Signal Processing 195 (2022) 108498

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Improving robust adaptive steganography via minimizing channel errors

Kai Zeng[a,b], Kejiang Chen[a,b,*], Weiming Zhang[a,b,*], Yaofei Wang[a,b], Nenghai Yu[a,b]

[a] School of Information Science and Technology, University of Science and Technology of China, Key Laboratory of Electromagnetic Space Information, Chinese Academy of Sciences, China
[b] Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, China

ARTICLE INFO

Article history:
Received 13 October 2021
Revised 11 December 2021
Accepted 6 February 2022
Available online 10 February 2022

Keywords:
Data hiding
Steganography
JPEG compression
Security
Robust

ABSTRACT

Online Social Networks (OSNs) are becoming increasingly entrenched in peoples lives and a huge number of images are shared on them every day, which are well-suited platforms for image steganography. Generally, the image can be regarded as the channel for steganography. However, OSNs usually perform lossy processing on uploaded images, which invalidates most of the current non-robust steganography algorithms. To solve the problem of poor detection resistance of existing robust steganography, we divide the causes of channel errors into two parts: steganography-related and steganography-independent. First, we propose a novel method to eliminate the effect of steganography-independent part. The message is embedded in the channel-processed cover which will modify some elements, and then the corresponding elements in the original image are replaced with the modified elements to generate the stego for transmission. Then for the steganography-related part, the wet paper model is employed to minimize the channel error rate. The proposed algorithm can not only resist JPEG recompression but also enhancement filtering which was not considered in previous algorithms. Experimental results show that the proposed algorithm surpasses previous methods in terms of both robustness and security by a clear margin. Besides, take the example of one of the most complex platforms, Facebook, this method can achieve error-free steganography without error correction codes.

1. Introduction

Steganography is a science and art of covert communication that conceals secret messages in digital media without being detected [1–3]. Many different mediums can be used in steganography, such as text, audio, video, or digital image. Among them, JPEG images are now widely used in Online Social Networks (OSNs), such as Facebook and Twitter, because they can provide high-level visual quality with less storage costs [4]. At present, the most remarkable steganographic schemes for JPEG images are based on Syndrome-Trellis Codes (STCs) [5] and recently developed Steganographic Polar Codes (SPC) [6] that can minimize the researcher-defined distortions while embedding messages.

With the powerful steganographic codes, the latest researches focused on how to define effective distortion functions. There are a lot of distortion functions defined for JPEG images, such as J-UNIWARD (JPEG UNIversal WAvelet Relative Distortion) [7], UERD (Uniform Embedding Revisited Distortion) [8], GUED (Generalized Uniform Embedding Distortion) [9], BET (Block Entropy Transformation) [10] and J-MiPOD (Minimizing the Power of Optimal Detector for JPEG domain) [11]. Their main mission is to assign low costs to the coefficients in complex areas and high costs to the coefficients in smooth areas of an image, which is also known as the "Complexity-First Rule" [12]. Recently, non-additive distortion functions for JPEG images which consider the interaction of modifications are defined to keep the continuity of adjacent image blocks in the spatial domain [13–15].

The aforementioned framework of "distortion functions + STCs" performs well on lossless channels. With the development of the Internet, OSNs are becoming increasingly entrenched in people's lives and a huge number of images are shared on them every day, which is a well-suited platform for image steganography. Steganographers and recipients can achieve behavioral security by disguising steganographic behavior as the everyday behavior of ordinary users. However, images transmitted over OSNs usually suffer from lossy processes such as resizing, JPEG compression, or image enhancement. Generally the cover image can be regarded as the channel for steganographic communication. The processing of the stego between sending and receiving is called chan-

* Corresponding authors.
E-mail addresses: zk0128@mail.ustc.edu.cn (K. Zeng), chenkj@mail.ustc.edu.cn (K. Chen), zhangwm@ustc.edu.cn (W. Zhang), yaofei@mail.ustc.edu.cn (Y. Wang), ynh@ustc.edu.cn (N. Yu).

https://doi.org/10.1016/j.sigpro.2022.108498
0165-1684/

nel processing. The channel processing in this paper is the processing of OSNs. These operations will fail the message extraction of STCs because the stego image is modified during transmission. Kin-Cleaves et al. [16] deeply analyzed the performance of STCs on lossy channels and pointed that previous frameworks are not available as well as there will be error diffusion after embedding with STCs, where an error bit in the stego image may affect multiple bits in the extracted sequence. To make the steganography robust to the channel processing, there has been a growing number of publications [16–28]. Overall, they supplemented the original framework with many robustness strengthening operations, which mainly considering two aspects: error correction codes and reducing the channel error rate.

A direct way to help correctly extract message is introducing error correction codes (ECC) to encode messages for error correction, which is called ECC-based operations in this paper. For example, the utilization of RS (Reed-Solomon) codes in [18,20] and BCH (Bose, Chaudhuri and Hocquenghem) codes in [17,22]. However, ECC-based operation can only provide limited robustness. Therefore, apart from ECC, other operations are utilized to enhance robustness.

Actually, considering error diffusion, reducing the channel error rate is the most effective method, in other words, to make the changes in the stego as few as possible during channel processing. These operations are the primary reasons why the corresponding algorithms can resist JPEG compression and we categorize them into three types. The first one is to select the coefficients of the cover image that remains essentially unchanged after channel processing as the cover sequence, which is called "Robust Domain Selection" [19]. For example, Zhang et al. [18] selected the medium frequency DCT coefficients of JPEG images as binary cover to embed with DMAS (Dither Modulation-based robust Adaptive Steganography). Yu et al. [20] proposed GMAS (Generalized dither Modulation-based robust Adaptive Steganography) by expanding robust domain and introducing ternary embedding. These operations are robust but their capacities are low due to cover selection. The second type is to preprocess the cover image to make it resistant to JPEG compression. The representative is TCM (Transport Channel Matching) proposed by Zhao et al. [17], which repeatedly processes the image by applying channel manipulations until the image is nearly identical before and after channel processing and uses the preprocessed images for steganography. These methods also provide robustness but preprocessing modifications will decrease security. Repeated uploading and downloading are also behaviorally insecure. The third type is to encode the stego for error correction. Zhang et al. [21] use part of the cover to embed the message first and then embed the cyclic redundancy check (CRC) codes [23] derived from the embedded sequence into the remaining cover. This type of operation can be used in combination with the preprocessing and robust domain selection in practice to further improve robustness.

Overall, robust steganography algorithms based on traditional framework are shown in Fig. 1. The core of these algorithms is to reduce the channel error rate with robust domain selection or preprocessing. Coding message or stego with ECC is just an additional robustness enhancement. These robustness strengthening operations sacrifice much security for robustness, thus cannot achieve satisfying performance in terms of both. To overcome this limitation, this paper first analyzes the key issues in robust steganography: how to reduce channel error rate while maintaining security. We divide the causes of channel errors into two parts: steganography-related and steganography-independent. Then we propose a novel method to eliminate the effect of steganography-independent part. The message is embedded in the channel-processed cover with STCs which will modify some elements, and then the corresponding elements in the original image are replaced with the modified elements to generate the stego for transmission. Thereafter, we applied this method for Facebook as an example. For the impact of steganography-related part, we set the elements where the modification may yield errors as wet elements to minimize channel error rate. Notably, the proposed algorithm is resistant not only to JPEG recompression but also to enhancement filtering which was not even considered in previous steganographic algorithms. We verify the robustness of the algorithm through experiments on both JPEG compression channels and Facebook. Security of the algorithm is examined by feature-based steganalysis with DCTR (Discrete Cosine Transform Residual) [29] and current CNN-based steganalysis [30]. Simulated and real-world experiments show that the proposed algorithm outperforms previous algorithms in terms of both security and robustness. Besides, the algorithm can achieve error-free steganography on one of the most complex channels, Facebook, without ECC-based operations.

The rest of the paper is organized as follows. The next section introduces the notations and provides preliminaries on robust steganography in the traditional framework. Section 3 analyzes the problems faced in reducing channel error rate and presents the novel method. The implementation of the method on Facebook is described in Section 4, which is resistant to both JPEG compression and enhancement filtering. The consequences of comparative experiments and performance tests are shown in Section 5. Finally, Section 6 concludes this paper.

2. Preliminaries

2.1. STC-based steganography

Denote a cover obtained from a JPEG image as $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{X} = \{\mathcal{I}\}^n$, where $\mathcal{I} = \{-1024, \dots, 1023\}$ is the range of DCT coefficient value and n is its length. The original message is $\mathbf{m}$. The process that the sender modifies $\mathbf{x}$ to stego $\mathbf{y} = (y_1, \dots, y_n) \in \mathcal{Y}$ to embed a message is represented as

$$\mathrm{Emb}(\mathbf{x}, \mathbf{m}) = \mathbf{y}, \tag{1}$$

where $\mathcal{Y} = \mathcal{I}_1 \times \mathcal{I}_2 \times \cdots \times \mathcal{I}_n$ and $\mathcal{I}_i \subset \mathcal{I}$. We call the embedding operation *binary* if $|\mathcal{I}_i| = 2$, or *ternary* if $|\mathcal{I}_i| = 3$. This paper considers the case of *ternary embedding*, where the possible values of stego elements are restricted to $\mathcal{I}_i = \{\max(x_i - 1, -1024), x_i, \min(x_i + 1, 1023)\}$. The impact of embedding modifications can be measured using an additive distortion function D which is in the form

$$\mathrm{D}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} \rho_i(\mathbf{x}, y_i), \tag{2}$$

where $\rho_i : \mathcal{X} \times \mathcal{I} \to [0, +\infty)$, are bounded functions expressing the cost of replacing the cover element x_i with y_i. A payload-limited sender can embed and extract a message with fixed-length l using STCs [5] while minimizing the total distortion.

$$\mathrm{Emb}(\mathbf{x}, \mathbf{m}) = \arg \min_{\mathcal{P}(\mathbf{y}) \in \mathcal{C}(\mathbf{m})} \mathrm{D}(\mathbf{x}, \mathbf{y}), \tag{3}$$

$$\mathrm{Ext}(\mathbf{y}) = \mathbb{H}\mathcal{P}(\mathbf{y}), \tag{4}$$

where $\mathcal{P} : \mathcal{I}_i \to \{0, 1\}$ is a parity function, $\mathcal{P}(\mathbf{y}) = (\mathcal{P}(y_1), \dots, \mathcal{P}(y_n))$, $\mathbb{H} \in \{0, 1\}^{l \times n}$ is a parity-check matrix of the code, $\mathcal{C}(\mathbf{m}) = \{\mathbf{z} \in \{0, 1\}^n | \mathbb{H}\mathbf{z} = \mathbf{m}\}$ is the coset corresponding to syndrome $\mathbf{m}$, and all operations are in binary arithmetic. Ternary STC can be implemented using double-layered STCs [31]. Eq. (1) can be considered as a simplified representation for the embedding process of Eq. (3). Eq. (4) is the extraction process and it is the inverse of the embedding process on the lossless channel,

$$\mathrm{Ext}(\mathrm{Emb}(\mathbf{x}, \mathbf{m})) = \mathbf{m} \quad \forall \mathbf{x} \in \mathcal{X}, \forall \mathbf{m} \in \{0, 1\}^l. \tag{5}$$

