Transform-Aware Content Adaptive Stegosystem for Social Networks

Klim Kireev, Grigorii Melnikov, Edgar Kaziakhmedov
Skolkovo Institute
of Science and Technology
{klim.kireev, grigorii.melnikov, edgar.kaziakhmedov}@skoltech.ru

***Abstract*—The aim of steganography is to hide information in the other data, such as images. Nowadays, the most common channels for sharing images are social networks and it makes them perfect for steganography. Usually they scale and recompress uploaded images for storage saving. Unfortunately, most of contemporary stegosystems are very fragile to any image modification, which limits their application in this case. In this paper we propose the system able to transfer information through such channels. It outperforms error correction code based approaches in terms of capacity and distortion since it takes into account the specific transform. We implemented the first steganographic algorithm able to restore the message after JPEG recompression in social networks without errors and successfully verified it in real world with Twitter.**

I. Introduction

The purpose of steganography is to conceal information in data sent through an open channel. Usually it is supposed that there is a person called a warden who analyses all data and tries to detect the fact of hidden information transmission. For digital image steganography, initial data is called a cover image, hidden information is called message, process of hiding is called embedding and resulting object is a stego image. Suspiciousness of a stego image depends on parts of image modified. Selection of the best areas and the best algorithm requires numerical measure of such suspiciousness. Such measure was proposed in [5] as distortion function. This function is minimized over stego image during embedding, which presumably leads to lower probability of being revealed. It is called content adaptive steganography. Syndrome-Trellis code (STC) was proposed as a practical construction close to optimal bounds. In fact this is a convolutional code adapted to steganography needs. Usual assumption in steganography is that a stego image is not modified during transmission. Sometimes authors consider the case of adversarial or (in most cases) non-adversarial noise. This assumption is very convenient, but in certain cases it is far from reality. Nowadays the most important and unsuspicious open channels are social networks, where people can share their images to the whole world. In this case we often cannot use most of the images since they are irreversibly recompressed with reduced quality to save storage space. The whole process of transmitting covert information in this case is depicted on Fig. 1. The sender embeds the message to the cover image and sends it to the social network. It is important to notice that in this pipeline we have two possible places to be attacked. One of them is just before the transform and we denoted it as warden 1, for example it can be social network itself which checks all images on image manipulation. Another one is warden 2 who can scan all the images in the channel after the modification. Since we have two different images which can be attacked it is reasonable to introduce two distortion functions D_1 and D_2. Depending on the potential attack sender can choose to optimize either one of them or both simultaneously.

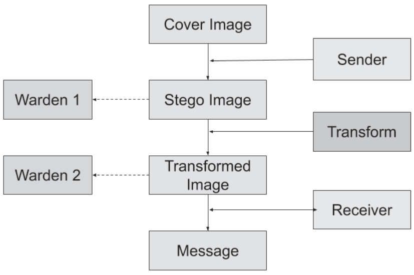


Fig. 1. Pipeline of steganography through social network

If we try to apply traditional stegosystem to this pipeline we have two variants:

1) Manually recompress image to quality which is allowed by social network.
2) Sender and receiver should come to consensus on default quality, and then sender can use error correction codes and receiver will recompress all images to this quality.

Since the task of steganography is to conceal the fact of embedding, the first method cannot be applied because any previous recompression can be easily exploited with high probability[16], also it doesn't always save an image from the social network recompression with the same quality, so error correction codes should also be applied. Social networks can also rescale the image, but this issue is easily solved by crop which is undetectable if it is aligned to dct blocks

The second method leads to high error rate, so strong error correction codes should be used in this case and amount of information significantly drops. We will discuss in detail this aspect in section V.

In this paper we propose the new solution. Our algorithm embeds information before the given transform and after the transform receiver can extract the message without errors. Moreover, the methods above allow only D_1, while our method described in section IV is able to minimize both distortion functions D_2, D_1.

