Robust Message Embedding via Attention Flow-Based Steganography

Huayuan Ye[1], Shenzhuo Zhang[1], Shiqi Jiang[1], Jing Liao[2], Shuhang Gu[3], Dejun Zheng[4], Changbo Wang[1], Chenhui Li[1*]
[1] East China Normal University [2] City University of Hong Kong
[3] University of Electronic Science and Technology of China, Chengdu, China
[4] Zhijiang College of Zhejiang Univeristy of Technology, Zhejiang, China
huayuan221@gmail.com; {10195102459, 52265901032}@stu.ecnu.edu.cn; jingliao@cityu.edu.hk;
shuhanggu@gmail.com; zhengdejun@zzjc.edu.cn; {cbwang, chli}@cs.ecnu.edu.cn

Abstract

*Image steganography can hide information in a host image and obtain a stego image that is perceptually indistinguishable from the original one. This technique has tremendous potential in scenarios like copyright protection and information retrospection. Some previous studies have proposed to enhance the robustness of the methods against image disturbances to increase their applicability. However, they generally cannot achieve a satisfying balance between the steganography quality and robustness. Instead of image-in-image steganography, we focus on the issue of message-in-image embedding that is robust to various real-world image distortions. This task aims to embed information into a natural image and the decoding result is required to be completely accurate, which increases the difficulty of data concealing and revealing. Inspired by the recent developments in transformer-based vision models, we discover that the tokenized representation of image is naturally suitable for steganography task. In this paper, we propose a novel message embedding framework, called **R**obust **M**essage **Steg**anography (RMSteg), which is competent to hide message via QR Code in a host image based on an normalizing flow-based model. The stego image derived by our method has imperceptible changes and the encoded message can be accurately restored even if the image is printed out and photographed. To our best knowledge, this is the first work that integrates the advantages of transformer models into normalizing flow. The code is available at https://github.com/huayuan4396/RMSteg.*

1. Introduction

Steganography, the art of hiding secret information in a carrier, has long been a prominent research direction. This technique is competent to embed information like images and text into target containers, thus achieving copyright protection [10, 37] and information retrospection [48, 52]. Steganography aims to prevent people from discovering the existence of secret data instead of obscuring meaning of data, differentiating it from cryptography. Specifically, image steganography uses image to carry secret information.

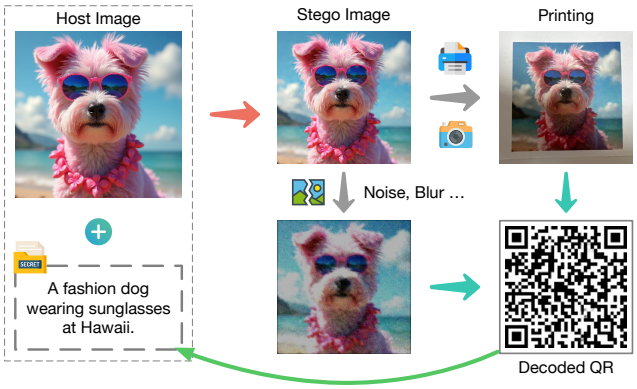

Figure 1. Compared with previous methods that can only embed limited bit-level information, RMSteg can achieve a much higher embedding capacity and meanwhile has better steganography quality. Also, it can survive various real-world distortions.

Traditional image steganography methods mainly modify the image in spatial domain [14, 17, 25, 28, 29] or transform domain [2, 36, 55]. This kind of method is easy to be detected by steganalysis techniques [9, 50], compromising security. Recently, with the developments of deep learning, some deep steganography methods have been proposed. Most of them are based on autoencoder [3, 44, 52, 54] and normalizing flow [5, 12, 16, 24, 47, 48].

Images can undergo various digital or real-world disturbances during dissemination. To enable the stego images to survive these distortions, some robust steganography methods have been proposed. They consider various distortion situations like light field messaging (LFM) [44], JPEG compression [47], etc. In the field of robust steganography, robust message embedding is very promising in many ap-

plication scenarios like hyperlink hiding [37] and metadata embedding [13, 52]. However, this task requires high decoding accuracy, which poses a challenge to the balance of decoding performance and stego image quality, especially when facing real-world distortions. Although some studies [10, 37] have proposed to hide messages in host images and try to make them survive printing and photography, which is among the most demanding situations that require high steganography robustness, they cannot achieve enough steganography quality and capacity at the same time.

As the most widely utilized method, normalizing flow-based model [6, 7, 18] has achieved impressive performance in various steganography tasks. Existing methods [5, 12, 16, 24, 47, 48] generally incorporate normalizing flow by utilizing a convolutional neural network (CNN) based backbone. However, according to our experiments, this kind of model design can lead to obvious artifacts in stego images when handling robust steganography tasks due to the lack of inner-channel feature fusion. Inspired by the transformer-based vision models, we discover that the tokenized representation of image is naturally suitable for robust steganography that requires highly abstract feature learning. As a result, we aim to take advantages of it to address the robust message-in-image steganography problem.

In this paper, we propose a new framework for message embedding, called **R**obust **M**essage **Steg**anography (RM-Steg), a simple demo is demonstrated in Figure 1. We use QR Code as the message carrier and encode it into the host image. Unlike previous methods that directly encode the secret image, we propose an invertible QR Code transition as a preprocessing step, which transforms the QR Code based on the features of the host image, lowering down the artifacts in stego images and meanwhile maintaining a high decoding accuracy. We outline a steganography model called AttnFlow, which integrates tokenized image representation into normalizing flow. We propose an attention affine coupling block (AACB) that leverages the attention mechanism [8, 41] instead of traditional CNN for invertible steganography function learning, thus significantly improving the stego image quality. Compared with previous methods, our method can overcome the aforementioned difficulties and achieve robust, high-quality and high-capacity message-in-image steganography. The main contributions of this paper include three aspects:

- We use QR Code as the message carrier and propose a transition scheme to transform the QR Code before steganography. This process can improve the stego image quality while maintaining decoding accuracy.
- We propose an invertible token fusion module that can effectively improve the steganography quality by simply including a small learnable matrix.
- We propose a normalizing flow-based steganography network that integrates the tokenized image representation. Our network can generate stego images with significantly higher quality and can survive extreme distortions. We use the case of printing and photography to validate our method's effectiveness.