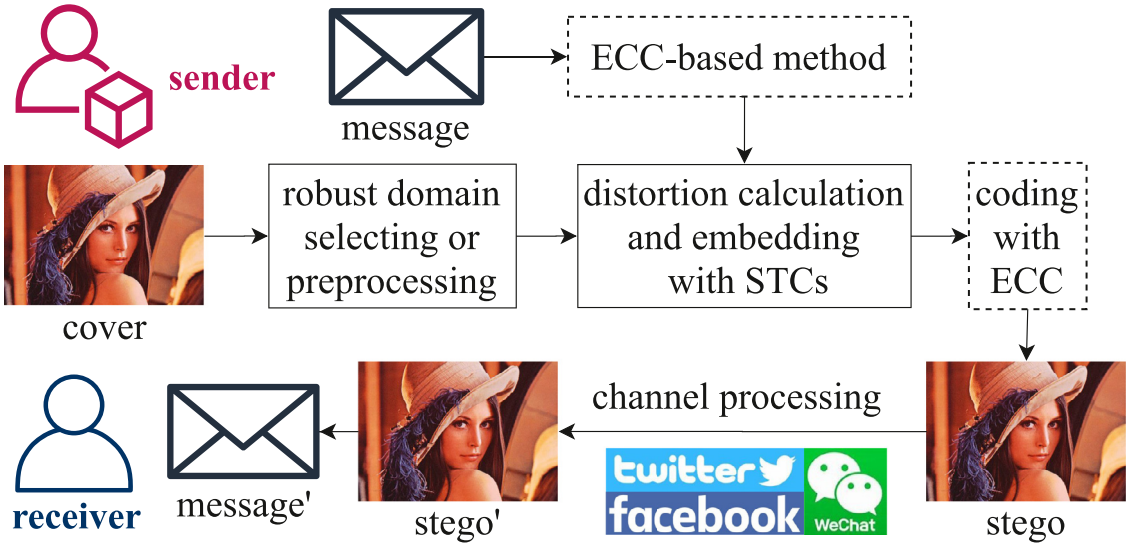

Fig. 1. The traditional framework of robust steganography.

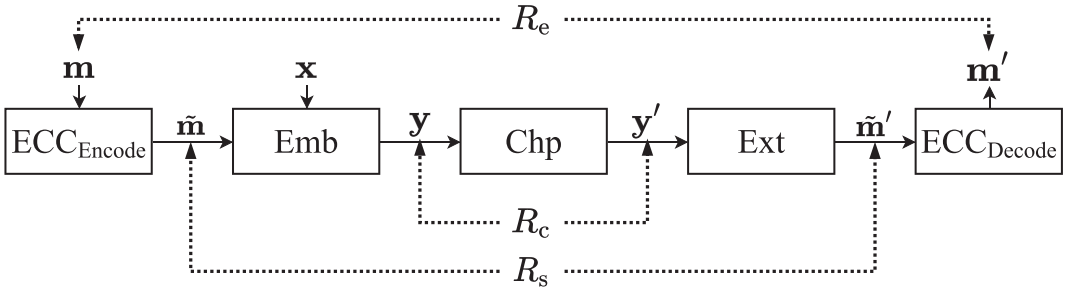

Fig. 2. The procedure that secret messages **m** are transmitted covertly by the robust steganography algorithms.

In this paper, we will mostly use the simple expression of embedding and extraction in Eqs. (3) and (5). The above is STC-based steganography in a lossless channel. The scenarios are more complex when it is applied to lossy channels.

2.2. Robust steganographic framework on OSNs

In practice, the stego image **y** is transmitted on the OSNs and **y** is changed to **y**′ after channel processing,

$$\mathrm{Chp}(\mathbf{y}) = \mathbf{y}'. \tag{6}$$

From Eq. (4), the message cannot be extracted correctly after **y** changes. To achieve both security and robustness, researchers equipped existing steganography frameworks with robustness strengthening operations and thus formed the robust steganography framework. As shown in the Fig. 1, what remains after removing all of the robustness strengthening operations from the robust steganography framework is the existing steganography framework. The state-of-the-art robust steganography algorithms [17,18,20] are built under this framework. Here we briefly describe how these algorithms covertly transmit messages through JPEG images.

As illustrated in the Fig. 2, the steganography sender encodes an original message **m** using the ECC-based operations. The encoded message of length $\tilde{l}$ is denoted by $\tilde{\mathbf{m}}$. The coefficients are chose from the cover image as a cover sequence **x** after the robust domain selection or preprocessing. STCs are used to generate a stego sequence **y** and then a stego image is generated by correspondingly modifying the coefficients in the cover image. Finally, the sender uploads the stego image to OSNs. Steganography receiver downloads the processed stego from OSNs. Then he can construct the stego sequence **y**′ in the same way as the sender, and decode $\tilde{\mathbf{m}}'$ and **m**′ using STCs and ECC-based operations sequentially.

2.3. Manipulation of OSNs on uploaded images

The processing of the uploaded images on different OSNs is complex and variable [32], but they all consist of several operations in resize, JPEG compression and enhancement filtering. This paper will consider one of the most complex OSNs, Facebook, as example. The processing of uploaded JPEG images on Facebook is shown in the Fig. 3. The platform first converts user-uploaded images to pixels for rounding and truncation. It then determines the quality factor (QF) for JPEG compression and whether to resize based on the content and size of the image, respectively. After that, the image content will be enhanced with filtering. Finally, the processed spatial image is compressed with a JPEG encoder.

In practice, not every image is subject to all of these lossy processes. Resizing can be avoided by selecting a cover image that meets the size criteria. JPEG recompression and enhancement filtering are unavoidable operations. To investigate the pattern of QF selection during Facebook recompression, we first generate two sets of images. Each set contains 51 different images which are randomly selected from UCID [33] and saved as JPEG images with the quality factor from 50 to 100, respectively. Notate the first set of JPEG images as $\mathcal{J}_1$ and the second set as $\mathcal{J}_2$. Upload $\mathcal{J}_1$ and $\mathcal{J}_2$ on Facebook and download to produce the corresponding Facebook-processed images $\mathcal{F}_1$ and $\mathcal{F}_2$. Record the QFs of the original JPEG images and corresponding downloaded images, as shown in Fig. 4. We can obtain the following conclusions from the experiments on this two sets of images:

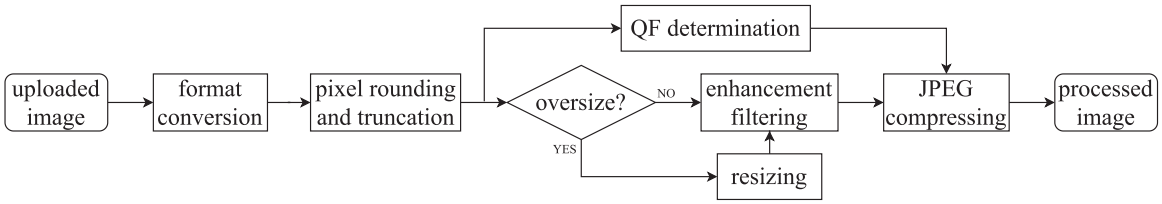

Fig. 3. The manipulation of uploaded JPEG images on Facebook. We discuss this in detail at Section 2.3.

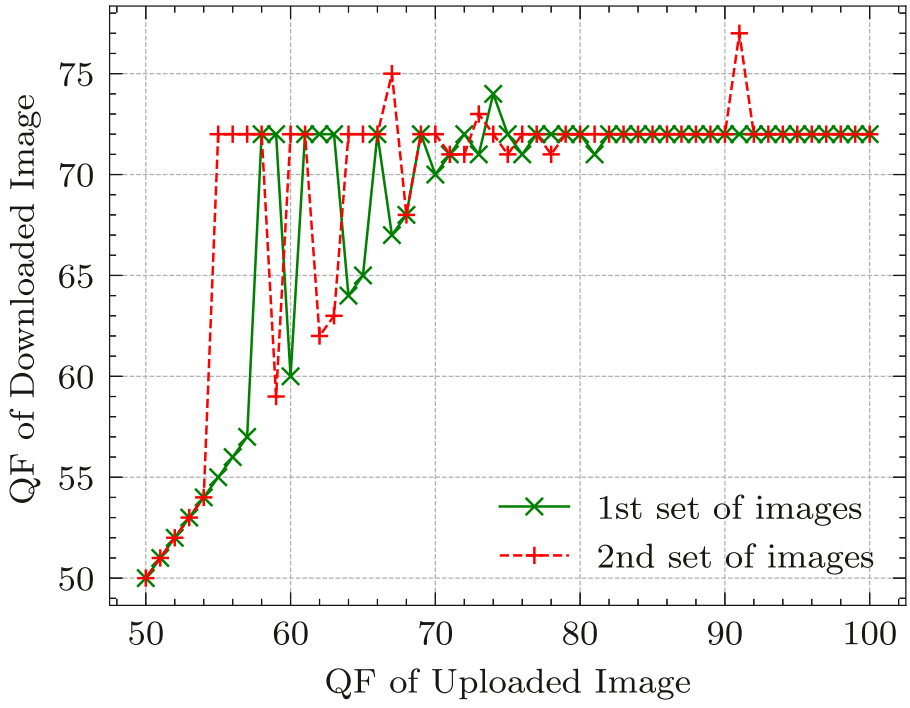

Fig. 4. Relationship between the QF of the upload images and the QF empolyed by Facebook.

- The original images with QF $\geqslant 72$ will usually be recompressed with QF $= 72$, and the original images with QF < 72 will be recompressed with the same QF or QF $= 72$.
- The same images are compressed by the same QF when uploaded and downloaded repeatedly, but different images with the same QF may be recompressed with different QFs.

To investigate the effects of enhancement filtering, we recompress the images in $\mathcal{J}$ ($\mathcal{J} = \mathcal{J}_1 \cup \mathcal{J}_2$) with the same quality factors as in Facebook processing. The processed images are denoted as $\mathcal{C}$. The different rates of non-zero coefficients are used to compare Facebook-processed and recompressed-only images, which is calculated as $R_{\mathrm{d}} = \delta(\mathbf{f} - \mathbf{c})/\delta(\mathbf{f})$, where $\mathbf{f}$ and $\mathbf{c}$ are the coefficient matrices of corresponding images in $\mathcal{F}$ ($\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$) and $\mathcal{C}$. $\delta(\mathbf{f})$ denotes the number of non-zero elements in $\mathbf{f}$, and δ is used to calculate the number of different bits. The average different rates of these images $\bar{R}_{\mathrm{d}} = 40\%$. This shows that the image after enhancement filtering and JPEG recompression differs a lot from the image only with JPEG recompression, which was not considered by previous robust steganography algorithms.

Overall, to achieve robust steganography that is resistant to OSNs processing, it is necessary to accomplish the resistance to JPEG recompression and enhancement filtering. Therefore, the lossy processes considered in this paper are the JPEG recompression with the same or different QFs and enhancement filtering processes. The analysis and solutions are presented in Section 4.