The brief outline of other sections: In section VI we describe how to apply this algorithm to transmit stego messages through twitter. In section V we compare our scheme with error correction codes approach.

All experiments are conducted on the first 100 pictures of Alaska dataset [2] if other environment is not stated explicitly.

II. Prior Arts

Despite the small number of publications the problem of transmission of stego images in channel with modifications is not new. For instance, in [8], [9] authors analyzed theoretical aspects of noise resistant steganography, but the codes proposed are not designed for adaptive steganography. Recently, in [13] it has been suggested applying error correcting code approach to existing STC codes. This approach will be analyzed further in next chapters. Apart from this class of approaches, it is worth mentioning the work done in [6], where authors proposed an embedding to the most ”uncertain” coefficients after quantization process in JPEG images, but with knowledge of a source uncompressed image. It was done there to improve statistical imperceptibility, without use of a distortion function, while in our case the main aim is to endure transform in channel. Also in [6] the transform was performed on the sender side, not in the channel. The main problem of majority of approaches is that the modifications in channel caused by transmission were assumed to be a noise, which might not be the case in real-world scenarios.

As it is shown in [12] the real-world scenarios might impose more challenging constraints which are, with a few exceptions, not taken into consideration in most papers. In our work, to deal with practical cases we consider a social network to be a transmission medium for the proposed steganographic method.

The idea of utilizing a social network as a transmission channel was first shown in [1]. Authors considered the most popular social networks for photo sharing, such as Facebook, Badoo, Google+. Social network is a promising place for steganography due to the fact that information embedded in pictures will be obscured within millions of others posted and shared daily. The main difference between an uploaded picture and an origin one within social network context is that the former is often subjected to the various transforms: resize, compression, requantization and format conversion. The main goal of applying transforms is to reduce size of the data which, in turn, can potentially destroy steganography. The particular set of transforms causes the aforementioned modifications. If the modifications can be studied within a given social network, then the platform can be suited to steganography. In this paper, the transforms applied to an uploaded image to Twitter were considered to be the main cause of the modifications.

The first social website to be analyzed profoundly for steganography purposes was Facebook in [11]. Authors conducted a series of experiments and managed to find the appropriate image parameters, which will not be altered by the network. To estimate them, the images were uploaded and then downloaded back from Facebook until the size ratio converged to 1. But despite promising results the algorithms for embedding were not aware of possible modifications to be applied. Moreover, the preprocessing which recompresses the input image before embedding results in a ”double JPEG” compression, which might arouse suspicion as it could be detected fairly straightforward [15], [14].

In addition to Facebook, there is not less popular online news and social networking service - Twitter, which allows to ”tweet” short messages as well as images. To the best of our knowledge there is no work considered Twitter for image steganography to date, so it was picked as the target social network in our study.

III. Definitions

In this paper we define images as capital letters
X - a cover image Y - a stego image

Sets are defined with different font: $\mathcal{X}, \mathcal{Y}$
$|\mathcal{X}|$ is cardinality of set $\mathcal{X}$

Over-line denotes an integer with inverted last bit:
$\overline{X_i} = X_i \oplus 1$, where $\oplus$ is bitwise XOR.

Small letters denote binary vector of last bits corresponding to image defined with capital vector:
x is a binary vector of last bits of the image X.

By H we define a parity check matrix of a linear code.

We notice here that images could be defined as vectors of pixels or some other units e.g. DCT coefficients in case of JPEG.