2. Related Work

2.1. Image Steganography

Traditional Steganography Image steganography hides information in an image by performing imperceptible changes on a host image. Traditional methods modify the image in spatial or transform domain [3]. Spatial-domain steganography generally leverages least-significant-bit (LSB) replacement [25], bit plane complexity segmentation (BPCS) [17, 28] and palette reordering [14, 29] to conceal information. However, this kind of scheme may raise statistical anomalies that can be detected by steganalysis techniques [9, 50]. Some methods utilize high-dimensional features [30] and distortion constraints [20] to improve steganography security and quality. Transform-based steganography can hide data in a transformed domain using discrete cosine transform (DCT) [2] and discrete wavelet transform (DWT) [36, 55]. Due to the limited ability of feature representation and transformation, traditional methods generally cannot achieve a satisfying quality.

Deep Steganography Recently, various deep learning-based image steganography schemes have been proposed and have achieved impressive performance. HiDDeN [54] adopted the autoencoder (AE) to embed binary messages. Baluja [3] first utilized an end-to-end network to hide a color image in another. Some studies [10, 31, 34, 38, 39, 51] incorporated generative adversarial network (GAN) [11] to reduce the image artifacts and defend steganalysis. More recently, the invertible neural network (INN) [6, 7, 18] has been widely used for steganography. These methods successfully hide single [5, 16, 24] or multiple [12, 45, 48] images in a carrier image. There are also some studies focusing on coverless steganography [21, 23, 26, 49] that directly transforms the secret information into a cover image. These methods mainly focus on improving the embedding capacity instead of robustness, as a result, they generally cannot survive image distortions.

Robust Steganography Robust steganography allows information decoding even if the images are interfered with by digital transmission or real-world distortions, which is meaningful for scenarios like copyright protection, secret communication, etc. VisCode [52] hides QR Codes in host images and can survive image brightness changes and slight tampering. LFM [44] is robust to light field messaging. RIIS [47] considers JPEG compression and various kinds of noise separately based on a conditional network. StegaStamp [37] and ChartStamp [10] take printing and photography into account but can only embed very little informa-

tion at a cost of visual quality loss. As far as we know, existing methods cannot achieve both high-quality and high-capacity message steganography that is robust to extreme image distortions. In this paper, we aim to take the advantages of transformer-based model to address this problem.

2.2. Normalizing Flow-Based Models

Normalizing flow model was first proposed as a generative model by Dinh et al. [6]. With further improvement by RealNVP [7] and GLOW [18], it is also known as the invertible neural network (INN). INN can learn an invertible function using a set of affine coupling layers with shared parameters to map the original data distribution to a simple distribution (e.g., Gaussian distribution). Chen et al. [4] proposed an unbiased estimation for normalizing flow model. i-RevNet [15] utilizes an explicit inversion to improve the invertible architecture.

Recently, normalizing flow has been applied to various downstream tasks in computer vision, such as image [46] and video [56] super-resolution, image-to-image translation [40]. Especially, in the field of steganography, normalizing flow-based methods [5, 24, 48] have shown promising performance. HiNet [16] introduces the discrete wavelet transform to guide channel squeezing and improve the steganography quality. DeepMIH [12] hides single or multiple images with a saliency detection module. Mou et al. [27] incorporated a key-controllable network design to implement secure video steganography. Xu et al. [47] simulated distortions during model training to improve the robustness and security of their method. Although previous studies have leveraged various methods to improve the network architecture for better performance, they cannot attend to both image quality and steganography robustness simultaneously.

3. Method

3.1. Overview

Given a secret message T_s, we first encode it into a QR Code image I_q. The concealing procedure aims to embed I_q into a host image I_h and derive a stego image I_s that is perceptually similar to I_h. Then, I_s can suffer from various real-world image distortions, resulting in a distorted image I'_s. After that, the revealing procedure aims to restore a QR Code $\hat{I}_q$ from I'_s that can be successfully recognized to obtain the original message.

To achieve the aforementioned targets, we first leverage a QR Code transition scheme (Sec. 3.3) to transform the original QR Code according to the host image, reducing the artifacts it causes in the subsequent steganography process. Then, we use an invertible token fusion (ITF) module (Sec. 3.4) to improve the stego image quality. After that, we propose an AttnFlow model (Sec. 3.5) to perform message embedding. To make our method robust to real-world distortions, we incorporate a distortion simulation module during the training stage, which will be described in detail in Sec. 3.6. Fig. 2 demonstrates an overview of the pipeline of our RMSteg.

3.2. Preliminary: Normalizing Flow

Normalizing Flow [6, 7, 18], also called the invertible neural network (INN), is proposed to model a bijective projection from a complex distribution (e.g., images) to a tractable distribution (e.g. Gaussian distribution and Dirac distribution). This kind of model generally comprises several invertible affine coupling blocks (ACBs). The most basic ACB architecture is proposed by NICE [6], in which the input u^i of the i^{th} ACB is split into two parts, u_1^i and u_2^i, whose corresponding outputs are u_1^{i+1} and u_2^{i+1}, respectively. For the forward process, the following transformation is performed:

$$u_1^{i+1} = u_1^i + \sigma(u_2^i), \quad u_2^{i+1} = u_2^i + \delta(u_1^{i+1}), \tag{1}$$

where $\sigma(\cdot)$ and $\delta(\cdot)$ are arbitrary functions. Obviously, the backward process can be formulated as:

$$u_2^i = u_2^{i+1} - \delta(u_1^{i+1}), \quad u_1^i = u_1^{i+1} - \sigma(u_2^i). \tag{2}$$

In the normalizing flow architecture, $\delta(\cdot)$ and $\sigma(\cdot)$ in Equation 1 and Equation 2 can be implemented by neural network modules with shared parameters and inverse calculation manner. By stacking multiple ACBs, the network can learn an invertible transformation between two distributions. Since this scheme is inherently suitable for steganography, many studies have utilized it for data hiding and proposed various improvements. In this paper, we further extend the ability of normalizing flow and propose a new network architecture for our robust message embedding task.

3.3. Invertible QR Code Transition

For the message embedding task in this paper, the hidden QR code needs to be restored with enough accuracy to be identified by common devices like cell phones, webcams, etc. To balance the trade-off between the stego image quality and decoding accuracy, VisCode [52] obtains a visual saliency map to guide the QR Code embedding while ChartStamp [10] utilizes the semantic segmentation result as the training loss guidance. Although this kind of rule-based strategy can improve the visual quality of the stego image, it does not consider the inherent relationship between QR Codes and the host image.