3. Basic concepts for minimizing channel error rate

3.1. Primary problem on robustness

During the transmission of messages over OSNs, three error rates can be calculated as shown in the Fig. 2:

- *channel error rate:*

$$R_{\mathrm{c}} = \frac{\delta(\mathbf{y} - \mathbf{y}')}{n}, \tag{7}$$

- *syndrome error rate:*

$$R_{\mathrm{s}} = \frac{\delta(\tilde{\mathbf{m}} - \tilde{\mathbf{m}}')}{\tilde{l}}, \tag{8}$$

- *message error rate:*

$$R_{\mathrm{e}} = \frac{\delta(\mathbf{m} - \mathbf{m}')}{l}, \tag{9}$$

where δ is used here to calculate the number of error bits. Minimizing message error rate R_{e} is the ultimate target of robustness strengthening operations and R_{e} is correlated with R_{s} and R_{c}. In the case where ECC-based operations are not considered, there are $\mathbb{H}\mathbf{y}' = \mathbf{m}'$ and $R_{\mathrm{s}} = R_{\mathrm{e}}$. Kin-Cleaves et al. [16] derived the relationship between R_{s} and correlation parameters on binary symmetric channel (BSC):

$$R_{\mathrm{s}} = R_{\mathrm{c}}\frac{an}{l} + O(R_{\mathrm{c}}^2), \tag{10}$$

where the submatrix of the parity-matrix $\mathbb{H}$ in STC is denoted as $\hat{\mathbb{H}}$, a is the average number of 1s in each column of $\hat{\mathbb{H}}$. R_s can be reduced by directly adjusting the a, l and n on the right side of Eq. (10), but this will drastically reduce security. Zhao et al. [17] yielded similar conclusions on the JPEG recompression channel. Therefore, Adjusting the last remaining R_c on the right side of the Eq. (10) is the key to reducing R_e.

The utilization of error correction codes (ECC) enhances robustness not only because of the properties of the correction code itself but also because of the increased length of the sequence encoded by the STCs. Under the circumstance, the message is extracted in two steps: $\mathbb{H}\mathbf{y}' = \tilde{\mathbf{m}}'$ and derive $\mathbf{m}'$ from $\tilde{\mathbf{m}}'$. $R_e < R_s$ which benefits from ECC. But this will reduce security because of the longer embedded message ($\tilde{l} > l$). More crucially, ECC-based operations cannot be used alone but only as additional tools to decrease R_e because of its limited error correction capability.

In summary, the key issue for robust steganography is how to reduce R_c while maintaining security. In this paper, we will minimize the channel error rate mainly by proposing a novel method rather than just using robustness strengthening operations. Let's start by decomposing the problem.

3.2. Problem decomposition

Consistent with the preceding, the cover $\mathbf{x}$ and the stego $\mathbf{y}$ can be obtained by Eq. (3). Divide the indices of the sequence into two non-overlapping sets according to whether the corresponding elements will be modified in embedding with STCs: $\boldsymbol{\zeta} = \{ i \mid x_i = y_i \}$ and $\boldsymbol{\eta} = \{ i \mid x_i \neq y_i \}$. Both cover and stego will be divided accordingly: $\mathbf{x} = \mathbf{x}_\zeta + \mathbf{x}_\eta$, $\mathbf{y} = \mathbf{y}_\zeta + \mathbf{y}_\eta$, where

$$\mathbf{x}_\zeta = (x_{\zeta 1}, \ldots, x_{\zeta n}), x_{\zeta i} = \begin{cases} x_i, & i \in \boldsymbol{\zeta} \\ 0, & else \end{cases}, \tag{11}$$

$$\mathbf{x}_\eta = (x_{\eta 1}, \ldots, x_{\eta n}), x_{\eta i} = \begin{cases} x_i, & i \in \boldsymbol{\eta} \\ 0, & else \end{cases}, \tag{12}$$

the definitions of $\mathbf{y}_\eta$ and $\mathbf{y}_\zeta$ are similar. As both $\mathbf{x}_\zeta$ and $\mathbf{y}_\zeta$ represent unmodified elements, they are the same before and after embedding. We denote the unmodified part before channel processing uniformly as $\mathbf{x}_\zeta$, so

$$\mathbf{y} = \mathbf{y}_\zeta + \mathbf{y}_\eta = \mathbf{x}_\zeta + \mathbf{y}_\eta. \tag{13}$$

We also have the same division mode for the channel processed sequences as the image before processing,

$$\mathrm{Chp}(\mathbf{x}) = \mathbf{x}' = \mathbf{x}'_\zeta + \mathbf{x}'_\eta, \tag{14}$$

$$\mathrm{Chp}(\mathbf{y}) = \mathrm{Chp}(\mathbf{x}_\zeta + \mathbf{y}_\eta) = \mathbf{y}' = \mathbf{x}^*_\zeta + \mathbf{y}'_\eta, \tag{15}$$

where the superscript and subscript have the same meaning as before. Although the unmodified parts are identical before channel processing, $\mathbf{x}_\zeta$ of stego may be affected by modifications of other elements during transmission. Distinguish from $\mathbf{x}'_\zeta$, the symbol $\mathbf{x}^*_\zeta$ is used to represent the channel processed $\mathbf{x}_\zeta$ of the stego in Eq. (15). In order to calculate the channel error rate, we first calculate the difference between the stego before and after channel processing,

$$\begin{aligned} \mathbf{y}' - \mathbf{y} &= \mathbf{x}^*_\zeta + \mathbf{y}'_\eta - (\mathbf{x}_\zeta + \mathbf{y}_\eta) \\ &= (\mathbf{x}^*_\zeta - \mathbf{x}_\zeta) + (\mathbf{y}'_\eta - \mathbf{y}_\eta) \\ &= (\mathbf{x}^*_\zeta - \mathbf{x}'_\zeta) + (\mathbf{x}'_\zeta - \mathbf{x}_\zeta) + (\mathbf{y}'_\eta - \mathbf{y}_\eta). \end{aligned} \tag{16}$$

According to Eq. (16), we can classify the changes in stego after channel processing into three types depending on their causes:

- $\mathbf{y}'_\eta - \mathbf{y}_\eta$: The difference between modified elements before and after channel processing.
- $\mathbf{x}'_\zeta - \mathbf{x}_\zeta$: The difference between unmodified elements of the cover before and after channel processing.
- $\mathbf{x}^*_\zeta - \mathbf{x}'_\zeta$: The difference between unmodified elements of the cover and of the stego after channel processing.

Modified or unmodified in the above description refers to elements whose indices are in sets $\boldsymbol{\eta}$ or $\boldsymbol{\zeta}$. Here we can express the channel error rate as follows,

$$\begin{aligned} R_c &= \frac{\delta(\mathbf{y}' - \mathbf{y})}{n} \\ &= \frac{\delta((\mathbf{x}^*_\zeta - \mathbf{x}'_\zeta) + (\mathbf{x}'_\zeta - \mathbf{x}_\zeta) + (\mathbf{y}'_\eta - \mathbf{y}_\eta))}{n}. \end{aligned} \tag{17}$$

In this way we have separated channel error bits into three parts. Notably, we divide the channel-processed changes in the elements of the stego that are not modified by steganography ($\mathbf{x}^*_\zeta - \mathbf{x}'_\zeta$) into two parts: affected by the channel only ($\mathbf{x}'_\zeta - \mathbf{x}_\zeta$) and affected by both the channel and steganography ($\mathbf{x}^*_\zeta - \mathbf{x}'_\zeta$). Therefore the above three parts can be grouped into two parts: steganography-related ($\mathbf{x}^*_\zeta - \mathbf{x}'_\zeta$ and $\mathbf{y}'_\eta - \mathbf{y}_\eta$) and steganography-independent ($\mathbf{x}'_\zeta - \mathbf{x}_\zeta$). The first three rows of Fig. 5 illustrate this decomposition. The following method will eliminate the effect of the steganography-independent part.

3.3. Eliminate the effect of steganography-independent part

The previous robust steganography algorithms reduced R_c as a whole. The problem decomposition reveals that the steganography-independent part ($\mathbf{x}'_\zeta - \mathbf{x}_\zeta$) has nothing to do with the embedding operation. The following tackles this part first.

Suppose we have obtained the original cover sequence $\mathbf{x}$ and the channel-processed sequence $\mathbf{x}'$. This assumption is easy to implement in practice and can be done offline if channel parameters are known. Then use STCs to modify the $\mathbf{x}'$ for embedding the message $\mathbf{m}$. To avoid introducing more symbols, we still use $\mathbf{y}$ as the stego sequence,

$$\mathrm{Emb}(\mathbf{x}', \mathbf{m}) = \mathbf{y}. \tag{18}$$

The previous symbols $\boldsymbol{\eta}$ and $\boldsymbol{\zeta}$ is followed, but here they are defined by whether the corresponding elements are modified or not when embedding the message in $\mathbf{x}'$ instead of $\mathbf{x}$. We still divide the elements in the sequences $\mathbf{x}'$ as before. Thus $\mathbf{x}' = \mathbf{x}'_\zeta + \mathbf{x}'_\eta$. Similar to Eq. (13),

$$\mathbf{y} = \mathbf{y}_\zeta + \mathbf{y}_\eta = \mathbf{x}'_\zeta + \mathbf{y}_\eta. \tag{19}$$

The above embedding process is shown in the third row of Fig. 5. Then we replace the corresponding elements in the original cover with the modified elements to generate the stego for transmission,

$$\mathbf{y}_t = \mathbf{x}_\zeta + \mathbf{y}_\eta. \tag{20}$$

To simplify the derivation and to make the method more widely applicable, we do not consider the diverse channel processing here and treat the channel processing as a black box. The details of the replacement will be explained in Section 4.1. $\mathbf{y}'_t$ is used to represent the stego after channel processing. As shown in the fourth row of Fig. 5,

$$\mathrm{Chp}(\mathbf{y}_t) = \mathrm{Chp}(\mathbf{x}_\zeta + \mathbf{y}_\eta) = \mathbf{y}'_t = \mathbf{x}^*_\zeta + \mathbf{y}'_\eta. \tag{21}$$

Then the extractor wants $\mathbf{y}'_t$ and $\mathbf{y}$ to be the same instead of $\mathbf{y}'_t$ and $\mathbf{y}_t$. The message can only be correctly extracted from the transmitted cover after it has been processed by the channel. The channel error rate in this situation is

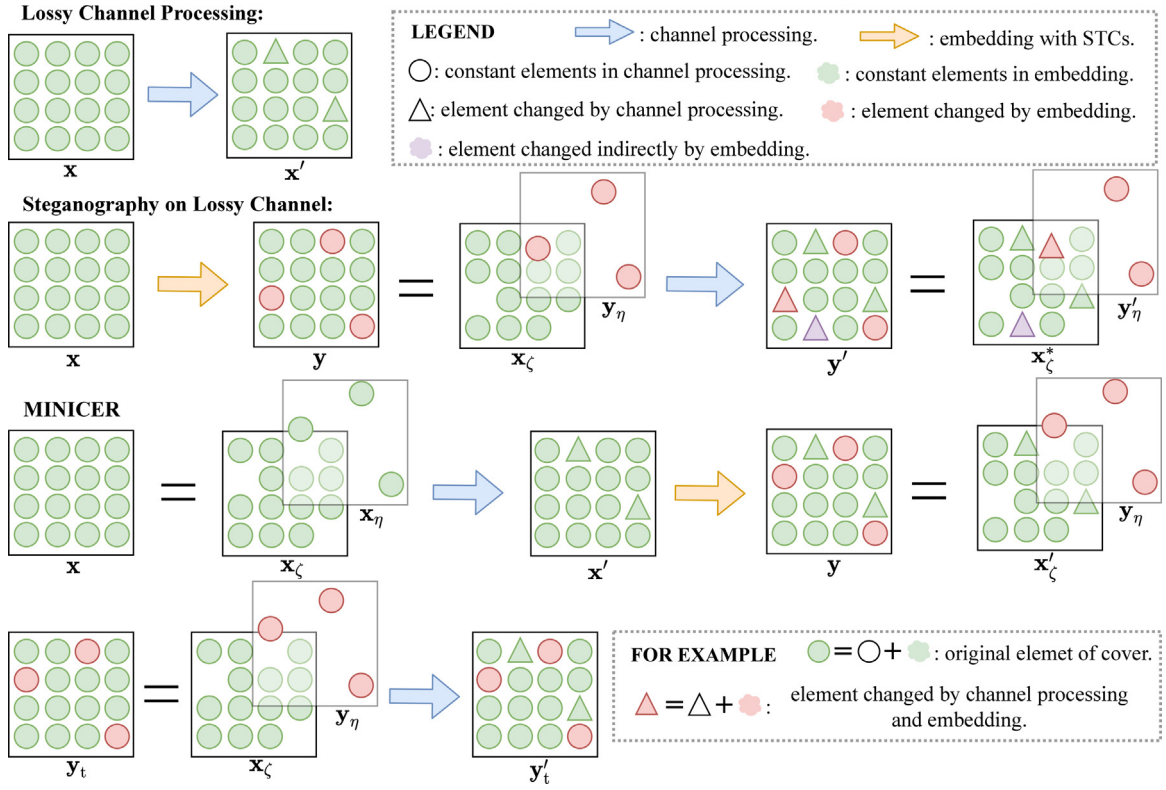

Fig. 5. Diagrammatic representation of problem decomposition and simplification. There is a detailed description in Sections 3.2 and 3.3.