IV. Transform-aware steganography

We would like to start this section with short introduction. Let $I = 0, 1, ..., 255$, X is image of size n: $X \in \mathcal{X} = I^n$, lets consider binary embedding operation when we hide information in last bits of pixels, in this case we denote Y as a stego image and $Y \in \mathcal{Y} = \prod_{i=1..n} x_i$ To optimally embed a message m to X is to find the solution to the optimization task:

$$D(X,Y) \longrightarrow min, Hy = m$$

In fact this problem is similar to decoding linear code. In general case such optimization problems is NP-hard, but for some cases of D, like for additive distortion:

$$D(X,Y) = \sum_{X_i \neq Y_i} \rho_i$$

and certain class of codes like STC it can be solved in O(n). In case with transform T this task need to be reformulated:

$$T(Y) = Z$$

$$\alpha D_1(X,Y) + \beta D_2(T(X), T(Y)) \longrightarrow min, Hz = m$$

$$0 \leq \alpha, \beta \leq 1$$

Lets consider image $X \in \mathcal{X}$ and lets transform be $T(X)$, for example lets consider binary embedding so $Y \in \mathcal{Y}$ it is obvious that if receiver knows X by transmitting Y we can send $log_2|T(\mathcal{Y})|$ bits of information without any error (by simple enumeration $|T(\mathcal{Y})|$), moreover, in fact according to

[10] knowledge of X does not give any substantial advantage to the receiver.

Even in this task we can still use code-based steganography method with usual restriction:$Hz = m$. Since there are $D_1(X,Y)$ and $D_2(T(X),T(Y))$ according to [5] there can be established bounds on its distortion and payload. The only problem in this approach is that in general case we do not know set $T(\mathcal{Y})$ and to find it experimentally exponential number of tries could be required. Moreover even if we did so, we still need to find optimal embedding function, and since this optimization task is defined on set without reasonable structure $T(\mathcal{Y})$ this task seems to be numerically unsolvable for big dimensions. To overcome this obstacle we propose not to change all components of x but its subset, with following property:

Let $Y' \in \mathcal{Y}'$ be a vector composed from subset of coordinates of Y with following property:

$$T(Y') = t_i(Y'_i, X)$$

where t_i is a set of scalar functions and each of them can be computed in $O(1)$. since t_i is a function over a small finite set, it can be defined with table. For clarification lets start with binary variant. For each bit in Y'_i we have two variants:

$$t_i(Y'_i, X) = t_i(\overline{Y'_i}, X)$$

or

$$t_i(Y'_i, X) \neq t_i(\overline{Y'_i}, X)$$

in the first case we cannot transmit anything, and in the second case we can transmit 1 bit. Positions in first cases we later will call "frozen positions". Let us denote number of frozen positions as a. The channel between sender and receiver becomes defective memory cells channel described in [18], [17]. Capacity of such channel asymptotically tends to $1 - a/n$. One of the arising questions is how to determine such a subset. This method must be known by receiver, but the receiver does not need to know whether this position frozen or not, his knowledge of that does not asymptotically impact capacity of the channel[18].

In fact, in [3], [4] authors introduced an asymptotically optimal class of codes with efficient construction for the problem. In addition to it, the wet paper codes were proposed in [7] in the scope of steganography. Despite wet paper codes allow information to be embedded in such channels, they cannot minimize additive distortion even in its simplest form (Hamming distance) in polynomial time. This is why we propose to use Syndrome-Trellis codes for such channels. In order to avoid modifications distortion function is set to be infinity for frozen positions. Since sender knows $t_i(Y'_i, X)$, he can embed information directly to $T(Y')$ using common STC and setting ρ_i to ∞ for frozen positions.

A. Example: JPEG recompression

This example is the most important in practical sense. For components of Y' in this case we can pick top-left coefficients of each dct 8x8 block (pointed red on Fig. 3). Changes of this coefficient do not affect other DCT blocks, so we can think that transform will be applied to them component wise. Sender can find each function t_i in embedding stage and define them as a table doing transform for Y_i and $\overline{Y_i}$ on each block separately. One can say that picking one coefficient per block we put off too much possible places for embedding, But later it will be shown that this approach wins in terms of capacity/distortion over error correction code. The percentage of frozen bit is depicted on Fig. 2.