In our method, we adopt a more direct approach, which is modifying the QR Code image according to the host image (shown in Figure 2 (a)). We call it invertible QR Code transition (IQRT). The key idea of IQRT is that, the QR Code used for steganography is not necessarily to be

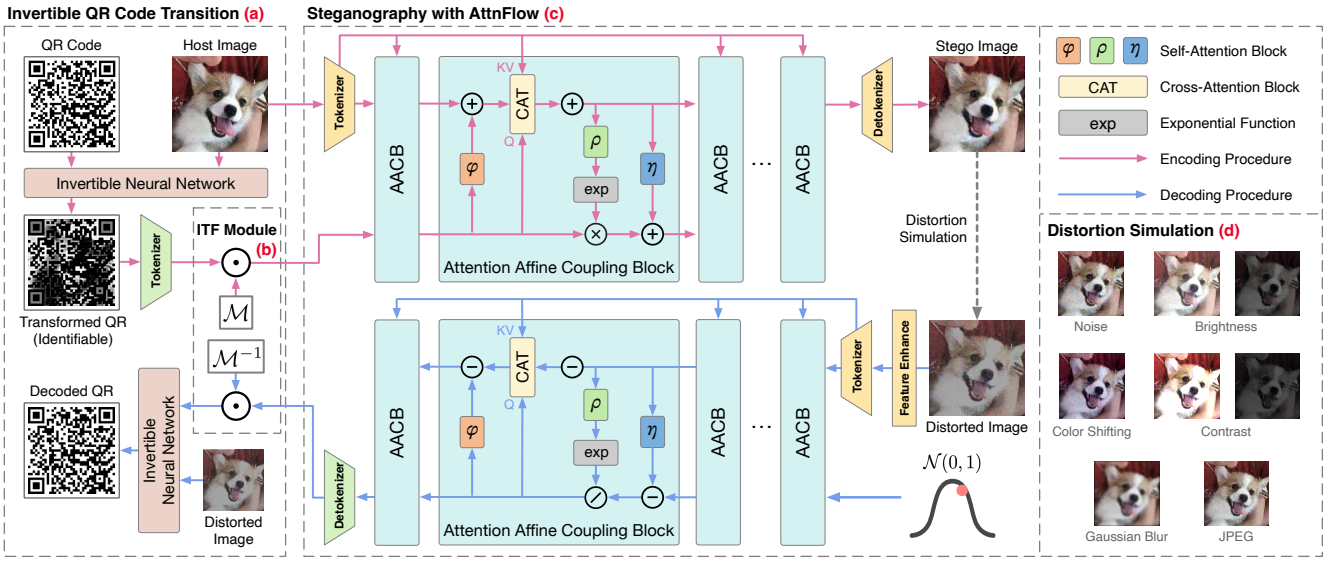

Figure 2. The pipeline of RMSteg. We first transform the QR Code encoded with the secret message to make it easier to hide through an invertible neural network (a). After that, we perform invertible token fusion (ITF) (b) on the tokenized QR Code. We then use a normalizing flow-based model with attention affine coupling blocks (AACBs) to implement data concealing and revealing (c). During training, we employ a distortion simulation module (d) to simulate real-world image disturbances.

black-and-white to keep its information. Thus, a learnable transformation can be applied to the QR Code for a better steganography quality as long as the transformed code is still identifiable. Formally, given a host image I_h and a QR Code I_q with the same size, we use an off-the-shelf INN architecture (with only two invertible blocks) proposed by ISN [24] to learn an invertible function $f(\cdot)$ that derives the transformed QR Code I_q^* by $I_q^* = f(I_q, I_h)$. In the reverse process, the restored QR Code $\hat{I}_q$ can be obtained by $\hat{I}_q = f^{-1}(I_q^*, I_s')$, where $f^{-1}(\cdot)$ is the inverse function of $f(\cdot)$ defined by normalizing flow and I_s' is the distorted stego image. Here I_s' is used instead of I_h since the latter is unknown in the decoding procedure.

During network training, the transition network is jointly trained with the subsequent steganography network. We employ the same constraint as ArtCoder [35] to the transformed QR Code to ensure that it is still identifiable. Specifically, a Gaussian convolution kernel is applied to each code module to simulate the QR Code scanning procedure. For more details, we suggest referring to the original paper [35]. We do not use extra constraint to the transition network so that it can learn the best transition strategy according to the overall optimization targets. Figure 3 shows some transition results, it can be observed that the transformed QR Codes have lower brightness. However, with the aforementioned constraint, the transformed QR Codes are still identifiable, guaranteeing almost no information loss.

3.4. Invertible Token Fusion

With the transformed QR Code, we first use a ViT [8] to obtain a tokenized representation $T_q \in \mathbb{R}^{N \times D}$, in which N is the number of tokens and D represents the token dimensionality. Inspired by the invertible 1×1 convolution proposed by GLOW [18], before feeding T_q to the subsequent steganography network, we put forth an invertible token fusion (ITF) module (as shown in Figure 2 (b)) to transform the QR Code tokens for a better steganography quality.

Figure 3. Some QR Code transition results, the transformed QR Codes are still identifiable.

Formally, we use a learnable matrix $\mathcal{M} \in \mathbb{R}^{N \times N}$, which is initialized as an orthogonal matrix using Cholesky decomposition [19], as a transform matrix for T_q. In the steganography process, T_q is transformed by performing a matrix multiplication: $T_q' = \mathcal{M} \cdot T_q$. Obviously, in the decoding procedure, the restored tokens $\hat{T}_q$ can be obtained by: $\hat{T}_q = \mathcal{M}^{-1} \cdot \hat{T}_q'$, where $\mathcal{M}^{-1}$ is the inverse matrix.

Different from GLOW [18] that utilizes the invertible convolution to learn the channel-wise fusion strategy, our ITF module learns a patch-wise transformation that enables inner-channel feature interaction. Our experiments also prove that ITF can efficiently and effectively improve the steganography quality by simply introducing the aforementioned learnable matrix.

3.5. Steganography with AttnFlow

Previous steganography studies based on normalizing flow generally adopt a convolutional neural network (CNN) based backbone, mostly DenseNet [42], to construct the affine coupling blocks (ACBs). This kind of design only considers the channel-wise feature fusion and can lead to perceptible artifacts in stego images, especially in the robust steganography task. Motivated by the impressive performance achieved by the transformer-based [8, 41] vision models recently, we propose a model called AttnFlow that introduces attention mechanism to normalizing flow to implement robust steganography.

