SKIN TONE BASED STEGANOGRAPHY IN VIDEO FILES EXPLOITING THE YCBCR COLOUR SPACE

Abbas Cheddad, Joan Condell, Kevin Curran and Paul Mc Kevitt

School of Computing and Intelligent Systems, Faculty of Computing & Engineering
University of Ulster, Londonderry, BT48 7JL, Northern Ireland, United Kingdom
Emails: {cheddad-a, j.condell, kj.curran, p.mckevitt}@ulster.ac.uk

ABSTRACT

Previous studies have shown that Steganography tools such as the S-Tools application outperforms counterpart tools in hiding data in the spatial domain. Another tool called F5 is identified as a robust tool in implementing Steganography in the frequency domain. This paper presents a Steganographic system which exploits the *YCbCr* colour space. *YCbCr* is intended to take advantage of human colour-response characteristics. Results show that our algorithm outperforms both *F5* and *S-Tools*. Moreover, our system demonstrates improved performance in retrieving the hidden data after applying image processing attacks in the form of additive artificial noise. As a performance measurement for image distortion Peak-Signal-to-Noise Ratio (*PSNR*), which is classified under the difference distortion metrics, can be applied to the Stego images. This study also shows that by adopting an object oriented Steganography mechanism, in the sense that we track skin tone objects in video frames, we get a higher *PSNR* value.

1. INTRODUCTION

Steganography is a method that involves hiding a message such as an image or an audio file in a suitable carrier. Changes are made to the carrier which represents the hidden message. If successful then no discernible change is made to the carrier. The carrier can then be sent to a receiver without anyone else knowing that it contains a hidden message. The embedding process is either applied in the image spatial domain, where the Least Significant Bits (*LSB*s) are altered, or in the image frequency domain by altering the *LSB*s of the coefficients. More detail on Steganography can be found in (http://www.infm.ulst.ac.uk/~abbasc/index.html). Next we discuss two popular algorithms.

1.1 S-Tools

S-Tools involves changing the least significant bit of each of the three colours in a pixel in a 24-bit image for example a 24-bit BMP file [1, 2]. The problem with 24-bit images is that they are not commonly used on the web and tend not to stand out (unlike GIF, JPEG, and PNG). This feature is not helpful to Steganography. It involves a pre-processing step to reduce the number of colour entries by using a distance measurement to identify neighbour colours in terms of intensity. After this stage each colour of the dithered image would be associated with two palette entries one of which will carry the hidden data. The software for S-Tools can reduce the number of colours in the image to 256 [1]. The software uses the algorithm developed by Heckbert [3] to reduce the number of colours in an image in a way that will not visually disrupt the image [1, 4]. The algorithm plots all the colours in three dimensions (RGB). It searches for a collection of *n* boxes, which contain all of the colours in one of the boxes. The process starts with the complete 256*256*256 space as one box. The boxes are then recursively subdivided by splitting them in the best possible way [1]. Splitting continues until there are *n* boxes representing the space. When it is finished the program chooses one colour to represent all the colours in each box. The colour may be chosen in different ways: the centre of the box, the average box colour or the average of the pixels in the box. S-Tools and other tools based on LSBs in the spatial domain, take for granted that least significant bits of image data are uncorrelated noise [5].

1.2 F5

F5 was the creation of Andreas Westfeld in 2001; it embeds messages by modifying the DCT coefficients. The central operation done by F5 is matrix embedding (subtraction and matrix encoding) with the aim of reducing the amount of changes made to the DCT coefficients. The algorithm [6] takes *n* DCT coefficients and hashes them to *k* bits, where *k* and *n* are computed based on the original image as well as the secret message length. If the hash value equals the message bits, then the next *n* coefficients are chosen, otherwise one of the *n* coefficients is modified and the hash is recalculated. The modifications are constrained by a threshold to estimate the acceptable hamming distance "*disT*" of *n* DCT coefficients and the original *n* DCT coefficients. This process is repeated until the hash value matches the message bits. A Java application version of the F5 code is publicly available at (http://wwwrn.inf.tu-dresden.de/~westfeld/f5.html). F5 first recompresses the image,

with a quality factor input by the user, after which the DCT coefficients are used for embedding the message. The embedded data should be no more than 14% of the cover's size [6] in order for it to be undetected by visual analysis. F5 and S-Tools scatter the secret message over the whole carrier medium.