$$
\begin{aligned}
R_{\mathrm{c}}^{*} &= \frac{\delta(\mathbf{y}_{\mathrm{t}}' - \mathbf{y})}{n} \\
&= \frac{\delta(\mathbf{x}_{\zeta}^{*} + \mathbf{y}_{\eta}' - (\mathbf{x}_{\zeta}' + \mathbf{y}_{\eta}))}{n} \\
&= \frac{\delta((\mathbf{x}_{\zeta}^{*} - \mathbf{x}_{\zeta}') + (\mathbf{y}_{\eta}' - \mathbf{y}_{\eta}))}{n}
\end{aligned} \tag{22}
$$

Since the sets $\boldsymbol{\eta}$ and $\boldsymbol{\zeta}$ of index division are non-overlapping, additionally, from the definition of δ function it is easy to derive that

$$
\delta((\mathbf{x}_{\zeta}^{*} - \mathbf{x}_{\zeta}') + (\mathbf{y}_{\eta}' - \mathbf{y}_{\eta})) = \delta(\mathbf{x}_{\zeta}^{*} - \mathbf{x}_{\zeta}') + \delta(\mathbf{y}_{\eta}' - \mathbf{y}_{\eta}). \tag{23}
$$

Substituting it into the Eq. (22), we have

$$
R_{\mathrm{c}}^{*} = \frac{\Delta_{\mathrm{E}} + \Delta_{\eta}}{n}, \tag{24}
$$

where $\Delta_{\mathrm{E}} = \delta(\mathbf{x}_{\zeta}^{*} - \mathbf{x}_{\zeta}')$ and $\Delta_{\eta} = \delta(\mathbf{y}_{\eta}' - \mathbf{y}_{\eta})$ are the number of previously mentioned two channel error causes. Comparing Eq. (17) and (22), the channel error rate is no longer affected by the original changes in the cover after channel processing. Since the number of steganographic modifications is always relatively small, keeping the unmodified part unchanged during transmission was a challenge for the previous algorithms. And $\mathbf{x}_{\zeta}' - \mathbf{x}_{\zeta}$ usually has a large impact on the R_{c}. Without the influence of this part, we can achieve stronger security. The remaining two parts (Δ_{E} and Δ_{η}) are both relevant to embedding operations. Here the problem is simplified by eliminating the effect of $\mathbf{x}_{\zeta}' - \mathbf{x}_{\zeta}$.

From the above description, we proposed a novel method called **MINImizing Channel Error Rate** (**MINICER**) and illustrate it in the last two rows of the Fig. 5. The embedding process based on MINICER is a two-step process.

i. Modification: embed the message in channel-processed cover $\mathbf{x}'$ with STCs, which will modify some elements in $\mathbf{x}'$.

ii. Replacement: replace the corresponding elements in the original cover $\mathbf{x}$ with the modified elements in $\mathbf{x}'$ to generate the stego for transmission. This is shown in Eq. (20).

To extract the message from the channel-processed cover, [34,35] proposed operations to map the modifications on the lossy processed cover back to the original cover. But they consider only one lossy process, such as DCT coefficient quantization in [34] or nearest-neighbor interpolation in [35], where DCT coefficients quantization is the quantization of DCT coefficients using different quantization tables. In these two cases, mapping can be achieved directly by correspondingly modifying the elements in the cover. In a real-world environment, there are usually multiple lossy processes and modification mapping is not available. So we employ the replacement that is feasible for complex channels. When applying MINICER to a real OSN, we need to consider the following issues.

- How to achieve a reasonable replacement? The replacement needs to be applicable to complex OSNs' processing. This is discussed in Section 4.1.
- How to define a suitable distortion on $\mathbf{x}'$? We usually minimize $\mathrm{D}(\mathbf{x}', \mathbf{y})$ when modifying $\mathbf{x}'$ to generate $\mathbf{y}$ with STCs. But in fact, $\mathrm{D}(\mathbf{x}, \mathbf{y}_{\mathrm{t}})$ should be minimized for security. The solution to this part is given in Section 4.2.
- How to minimize the channel error rate? From Eq. (24), there are still two parts (Δ_{E} and Δ_{η}) to deal with in R_{c}^{*}. We address this in Section 4.3.

Next, we will give some insights about how to answer the three questions and design a robust steganography algorithm based on MINICER with applications in channel introduced in Section 2.3.

4. The implementation of MINICER

In this section, we introduce the lossy operations that are present on OSNs first. Then we describe the robust steganographic method MINICER.

In the experiments in Section 2.3, the main lossy processes for uploaded images are JPEG recompression and enhancement filtering. The process of recompressing an image with QF $= q_1$ to an image with QF $= q_2$ is as follows:

$$\mathbf{s} = \mathrm{TRU}([\mathrm{IDCT}(\mathbf{d} \times \mathbf{q}_1)] + 128), \tag{25}$$

$$\mathbf{d}' = [(\mathrm{DCT}(\mathbf{s} - 128)/\mathbf{q}_2], \tag{26}$$

where $\mathbf{d} = (d_1, \ldots, d_{64})$, $(d_i \in \mathcal{I},\ i = 1, \ldots, 64)$ is an 8×8 coefficient block in the image to be recompressed, $\mathbf{s}$ and $\mathbf{d}'$ are the corresponding pixel block and recompressed coefficient block, $\mathbf{q}_1$ and $\mathbf{q}_2$ are quantization tables corresponding to QF $= q_1$ and QF $= q_2$, respectively. DCT$(\cdot)$ and IDCT$(\cdot)$ are the Discrete Cosine Transform (DCT) and inverse DCT, $[\cdot]$ and TRU$(\cdot)$ are rounding and truncation operation. In this paper, the multiplication and division between blocks and quantization tables are performed between the corresponding elements. Therefore, JPEG recompression is a known process, while the details of enhancement filtering are not known. We regard it as a perturbation imposed on the pixels depending on the image content. The experiments in Section 2.3 prove that the effect of enhanced filtering is the same for the same image. Steganographic modifications have only a minor impact on the image content and therefore have a minimal impact on the effect of enhancement filtering. Subsequently, based on the above analysis of the two lossy processes, we will design some robustness strengthening operations based on the method proposed in Section 3.3.

4.1. Quality factor synchronization

First, we discuss the basic replacement and its implementation. In the MINICER, we directly replace the corresponding elements in the original cover after the steganographic modification is performed on the channel-processed cover. Obviously, the direct replacement of the coefficients introduced in Section 3.3 is available on the channels that compress the uploaded images with only the same quality factor. However, when the channel compresses the uploaded images with different quality factors, simple replacement of coefficient will not work. To solve this problem, we decompose the JPEG recompression process, and further divide the process of Facebook into four lossy processes: DCT coefficient quantization, pixel rounding, pixel truncation, and enhancement filtering. All processes except the DCT coefficient quantization work on the pixels, but the quantization of DCT coefficients plays a major role among them. Therefore, we utilize quality factor (QF) synchronization to avoid the influence of DCT coefficient quantization. This synchronization is not a complete recompression process but just a quantization of the DCT coefficients with different quantization tables as follows,

$$\mathbf{x}'' = \left[(\mathbf{x}' \times \mathbf{q}_\mathrm{c})/\mathbf{q}_\mathrm{o}\right], \tag{27}$$

where $\mathbf{q}_\mathrm{o}$ and $\mathbf{q}_\mathrm{c}$ are quantization tables of the original cover $\mathbf{x}$ and the channel-processed cover $\mathbf{x}'$, respectively. $\mathbf{x}'' = (x_1'', \ldots, x_n'') \in \mathcal{X}$ is QF synchronized cover. After this, the modification and replacement will be performed on $\mathbf{x}''$ instead of $\mathbf{x}'$. Based on this, we will discuss the remaining two problems proposed in Section 3.3.

4.2. Realistic distortion calculation

STCs can minimize distortion caused by modifications while embedding message. And quality factor synchronization enables the replacement of modified elements and original cover elements. Therefore, to achieve security is to obtain the proper distortion on $\mathbf{x}''$. After quality factor synchronization and embedding with STCs, the corresponding element in the original cover will be replaced with the modified element. The final modifications are performed on the original cover. Hence, we first use the existing algorithm to calculate the distortion of modification on the original cover. Suppose we have $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)$ by the existing distortion function as initial distortion, where μ_i denotes the distortion of modifying x_i to $x_i + 1$ or $x_i - 1$ (symmetric distortion is used for narrative brevity and asymmetric distortion can also be used here). The replacement can be considered as modifying x_i to x_i'' on original cover. Then the distortion of the modifications on $\mathbf{x}''$ is

$$\mu_i^+ = \left|(x_i'' + 1) - x_i\right| \times \mu_i, \tag{28}$$

$$\mu_i^- = \left|(x_i'' - 1) - x_i\right| \times \mu_i, \tag{29}$$

where μ_i^+ and μ_i^- are the distortion on $\mathbf{x}''$ of modifying x_i'' to $x_i'' + 1$ and $x_i'' - 1$, $\left|(x_i'' + 1) - x_i\right|$ and $\left|(x_i'' - 1) - x_i\right|$ denote corresponding modification amplitude on x_i after replacement. To distinguish it from the initial distortion, we refer to this distortion as realistic distortion. This way, embedding on $\mathbf{x}''$ with STCs can minimize the distortion of the final modification on $\mathbf{x}$.

4.3. Wet elements assignment

According to the requirement of the modification in MINICER, it is necessary to minimize the channel error rate while minimizing the distortion. Specifically, there are two objectives: to maintain the modified elements unchanged before and after channel processing (minimizing Δ_η) and to maintain unmodified elements unchanged from the steganography modification (minimizing Δ_E). To attain these, we regard the practical channel as a wet paper channel [36]. That is, the steganography sender writes on a cover that has several elements that cannot be modified and calls these elements wet elements. During transmission, the stego image dries out, the receiver does not need to determine which points were wet to read the message. Normally make the distortion of the modification on the wet point infinite, i.e., $\rho_i = \infty$ if x_i is a wet element. Then effective wet paper coding can be achieved with STCs. Alternatively, the sender may allow a small number of wet elements to be modified. One can set the distortion of all wet cover elements to $\hat{\rho}_i = W$, $W > \sum_{\rho_i < \infty} \rho_i$ and $\hat{\rho}_i = \rho_i$ for i dry. Such a setting has almost no impact on the embedding efficiency [5]. Therefore, according to the objective of minimizing the channel error rate, we need to set the coefficients that may change before and after channel processing as wet elements (minimizing Δ_η) and set the coefficients whose modification will cause other coefficients to change as wet elements (minimizing Δ_E). Since the wet elements set in this paper are coefficients where errors *may* occur, the latter setting is adopted and makes $W = 10^{13}$. In the following, we will analyze them in detail.

