Steganography in a Video Conferencing System*

Andreas Westfeld[1] and Gritta Wolf[2]

[1] Institute for Theoretical Computer Science
[2] Institute for Operating Systems, Databases and Computer Networks

Dresden University of Technology
D-01062 Dresden, Germany
{westfeld, g.wolf}@inf.tu-dresden.de

Abstract. We describe a steganographic system which embeds secret messages into a video stream. We examine the signal path which typically includes discrete cosine transformation (DCT) based, lossy compression. Result is the technical realisation of a steganographic algorithm whose security is established by indeterminism within the signal path.

1 Introduction

The escalation of communication via computer network has been linked to the increasing use of computer aided steganography [5,6]. Steganographic methods usually hide ciphered messages in other, harmless-looking data in such a way that a third person can not detect or even prove this process. Examples for information hiding exist for digital image files, audio files, and in background sounds of phone calls [2]. There are more than 20 programs on the Internet (for examples see the list below).

- S-Tools by Andy Brown embeds data as least significant bits in audio files (.wav) or as least significant bits of the RGB color values in graphic files (.bmp). A third method hides data in free sectors of diskettes. Several symmetric encryption methods (DES, IDEA, ...) are offered for additional encryption of the secret data [9].
- Jsteg by Derek Upham embeds data in JFIF images. It overwrites the least significant bits of the coefficients [8,10].
- Hide and Seek by Colin Maroney hides data (encrypted with IDEA) in GIF files [11].
- PGE (Pretty Good Envelope) by Roche-Crypt packs data in GIF or JPEG files. The use of an additional secure encryption method is recommended [12].
- Mandelsteg by Henry Hastur calculates a GIF fractal from a file. The resulting images are very similar. Differences can only be seen when comparing their color values [13].

* This work is sponsored by the German Federal Ministry of Education, Science, Research and Technology (BMBF).

David Aucsmith (Ed.): Information Hiding 1998, LNCS 1525, pp. 32–47, 1998.

- Stego by John Walker transforms any file to a nonsensical text by means of a free choosable dictionary.
- Texto by Kevin Maher transfers files into poetic English sentences (comparable with stego, which produces nonsensical texts) [14].

Data camouflage is also used for compatible enlargement of norms, such as stereophony, color TV, videotext, traffic control system (TCS), and radio data system (RDS) at FM radio.

This paper does not deal with watermarking systems at all [1].

2 Video Conferencing Systems

Video conferences use compression algorithms to ensure an acceptable quality even on low data rate systems like ISDN. Usually, compression methods are lossy which means that the reconstructed image is not identical with the original.

The video conference used for the implementation of the steganographic system presented in this paper works on the H.261 standard. This is the most common standard for compression in video conferences and is recommended by the *Comité Consultatif International Télégraphique et Téléphonique*[1] (CCITT) [3]. In Fig. 4, we can see the points for embedding and extracting within the H.261 information flow.

Compression and data embedding have contrary goals. For data embedding we need a carrier that allows the possibility of unnoticeable modifications. Signal noise and irrelevance are common examples for it. Compression methods try to remove signal noise and irrelevance. The better a signal is compressed, the less possibilities for data embedding we have. In section 5 we investigate a typical signal path for data embedding.

3 Discrete Cosine Transformation

The steganographic algorithm described in section 6 embeds data in transformed blocks. Therefore we describe the transformation process used in this video conferencing system.

A suitable transformation is a means to separate essential information (visible for the human eye) from marginal parts (invisible for the human eye) of the image. The subsequent quantization removes the insignificant parts of the image. The transformation employed has to be invertible in order to regain the essential parts of the image.

Many digital video conferencing systems, for instance based on the standards H.261, M-JPEG, MPEG, use the two-dimensional discrete cosine transformation (DCT). It transformes an image of 8×8 pixels with $8 \cdot 8 = 64$ brightness values $F(0,0) \ldots F(7,7)$ into 64 values (so-called DCT coefficients) $f(0,0) \ldots f(7,7)$ (see Equation 1). The transformation causes no significant loss (rounding errors

[1] the former CCITT is now the International Telecommunication Union (ITU)

only). The retransformed image results from back transformation of the DCT coefficients (see Equation 2). It can also be understood as linear combination of the DCT coefficients (see Equation 3 and Fig. 2) with the DCT base images $B_{k,n}$ (see Fig. 1).