2. BRIEF DEFINITION OF THE COLOUR TRANSFORMATION USED

Colour images comprise three channels of colours, namely Red (R), Green (G) and Blue (B). There are two types, one of which is called indexed colour. Its RGB map is represented by three vectors where each intensity pixel points to one raw (R_i,G_i,B_i) where i denotes the i^{th} entry in the 3D colour matrix. Another type is known as true colour where each channel is referenced by a matrix having the same image dimension; the three matrices (RGB) are then blended to yield a true colour output.

YCbCr (*Y*: intensity, *Cb*: Chromatic blue and *Cr*: Chromatic red) is a well known transformation colour space where it is applied in image compression [7] and in object detection (human skin tone[1]) [8]. The transformation invokes a rounding operator to perform a series of arithmetic operations. This phenomenon classifies it as a non-reversible lossy system. The loss of information happens only during the first transformation cycle (Fig. 1) and will show an extremely slight effect, if any, on any additional cycles (Fig. 2). This was based on our own experiments and was highlighted by Domanski and Rakowski in their work [7]. The convergence between the two colour spaces is given by the following system (Note: the transformation formulae for this colour space depends on the used recommendation):

$$YC_bC_r: \begin{cases} Y = 0.299R + 0.587G + 0.114B \\ C_b = 0.56(B - Y) \\ C_r = 0.71(R - Y) \end{cases} \quad (1)$$

Hsu et al. [8] provided a soultion with one of the earliest initiatives that used *Cb/Y* and *Cr/Y* for their aim of face detection in colour images. This method is adopted here as a skin colour discriminator. We report in this work some of our test images which appear in Fig 3. We mainly show results of a frame from an interview with a Spiderman 3 actress and of the commonly used "*lady*" and "*kodim15*" images.

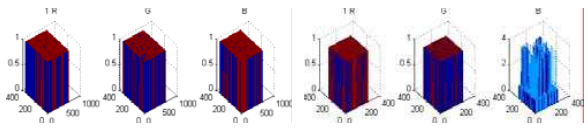

Fig. 1. Cycle 1 - RGB triplet of Spiderman (left) and of Lady (right). Signal error introduced by converting true colour images into *YCbCr* colour space. Even though these changes are small their effect appears in a reasonable compression.

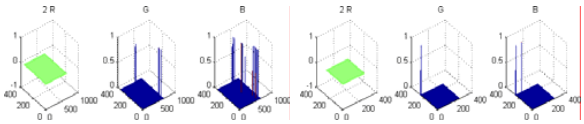

Fig. 2. Cycle 2. RGB triplet of Spiderman (left) and of Lady (right). Signal error introduced by converting the resulting images from Cycle 1 into *YCbCr* colour space. Cycle 2 introduces no further image degradation except the slight error appearing in the Green and Blue (G, B) channels. All additional cycles produce no changes.

Embedding in such a colour space is meaningless if we do not correct these errors and restore the original values of the original image when we inverse transform the *YCbCr* colours to RGB colour space. In other words, we transform only the series of pixels which will carry our hidden data. By this means we can increase the PSNR value by minimizing the errors caused by the transformation.