4.3.1. Minimizing Δ_η

From Section 3.2, $\Delta_\eta \triangleq \delta(\mathbf{y}_\eta' - \mathbf{y}_\eta)$. To minimize this, we need to set the coefficients that are likely to change before and after channel processing as wet elements. As described in Section 4.1, there are four lossy processes on uploaded images to Facebook. With the quality factor synchronization in place, we will consider only three operations that perform on pixels: pixel rounding, pixel truncation, and enhancement filtering. We will deal with each operation separately according to the magnitude of its impact.

All coefficients are affected by pixel rounding, but the operation changes the pixel by only $(-0.5, 0.5]$. If each element of the quantization matrix is large than 1 (QF < 92), pixel rounding has almost

no effect on the coefficient [37]. From the Fig. 4, Facebook usually uses QF $\leqslant 72 < 92$, so we also don't consider the effect of pixel rounding. In the absence of enhancement filtering, pixel truncation results in the majority of DCT coefficient changes. Since JPEG compressing is processed in blocks, the truncation operation affects all coefficients in the block. Consequently, we will set the entire block affected by the truncation as a *wet block*, that is, all coefficients within the block are wet elements. As before, $\mathbf{d}$ is an 8×8 coefficient block in the original cover, calculate spatial values without truncation and rounding,

$$\hat{\mathbf{s}} = \mathrm{IDCT}(\mathbf{d} \times \mathbf{q}_{\mathrm{o}}) + 128. \tag{30}$$

Determine if $\hat{s}_i \in \hat{\mathbf{s}}$ satisfies $\hat{s}_i > 255$ or $\hat{s}_i < 0$, if so, set the block as the wet block. And we call such $\mathbf{d}$ and $\hat{\mathbf{s}}$ "overflow". For the distortion μ_i^+ and μ_i^- corresponding to the coefficients of this block, $\mu_i^+ = \mu_i^- = W$.

Finally, we also need to consider the effect of enhancement filtering on the modified coefficients. The operation is performed on pixels and its details are not known [32]. But we are only concerned with whether the operation results in error bits. Thus, for the original cover and the synchronized cover, if $x_i \neq x_i''$, it means that this block is strongly influenced by the enhancement filtering. The corresponding block also needs to be set as a wet block and the detailed manipulation is as described before.

4.3.2. Minimizing Δ_{E}

From Section 3.2, $\Delta_{\mathrm{E}} \triangleq \delta(\mathbf{x}_\zeta^* - \mathbf{x}_\zeta')$. To minimize this, we set the coefficients whose modification will cause other coefficients to change as wet elements. First, we also disregard the DCT coefficients quantization and pixel rounding here. Moreover, because the region that is strongly affected by the enhancement filtering is also set as a wet block, only the truncation after the steganography modification is considered here. In the previous section, we set the coefficients in the original cover that may change considerably after channel processing and quality factor synchronization as wet elements. Therefore, the "± 1" modification on $\mathbf{x}''$ is equivalent to the same modification on corresponding elements of $\mathbf{x}$. Let $\mathbf{a}^j = (a_1^j, \ldots, a_{64}^j)$ is a 8×8 block, $a_i^j = 1$ if $i = j$ and $a_i^j = 0$ if $i \neq j$ ($a_i^j \in \mathbf{a}^j,\ i = 1, 2, \ldots, 64$). Calculate the spatial values corresponding to the block $\mathbf{d}$ after the modification for each dry element,

$$\hat{\mathbf{s}}^{\pm j} = \mathrm{IDCT}((\mathbf{d} \pm \mathbf{a}^j) \times \mathbf{q}_{\mathrm{o}}) + 128,\ (\mathrm{j} = 1, \ldots, 64). \tag{31}$$

Determine as before whether $\hat{\mathbf{s}}^{\pm j}$ will be truncated, then, for example, if $\hat{\mathbf{s}}^{+5}$ will be truncated, the distortion of "$+1$" corresponding to d_5 will be $\mu^+ = W$. Unlike before, we only set *wet point* here. Because in this case, the key is the influence of the change in a certain DCT coefficient.

However, this method requires an inverse DCT for each possible modification. This can significantly increase the running time of the algorithm. We can calculate the spatial values corresponding to each possible modification in advance: $\mathbf{A}^{\pm j} = \mathrm{IDCT}((\pm \mathbf{a}^j) \times \mathbf{q}_{\mathrm{c}})$. Store the obtained data, then $\hat{\mathbf{s}}^{\pm j} = \hat{\mathbf{s}} + \mathbf{A}^{\pm j}$. Thus, we only need to perform the inverse DCT once for each block and this will greatly reduce the time consumption.

4.4. Stego image generation

The complete procedure of the proposed method is shown in Fig. 6 and Algorithm 1. When we want to communicate covertly through OSNs, we need to prepare an original cover image and a channel-processed version of that image. All coefficients in the image can be employed as cover elements. ECC-based operations can also be used for this algorithm in practice. Next, we synchronize the quality factors of the processed image and the original image as well as calculate the realistic distortion with this two images.

Algorithm 1 Robust steganography MINICER.

Require: Embedded messages $\mathbf{m}$, the original cover $\mathbf{x}$, the quantization tables of the original cover $\mathbf{q}_{\mathrm{o}}$, the channel-processed cover $\mathbf{x}\&'$, the quantization tables of the channel-processed cover $\mathbf{q}_{\mathrm{c}}$, the distortion function $\mathrm{D}_{dsf}(\cdot)$ and the coding scheme $\mathrm{STC}(\cdot)$;

Ensure: Stego for transmission $\mathbf{y}_{\mathrm{t}}$;

procedure EMBEDDING
  $\mathbf{x}\&'' = (\mathbf{x}' \times \mathbf{q}_{\mathrm{c}})/\mathbf{q}_{\mathrm{o}}$ ▷ Synchronize quality factor
  $\boldsymbol{\mu} = \mathrm{D}_{dsf}(\mathbf{x})$ ▷ Calculate initial distortion
  $\boldsymbol{\mu}^+ = \left|(\mathbf{x}\&'' + 1) - \mathbf{x}\right| \times \boldsymbol{\mu},\ \boldsymbol{\mu}^- = \left|(\mathbf{x}\&'' - 1) - \mathbf{x}\right| \times \boldsymbol{\mu}$
  ▷ Calculate realistic distortion
  $\mathbf{x} = (\mathbf{d}_i)_{1\times n},\ \mathbf{x}\&'' = (\mathbf{d}_i)\&''_{1\times n},\ \boldsymbol{\mu}^+ = (\boldsymbol{\mu}_i^+)_{1\times n},\ \boldsymbol{\mu}^- = (\boldsymbol{\mu}_i^-)_{1\times n}$ ▷ Divided into 8×8 blocks
  for $i = 1$ to n **do** ▷ Assign wet element
    if isOverflow($\mathbf{d}_i$) **then**
      $\boldsymbol{\mu}_i^+ = \boldsymbol{\mu}_i^+ = \boldsymbol{W}$ ▷ Wet blocks
    else if $\mathbf{d}_i \neq \mathbf{d}_i\&''$ **then**
      $\boldsymbol{\mu}_i^+ = \boldsymbol{\mu}_i^+ = \boldsymbol{W}$ ▷ Wet blocks
    else
      for $j = 1$ to 64 **do** ▷ Wet point
        if isOverflow($\mathbf{d}_i + \mathbf{a}^j$) **then** $\mu_{ij}^+ = W$
        else if isOverflow($\mathbf{d}_i - \mathbf{a}^j$) **then** $\mu_{ij}^- = W$
        end if
      end for
    end if
  end for
  $\tilde{\mathbf{m}} = \mathrm{ECC}(\mathbf{m})$ ▷ Encoding message with ECC
  $\mathbf{y} = \mathrm{STC}(\mathbf{x}\ \&'', \boldsymbol{\mu}^+, \boldsymbol{\mu}^-, \tilde{\mathbf{m}})$ ▷ Modification
  $\mathbf{y} = \mathbf{y}_\zeta + \mathbf{y}_\eta,\ \mathbf{x} = \mathbf{x}_\zeta + \mathbf{x}_\eta$
  $\mathbf{y}_{\mathrm{t}} = \mathbf{x}_\zeta + \mathbf{y}_\eta$ ▷ Replacement
  return $\mathbf{y}_{\mathrm{t}}$
end procedure

Then we set the wet blocks and wet points according to the principle proposed in Section 3.3. After setting up all the wet elements, embedding on synchronized cover $\mathbf{x}''$ with STCs can minimize both channel error rate and distortion. Finally, replace the corresponding element in original cover $\mathbf{x}$ to generate the stego $\mathbf{y}_{\mathrm{t}}$ for transmission.

Similarly, the steganography receiver has to synchronize the quality factor and then extract the message. This can be expressed as $\mathbf{y}_{\mathrm{t}}'' = (\mathbf{y}_{\mathrm{t}}' \times \mathbf{q}_{\mathrm{s}})/\mathbf{q}_{\mathrm{c}}$, $\mathbb{H}\mathbf{y}_{\mathrm{t}}'' = \tilde{\mathbf{m}}'$, where $\mathbf{y}_{\mathrm{t}}'$ is the channel processed stego. It requires that both sides of the covert communication have to share the quality factor of the cover image. Therefore, the quality factor of the cover images should be agreed upon in advance or transmitted together with other parameters of the STCs.

5. Experiment

The performance of the algorithm will be experimentally investigated in this section. First, we introduce the experimental setup. Then we test the resistance to JPEG recompression and security of the proposed algorithm. We also analyze the relative wetness of the wet paper model introduced in Section 4.3 and the respective contribution of wet blocks and wet elements. Finally, we test the algorithm on Facebook, that is, the ability of the algorithm to simultaneously resist JPEG compression and enhance filtering is verified.

5.1. Experimental setup

The images used in the local simulation contain two parts. The first part is BOSSBase 1.01 [38] containing 10,000 grayscale

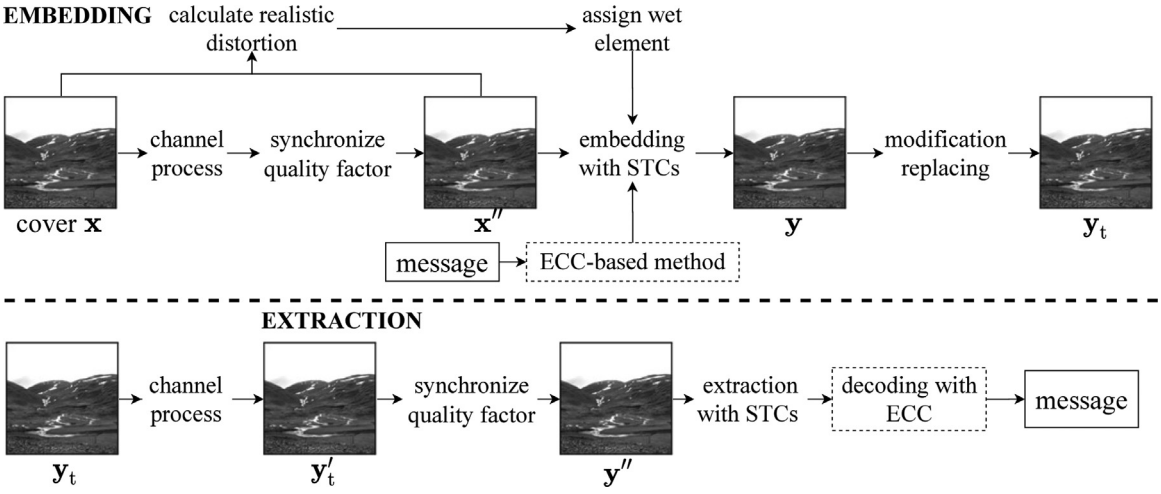

Fig. 6. Embedding and extraction of the proposed algorithm.