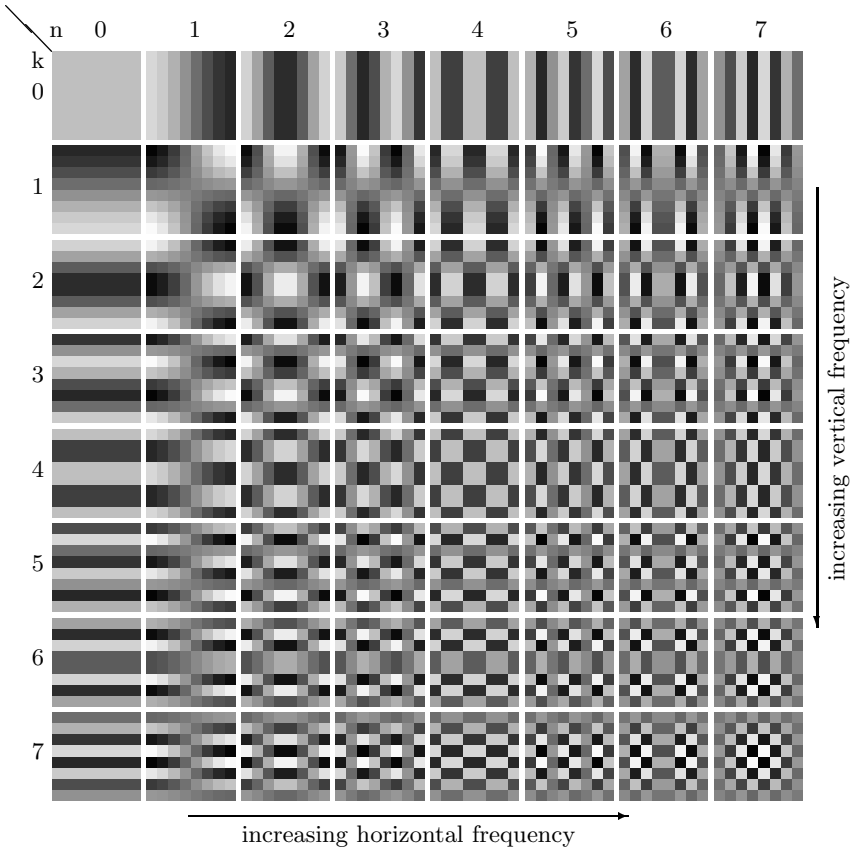


Fig. 1. DCT base images $B_{k,n}$

$$f(k,n) = \frac{C(k)}{2}\frac{C(n)}{2}\sum_{x=0}^{7}\sum_{y=0}^{7} F(x,y)\cos\left(\frac{\pi(2x+1)k}{16}\right)\cos\left(\frac{\pi(2y+1)n}{16}\right) \quad (1)$$

$$F(x,y) = \sum_{k=0}^{7}\sum_{n=0}^{7}\frac{C(k)}{2}\frac{C(n)}{2} f(k,n)\cos\left(\frac{\pi(2x+1)k}{16}\right)\cos\left(\frac{\pi(2y+1)n}{16}\right) \quad (2)$$

$$F(x,y) = \sum_{k=0}^{7}\sum_{n=0}^{7} f(k,n)B_{k,n}(x,y) \quad (3)$$

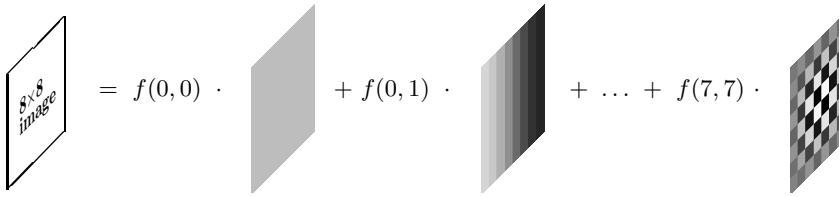


Fig. 2. Presentation of an 8×8 image by 64 base image parts

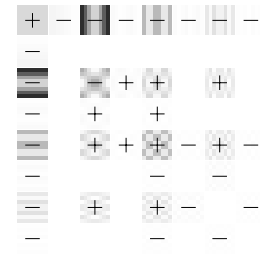
with

$$C(z) = \begin{cases} \frac{1}{2}\sqrt{2} \text{ for } z = 0 \\ 1 \quad \text{else} \end{cases}$$

4 An Example

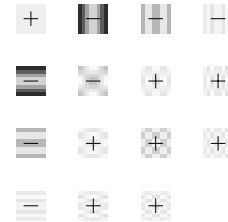
In order to illustrate the DCT, we transform a dot over the i. Fig. 3 shows it strongly enlarged. As presented in Fig. 3 b) the dot over the i has a grating of 64 brightness values. Let's look at its transformed matrix:

$$\begin{bmatrix} 50 & -4 & -84 & -2 & -31 & -2 & -12 & -1 \\ -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -84 & 0 & -29 & 2 & 15 & 0 & 10 & 0 \\ -2 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ -31 & 0 & 15 & 1 & 28 & -1 & 10 & -1 \\ -2 & 0 & 0 & 0 & -1 & 0 & -2 & 0 \\ -12 & 0 & 10 & 0 & 10 & -2 & 0 & -1 \\ -1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \end{bmatrix}$$



The quantization causes an accumulation of zeros by applying a step function (dividing and rounding) to the DCT coefficients:

$$\begin{bmatrix} 2 & 0 & -8 & 0 & -3 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -8 & 0 & -2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & 1 & 0 & 2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



The example demonstrates that only 15 coefficients different from zero are left of the initial 64 brightness values. The coefficients are arranged in linear ordering and then are run and level coded. The new created sequence will be Huffman-coded. [3] and [4] enclose a description of run and level and Huffman coding. The significant fact is that they are **loss-free codings**. Fig. 3 c) shows the result of the back transformation.