Algorithm 1 Embedding for transform

```
1. Compose vector from subset $Y'$;
2. Initialize distortion as $\rho_i = \alpha\rho_1 + \beta\rho_2$;
3. Check each position and set in frozen positions $\rho_i = \infty$;
4. Calculate initial vector $z'$ from $x'$;
5. Embed message to $z'$ using Viterbi algorithm with STC;
6. With table $t_i$ restore $y'$ from $z'$;
```

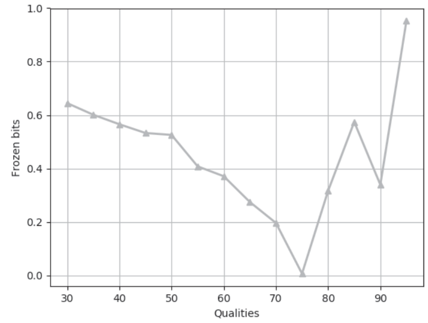


Fig. 2. Percentage of frozen coefficients

B. Example: JPEG recompression (non-binary case)

Although in this paper we focus on binary code usage, this scheme can successfully be extended to q-ary case. For example, to utilize DCT coefficients more efficiently, 3 coefficients instead of 1 can be used (green + red on Fig. 3). In this case instead of using binary code, we can use code over GF(8). We can map last bits of these 3 coefficients to elements of GF(8) in this case t_i is also defined as a table, but with 8 entities. In this case additive distortion function will be defined as:

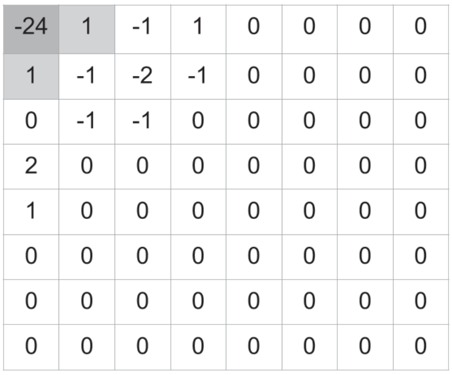
$$D_1 = \sum_{Y'_i} \rho_i$$

$$\rho_i = \rho_{DCT_{11}} * (y_{DCT_{11}} \neq x_{DCT_{11}}) + \rho_{DCT_{21}} * (y_{DCT_{21}} \neq x_{DCT_{21}}) + \rho_{DCT_{12}} * (y_{DCT_{12}} \neq x_{DCT_{12}})$$

Note, that some of these bits can be frozen. In this case to embed a message in optimal way we can use STC over GF(8).

V. Error correcting codes approach

Steganography with error-correction codes (ECC) is quite researched topic, which was repeatedly proposed to solve the problem of social networks. So we devoted this section to analyze this approach in details. For example, Transmitter can reach consensus with receiver on default quality factor, and then send photos of only agreed quality. The receiver



-24	1	-1	1	0	0	0	0
1	-1	-2	-1	0	0	0	0
0	-1	-1	0	0	0	0	0
2	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Fig. 3. Top-left coefficient of dct matrix

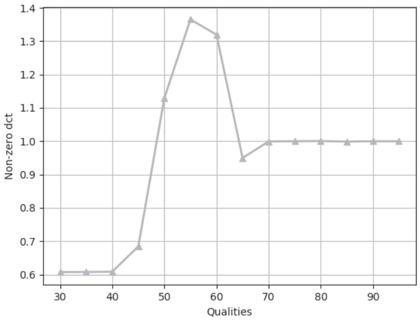


Fig. 4. Relative number of unused (zeros and ones) DCT coefficients. If this number is not equal to 1.0 length of binary vector before and after transform is different

can compress all incoming photos to this quality, and correct emerging errors. First of all, error-correction works well if size of the data does not change, i.e. the modification are limited to errors and erasures. Although, there are error correction codes correcting insertions and deletions, their performance is very poor and they are not applicable in practice. In case of JPEG steganography the usual assumption is a preserving zero DCT coefficients since they are crucial in terms of detectability. Usually the largest part of DCT matrix is filled with zeroes and then avoided during embedding. In fact in case of binary embedding ones are also avoided because they can be changed to zeros. Since number of zeros and ones do not change during embedding, on receiver side binary vector of the same size can be restored. In case of image transform such as compression to certain quality and then back the number of non-zero DCT coefficients can change. On Fig. 4 you can see an average percent of non-zero DCT coefficients after converting from quality 75 and back. This number was averaged over the first 100 images from Alaska dataset.