As shown in Figure 2 (c), similar to ordinary normalizing flow, AttnFlow contains several attention affine coupling blocks (AACBs) for invertible function learning. Assume that the input of the i^{th} AACB is split into $T_h^{(i-1)}$ and $T_q^{(i-1)}$, corresponding to the host image tokens and QR Code tokens, respectively. Specifically, $T_h^{(0)}$ is the tokenized host image obtained with a basic ViT [8] and $T_q^{(0)}$ represents the QR Code image tokens output by the ITF module. For the i^{th} AACB, we perform the following affine transformation:

$$\begin{aligned} T_h^{(i)} &= T_h^{(i-1)} + \phi(T_q^{(i-1)}) + \mathcal{C}(T_q^{(i-1)}, T_h^{(0)}) \times \alpha_i, \\ T_q^{(i)} &= \eta(T_h^{(i)}) + T_q^{(i-1)} \odot exp(\rho(T_h^{(i)})), \end{aligned} \tag{3}$$

in which $\phi(\cdot)$, $\eta(\cdot)$, $\rho(\cdot)$ are self-attention blocks [41] followed by a feedforward multilayer perceptron (MLP), $\mathcal{C}(q, kv)$ represents the cross-attention block [41], $exp(\cdot)$ is the exponential function, $\odot$ indicates the Hadamard product and α_i is a dependent trainable coefficient for each AACB. We calculate the attention value with:

$$Attn(Q, K, V) = M \cdot V, \quad M = Softmax(\frac{QK^T}{\sqrt{d}}), \tag{4}$$

where Q, K, V are derived from the learned projections and d is the dimension of the projected tokens. As described in Equation 3, in addition to the self-attention value, we also calculate the cross-attention value of the initial host image tokens $T_h^{(0)}$ upon the QR code tokens $T_q^{(i-1)}$ for each AACB. Then, we add these values to $T_h^{(i-1)}$ to help AACBs gradually integrate the information from the QR code into the image. For the QR Code tokens, we choose to adopt a generally incorporated [5, 16, 24, 48] affine transformation and replace the original convolutional blocks with $\eta(\cdot)$ and $\rho(\cdot)$. After n AACBs, $T_h^{(n)}$ further goes through a detokenizer[1], resulting in the final stego image. Although some methods further map $T_h^{(n)}$ and $T_q^{(n)}$ as a conditional distribution for better performance, here we choose to adopt the same assumption as HiNet [16], which is simply positing that $T_q^{(n)}$ obeys a Gaussian distribution. We follow this scheme since it would not increase calculation overload by including extra networks and we empirically find it would make almost no performance loss in our task.

In the revealing process, we aim to restore the original QR Code from a distorted stego image I_s', we first use a UNet [32] for feature enhancement and tokenize it to derive $\hat{T}_h^{(n)}$. Then, we obtain $\hat{T}_q^{(n)}$ by sampling from a standard Gaussian distribution. After that, we perform the inverse AACB transformation by going through them with inverse calculation manner:

$$\begin{aligned} \hat{T}_q^{(i-1)} &= (\hat{T}_q^{(i)} - \eta(\hat{T}_h^{(i)})) \odot exp(-\rho(\hat{T}_h^{(i)})), \\ \hat{T}_h^{(i-1)} &= \hat{T}_h^{(i)} - \phi(\hat{T}_q^{(i-1)}) - \mathcal{C}(\hat{T}_q^{(i-1)}, \hat{T}_h^{(0)}) \times \alpha_i, \end{aligned} \tag{5}$$

in which $\hat{T}_h^{(0)}$ is obtained by tokenizing I_s' since I_h is unknown in the decoding process. Then, $\hat{T}_q^{(0)}$ is detokenized and fed into the reversed QR Code transition (introduced in Sec. 3.3) with I_s' to get the final decoded QR Code image.

3.6. Optimization Target and Training Strategy

Distortion Simulation Module We use a module to simulate the distortions that the stego images may undergo during printing and photography. In this paper, we choose to use the same simulation module proposed by StegaStamp [37] that considers color shifting, blurring, noising, etc. We mainly modify the standard deviation of Gaussian noise from 0.02 to 0.07 and increase the JPEG compression quality from 25 to 60 for our task. During training, we perform random distortion combinations on stego images to simulate real-world image disturbances.

Loss Function The aforementioned three networks (IQRT, ITF and AttnFlow) are trained jointly. We use the following loss functions to guide the training process:

$$\mathcal{L}_{steg}^{L1} = \|I_h - I_s\|_1, \tag{6}$$

$$\mathcal{L}_{steg}^{ssim} = ssim(I_h, I_s), \tag{7}$$

$$\mathcal{L}_{steg}^{lpips} = lpips(I_h, I_s), \tag{8}$$

$$\mathcal{L}_{qr} = \|I_q - \hat{I}_q\|_1, \tag{9}$$

in which $ssim(\cdot)$ represents the structural similarity index [43] and $lpips(\cdot)$ indicates the perception loss [53]. Besides, as introduced in Sec. 3.3, an additional QR Code transition loss $\mathcal{L}_t$ is incorporated. The overall loss function is the weighted sum of the above functions:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{steg}^{L1} + \beta\mathcal{L}_{steg}^{ssim} + \gamma\mathcal{L}_{steg}^{lpips} + \delta\mathcal{L}_{qr} + \epsilon\mathcal{L}_t, \tag{10}$$

where α, β, γ, δ, ϵ are weight coefficients.

[1]The detailed architecture of the tokenizers and detokenizers are described in the appendix.

Table 1. Steganography quality under different situations. Here σ represents the standard deviation of Gaussian noise (given the image pixel values range in $[0, 1]$). The best and second-best results are marked in red and blue colors, respectively.

Method	Stego Image			$\sigma = 0.1$		$\sigma = 0.15$		JPEG Q = 20		JPEG Q = 40		Mixed		Printing	
	PSNR↑	SSIM↑	LPIPS↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓
ISN†	32.175	0.8765	0.3266	0.728	1.563	0.178	5.020	0.991	0.721	0.999	0.184	0.713	3.131	0.960	1.125
HiNet†	31.629	0.8662	0.3423	0.827	1.077	0.162	3.724	0.986	0.573	0.997	0.099	0.677	3.426	0.970	1.619
StegaStamp	21.215	0.7027	0.3055	0.051	6.152	0.000	10.57	0.951	1.259	0.977	0.798	0.557	3.843	0.750	3.214
StegaStamp†	21.173	0.6903	0.3418	0.481	3.298	0.015	6.500	0.953	1.104	0.969	0.833	0.693	2.975	0.900	1.917
Ours	32.883	0.9109	0.0707	0.794	1.235	0.216	3.306	0.995	0.117	1.000	0.038	0.859	0.861	1.000	0.606