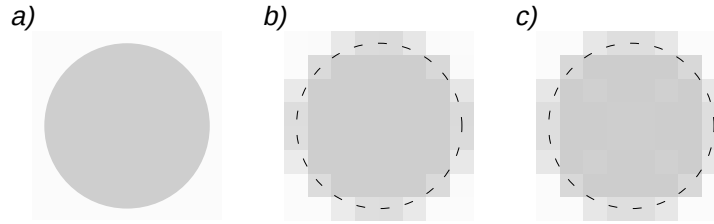


Fig. 3. "Dot over the i": a) original, b) rastered c) after decompression

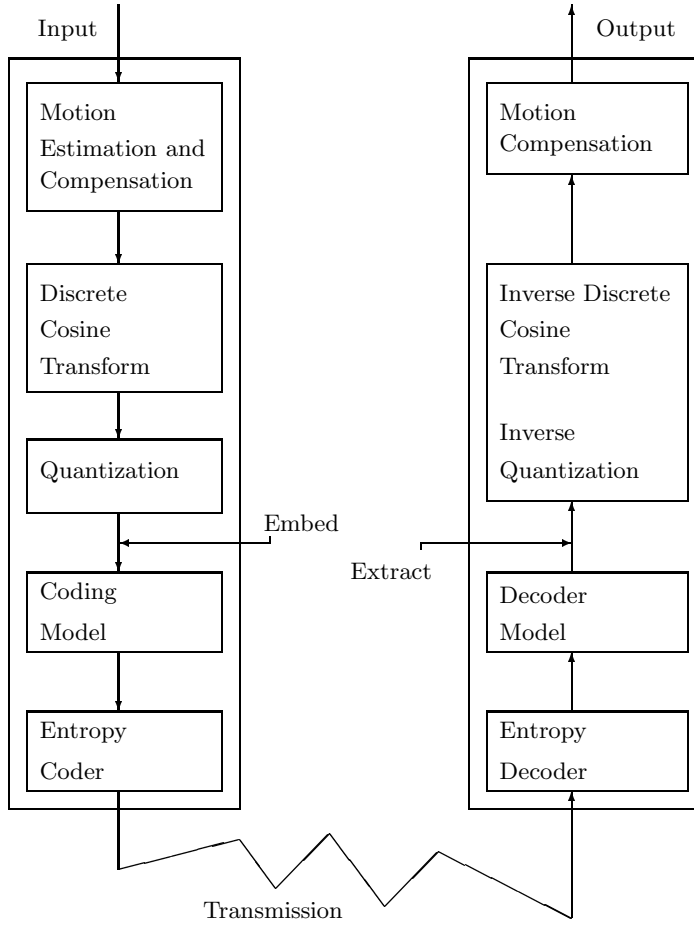


Fig. 4. Information flow in the H.261 Codec [4]

5 Signal Path

A precise knowledge of the signal path is important in order to be able to estimate the safety of a steganographic technique. From the camera to the coded sequence of pictures, the signal path is subject to losses by transformations as well as to influences and to disturbances. In the following, some transformation points on the path are designated, and in parentheses, the altered quantity.

The appearance of the original is influenced by the lighting conditions. The image of the original is preprocessed optically by the lens of the camera. Additionally, attitude aperture setting (depth of focus), the focuses (part of high video frequencies), the focal length (detail, video depth) and the quality of the lens (distortion) contribute essentially. Through dispersion, the focuses are dependent on the color of light. The light is usually transformed into an electrical signal in the camera by a charge coupled device (CCD). The tiny CCDs are characterized through their high sensitivity to light. The light in front of the about 380 000 photosensitive points is filtered by many colored, narrow, vertical stripes. Each three adjacent sensors, receiving respectively red, green, and blue filtered light, make one pixel. The horizontal distance of the three sensors is only a partial pixel distance and thus, is neglected. A CCD has a temporal inertness (the reader possibly observed the "tracing" in the case of a camera pan shot) and is operated with a specific sampling frequency. Afterwards, the altered and rastered image is available in the form of an electrical signal. In this form, it runs through a circuit which contains semiconductors (temperature dependence, noise), and it is changed into a NTSC or PAL signal. The signal path now leads to the computer over a coaxial cable (spectral phase shift, attenuation) from the camera. On the videocard in the computer the picture is locked, digitized and transmitted to the device driver.[2] A data structure in the RGB format results, which contains the image which is now even more coarsely rastered than in the NTSC or PAL signal. With the last step on the signal path, we exceeded the boundary to determinism (see Fig. 5). All further processing steps are digital and deterministic. With the transformation and quantization, desirable rounding errors occur. A loss-free entropy coding compresses the data between quantization and transfer.

