterFirst algorithm is very effective at hiding information. FilterFirst beat all the steganalysis techniques until embedding rates became greater than 7% and performed better than all other steganography algorithms tested. The results indicate that using features of the cover, such as edges, is a better way of hiding information than in scan lines or randomly across the image.

The steganalysis and steganography algorithms discussed in this paper have been implemented into a tool for ease of use during this research. This tool has been made available for free under the Gnu General Public License at: http://diit.sourceforge.net

7. REFERENCES

[1] Maroney, C. Hide and Seek 5 for Windows 95, computer software and documentation, originally released in Finland and the UK, n.d. Downloadable from: http://www.rugeley.demon.co.uk/security/hdsk50.zip

[2] Katzenbeisser, S. and Petitcolas F.A.P. *Information hiding techniques for steganography and digital watermarking*. Artech House, Norwood, MA 02062, USA, 1999.

[3] Wang, S. and Wang, H. Cyber Warfare: Steganography vs. Steganalysis, *Communications of the ACM Volume 47, Number 10*, pp 76-82, 2004

[4] Fridrich, J., Goljan, M. and Du,R. (2001) Reliable Detection of LSB Steganography in Color and Grayscale Images, *Proceedings of ACM Workshop on Multimedia and Security,* Ottawa, October 5, 2001, pp. 27-30.

[5] Wang, Z., Dumitrescu, S. and Wu, X. Detection of LSB Steganography via Sample Pair Analysis, *Information Hiding 2002, LCNS 2578*, pp 355-372, 2003.

[6] Platt, J. *Fast Training of Support Vector Machines using Sequential Minimal Optimization*. Advances in Kernel Methods - Support Vector Learning, B. Schoelkopf, C. Burges, and A. Smola, eds., MIT Press, 1998.

[7] Keerthi, S.S., Shevade, S.K., Bhattacharyya, C. and Murthy, K.R.K. *Improvements to Platt's SMO Algorithm for SVM Classifier Design*. Neural Computation, 13(3), pp 637-649, 2001.

[8] Frank, E. and Witten, I.H. *Data Mining*, Morgan Kaufmann Publishers, San Francisco, USA, 1999.

[9] Green, B. Edge Detection Tutorial, 2002, Available at: http://www.pages.drexel.edu/~weg22/edge.html (May 2005)