512×512 images. The second part is a color version of BOSSBase 1.01 containing 10,000 color $512 \times 512 \times 3$ images, it is obtained as follows. We utilize Patterned Pixel Grouping (PPG) in dcraw (ver.9.26) on full-resolution raw images for demosaicking and downsample the obtained images, so the dimension of smaller image is 512 and for central cropping to 512×512. By default, dcraw writes PPM with 8-bit samples, a BT.709 gamma curve, a histogram-based white level and the sRGB colorspace. The downsampling algorithm uses the Bicubic kernel in MATLAB.

The proposed algorithm needs to use the existing distortion function on JPEG images to calculate the initial distortion. It is feasible to adopt any distortion function here. In the experiments, we employ J-UNIWARD [7] and UERD [8]. These two algorithms treat all coefficients as modifiable elements and define the same distortion for "+1" or "−1" on cover coefficients (symmetric distortion). The coding scheme is a ternary version of multi-layered STCs [5]. The larger height h of the submatrix improves the security, but it also increases the computational complexity and reduces the robustness due to error diffusion. Similarly, preprocessing messages with ECC-based operations can sacrifice security to improve robustness. However, neither of these is the focus of this paper. In the experiment, we fix $h = 10$ and do not employ the ECC-based operations to avoid these effects.

When we need to perform the recompression in local simulation, we use MATLAB and Phil Sallee's JPEG toolbox [39] to execute the recompression process expressed by the formula Eqs. (25) and (26). The relative payload $p = l/n_{\text{nzac}}$, where n_{nzac} is the number of non-zero AC DCT coefficients of the original cover image. The robustness of the algorithm is measured by the message error rate in Eq. (9). Since the ECC-based operations are not applied ($\tilde{\mathbf{m}} = \tilde{\mathbf{m}}'$), $R_e = R_s$.

To test the security performance, we employ the FLD ensemble classifier [40] trained with DCTR feature [29] and the SRNet [30]. The input of the ensemble classifier is the extracted features from cover and stego images. Its output is the classification error probability, which is the mean of the false alarm rate and the missed detection rate over 10 runs using 5000/5000 database splits. SRNet is fed the cover and stego images and the image set is randomly split into a training set with 7000 cover and stego image pairs, a validation set with 500 image pairs and the remaining 2500 images were used for testing. The training first runs for 300 epochs with an initial learning rate of $r_1 = 0.001$ and then for an additional 100 epochs with a learning rate of $r_2 = 0.0001$. A separate classifier and a deep network are trained for each embedding algorithm and payload. The security of the algorithm is evaluated by the classification error probability and the testing error probability. We denote them both as $\bar{P}_E$.

We compare the proposed algorithm with the state-of-the-art robust steganographic algorithms GMAS [20] and JCRIS (JPEG Compression ResIstant Solution) [17]. These two algorithms are based on "Robust Domain Selection" and "Cover Preprocessing", respectively. All above experimental environments will stay the same when comparing. This is why we chose JCRIS which employs TCM without ECC. GMAS here also has no ECC.

5.2. JPEG recompression resistant comparison

Although the processing of uploaded images in OSNs may also involve enhancement filtering and the quality factors of recompression may be also different as stated in Section 2.3, it is still necessary to examine the JPEG recompression resistance of robust steganography algorithms and the reasons are two-fold.

- The first is about JPEG recompression. The unknown quality factors in recompression can be divided into two cases: the same or a different quality factor as the original image. The performance in these two cases can illustrate the resistance of the algorithm to recompression with an unknown quality factor.
- The second is about enhanced filtering. Not all OSNs have this operation, such as Twitter and Wechat. Even if the process is unknowable to us and subject to change, its impact is relatively small compared to recompression with different quality factors.

We will therefore test the resistance to recompression with the same quality factor and different quality factors.

The images in BOSSBase 1.01 are compressed as cover images using *imwrite* in MATLAB with quality factors 70, 80, and 90, respectively. After that, different algorithms are employed to obtain the stego images. Recompress the stego images with different quality factors, extract the message, and calculate the average message error rate. The experimental results are shown in Table 1, where "Proposed-WP" and "Proposed-WB" denotes the proposed algorithm for setting only wet point or wet block when performing wet element assignment, respectively. Notably, JCRIS can only resist recompression of the same quality factor, whereas the proposed algorithm and GMAS are available in both cases. It can be seen that the proposed algorithm achieves comparable robustness to GMAS when Cover QF is 90 and recompression with the same QF. This is due to the influence of pixel rounding when the QF is large as described in Section 4.3.1. In other cases, the robustness is greatly improved compared to previous algorithms. Comparing

Table 1
Comparison of the Resistance to JPEG Recompression.

Cover QF	Recompression QF	Algorithms	Average Message Error Rate									
			J-UNIWARD					UERD				
			0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
90	90	GMAS	**0.0047**	**0.0037**	**0.0034**	**0.0033**	**0.0034**	0.0064	**0.0056**	**0.0056**	**0.0058**	**0.0059**
		JCRIS	0.0182	0.0170	0.0173	0.0183	0.0202	0.0261	0.0247	0.0264	0.0300	0.0332
		Proposed-WP	0.0579	0.0444	0.0382	0.0345	0.0328	0.0460	0.0373	0.0329	0.0304	0.0292
		Proposed-WB	0.0056	0.0064	0.0074	0.0086	0.0098	0.0058	0.0060	0.0062	0.0065	0.0069
		Proposed	0.0056	0.0064	0.0073	0.0085	0.0097	**0.0057**	0.0059	0.0060	0.0063	0.0067
80	80	GMAS	0.0022	0.0028	0.0024	0.0027	0.0029	0.0033	0.0037	0.0042	0.0047	0.0052
		JCRIS	0.0144	0.0146	0.067	0.0180	0.0212	0.0230	0.0251	0.0292	0.0343	0.0399
		Proposed-WP	0.0310	0.0248	0.0218	0.0199	0.0189	0.0411	0.0319	0.0283	0.266	0.0260
		Proposed-WB	0.0005	0.0009	0.0012	0.0016	0.0019	0.0015	0.0021	0.0028	0.0035	0.0043
		Proposed	**0.0005**	**0.0008**	**0.0011**	**0.0014**	**0.0017**	**0.0009**	**0.0013**	**0.0018**	**0.0024**	**0.0031**
70	70	GMAS	0.0023	0.0024	0.0028	0.0030	0.0033	0.0035	0.0038	0.0044	0.0050	0.0056
		JCRIS	0.0155	0.0167	0.0196	0.0211	0.0252	0.0253	0.0284	0.0335	0.0393	0.0460
		Proposed-WP	0.0256	0.0205	0.0182	0.0169	0.0162	0.0353	0.0281	0.0253	0.0243	0.0242
		Proposed-WB	0.0001	0.0003	0.0004	0.0006	0.0008	0.0004	0.0008	0.0013	0.0019	0.0028
		Proposed	**0.0001**	**0.0002**	**0.0004**	**0.0005**	**0.0007**	**0.0002**	**0.0005**	**0.0009**	**0.0015**	**0.0021**
	80	GMAS	0.0175	0.0117	0.0100	0.0095	0.0093	0.0120	0.0134	0.0122	0.0120	0.0123
		Proposed-WP	0.0627	0.0497	0.0436	0.0399	0.0376	0.0743	0.0579	0.0512	0.0475	0.0457
		Proposed-WB	0.0073	0.0064	0.0063	0.0065	0.0067	0.0043	0.0044	0.0050	0.0059	0.0071
		Proposed	**0.0072**	**0.0063**	**0.0062**	**0.0064**	**0.0066**	**0.0040**	**0.0041**	**0.0046**	**0.0055**	**0.0056**
	90	GMAS	0.0112	0.0079	0.0071	0.0070	0.0071	0.0120	0.0094	0.0091	0.0094	0.0100
		Proposed-WP	0.0284	0.0226	0.0200	0.0184	0.0177	0.0388	0.0305	0.0274	0.0262	0.0259
		Proposed-WB	0.0001	0.0003	0.0005	0.0006	0.0009	0.0004	0.0008	0.0013	0.0020	0.0029
		Proposed	**0.0001**	**0.0002**	**0.0004**	**0.0005**	**0.0007**	**0.0002**	**0.0005**	**0.0010**	**0.0016**	**0.0023**

Table 2
Detection Resistance of the Proposed Algorithm to Ensemble Classifier Trained with DCTR Features.

Cover QF	Recompression QF	Algorithms	Detection Error Probability									
			J-UNIWARD					UERD				
			0.1	0.2	0.3	0.4	0.5	0.1	0.2	0.3	0.4	0.5
70	70	Proposed	0.4091	0.2843	0.1745	0.0964	0.0485	0.3980	0.2782	0.1741	0.0999	0.0490
		Proposed-WB	0.4112	0.2837	0.1746	0.0961	0.0507	0.4008	0.2766	0.1760	0.1023	0.0491
		Proposed-WP	**0.4165**	**0.2985**	**0.1898**	**0.1118**	**0.0597**	**0.4043**	**0.2899**	**0.1842**	**0.1113**	**0.0058**
	80	Proposed	0.4105	0.2853	0.1755	0.0977	0.0490	**0.4055**	0.2775	0.1750	0.0987	0.0519
		Proposed-WB	0.4137	0.2856	0.1781	0.0978	0.0485	0.4008	0.2775	0.1759	0.1009	0.0476
		Proposed-WP	**0.4156**	**0.3002**	**0.1943**	**0.1127**	**0.0599**	0.4045	**0.2906**	**0.1820**	**0.1133**	**0.0575**
	90	Proposed	0.4151	0.2848	0.1758	0.0960	0.0500	0.4021	0.2779	0.1747	0.0985	0.0505
		Proposed-WB	**0.4203**	0.2843	0.1758	0.0940	0.0500	0.4064	0.2824	0.1741	0.1019	0.0499
		Proposed-WP	0.4182	**0.3081**	**0.1936**	**0.1155**	**0.0594**	**0.4067**	**0.2857**	**0.1873**	**0.1105**	**0.0560**
	Original (without WB&WP)		0.4200	0.3045	0.1974	0.1183	0.0628	0.4014	0.2890	0.1891	0.1168	0.0585
80	80	Proposed	0.4326	0.3260	0.2192	0.1345	0.0741	0.4197	0.3102	0.2075	0.1257	0.0719
		Proposed-WB	0.4321	0.3292	0.2208	0.1348	0.0709	**0.4256**	0.3106	0.2068	0.1244	0.0687
		Proposed-WP	**0.4426**	**0.3370**	**0.2344**	**0.1538**	**0.0862**	0.4255	**0.3169**	**0.2188**	**0.1376**	**0.0736**
	Original (without WB&WP)		0.4396	0.3353	0.2388	0.1539	0.0902	0.4288	0.2890	0.2207	0.1396	0.0791
90	90	Proposed	0.4661	0.3873	0.2999	0.2075	0.1300	0.4542	0.3690	**0.2819**	0.1884	0.1154
		Proposed-WB	0.4620	0.3918	0.2967	0.2092	0.1256	0.4533	0.3708	0.2787	0.1856	0.1158
		Proposed-WP	**0.4677**	**0.4013**	**0.3097**	**0.2190**	**0.1437**	**0.4580**	**0.3771**	0.2810	**0.1897**	**0.1216**
	Original (without WB&WP)		0.4701	0.4046	0.3166	0.2300	0.1490	0.4590	0.2890	0.1891	0.2007	0.1313

the results of "Proposed-WP" and "Proposed-WB" reveals that wet blocks have a stronger effect on robustness than wet points.