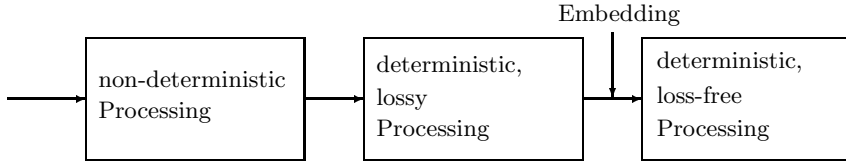


Fig. 5. Sections of the signal path

[2] Often only the interface of the device driver is documented. The programmer is unable to separate the activities of the videocard and those of the device driver.

The signal path can be divided into three parts. Non-deterministic processing means that resulting output signals differ from each other with high probability in the case of identical input signals.

An example is the noise of the semiconductor devices mentioned already. The noise level is dependent of the considered bandwidth. The image incorporated by the camera is filed in the PAL signal line by line from top to bottom, 15 625 lines per second (for 25 full frames). The voltage within a line varies during horizontal color change, with vertical from line to line. Therefore, upright brightness modifications are put in the signal at a bandwidth a maximum of 8 kHz (one line is white, one line is black, alternating). Up to 800 image elements per line can be placed in comparison to this, which corresponds to a bandwidth of 6 MHz. The bandwidth for horizontal video frequencies – and therefore the noise level – is up to 800 times as large as the bandwidth for vertical video frequencies.

Two scan lines are aligned by means of the horizontal synchronization pulse which is included in the CCVS signal. Line interlacing divides a full frame into two half frames. If one numbers the lines of a full frame from top to bottom, the first frame contains all odd lines and the second frame all even lines. Therefore, two straight adjacent lines of the full frame are $\frac{1}{50}$ frame separated from each other. Since the half frames are also registered at the frame rate, the screen content can already have changed.

Here follows an example with numbers. The about 380 000 sensors of a CCD might be placed in format 720 by 540. Three adjacent sensors will be summarized as one image point (RGB), although the sensors have a distance of $\frac{1}{720}$ line length. This horizontal distance is, referring to the smallest H.261 format (176 by 144), $\frac{1}{4}$ pixel distance. Table 1 shows, that already a horizontal displacement of the image content of $\frac{1}{10}$ pixels allows considerable modifications of the frequency spectrum.

We developed a little program which creates Table 1 when base images are generated with horizontal dephasing and transformed. The dephasing results if term $2y + 1$ is replaced by $2y + 1.2$ in Equation 1. Through consideration of Fig. 1, it is obvious that a horizontal "dephasing" of the base images $B_{k,0}$ brings no change. Under exclusion of the coefficients $f(0, 0)$ (base brightness) and $f(k, n)$ (appropriate for the base image), the coefficient $f(k', n')$ with the largest absolute value was always searched in the transformed matrix. The values in brackets are not to be traced back to displacement and can be explained by truncated values during computation. In the line for $B_{0,1}$, the coefficient with the strongest differing amount probably is $f(0, 0)$. Since this coefficient is excluded from consideration, the smaller next appears. Otherwise, for $n > 0$, the following pattern is valid:

$$\begin{matrix} k' = k \\ n' = n - 1 \end{matrix} \quad \text{and} \quad \frac{\Delta f(k', n')}{f(k, n)} \approx n \cdot 3\%$$

Column $f_{\min}$ contains the minimum amount for the coefficient $f(k, n)$ from that coefficient $f(k', n')$ currently changes by 1. Since the coefficients are integer, a modification less than 1 is not possible.

Table 1. Relative change of DCT coefficients $f(k', n')$ while horizontal dephasing of base images $B_{k,n}$ by $\frac{1}{10}$ pixel

k	n	k'	n'	$\frac{\Delta f(k',n')}{f(k,n)}$	$f_{\min}$
0	0	–	–	(0.00) %	(∞)
0	1	0	2	1.51 %	67
0	2	0	1	6.75 %	15
0	3	0	2	9.19 %	11
0	4	0	3	12.27 %	9
0	5	0	4	15.16 %	7
0	6	0	5	18.90 %	6
0	7	0	6	24.60 %	5
1	0	3	0	(0.27) %	(369)
1	1	1	0	3.61 %	28
1	2	1	1	6.79 %	15
1	3	1	2	9.28 %	11
1	4	1	3	12.28 %	9
1	5	1	4	15.13 %	7
1	6	1	5	18.98 %	6
1	7	1	6	24.55 %	5
2	0	6	0	(0.13) %	(780)
2	1	2	0	3.58 %	28
2	2	2	1	6.75 %	15
2	3	2	2	9.29 %	11
2	4	2	3	12.18 %	9
2	5	2	4	15.19 %	7
2	6	2	5	18.82 %	6
2	7	2	6	24.62 %	5
3	0	7	0	(0.14) %	(737)
3	1	3	0	3.61 %	28
3	2	3	1	6.79 %	15
3	3	3	2	9.28 %	11
3	4	3	3	12.30 %	9
3	5	3	4	15.16 %	7
3	6	3	5	18.98 %	6
3	7	3	6	24.55 %	5