Another major problem of usage ECC in this task is pointed out in [13]. It is called error amplification: to minimize distortion function more effectively code matrix H must have big weights of columns, it means that each error in stego vector will lead to error with big weight in the syndrome. In fact error rate in syndrome will also depend on its size: small syndromes will be more fragile. It leads us to contradictory trade-off: to remain undiscovered we need to embed small amount of information which will lead to big error rate. Our scheme does not have this disadvantage.

TABLE I. COMPARISON TABLE BETWEEN OUR APPROACH AND ERROR CORRECTION CODES

	Our algorithm	ECC
Resistance to changing length transforms	Yes	No
Probability of errors	Zero [a]	Low
Transform independence	No	Yes [b]
Error amplification	No	Yes
Distortion/robustness trade-off free	Yes	No
Ability to optimize distortion after transform	Yes	No

[a] Even if embedding fails it is known on transmitter side
[b] Only if transform keeps the length

VI. Application to Twitter

We discovered that Twitter converts all incoming JPEG images with quality more than 85 to quality 85. So we chose twitter as potential Social network for testing our algorithm. We found that Twitter compression can be reproduced with libjpeg library with integer slow method and 2x2 sampling factor. So we successfully transmitted some sample images like lena, peppers, etc. multiple times with random messages without any errors.

To embed message we apply Algorithm 1. As base distortion function we use J-UNIWARD with modifications mentioned before: it returns ∞ on ”frozen positions” and calculated distortion on others. Procedure of finding ”frozen positions” is described in Algorithm 2.

Some of the DCT-coefficients turn into 0 or 1 after transform. We denote such DCT-coefficient as ”dead”. Positions where DCT-coefficient is ”dead” for any value of the last bit (0 or 1), we denote as ”dead positions”. Position where initial DCT-coefficient (without modification of last bit) is ”dead” we also call ”dead positions”. On the transmitter side we do not include ”dead positions” into binary vector for embedding.

Algorithm 2 Frozen positions detection

```
Original image - $X$, last bits of top-left coefficients - $x$ 1. Initialize all last bits of top-left coefficients of dct matrices with 0(1) (output: $X^0$ and $X^1$ - modified images)
2. Apply to changed images the same transformations which Twitter does (output: $\hat{X}^0_{DCT}$, $\hat{X}^1_{DCT}$ - top-left dct coefficients vectors of transformed images)
3. Extract last bits of top-left coefficients of dct matrices of transformed images (output: $T^1$ and $T^0$ - transition vectors for $X^1$ and $X^0$)
4. If $T^1_i = T^0_i$: yield $i$ as ”frozen position”
5. If $\hat{X}^0_{DCT_i} - ”dead”$ and $x_i = 1$: yield $i$ as ”frozen position”
6. If $\hat{X}^1_{DCT_i} - ”dead”$ and $x_i = 0$: yield $i$ as ”frozen position”
```

VII. Experiments

As we discussed in previous parts there are no straightforward alternatives for our approach, since converting to different quality usually change the number of non-zero DCT

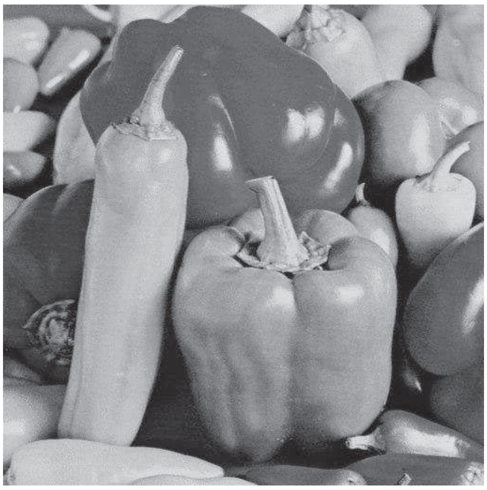


Fig. 5. One of the sample images we transferred on Twitter.