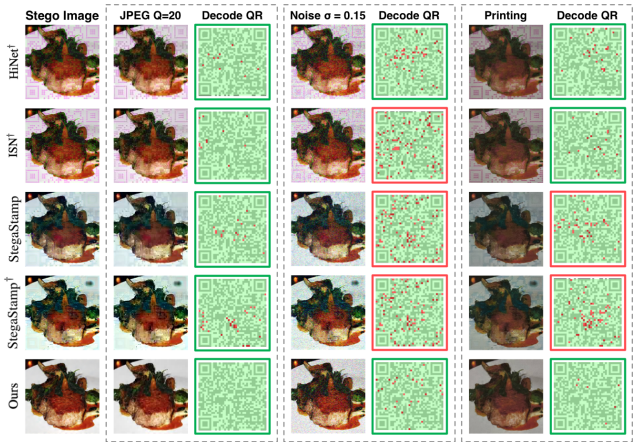

Figure 4. Stego images and decoded QR Codes under different distortions. QR Codes with green borders can be recognized while those with red borders cannot. Zoom in for better observation.

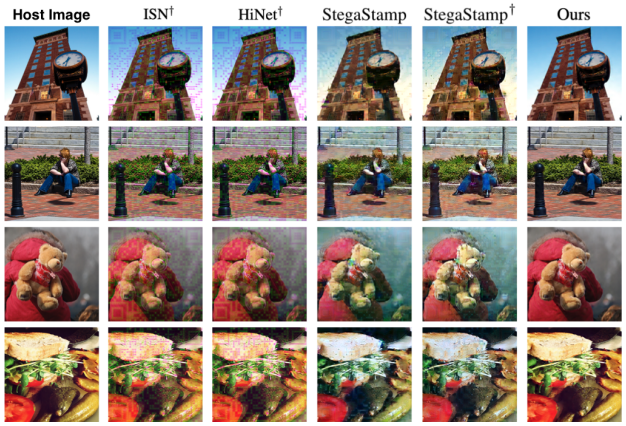

Figure 5. Stego images generated by different methods.

4. Experiment

4.1. Experimental Settings

Datasets Our training and testing datasets of host images are the *train2017* (118K) and *test2017* (41K) datasets of COCO [22], respectively. For QR Code images, we manually construct the training (50K) and testing (41K) datasets with random encoded messages. We generate the QR Code images by adopting the scheme of QR Code version 5 [1] with highest error correction (ECC) level of 'H'. We incorporate this code version for most of our experiments except the evaluation in Sec. 4.3. The image used for training and testing is 224×224 and the patch size of ViT [8] is 16.

Metrics Our experiments focus on two aspects: stego image quality and decoding accuracy. For stego image quality, we use the peak signal-to-noise ratio (PSNR), SSIM [43] and LPIPS [53] to measure the difference between host images and stego images. For decoding accuracy, we adopt the text recovery accuracy (TRA) [48, 52], which is the ratio of the successfully decoded QR Codes. In addition, we calculate the error module rate (EMR), which represents the error rate (in percentage) of the modules in the QR Code.

Baselines We compare our method with some state-of-the-art methods[2], including ISN [24], HiNet [16] and StegaStamp [37]. Since these methods are not designed specifically for our task, we train these models on our datasets for a fair comparison. Moreover, for ISN and HiNet, since they are not robust steganography methods, we incorporate the distortion simulation module when training them. The retrained models of these two methods are illustrated as ISN† and HiNet†, respectively. For StegaStamp, we also additionally train it by using the same distortion level as our method, represented as StegaStamp†.

4.2. Steganography Quality

Steganography quality indicates both stego image quality and decoding accuracy. To compare the robustness of different methods against image distortions, we first consider several manually created situations[3]: Gaussian noise, JPEG compression and random noise combinations. We generate random noise combinations (represented by *Mixed*) with the distortion simulation module introduced in Sec. 3.6. We then consider the printing and photography case to validate the methods' robustness against real-world distortions since it is one of the most extremely severe distortion situations and contains mixed disturbance factors. We randomly select 100 host images and embed random message in them. We then use a inkjet printer to print the encoded images out and take photos with a cell phone. To eliminate the potential errors caused by factors like print quality, we repeat the experiment for 5 times and choose the best result.

Table 1 shows the experiment results, Figure 4 and Figure 5 demonstrate some qualitative results. It can be observed that our method can achieve higher stego image quality, especially for LPIPS that represents the perceptual sim-

[2]Since RIIS [47] and StampOne [33] has not released its source code or pre-trained model, we cannot compare with it.

[3]The experiments under more situations are presented in the appendix.

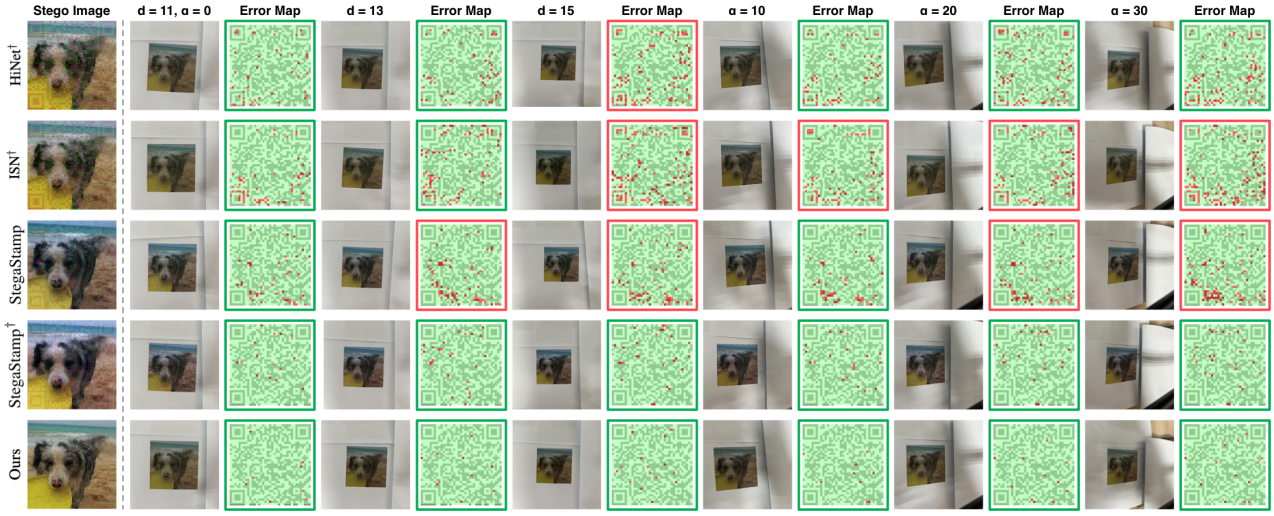

Figure 6. Photos of printed stego images and their decoding errors under different shooting situations. The actual image used for decoding is cropped out and resized from the photo. Photos shown in this figure are intended for an intuitive demonstration of the shooting results. QR Codes with green borders can be recognized while those with red borders cannot. Zoom in for better observation.