k	n	k'	n'	$\frac{\Delta f(k',n')}{f(k,n)}$	$f_{\min}$
4	0	3	6	(0.00) %	(∞)
4	1	4	0	3.57 %	29
4	2	4	1	6.75 %	15
4	3	4	2	9.19 %	11
4	4	4	3	12.29 %	9
4	5	4	4	15.16 %	7
4	6	4	5	18.90 %	6
4	7	4	6	24.60 %	5
5	0	1	0	(0.14) %	(736)
5	1	5	0	3.62 %	28
5	2	5	1	6.80 %	15
5	3	5	2	9.28 %	11
5	4	5	3	12.30 %	9
5	5	5	4	15.16 %	7
5	6	5	5	18.98 %	6
5	7	5	6	24.55 %	5
6	0	2	0	(0.26) %	(390)
6	1	6	0	3.59 %	28
6	2	6	1	6.76 %	15
6	3	6	2	9.29 %	11
6	4	6	3	12.20 %	9
6	5	6	4	15.21 %	7
6	6	6	5	18.82 %	6
6	7	6	6	24.67 %	5
7	0	3	0	(0.14) %	(736)
7	1	7	0	3.62 %	28
7	2	7	1	6.80 %	15
7	3	7	2	9.28 %	11
7	4	7	3	12.30 %	9
7	5	7	4	15.16 %	7
7	6	7	5	18.98 %	6
7	7	7	6	24.60 %	5

6 Algorithm

In this section we will discuss the image (dot over the i) of section 4 again. This time, we want to show an unverifiable modification, unverifiable in the sense of the calculated example in section 5. When embedding something, we have to change the carrier signal. The heart of a steganographic algorithm is a process that changes the signal. In our case it changes DCT coefficients. Although the changes cause an imperceptible horizontal dislocation of the image, the algorithm does not influence so-called motion vectors, at least not in a direct way. A motion compensation step (see Fig. 4) is not necessary for the implementation of this steganographic videoconferencing system. An attacker could get more precise image data by interpolating consecutive frames of an unchanging picture which he could match against the actual frame. This would reduce the space for embedding. However, delta frame coding in case of still images makes life easier for steganographers. An unchanging picture in front of the camera comes only once as a key frame. Hence, it is transmitted and used for steganography only once. It is very unlikely that the difference (or delta) frames of a still image contain a big coefficient (see $f_{\min}$ in Table 1) making them suitable for steganographic processing.

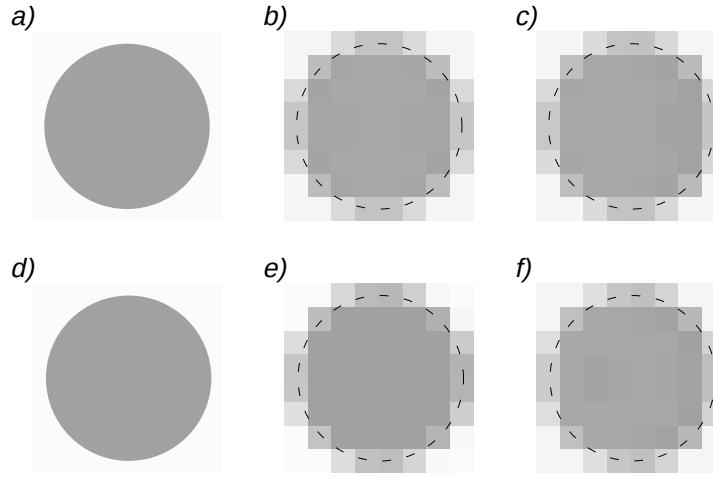


Fig. 6. "Dot over the i" and horizontal shifting: a) original, b) after decompression (unchanged), c) after decompression (changed by algorithm), d) by $\frac{1}{15}$ pixel shifted original, e) moved image after grating, f) moved image after decompression

The contrast of the original image has been increased (see Fig. 6 a)). As a result, the absolute value of one of the DCT coefficients according to Table 1 is large enough to allow a modification. (Refer to coefficient $f(0, 2)$ in the example.)