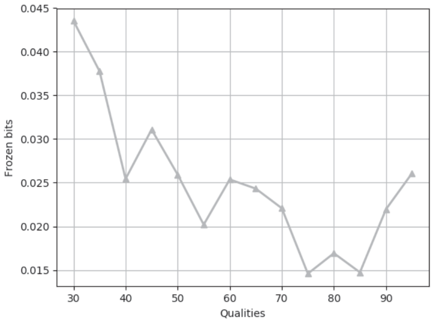


Fig. 6. Frozen bits for recompression with the same quality. The percent of such positions is small, but if we treat them as errors, it amplifies the percentage many times.

coefficients. So for comparison we found the closest method described in [13]. Authors considered recompression with the same quality as the simplest and less harmful type of transform and used the DSTC code for error correction. Recompression with the same quality also fits to our algorithm, moreover, since the transform introduces small numbers of errors the number of frozen bits is also small. On Fig. 6 the number of frozen bits for recompression with the same quality is provided.

This number is quite small so they almost do not affect STC performance. For comparison with DSTC we took the first 1000 images of BOSS competition and converted them to JPEG with quality 75 as it was did in [13]. It is important to mention that DSTC codes proposed there cannot provide error-free results, they just keep error rate on 10^{-3}, while our method preserve an entire message. Although efficiency of DSTC decreases with decreasing size of the message and therefore distortion grows. In our method it works in opposite way. Since we use only a small number of coefficients, the maximum number of embedded information for 512x512 picture is about 2000 bits, so we compared for this number. Results are depicted on Fig. 7. In our case distortion is more than ten times less for the same amount of information.It would make it hardly detectable for modern steganalyzers. If it requires more than 2000 bits to transmit one can use q-ary scheme proposed above.

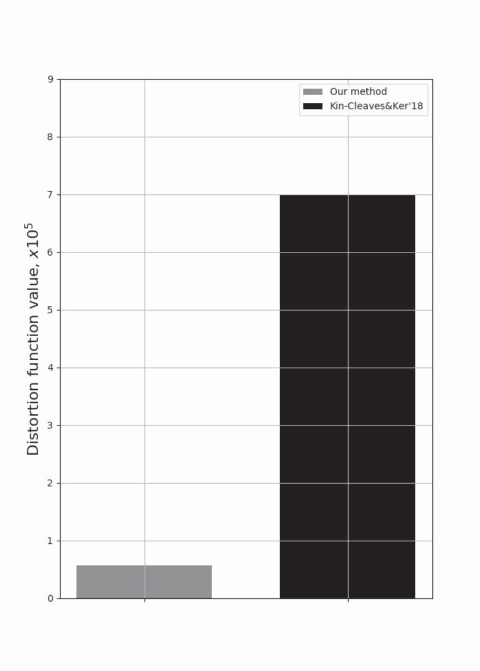


Fig. 7. Results of comparison for 2000 bit message, for [13] and our method. The distortion decreased drastically for the same message length.

VIII. Conclusion

In this paper, we did the following work:

- studied the problem of steganography for channels with modifications,
- discovered that we can efficiently utilize the knowledge of the particular transforms within a given social network,
- tested the proposed method in Twitter,
- compared our method with error correction based approaches and verified that it outperforms them in terms of distortion value.

Furthermore, error correction codes are not applicable in a wide range of cases such as compression to a different quality (like in Twitter), while our method is able to address them.