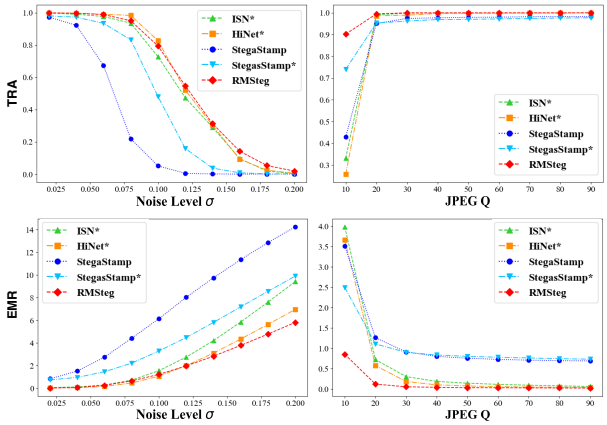

Figure 7. TRA and EMR under different levels of distortions.

ilarity. StegaStamp incorporates adversarial training [11] to make the generated stego image more realistic and it does work when facing low-level distortions. However, with the distortion used during training increases, StegaStamp† fails to preserve a sound visual quality and instead lead to hue shifting and artifacts. In addition, adversarial training can sometimes bring severe artifacts in some regions, as shown in the 3rd and 4th row of Figure 4. For HiNet and ISN that both leverage normalizing flow, due to the lack of inner-channel interaction by using CNN-based affine block, the stego images they derive have obvious QR Code-like artifacts, making the existence of secret message easy to detect. In terms of decoding accuracy, although HiNet† outperforms our method in some cases, we achieve the best performance in the mixed noise and printing tests, which are more close to the real-world application scenarios.

Figure 7 shows the decoding accuracy under more levels of distortions, which are Gaussian Noise whose standard deviation ranges from 0.02 to 0.2 and JPEG compression with quality ranging from 10 to 90. It can be observed that our method demonstrates a stable and good performance under these situations.

In practical application scenarios, the shooting condition may vary from time to time and a good message embedding method should keep its robustness in most cases. As a result, we further measure the decoding accuracy under different shooting situations. We mainly consider the shooting distance and angle (the offset relative to vertical shooting). Given the default printed image side length / shooting distance ratio and angle of this paper are $5.4cm$ / $11cm$ and $0°$, we gradually increase the distance and angle, the test results are shown in Table 2. It can be found that, with the shooting distance and angle grow, the EMRs exhibit a significant increase for all methods. However, for TRA that directly reflects the identifiability of QR Codes, our method maintains a fair performance. The comparison shown in Figure 6 also indicates that RMSteg can achieve high decoding accuracy under different shooting situations.

4.3. Quality under Different Embedding Capacity

To validate the generality of our method, we test it under different embedding capacity, i.e., using QR Codes with different versions for training. We demonstrate the model performance on code versions from *v5* to *v8* in Table 3. Although the steganography quality is getting worse with the embedding capacity increases, the artifacts in the stego images are still not perceptible, especially when the image is printed out, as shown in Figure 8. In addition, RMSteg keeps a TRA of more than 0.8 even for code *v8* whose embedding capacity is two more times higher than *v5*. Compared with StegaStamp [37] that is also designed for robust message embedding, its PSNR is lower than 25 when en-

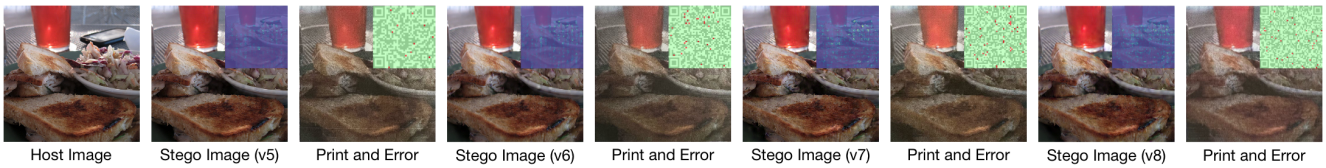

Figure 8. Stego images (the residual is shown in the upper right) and decoding results for printing situation with different code versions.

Table 2. Decoding accuracy under different shooting situations. Here d indicates the shooting distance (measured by *cm*) and α is the shooting angle offset (measured by *degree*). The best and second-best results are marked in red and blue colors, respectively.

Method	$d = 11, \alpha = 0$		$d = 13$		$d = 15$		$\alpha = 10$		$\alpha = 20$		$\alpha = 30$	
	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓	TRA↑	EMR↓
ISN†	0.960	1.125	0.920	2.173	0.790	2.780	0.910	1.859	0.820	2.394	0.590	3.389
HiNet†	0.970	1.619	0.850	3.033	0.680	3.909	0.880	2.396	0.800	3.362	0.660	3.828
StegaStamp	0.750	3.214	0.610	4.131	0.350	5.321	0.540	3.784	0.360	4.258	0.040	5.916
StegaStamp†	0.900	1.917	0.790	2.469	0.710	2.922	0.860	2.272	0.840	2.413	0.570	3.978
Ours	1.000	0.606	1.000	0.891	0.980	1.269	1.000	0.953	1.000	1.163	0.960	1.680

Table 3. Model performance under different QR Code versions. The numbers in parentheses indicate the encoding capacity in *bit*.

Version	Stego Image			Mixed		Printing	
	PSNR↑	SSIM↑	LPIPS↓	TRA↑	EMR↓	TRA↑	EMR↓
v5 (368)	32.883	0.9109	0.0707	0.859	0.861	1.000	0.606
v6 (480)	31.363	0.8903	0.0892	0.859	0.934	1.000	0.877
v7 (528)	31.167	0.8880	0.0902	0.782	1.216	0.890	1.290
v8 (688)	30.765	0.8762	0.1020	0.743	1.370	0.820	1.476

coding 200 bits in a 400×400 image according to the original paper. In contrast, our method is able to keep a PSNR of around 30 when encoding more than 600 bits in a 224×224 image. Thus, RMSteg can achieve a higher steganography quality and meanwhile a much larger embedding capacity.