The following matrix includes the 64 brightness values of the original image with higher contrast.

$$\begin{bmatrix} 20 & 20 & 53 & 79 & 80 & 56 & 21 & 20 \\ 20 & 82 & 110 & 110 & 110 & 110 & 86 & 22 \\ 53 & 110 & 110 & 110 & 110 & 110 & 110 & 59 \\ 79 & 110 & 110 & 110 & 110 & 110 & 110 & 85 \\ 80 & 110 & 110 & 110 & 110 & 110 & 110 & 86 \\ 56 & 110 & 110 & 110 & 110 & 110 & 110 & 62 \\ 21 & 86 & 110 & 110 & 110 & 110 & 91 & 23 \\ 20 & 22 & 59 & 85 & 86 & 62 & 23 & 20 \end{bmatrix}$$

The following left matrix includes the DCT coefficients after quantization. The bold highlighted coefficient $f(0,2)$ allows a modification of 6.75 %, which means $16 \cdot 0.0675 = 1.08$. The right matrix shows this modification for $f(0,1)$.

$$\begin{bmatrix} 4 & 0 & \mathbf{-16} & 0 & -6 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -16 & 0 & -5 & 0 & 3 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -6 & 0 & 2 & 0 & 5 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 2 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 4 & \mathbf{-1} & -16 & 0 & -6 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -16 & 0 & -5 & 0 & 3 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -6 & 0 & 2 & 0 & 5 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 2 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

After recovery, the following matrixes of brightness values result, presented in Fig. 6 b) and c), too. The modification leads to a slight shifting to the right.

$$\begin{bmatrix} 26 & 26 & 53 & 76 & 76 & 53 & 26 & 26 \\ 26 & 83 & 105 & 104 & 104 & 105 & 83 & 26 \\ 54 & 107 & 103 & 104 & 104 & 103 & 107 & 54 \\ 73 & 105 & 106 & 104 & 104 & 106 & 105 & 73 \\ 73 & 105 & 106 & 104 & 104 & 106 & 105 & 73 \\ 54 & 107 & 103 & 104 & 104 & 103 & 107 & 54 \\ 26 & 83 & 105 & 104 & 104 & 105 & 83 & 26 \\ 26 & 26 & 53 & 76 & 76 & 53 & 26 & 26 \end{bmatrix} \begin{bmatrix} 26 & 26 & 52 & 76 & 77 & 55 & 26 & 26 \\ 26 & 81 & 104 & 104 & 105 & 107 & 85 & 26 \\ 52 & 105 & 102 & 103 & 104 & 105 & 109 & 57 \\ 71 & 103 & 104 & 103 & 104 & 107 & 108 & 75 \\ 71 & 103 & 104 & 103 & 104 & 107 & 108 & 75 \\ 52 & 105 & 102 & 103 & 104 & 105 & 109 & 57 \\ 26 & 81 & 104 & 104 & 105 & 107 & 85 & 26 \\ 26 & 26 & 52 & 76 & 77 & 55 & 26 & 26 \end{bmatrix}$$

The "natural" shifting as a comparison: Fig. 6 d) shows the original, shifted by $\frac{1}{15}$ pixel. The coefficients presented in the right following matrix result from transformation and quantization of the left following matrix (see also Fig. 6 e). The shifting of the original image would have caused a more intensive modification of the coefficient.

$$\begin{bmatrix} 20 & 20 & 50 & 78 & 81 & 58 & 22 & 20 \\ 20 & 76 & 110 & 110 & 110 & 110 & 91 & 23 \\ 47 & 110 & 110 & 110 & 110 & 110 & 110 & 65 \\ 73 & 110 & 110 & 110 & 110 & 110 & 110 & 91 \\ 74 & 110 & 110 & 110 & 110 & 110 & 110 & 92 \\ 50 & 110 & 110 & 110 & 110 & 110 & 110 & 68 \\ 20 & 82 & 110 & 110 & 110 & 110 & 95 & 25 \\ 20 & 21 & 56 & 84 & 87 & 64 & 24 & 20 \end{bmatrix} \begin{bmatrix} 4 & \mathbf{-2} & -16 & 0 & -6 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -16 & 0 & -5 & \mathbf{1} & 3 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -6 & 0 & 2 & 0 & 5 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 2 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Finally, Fig. 6 f) shows the following, recovered matrix.

$$\begin{bmatrix} 26 & 26 & 48 & 73 & 79 & 59 & 26 & 26 \\ 26 & 79 & 102 & 103 & 106 & 109 & 87 & 26 \\ 49 & 103 & 102 & 103 & 104 & 104 & 110 & 60 \\ 67 & 102 & 107 & 105 & 103 & 105 & 108 & 80 \\ 67 & 102 & 107 & 105 & 103 & 105 & 108 & 80 \\ 49 & 103 & 102 & 103 & 104 & 104 & 110 & 60 \\ 26 & 79 & 102 & 103 & 106 & 109 & 87 & 26 \\ 26 & 26 & 48 & 73 & 79 & 59 & 26 & 26 \end{bmatrix}$$

As the example shows, early, non-deterministic effects at the beginning of the signal path can be reproduced in a later part (see Fig. 5).