3. METHODOLOGY

Usage of the $RGB \rightarrow YCbCr$ transformation is twofold; first to segment homogeneous objects in the cover image namely human skin regions in this study, and second to embed our data using the chrominance red (*Cr*). The *YCbCr* space can remove the correlation of R, G, and B in a given image. This phenomenon is what interests us as less correlation between colours means less noticeable distortion. There are different algorithms exploiting different colour spaces to detect human skin tone in colour images [8, 9]. In our approach, the concentration on skin tone is motivated by some interesting applications of the final product. For instance, to combat the use of forged passport documents or national identity cards, a security measure would be to embed individuals' information in their photos. This can also reduce the cost of chip production since many contemporary identity cards use chips; moreover, it enhances the portability as the decoding phase can take place anywhere using a tiny applet application installed on mobile devices.

Based on our experiments we found that embedding into these regions produces less distortion to the carrier image compared to embedding in a sequential order or even in a noise-like fashion as adopted by S-Tools (see Fig. 3). In addition to this, our algorithm yields a robust output against reasonable noise attacks and translation (Fig. 4). Future work is planned for addressing rotation caused distortion. Robustness against noise is due to our selection of featured points, surviving *YCbCr* compression, which are salient dot patterns in images. The resistance to geometric distortion is feasible since, unlike S-Tools and F5, when we select skin tone blobs we can detect eye coordinates which act as our reference points to recover the initial position and orientation and thus make our method invariant to both

[1] Skin tone refers to variations in colour that characterise human skin.

rotation and translation. The next section highlights the statistical measurement used to contrast our work to S-Tools and F5.

3.1 Some Important Notes on the PSNR Calculation

As a performance measurement for image distortion, the well known Peak-Signal-to-Noise Ratio (PSNR) which is classified under the difference distortion metrics is applied on the Stego and the Original images. It is defined as:

$$PSNR = 10\log_{10}\left(\frac{C_{\max}^2}{MSE}\right) \quad (2)$$

where MSE denotes Mean Square Error given as:

$$MSE = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}\left(S_{xy} - C_{xy}\right)^2$$

and $C_{\max}$ holds the maximum value in the original image, for example:

$$C_{\max} \leq \begin{cases} 1 & \textit{in double precision intensity images} \\ 255 & \textit{in 8-bit unsigned integer intensity images} \end{cases}$$

x and y are the image coordinates, M and N are the dimensions of the image, S_{xy} is the generated Stego image and C_{xy} is the cover image.

PSNR is often expressed on a logarithmic scale in decibels (dB). PSNR values falling below 30dB indicate a fairly low quality (i.e., distortion caused by embedding can be obvious); however, a high quality Stego should strive for 40dB and above.

Many authors take the above values (1, 255) as the default values for $C_{\max}$, in binary and 8-bit images respectively, regardless of the range of the examined intensity values but it can be the case for example that an 8-bit original image has its values range only from 3 to 230 and thus its $C_{\max}$ would be 230. The PSNR is a universal formula which can be straightforwardly applied when we are dealing with gray-scale images; however, one can face a problem when confronting true RGB colour images. Some authors generate the average MSE of the three colour channels [10, 11], MATLAB, on the other hand, advises that the RGB model be completely converted into *YCbCr* colour space where the *Y* component is used to calculate the PSNR [12].

3.2 Experiments

Several simulations were performed to evaluate the performance of the proposed system. A set of RGB images were used for this purpose; however, we show examples of three images (one of which is a frame grabbed from a video) with two different binary secret images (Fig. 3). It is worth noting here that the frame appearing in Fig. 3 (top) is grabbed from a 16 sec duration video of size 2.76 MB and having 485 frames. The approximate embedding capacity would be about 34% of the total file size or 0.9407 MB.