4.4. Ablation Study

We conduct an ablation study to validate the effectiveness of the IQRT procedure, the ITF module and the AttnFlow model. The results are shown in Table 4.

IQRT The model without IQRT performs slightly better than the full model in decoding accuracy. This is because IQRT may sometimes cause information loss, e.g., some code module could be wrongly transformed during this procedure, although the QR Code is still identifiable. On the other hand, IQRT can largely improve stego image quality.

ITF Module As discussed in Sec. 3.4, the ITF module can learn a transformation for image tokens and thus leading to better stego image quality. We also find that the ITF module can help derive a better distribution of artifacts brought by message embedding. As shown in Figure 9, the stego image generated using ITF has less distortion in homogenous regions (the sky), achieving a better visual quality.

Tokenized Representation We replace the AttnFlow model with ISN and HiNet (two CNN-based normalizing flow model), respectively, to validate the effectiveness of introducing tokenized image representation (the IQRT module is retained). The result shows that, compared with CNN-based scheme, incorporating tokenized image representation makes normalizing flow more competent for robust steganography task.

AACB Number We also train our model with different AACB numbers ranging from 1 to 4. The result shows that, with the increase of the block number, both the stago image quality and the recovery accuracy are gradually improved.

Table 4. Ablation study result. Here *CAT* indicates cross attention, *TR* represents tokenize representation. The best and second-best results are marked in red and blue colors, respectively.

Method	Stego Image			Mixed	
	PSNR↑	SSIM↑	LPIPS↓	TRA↑	EMR↓
w/o IQRT	30.662	0.8651	0.1059	0.871	0.828
w/o ITF	31.422	0.8771	0.0919	0.833	0.947
w/o TR + ISN	32.444	0.8856	0.3209	0.692	3.698
w/o TR + HiNet	31.513	0.8674	0.3370	0.714	3.321
1 AACB	30.426	0.8728	0.1076	0.798	1.135
2 AACBs	31.083	0.8972	0.0831	0.819	0.924
3 AACBs	31.649	0.9008	0.0796	0.819	0.924
Ours Full Model	32.883	0.9109	0.0707	0.856	0.861

Figure 9. The generated stego images using and not using ITF.

5. Conclusion

We propose a robust message embedding framework based on an attention flow-based model, called RMSteg. Our method is capable of generating stego images that can survive various real-world distortions, especially for printing and photography. To our best knowledge, RMSteg is the first method that introduces the transformer-based attention mechanism into normalizing flow. Our experiments show that this scheme is competent in steganography tasks. Compared with existing methods, RMSteg can achieve a better performance in robust and high-quality message embedding. We believe this is to a large extent due to the incorporation of the tokenized image representation and we hope this scheme can inspire subsequent studies.

Acknowledgement This work is supported by the Natural Science Foundation of Shanghai under Grant No. 24ZR1418300 .

References

[1] Information technology — automatic identification and data capture techniques — qr code bar code symbology specification. *ISO/IEC 18004:2015*, 2015. 6

[2] Adel Almohammad, Robert M Hierons, and Gheorghita Ghinea. High capacity steganographic method based upon jpeg. In *Intl. Conf. Availability Reliability Security*, pages 544–549. IEEE, 2008. 1, 2

[3] Shumeet Baluja. Hiding images in plain sight: Deep steganography. *Adv. Neural Inf. Process. Syst. (NIPS)*, 30, 2017. 1, 2

[4] Ricky TQ Chen, Jens Behrmann, David K Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[5] Ka Leong Cheng, Yueqi Xie, and Qifeng Chen. Iicnet: A generic framework for reversible image conversion. In *Proc. IEEE/CVF Intl. Conf. Comput. Vis.*, pages 1991–2000, 2021. 1, 2, 3, 5

[6] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. In *ICLR*, 2014. 2, 3

[7] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *ICLR*, 2016. 2, 3

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 4, 5, 6

[9] Jessica Fridrich, Miroslav Goljan, and Rui Du. Detecting lsb steganography in color, and gray-scale images. *IEEE Multimed.*, 8(4):22–28, 2001. 1, 2

[10] Jiayun Fu, Bin B Zhu, Haidong Zhang, Yayi Zou, Song Ge, Weiwei Cui, Yun Wang, Dongmei Zhang, Xiaojing Ma, and Hai Jin. Chartstamp: Robust chart embedding for real-world applications. In *Proc. 30th ACM Intl. Conf. Multimedia*, pages 2786–2795, 2022. 1, 2, 3

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2, 7

[12] Zhenyu Guan, Junpeng Jing, Xin Deng, Mai Xu, Lai Jiang, Zhou Zhang, and Yipeng Li. Deepmih: Deep invertible network for multiple image hiding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(1):372–390, 2022. 1, 2, 3

[13] Alok Hota and Jian Huang. Embedding meta information into visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3189–3203, 2019. 2

[14] Shoko Imaizumi and Kei Ozawa. Multibit embedding algorithm for steganography of palette-based images. In *Image and Video Technology: 6th Pacific-Rim Symposium, PSIVT 2013, Guanajuato, Mexico, October 28-November 1, 2013. Proceedings 6*, pages 99–110, 2014. 1, 2

[15] Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-revnet: Deep invertible networks. *arXiv preprint arXiv:1802.07088*, 2018. 3

[16] Junpeng Jing, Xin Deng, Mai Xu, Jianyi Wang, and Zhenyu Guan. Hinet: Deep image hiding by invertible network. In *Proc. IEEE/CVF Intl. Conf. Comput. Vis.*, pages 4733–4742, 2021. 1, 2, 3, 5, 6

[17] Eiji Kawaguchi and Richard O Eason. Principles and applications of bpcs steganography. In *Multimedia Syst. Appl.*, pages 464–473. SPIE, 1999. 1, 2

[18] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018. 2, 3, 4

[19] Aravindh Krishnamoorthy and Deepak Menon. Matrix inversion using cholesky decomposition. In *2013 signal processing: Algorithms, architectures, arrangements, and applications (SPA)*, pages 70–72. IEEE, 2013. 4

[20] Bin Li, Ming Wang, Jiwu Huang, and Xiaolong Li. A new cost function for spatial image steganography. In *ICIP*, 2014. 2