5.3. Comparison of security

In this section, we test the detection resistance of the generated stego images in Section 5.2. The cover images used for steganalysis are the original images compressed from BOSSBase 1.01, while the stego images are the corresponding images generated by different algorithms.

First, we investigate the correlation between the security of the stego produced by the proposed algorithm and the quality factor as well as the wet element assignment. To avoid redundant experiments, only the ensemble classifier trained with DCTR features is employed here as the steganalyzer. The experimental results are shown in Table 2. It can be seen that as the quality factor of cover images becomes larger, the security of the corresponding stego images increases. And when the quality factor of cover is identical, the security of the stego is similar for different recompression quality factors. The reason is that as the quality factor of cover decreases, "±1" in the coefficients will have a greater impact on the pixels, which is easier to detect. The different recompression quality factors only change the cover of the STCs and have little influence on security. The security of algorithms with different wet element assignments also differs. Simply put, the less robust, the stronger the security. This will be explained in Section 5.4. In addition, by comparing the security with that of steganography without wet elements, we think that the impact of wet elements assignment on security is acceptable.

Then we compare the security of different algorithms against feature-based and CNN-based steganalyzer. The experimental results are displayed in Figs. 7 and 8. Note that the stego used for the cover quality factor of 70 comparison is generated at a recompression quality factor of 70. It can be seen that the proposed al-

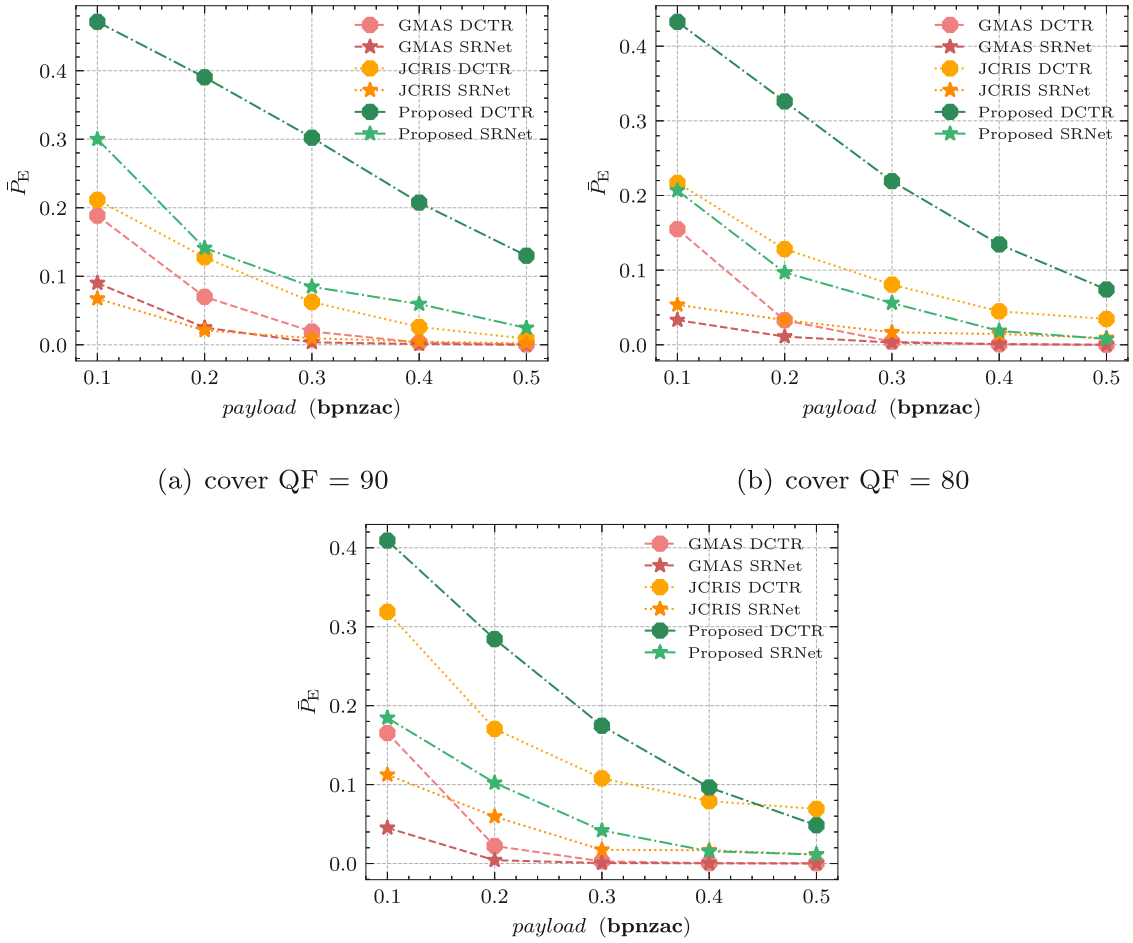

Fig. 7. Comparison of the security among various algorithms with initial distortion function J-UNIWARD and (a) cover QF = 90, (b) cover QF = 80, (c) cover QF = 70.

gorithm has a considerable improvement in security over the previous algorithms. Specifically, when cover QF = 90, *payload* = 0.1 and employing J-UNIWARD, the detection error probability can be improved by 26% when against DCTR and 20% when against SRNet. Moreover, calculating the initial distortion with J-UNIWARD is better than UERD, especially when against SRNet.

5.4. Investigation on proposed wet paper channel

To minimize the channel error rate, we consider the cover as wet paper and have two options for setting the wet elements: wet block and wet point. The wet paper channel is described by relative wetness, which is calculated like this,

$$\tau = \frac{\left|\{i|\mu_i^+ = W\}\right| + \left|\{i|\mu_i^- = W\}\right|}{2n}, \tag{32}$$

where n is the length of the cover. Since only one of the plus or minus modifications is not allowed when setting wet points. Unlike the previous calculation of relative wetness, $\tau_p = |\{i|\rho_i = \infty\}|/n$, here the distortion of plus one and minus one are calculated separately and the denominator is also $2n$ correspondingly. Eq. (32) gives the same results as τ_p when only wet blocks are considered. The relative wetness of the three different wet elements assignments are shown in Table 3, where "WP", "WB" and "WP+WB" denotes the wet paper channels with setting wet point, wet block only, and both of them. The images of the three different quality factors used in the experiment are the same as before.

Table 3
Average Relative Wetness under Different Wet Element Assignment Schemes.

Distortion Function	Wet Element Assignment	Relative Wetness		
		70	80	90
J-UNIWARD	WP	0.0332	0.0309	0.0253
	WB	0.0986	0.0840	0.0603
	WP+WB	0.1010	0.0852	0.0612
UERD	WP	0.0332	0.0303	0.0253
	WB	0.0986	0.0840	0.0603
	WP+WB	0.1010	0.0855	0.0612

It can be seen that the relative wetness of the "WB" is obviously larger than that of the "WP" and the relative wetness of "WB" and "WP+WB" is similar. This explains the experimental results of the algorithm corresponding to these wet paper channels. That is, setting wet blocks can clearly improve robustness and reduce security, while setting wet points has a relatively small impact. Notably, the relative wetness here is small and does not interfere with the feasibility of the STCs [5].

5.5. Practical application on facebook

In this section, we test the performance of the proposed algorithm on Facebook as an example of OSNs. Notably the proposed algorithm needs a channel-processed cover, this will be realized

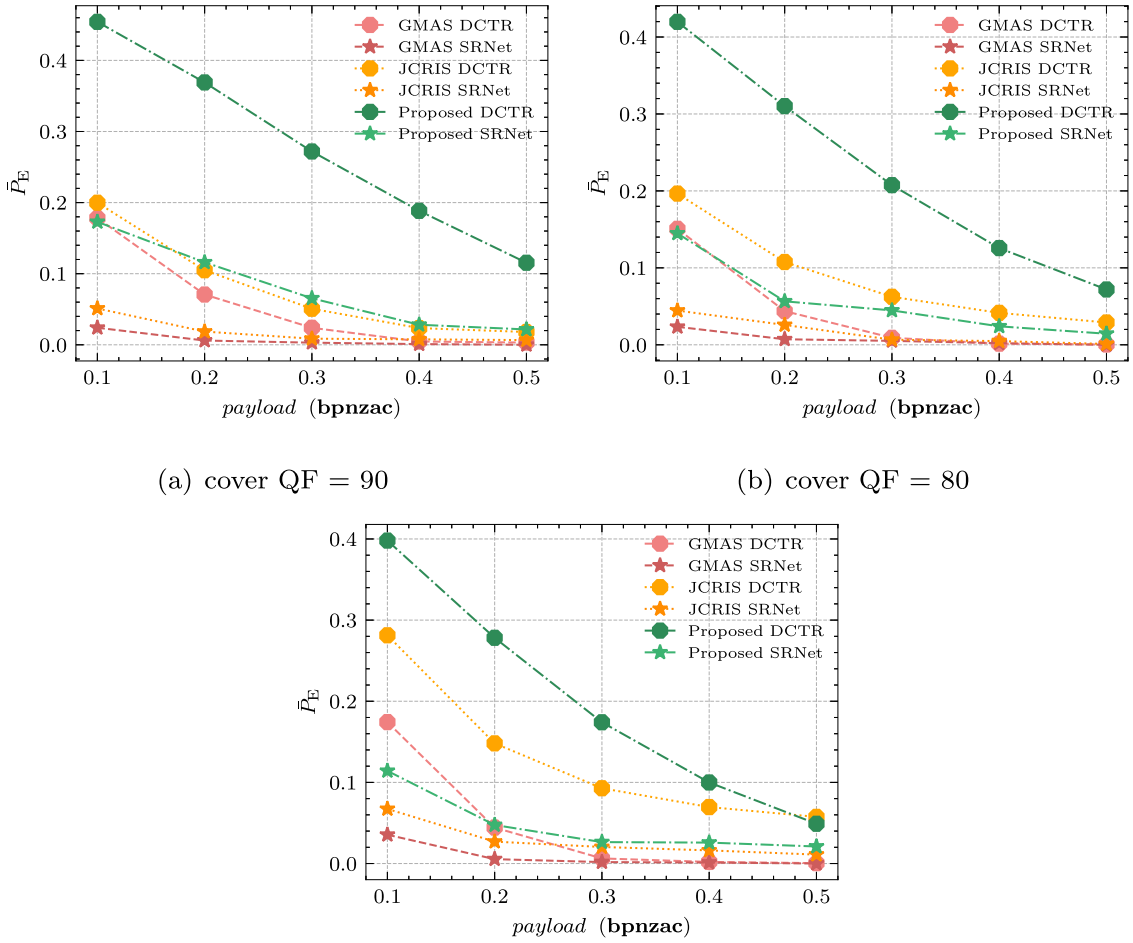

Fig. 8. Comparison of the security among various algorithms with initial distortion function UERD and (a) cover QF = 90, (b) cover QF = 80, (c) cover QF = 70.

through a upload and download on Facebook and the algorithm in [17] (JCRIS in previous experiments) requires about 10 times of upload and download. Thus the algorithm in this paper is also more secure in its behavior.