7 Implementation

The implemented steganographic function "Embedding" (see the model in [7]) exploits the effect described in section 6: the frequency spectrum changes considerably already at minor changings of the phasing of the image.

At first, we distinguish between "suitable" and "unsuitable" blocks of DCT coefficients. Blocks are "suitable" if they include a coefficient which is larger than its minimum amount $f_{\min}$ (see Table 1). In the source code, all minimum amounts are represented by `delta[]`. All other blocks are "unsuitable" and will be transmitted without steganographic modification.

```
/*
    To be classified as "suitable", a block must contain one
    coefficient greater or equal to its correspondent value
    in the following matrix.
*/
unsigned int delta[64]={
    -1,-1,-1,-1,-1,-1,-1,-1,    /* -1 means infinity */
    -1,28,28,28,29,28,28,28,
    15,15,15,15,15,15,15,15,
    11,11,11,11,11,11,11,11,
     9, 9, 9, 9, 9, 9, 9, 9,
     7, 7, 7, 7, 7, 7, 7, 7,
```

```
     6, 6, 6, 6, 6, 6, 6, 6,
     5, 5, 5, 5, 5, 5, 5, 5
};

/*
    stego_in(p) is the steganographic function "embedding".
    The parameter p points to a matrix of 64 coefficients.
*/

void stego_in(int *p)
{
    int i, most_suitable, sum_of_block, is_stego, *steg_ptr;

    sum_of_block = 0; /* for sum (mod 2) */
    is_stego = 0;     /* 1 means "suitable" block */
    for (i=1; i<64; i++) {  /* skip DC coefficient p[0] */
        if (p[i]) {    /* consider non-zero coefficients */
            sum_of_block += abs(p[i]);    /* sum up */
            /* coefficient large enough? */
            if (abs(p[i]) >= delta[i]) {
                is_stego = 1;       /* "suitable" block */
                /* more suitable? then keep the pointer */
                if (abs(p[i])-delta[i] >= most_suitable) {
                    steg_ptr = &p[i-8]; /* this is f(k',n') */
                    most_suitable = abs(p[i]) - delta[i];
                }
            }
        }
    }
...
```

For a block classified as suitable, its further treatment depends on the modulo-2 sum of its coefficients (a kind of parity). If the parity is equal to the next bit for embedding, the block will be transmitted unchanged. If the parity is not equal, it has to be changed.

```
...
    if (is_stego)   /* suitable block? */
        /* compare the modulo-2 sum with the next bit to embed */
        if ((sum_of_block&1) != get_bit_to_embed()) {
            /* decrement abs(*steg_ptr), the coefficient */
            if (*steg_ptr > 0)
                (*steg_ptr)--;
            else if (*steg_ptr < 0)
                (*steg_ptr)++;
```

```
            else /* 0 ==> 1 */
                *steg_ptr = 1;
        }
}
```

Let $f(k, n)$ be the coefficient of a suitable block which, corresponding to Table 1, allows the maximum modification. In this case, the absolute value of the coefficient $f(k, n-1)$ will be decreased by 1 or if it is zero, set to 1. This way, a coefficient of a block is changed by 1 and its parity flips.

All changed blocks are transmitted as well as those, where no change was necessary. The whole scenario is shown in Fig. 7. The recipient receives suitable and unsuitable blocks which are separated by the same criteria as at the sender. It has to be remarked that changed suitable blocks (suitable blocks with flipped parity) will always stay suitable blocks because the coefficient $f(k, n)$ has not been changed, but fulfills the criterion "suitable".

The recipient can extract the embedded data through reading out the parity bits of the suitable blocks sequentially. The recipient system uses all blocks for image reconstruction. The steganographic algorithm presented here acts like a quantization with a higher divisor. It increases the compression rate so that the introduced error looks natural. A lower quantizer should equalize the effect of the algorithm.

```
void stego_out(int *p) /* steganographic function "Extraction" */
{
    int i, sum_of_block, is_stego;

    sum_of_block = 0;     /* for mod-2 sum */
    is_stego = 0;     /* 1 means "suitable" block */
    for (i=1; i<64; i++) {
        if (p[i]) {     /* consider non-zero coefficients */
            sum_of_block += abs(p[i]); /* sum up */
            /* coefficient large enough? */
            if (abs(p[i]) >= delta[i])
                is_stego=1;     /* YES! suitable block! */
        }
    }
    if (is_stego) put_embedded_bit(sum_of_block & 1);
}
```

In Fig. 8 we show the surface of the application. `ivsd` is the daemon which receives conference calls. Per mouse click it is possible to open a window to call other daemons (single- or multicast-addresses). After establishing the video conference connection, the user of the conference *with* steganographic enhancement has two additional windows: one for the input of text (the secret message to hide) and one for displaying the embedded messages of the communicating partner.