[21] Yung-Hui Li, Ching-Chun Chang, Guo-Dong Su, Kai-Lin Yang, Muhammad Saqlain Aslam, and Yanjun Liu. Coverless image steganography using morphed face recognition based on convolutional neural network. *EURASIP Journal on Wireless Communications and Networking*, 2022(1):1–21, 2022. 2

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 6

[23] Qiang Liu, Xuyu Xiang, Jiaohua Qin, Yun Tan, and Yao Qiu. Coverless image steganography based on densenet feature mapping. *EURASIP Journal on Image and Video Processing*, 2020:1–18, 2020. 2

[24] Shao-Ping Lu, Rong Wang, Tao Zhong, and Paul L Rosin. Large-capacity image steganography based on invertible neural networks. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 10816–10825, 2021. 1, 2, 3, 4, 5, 6

[25] Jarno Mielikainen. Lsb matching revisited. *IEEE Signal Process. Lett.*, 13(5):285–287, 2006. 1, 2

[26] Mohammed Saad Mohamed, EH Hafez, et al. Coverless image steganography based on jigsaw puzzle image generation. *Computers, Materials and Continua*, 67(2):2077–2091, 2021. 2

[27] Chong Mou, Youmin Xu, Jiechong Song, Chen Zhao, Bernard Ghanem, and Jian Zhang. Large-capacity and flexible video steganography via invertible neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22606–22615, 2023. 3

[28] Bui Cong Nguyen, Sang Moon Yoon, and Heung-Kyu Lee. Multi bit plane image steganography. In *Digital Watermarking: 5th International Workshop, IWDW 2006, Jeju Island, Korea, November 8-10, 2006. Proceedings 5*, pages 61–70, 2006. 1, 2

[29] Michiharu Niimi, Hideki Noda, Eiji Kawaguchi, and Richard O Eason. High capacity and secure digital steganography to palette-based images. In *Proceedings. International conference on image processing*, pages II–II. IEEE, 2002. 1, 2

[30] Tomáš Pevnỳ, Tomáš Filler, and Patrick Bas. Using high-dimensional image models to perform highly undetectable steganography. In *Information Hiding, IH*, pages 161–177. Springer, 2010. 2

[31] Jiaohua Qin, Jing Wang, Yun Tan, Huajun Huang, Xuyu Xiang, and Zhibin He. Coverless image steganography based on generative adversarial network. *Math.*, 8(9):1394, 2020. 2

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 5

[33] Farhad Shadmand, Iurii Medvedev, Luiz Schirmer, João Marcos, and Nuno Gonçalves. Stampone: Addressing frequency balance in printer-proof steganography. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4367–4376, 2024. 6

[34] Haichao Shi, Jing Dong, Wei Wang, Yinlong Qian, and Xiaoyu Zhang. Ssgan: secure steganography based on generative adversarial networks. In *Pacific Rim Conference on Multimedia*, 2017. 2

[35] Hao Su, Jianwei Niu, Xuefeng Liu, Qingfeng Li, Ji Wan, Mingliang Xu, and Tao Ren. Artcoder: An end-to-end method for generating scanning-robust stylized qr codes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2277–2286, 2021. 4

[36] Mitchell D Swanson, Bin Zhu, Benson Chau, and Ahmed H Tewfik. Multiresolution video watermarking using perceptual models and scene segmentation. In *Proc. Intl. Conf. Image Process.*, pages 558–561. IEEE, 1997. 1, 2

[37] Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 2117–2126, 2020. 1, 2, 5, 6, 7

[38] Weixuan Tang, Shunquan Tan, Bin Li, and Jiwu Huang. Automatic steganographic distortion learning using a generative adversarial network. *IEEE Signal Processing Letters*, 2017. 2

[39] Weixuan Tang, Bin Li, Shunquan Tan, Mauro Barni, and Jiwu Huang. Cnn-based adversarial embedding for image steganography. *TIFS*, 2019. 2

[40] Tycho FA van der Ouderaa and Daniel E Worrall. Reversible gans for memory-efficient image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4720–4728, 2019. 3

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 5

[42] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, pages 0–0, 2018. 5

[43] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4): 600–612, 2004. 5, 6

[44] Eric Wengrowski and Kristin Dana. Light field messaging with deep photographic steganography. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 1515–1524, 2019. 1, 2

[45] Yue Wu, Guotao Meng, and Qifeng Chen. Embedding novel views in a single jpeg image. In *Proc. IEEE/CVF Intl. Conf. Comput. Vis.*, pages 14519–14527, 2021. 2

[46] Mingqing Xiao, Shuxin Zheng, Chang Liu, Yaolong Wang, Di He, Guolin Ke, Jiang Bian, Zhouchen Lin, and Tie-Yan Liu. Invertible image rescaling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 126–144. Springer, 2020. 3

[47] Youmin Xu, Chong Mou, Yujie Hu, Jingfen Xie, and Jian Zhang. Robust invertible image steganography. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7875–7884, 2022. 1, 2, 3, 6

[48] Huayuan Ye, Chenhui Li, Yang Li, and Changbo Wang. Invvis: Large-scale data embedding for invertible visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 1, 2, 3, 5, 6

[49] Jiwen Yu, Xuanyu Zhang, Youmin Xu, and Jian Zhang. Cross: Diffusion model makes controllable, robust and secure image steganography. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[50] Xiaoyi Yu, Tieniu Tan, and Yunhong Wang. Reliable detection of bpcs-steganography in natural images. In *Intl. Conf. Image Graphics (ICIG)*, pages 333–336. IEEE, 2004. 1, 2

[51] Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. Steganogan: High capacity image steganography with gans. *arXiv:1901.03892*, 2019. 2

[52] Peiying Zhang, Chenhui Li, and Changbo Wang. Viscode: Embedding information in visualization images using encoder-decoder network. *IEEE Trans. Visual. Comput. Graph.*, 27(2):326–336, 2020. 1, 2, 3, 6

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 586–595, 2018. 5, 6

[54] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proc. Eur. Conf. Comput. Vis. (ECCV)*, pages 657–672, 2018. 1, 2

[55] Wenwu Zhu, Zixiang Xiong, and Ya-Qin Zhang. Multiresolution watermarking for images and video. *IEEE Trans. Circ. Syst. Video Technol.*, 9(4):545–550, 1999. 1, 2

[56] Xiaobin Zhu, Zhuangzi Li, Xiao-Yu Zhang, Changsheng Li, Yaqi Liu, and Ziyu Xue. Residual invertible spatio-temporal network for video super-resolution. In *Proceedings of the AAAI conference on artificial intelligence*, pages 5981–5988, 2019. 3