Since the images uploaded by normal users on OSNs are almost color images, we experiment with the images in the color version of BOSSBase 1.01. First, we produce two sets of covers with different quality factors. Randomly select 80 images from these and save them as JPEG images with QF = 60. Randomly select 80 images again and save them with QF = 71. From the previous pattern of Facebook's choice for recompression quality factors, it is clear that most of the images on Facebook are of quality factor less than 72. The steganographer has to disguise the cover image as a downloaded image from the Internet, because using the original image directly would reveal his privacy, such as photography time, location, and camera parameters. However, a too low quality factor will degrade the image quality, so we choose 60 and 71 as examples. In this experiment, only the Y channel of the color image is adopted for steganography and calculating the initial distortion with J-UNIWARD. Then we subdivide the images with different quality factors into two groups and embed them with $p = 0.1$ and $p = 0.3$, respectively. The message error rate of this algorithm applied to Facebook is shown in Table 4. It is observed that the proposed algorithm achieves error-free transmission when the quality factor of cover is 60, and also works well when it is 71. Besides, we record the relative wetness of the algorithm on Facebook in Table 4. The result is that the relative wetness is low and therefore does not interfere with the STCs.

Table 4
Performance of the Proposed Algorithm on Facebook.

Cover QF	p	$\bar{R}_e$	$\bar{\tau}$
60	0.1	0	0.0641
	0.3	0	0.0455
71	0.1	0.0003	0.0511
	0.3	0.0101	0.0282

6. Conclusions

Since online social networks (OSNs) usually perform lossy operations on the uploaded images. Steganography algorithms used over OSNs require robustness along with undetectability. However, previous algorithms need to sacrifice much security for robustness.

This paper proposes MINICER to upgrade the performance. According to MINICER, we only need to focus on those modified elements in steganography. After that, this paper proposes a practical steganography algorithm based on MINICER and manipulation of OSNs. In addition to recompression, the algorithm also is resistant to the enhancement filtering which was not considered by previous algorithms. Experiments on both the simulated JPEG recompressed channel and Facebook demonstrate the effec-

tiveness of these operations. Notably, for minimizing R_c^*, we employ the wet paper model to transform the problem into finding potential error elements. In fact, previous algorithms also look for these elements. However, the treatment of these elements based on the previous method is to avoid these elements by robust domain selection [18,20] or to make them less error-prone by preprocessing [17]. These two operations will reduce the capacity and security of the algorithm. More importantly, for unknown processes such as enhancement filtering, it may be easy to find error-prone elements, but more difficult to design the suitable preprocessing or select the robust domain. This is the primary benefit of working with the proposed method and wet paper model.

Credit Author Statement

Kai Zeng: methodology, data curation, software, writing original draft, review&editing. **Kejiang Chen**: formal analysis, validation, software, review&editing. **Weiming Zhang**: conceptualization, project administration, review&editing. **Yaofei Wang**: software, review&editing. **Nenghai Yu**: resources, funding acquisition.

Declaration of Competing Interest

None.

Acknowledgement

This work was supported in part by the Natural Science Foundation of China under Grant 62102386, 62002334, 62072421, and 62121002, and by China Postdoctoral Science Foundation under Grant 2021M693091, and by Anhui Science Foundation of China under Grant 2008085QF296, and by Open Fund of Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation.

References

[1] W. Su, J. Ni, X. Hu, J. Huang, New design paradigm of distortion cost function for efficient JPEG steganography, Signal Process. 190 (2022) 108319.
[2] F. Li, Z. Yu, C. Qin, Gan-based spatial image steganography with cross feedback mechanism, Signal Process. 190 (2022) 108341.
[3] M. Yu, X. Yin, W. Liu, W. Lu, Secure halftone image steganography based on density preserving and distortion fusion, Signal Process. 188 (2021) 108227.
[4] G.K. Wallace, The JPEG still picture compression standard, IEEE Trans. Consum. Electron. 38 (1) (1992) xviii–xxxiv.
[5] T. Filler, J. Judas, J. Fridrich, Minimizing additive distortion in steganography using syndrome-trellis codes, IEEE Trans. Inf. ForensicsSecur. 6 (3–2) (2011) 920–935.
[6] W. Li, W. Zhang, L. Li, H. Zhou, N. Yu, Designing near-optimal steganographic codes in practice based on polar codes, IEEE Trans. Commun. 68 (7) (2020) 3948–3962.
[7] V. Holub, J.J. Fridrich, T. Denemark, Universal distortion function for steganography in an arbitrary domain, EURASIP J. Inf. Secur. 2014 (2014) 1.
[8] L. Guo, J. Ni, W. Su, C. Tang, Y.-Q. Shi, Using statistical image model for JPEG steganography: uniform embedding revisited, IEEE Trans. Inf. ForensicsSecur. 10 (12) (2015) 2669–2680.
[9] W. Su, J. Ni, X. Li, Y.-Q. Shi, A new distortion function design for JPEG steganography using the generalized uniform embedding strategy, IEEE Trans. Circuits Syst. Video Technol. 28 (12) (2018) 3545–3549.
[10] X. Hu, J. Ni, Y.-Q. Shi, Efficient JPEG steganography using domain transformation of embedding entropy, IEEE Signal Process. Lett. 25 (6) (2018) 773–777.
[11] R. Cogranne, Q. Giboulot, P. Bas, Steganography by minimizing statistical detectability: the cases of JPEG and color images, ACM Information Hiding and MultiMedia Security (IH&MMSec), 2020.
[12] B. Li, S. Tan, M. Wang, J. Huang, Investigation on cost assignment in spatial image steganography, IEEE Trans. Inf. ForensicsSecur. 9 (8) (2014) 1264–1277.
[13] W. Li, W. Zhang, K. Chen, W. Zhou, N. Yu, Defining joint distortion for JPEG steganography, in: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, Innsbruck, Austria, June 20–22, 2018, ACM, 2018, pp. 5–16.
[14] Y. Wang, W. Li, W. Zhang, X. Yu, K. Liu, N. Yu, BBC++: enhanced block boundary continuity on defining non-additive distortion for JPEG steganography, IEEE Trans. Circuits Syst.Video Technol. (2020). 1–1
[15] Y. Wang, W. Zhang, W. Li, N. Yu, Non-additive cost functions for JPEG steganography based on block boundary maintenance, IEEE Trans. Inf. ForensicsSecur. (2020). 1–1
[16] C. Kin-Cleaves, 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), IEEE, 2018, pp. 1–7.
[17] Z. Zhao, Q. Guan, H. Zhang, X. Zhao, Improving the robustness of adaptive steganographic algorithms based on transport channel matching, IEEE Trans. Inf. ForensicsSecur. 14 (7) (2019) 1843–1856.
[18] Y. Zhang, X. Zhu, C. Qin, C. Yang, X. Luo, Dither modulation based adaptive steganography resisting JPEG compression and statistic detection, Multimedia Tools Appl. 77 (14) (2018) 17913–17935.
[19] Y. Zhang, X. Luo, C. Yang, D. Ye, F. Liu, A framework of adaptive steganography resisting JPEG compression and detection, Secur. Commun. Netw. 9 (15) (2016) 2957–2971.
[20] X. Yu, K. Chen, Y. Wang, W. Li, W. Zhang, N. Yu, Robust adaptive steganography based on generalized dither modulation and expanded embedding domain, Signal Process. 168 (2020).
[21] Y. Zhang, X. Luo, X. Zhu, Z. Li, A.G. Bors, Enhancing reliability and efficiency for real-time robust adaptive steganography using cyclic redundancy check codes, J. Real-Time Image Process. 17 (1) (2020) 115–123.
[22] W. Lu, J. Zhang, X. Zhao, W. Zhang, J. Huang, Secure robust JPEG steganography based on autoencoder with adaptive BCH encoding, IEEE Trans. Circuits Syst. Video Technol. (2020).
[23] W.W. Peterson, D.T. Brown, Cyclic codes for error detection, Proc. IRE 49 (1) (1961) 228–235.
[24] Y. Zhang, X. Luo, Y. Guo, C. Qin, F. Liu, Multiple robustness enhancements for image adaptive steganography in lossy channels, IEEE Trans. Circuits Syst. Video Technol. 30 (8) (2019) 2750–2764.
[25] T. Qiao, S. Wang, X. Luo, Z. Zhu, Robust steganography resisting JPEG compression by improving selection of cover element, Signal Process. 183 (2021) 108048.
[26] L. Zhu, X. Luo, C. Yang, Y. Zhang, F. Liu, Invariances of JPEG-quantized DCT coefficients and their application in robust image steganography, Signal Process. 183 (2021) 108015.
[27] Y. Zhang, C. Qin, W. Zhang, F. Liu, X. Luo, On the fault-tolerant performance for a class of robust image steganography, Signal Process. 146 (2018) 99–111.
[28] F. Li, K. Wu, C. Qin, J. Lei, Anti-compression JPEG steganography over repetitive compression networks, Signal Process. 170 (2020) 107454.
[29] V. Holub, J.J. Fridrich, Low-complexity features for JPEG steganalysis using undecimated DCT, IEEE Trans. Inf. ForensicsSecur. 10 (2) (2015) 219–228.
[30] M. Boroumand, M. Chen, J. Fridrich, Deep residual network for steganalysis of digital images, IEEE Trans. Inf. Forensics Secur. 14 (5) (2018) 1181–1193.
[31] W. Zhang, X. Zhang, S. Wang, A double layered "plus-minus one" data embedding scheme, IEEE Signal Process. Lett. 14 (11) (2007) 848–851.
[32] W. Sun, J. Zhou, Y. Li, M. Cheung, J. She, Robust high-capacity watermarking over online social network shared images, IEEE Trans. Circuits Syst.Video Technol. (2020).
[33] G. Schaefer, M. Stich, UCID: an uncompressed color image database, in: Storage and Retrieval Methods and Applications for Multimedia 2004, San Jose, CA, USA, January 20, 2004, in: SPIE Proceedings, vol. 5307, SPIE, 2004, pp. 472–480.
[34] J. Tao, S. Li, X. Zhang, Z. Wang, Towards robust image steganography, IEEE Trans. Circuits Syst.Video Technol. 29 (2) (2019) 594–600.
[35] Y. Zhang, X. Luo, J. Wang, C. Yang, F. Liu, A robust image steganography method resistant to scaling and detection, J. Internet Technol. 19 (2) (2018) 607–618.
[36] J. Fridrich, M. Goljan, P. Lisonek, D. Soukal, Writing on wet paper, IEEE Trans. Signal Process. 53 (10) (2005) 3923–3935.
[37] Y. Niu, X. Li, Y. Zhao, R. Ni, An enhanced approach for detecting double JPEG compression with the same quantization matrix, Signal Process. Image Commun. 76 (2019) 89–96.
[38] P. Bas, T. Filler, T. Pevnỳ, "Break our steganographic system": the ins and outs of organizing BOSS, in: International Workshop on Information Hiding, Springer, 2011, pp. 59–70.
[39] P. Sallee, MATLAB JPEG toolbox, 2003, (http://www.philsallee.com/jpegtbx/index.html).
[40] J. Kodovský, J.J. Fridrich, V. Holub, Ensemble classifiers for steganalysis of digital media, IEEE Trans. Inf. ForensicsSecur. 7 (2) (2012) 432–444.