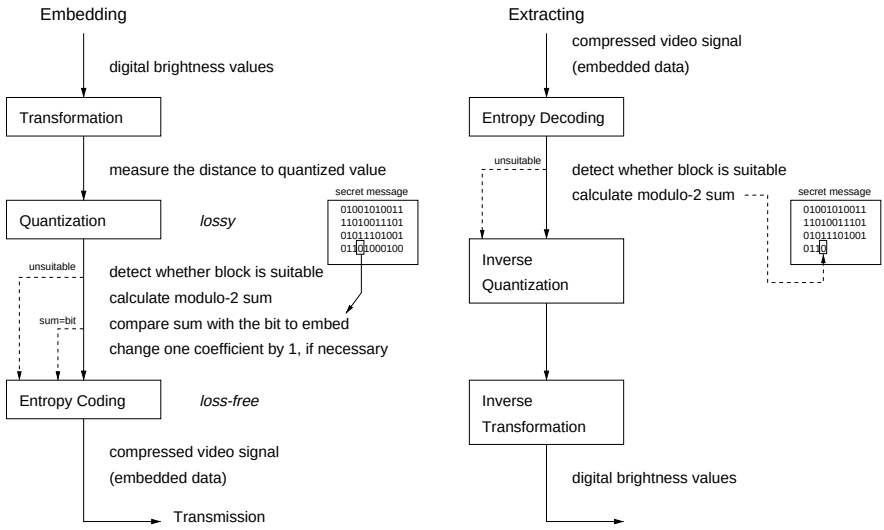


Fig. 7. Embedding and Extracting

The application is comparable to a combination of a video conference and the Unix standard command `talk`[3].

8 Conclusion

Through compression, as used with videoconferencing, the least significant bits gain importance, so every bit of the compressed signal contributes a significant part to the picture. The detection of a random replacement of these bits is possible as shown for Jsteg in [8]. However, it is possible to change parts of the carrier, making it impossible to detect these changes without direct comparison to the unchanged carrier, which should never leave the security domain. We use special features of the input devices, such as a camera or scanner. The analysis of the input devices shows free spaces permitting embedded data. If steganographic techniques simulate peculiarities of a camera, the changes do not raise any suspicion for a possible attacker. For this reason, we scrutinised the picture reception closely.

Our algorithm reproduces these effects artificially; the signal changes imperceptibly. A direct comparison with the original allows differentiation, but this still does not enable the observer to discern between the original and the altered signals. Furthermore, the sender merely transmits the changed frames. In this manner, a secret message can be embedded. The slight horizontal dephasing is unnoticeable.

[3] `talk` is a communication program for terminals.

Fig. 8. User interface of the implemented application

Algorithms are only trustworthy, when they are open to public scrutiny. For this it is necessary to separate the algorithm from the secrets. The simplest possibility is the generation of pseudo random bits. Both the sender and the receiver need the same key and procedure to generate these bits and use them as a pseudo one-time pad. Because the distribution of these bits has the same random uniformity as bits extracted from any video conference, the attacker can not discern between a normal video conference and one in which secret data has been embedded after encryption.

In an ISDN videoconferencing system it is possible to embed a GSM telephone conversation (up to 8 kBit/s). This depends upon the texture of the picture, because it is impossible to embed data in black frames.

References

1. Ingemar J. Cox, Joe Kilian, Tom Leighton, Tatal Shamoon, A Secure, Robust Watermark for Multimedia, In: Proceedings: Information Hiding. Workshop, Cambridge, U.K., May/June, 1996, LNCS 1174.
2. Elke Franz, Anja Jerichow, Steffen Möller, Andreas Pfitzmann, Ingo Stierand: Computer Based Steganography. In: Proceedings: Information Hiding. Workshop, Cambridge, U.K., May/June, 1996, LNCS 1174.
3. CCITT Recommendation H.261, Video Codec For Audiovisual Services At $p \times 64$ kbit/s, Genf, 1990
4. Andy C. Hung, PVRG-P64 Codec 1.1, Stanford University, 1993
5. Neil F. Johnson, Steganography, George Mason University, 1996
6. Marit Köhntopp, Steganographie als Verschlüsselungstechnik, iX 4/1996
7. Birgit Pfitzmann: Information Hiding Terminology. In: Proceedings: Information Hiding. Workshop, Cambridge, U.K., May/June, 1996, LNCS.
8. Robert Tinsley, Steganography and JPEG Compression, Final Year Project Report, University of Warwick, 1996
9. `ftp://idea.sec.dsi.unimi.it/pub/security/crypt/code/s-tools4.zip`
10. `ftp://ftp.funet.fi/pub/crypt/steganography/`
11. `http://www.rugeley.demon.co.uk/security/hdsk50.zip`
12. `http://www.rugeley.demon.co.uk/security/`
13. `ftp://ftp.funet.fi/pub/crypt/mirrors/idea.sec.dsi.unimi.it/cypherpunks/steganography/MandelSteg1.0.tar.gz`
14. `http://ftp.giga.or.at/pub/hacker/stego/texto_os2.zip`