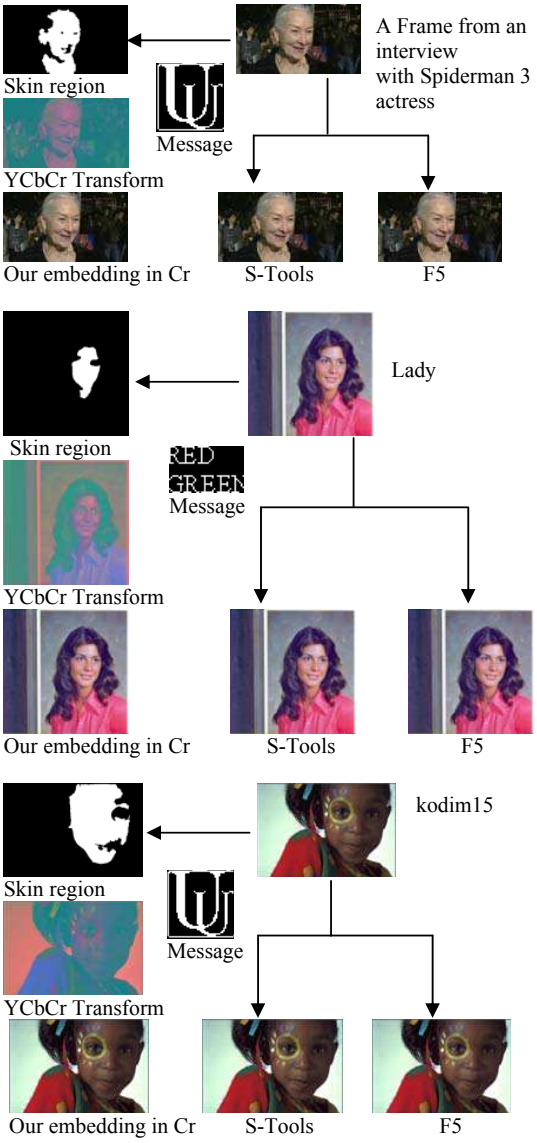

Fig. 3. Embedding results.

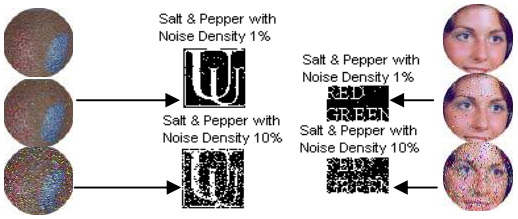

Fig. 4. Results of applying Salt&Pepper noise. Note that S-Tools and F5 are not reported here for they cannot tolerate any kind of alteration to the Stego image.

4. DISCUSSION

The experimental results are promising as can be concluded from Table 1. Notice that our method maintains higher PSNR despite the size of embedded bits which is more than that of S-Tools. Nevertheless, the deficiency of the proposed framework is the limited payload caused by our careful selection of salient points that can resist *YCbCr* compression and because of the region based spirit that the method follows. An obvious workaround would be to target video files, and this forms one of our objectives for the following reasons:

1- Video files provide the ideal scenario for our proposed method to compensate for the limited payload by providing a high-dimensional visual data embedding carrier medium.

2- Intra-frame and Inter-frame properties in videos provide a unique environment to deploy a secure mechanism for image based Steganography. We could embed in the first frame e.g., #116 an encrypted password and a link to the next frame holding the next portion of the hidden data in the video. Note this link does not necessarily need to be in a linear fashion (e.g., frames $116 \xrightarrow{embed} 24 \xrightarrow{embed} 1 \ldots \xrightarrow{embed} n$).

3- Videos are one of the main multimedia files available to public on the net thanks to the giant free web-hosting companies (e.g., *YouTube*, Google Videos, etc). Every day a mass of these files is uploaded online and human factors are usually present. *YouTube* has about 10 million videos, with more than 65 thousand new videos added daily [13].

TABLE I
PERFORMANCE COMPARISON BETWEEN OUR METHOD (Cr), S-TOOLS (S-T) AND F5.