References

[1] A. Castiglione, G. Cattaneo, and A. De Santis. A forensic analysis of images on online social networks. In *2011 Third International Conference on Intelligent Networking and Collaborative Systems*, pages 679–684, Nov 2011.

[2] Rmi Cogranne, Quentin Giboulot, and Patrick Bas. The alaska steganalysis challenge: A first step towards steganalysis. In *The 7th ACM Workshop on Information Hiding and Multimedia Security*, pages 125–137. ACM Press, 07 2019.

[3] Ilya Dumer. Asymptotically optimal codes correcting memory defects of fixed multiplicity. *Problems of Information Transmission*, 25(4):132–138, 10 1989.

[4] Ilya I. Dumer. Asymptotically optimal linear codes for correcting defects of linearly increasing multiplicity. *Problems of Information Transmission*, 26(2):93–104, 04 1990.

[5] T. Filler, J. Judas, and J. Fridrich. Minimizing additive distortion in steganography using syndrome-trellis codes. volume 6, pages 920–935, Sep. 2011.

[6] Jessica Fridrich, Miroslav Goljan, and David Soukal. Perturbed quantization steganography with wet paper codes. In *Proceedings of the 2004 Workshop on Multimedia and Security*, MM&#38;Sec '04, pages 4–15, New York, NY, USA, 2004. ACM.

[7] Jessica Fridrich, Miroslav Goljan, and David Soukal. Efficient wet paper codes. In *Proceedings of the 7th International Conference on Information Hiding*, IH'05, pages 204–218, Berlin, Heidelberg, 2005. Springer-Verlag.

[8] F. Galand and G. Kabatiansky. Information hiding by coverings. In *Proceedings 2003 IEEE Information Theory Workshop (Cat. No.03EX674)*, pages 151–154, March 2003.

[9] F. Galand and G. A. Kabatiansky. Coverings, centered codes, and combinatorial steganography. *Probl. Inf. Transm.*, 45(3):289–294, September 2009.

[10] S. I. Gelfand and M. S. Pinsker. Coding for channel with random parameters. *Probl. Contr. and Inform. Theory*, 9(1):19–31, 1980.

[11] Jason Hiney, Tejas Dakve, Krzysztof Szczypiorski, and Kris Gaj. Using facebook for image steganography. In *Proceedings of the 2015 10th International Conference on Availability, Reliability and Security*, ARES '15, pages 442–447, Washington, DC, USA, 2015. IEEE Computer Society.

[12] Andrew D. Ker, Patrick Bas, Rainer Böhme, Rémi Cogranne, Scott Craver, Tomáš Filler, Jessica Fridrich, and Tomáš Pevný. Moving steganography and steganalysis from the laboratory into the real world. In *Proceedings of the First ACM Workshop on Information Hiding and Multimedia Security*, IH&#38;MMSec '13, pages 45–58, New York, NY, USA, 2013. ACM.

[13] C. Kin-Cleaves and A. D. Ker. Adaptive steganography in the noisy channel with dual-syndrome trellis codes. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7, Dec 2018.

[14] T. Pevny and J. Fridrich. Detection of double-compression in jpeg images for applications in steganography. *IEEE Transactions on Information Forensics and Security*, 3(2):247–258, June 2008.

[15] Alin C. Popescu and Hany Farid. Statistical tools for digital forensics. In *Proceedings of the 6th International Conference on Information Hiding*, IH'04, pages 128–147, Berlin, Heidelberg, 2004. Springer-Verlag.

[16] T. H. Thai, R. Cogranne, F. Retraint, and T. Doan. Jpeg quantization step estimation and its applications to digital image forensics. *IEEE Transactions on Information Forensics and Security*, 12(1):123–133, Jan 2017.

[17] B.S. Tsybakov. Additive group codes for defect correction. *Problems Inform. Transmission*, 11(1):88–90, 01 1975.

[18] A V. Kuznetsov and B S. Tsybakov. Coding for memories with defective cells. *Problems Inform. Transmission*, 10(2):132–138, 01 1974.