Lady vs. Red Green				
	PSNR AMSE	PSNR (*Y*) *	Embedded Bits	Size Original /Stego-KB
Cr	67.3257	Inf **	1368	192/102
S-T	64.1924	64.0762	1624	192/192
F5	45.6980	53.9535	2384	51.6/49.1
Spiderman vs. UU				
	PSNR AMSE	PSNR (*Y*)	Embedded Bits	Size Original /Stego-KB
Cr	70.3956	Inf	2264	315/222
S-T	69.5629	68.7397	1688	660/ 660
F5	49.4457	54.5660	3600	101/102
kodim15 vs. UU				
	PSNR AMSE	PSNR (*Y*)	Embedded Bits	Size Original /Stego-KB
Cr	73.3350	Inf	2264	598/604
S-T	72.0089	72.1096	1688	1120/1120
F5	47.4984	54.3740	3600	281/ 273

(*) PSNR calculated based on *Y (YCbCr)*, refer to the note on Sec 3.1. (**) Infinity, this indicates the Stego image is perceptually identical to the Original.

5. CONCLUSION

In this paper we propose a new colour image Steganography method which outperforms S-Tools and F5 in many aspects. A number of experiments were discussed and a table of comparison results was given. In previous work we tested this adaptive algorithm in the wavelets domain [14] and for future work, we are considering the use of face features as reference points to recover from any rotational distortion.

REFERENCES

[1] Wayner, P., (2002). Disappearing Cryptography. 2nd ed. USA: Morgan Kaufmann Publishers.

[2] Johnson N, F., Zoran D, Sushil J. (2001). Information Hiding: Steganography and Watermarking-Attacks and Countermeasures. Kluwer Academic Publishers.

[3] Heckbert P. (1982).Colour Image Quantization for Frame Buffer Display. Proc of SIGGRAPH 82, 1982. pp 297-307.

[4] Martin, A., Sapiro, G. and Seroussi, G., (2005). Is Image Steganography natural? IEEE Trans on Image Processing, 14(12): pp. 2040-2050, 2005.

[5] Westfield A and Pfitzmann, A., (1999). Attacks on Steganographic Systems Breaking the Steganography Utilities EzStego, Jsteg, Steganos and S-Tools and Some Lessons Learned. Proc of 3rd International Workshop on Information Hiding, IH'99. LNCS 1768. pp. 61- 76, 1999.

[6] Westfeld, A. (2001). F5-steganographic algorithm: High capacity despite better Steganalysis. LNCS 2137. 2001, pp. 289-302.

[7] Domanski, M and Rakowski, K. (2001). Near-Lossless Colour Image Compression for Multimedia Applications. Proceedings of EURASIP ECMCS 2001. Budapest: September 13 2001, pp. 181-184.

[8] Hsu R.L., Abdel-Mottaleb M. and Jain A. K. (2002)."Face Detection in Color Images".IEEE Transactions on Pattern Analysis and Machine Intelligence. 24 (5), pp 696-706, 2002.

[9]Osamu Ikeda. (2003). Segmentation of Faces in Video Footage Using HSV Color for Face Detection and Image Retrieval. Proceedings International Conference on Image Processing, 2003. Volume 3, 14-17 Sept. pp.III-913-916.

[10] Saenz M, Oktem R, Egiazarian K and Delp E., (2000). Colour image wavelet compression using vector morphology. Proc. of the European Signal Processing Conference, September 5-8 2000.

[11] Yuan-Hui Yu, Chin-Chen Chang, Feng Chia,Iuon-Chang Lin. A. (2007). New steganographic method for colour and grayscale image hiding. Computer Vision and Image Understanding. 107, (3). September 2007, pp: 183-194.

[12]http://www.mathworks.com/access/helpdesk/help/toolbox/vipblks/index.html?/access/helpdesk/help/toolbox/vipblks/ref/psnr.htm. Accessed on 28-11-2007 at 18:55.

[13] Zisserman A. Content Based Retrieval. Visual Geometry Group. University of Oxford. Seminar at the Vision Video and Graphics (VVG) EPSRC Summer School 2007, University of Bath, UK.

[14] Cheddad A., Condell J., Curran K. and Mc Kevitt P. (2008). Biometric Inspired Digital Image Steganography. Proc of the 15th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'08). pp. 159-168. Belfast, UK.