This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

JPEG Robust Invertible Grayscale

Kunlin Liu*, Dongdong Chen*, Jing Liao, Weiming Zhang, Hang Zhou, Jie Zhang, Wenbo Zhou, and Nenghai Yu

***Abstract*—Invertible grayscale is a special kind of grayscale from which the original color can be recovered. Given an input color image, this seminal work tries to hide the color information into its grayscale counterpart while making it hard to recognize any anomalies. This powerful functionality is enabled by training a hiding sub-network and restoring sub-network in an end-to-end way. Despite its expressive results, two key limitations exist: 1) The restored color image often suffers from some noticeable visual artifacts in the smooth regions. 2) It is very sensitive to JPEG compression, i.e., the original color information cannot be well recovered once the intermediate grayscale image is compressed by JPEG. To overcome these two limitations, this paper introduces adversarial training and JPEG simulator respectively. Specifically, two auxiliary adversarial networks are incorporated to make the intermediate grayscale images and final restored color images indistinguishable from normal grayscale and color images. And the JPEG simulator is utilized to simulate real JPEG compression during the online training so that the hiding and restoring sub-networks can automatically learn to be JPEG robust. Extensive experiments demonstrate that the proposed method is superior to the original invertible grayscale work both qualitatively and quantitatively while ensuring the JPEG robustness. We further show that the proposed framework can be applied under different types of grayscale constraints and achieve excellent results.**

***Index Terms*—Invertible Grayscale, Adversarial Training, JPEG Robust**

I. Introduction

NOwadays the vast majority of images are shot as color images. However, in some real user scenarios such as black-and-white digital printing, photography rendering, and abstract stylization, they need to be converted to the grayscale counterparts instead. Essentially, converting a color image to a grayscale image is a dimensionality reduction problem. In the past decades, a lot of different methods have been proposed for color-to-grayscale conversion. Among them, the most commonly used techniques are based on simple weighted averages of the red, green and blue channels, such as extracting the lightness channel in the CIELab color space [1]. There also exist some other advanced methods designed by considering different perceptual factors or constraints, like contrast preserving [2], [3] and saliency preserving [4]. However, all conversion procedures proposed in these methods are irreversible. In other words, it is difficult to get the original color information back from the converted grayscale image.

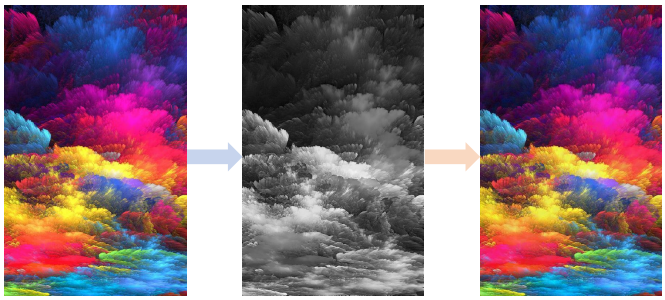


Fig. 1: The illustration of invertible grayscale. Given an input color image, this special type of grayscale image can be inverted back to a similar color image.

A lot of methods[5], [6], [7], [8], [9] have been proposed for reversible color-to-gray conversion. However, most of them are not robust to regular distortion. Recently, the pioneering work [10] proposes an innovative CNN-based color-to-grayscale conversion method named "Invertible GrayScale" (IG). As shown in Figure 1, different from previous methods, it aims to propose an invertible grayscale image that can fully restore its original color and ensure the users cannot recognize any anomalies in it. To achieve this goal, they leverage a deep convolutional neural network (CNN) to learn this conversion process rather than using pre-defined or handcrafted rules. Thanks to the strong ability of CNN, invertible grayscale images are so robust that they can revert themselves back to the color version with Gaussian noise. The whole system consists of two parts: one hiding sub-network to convert a color image to grayscale, and one restoring sub-network to invert the grayscale back to the color image correspondingly. These two sub-networks are jointly trained in an end-to-end way so that the restored color image should be similar to the original color image as much as possible.

Notwithstanding its demonstrated expressive results, we find two key limitations still exist in [10], which are shown in Figure 2. The first one is that some noticeable visual artifacts often appear in the smooth regions of the grayscale and restored color images. This is because, compared to texture regions, it is more difficult to hide color information in smooth regions while ensuring unnoticeable. Thus some explicit supervision should be provided to instruct the system to put more effort into smooth regions to generate visually

This work was supported in part by the National Natural Science Foundation of China under Grant 62002334U20B2047 and 62072421, by Anhui Science Foundation of China under Grant 2008085QF296, by Anhui Initiative in Quantum Information Technologies under Grant AHY15040and by Fundamental Research Funds for Central Universities of China under Grant WK2100000018.

Kunlin Liu, Weiming Zhang, Hang Zhou, Jie Zhang, Wenbo Zhou and Nenghai Yu are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, Anhui 230026, China. E-mail: lkl6949@mail.ustc.edu.cn, zhangwm@ustc.edu.cn, zh2991@mail.ustc.edu.cn, zjzac@mail.ustc.edu.cn, welbeckz@mail.ustc.edu.cn, ynh@ustc.edu.cn

Dongdong Chen is with Microsoft Research, Redmond, Washington 98052, USA. Email: cddlyf@gmail.com

Jing Liao is with the Department of Computer Science, City University of Hong Kong. Email: jingliao@cityu.edu.hk

Kunlin Liu and Dongdong Chen are co-first authors.

Corresponding author: Weiming Zhang.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

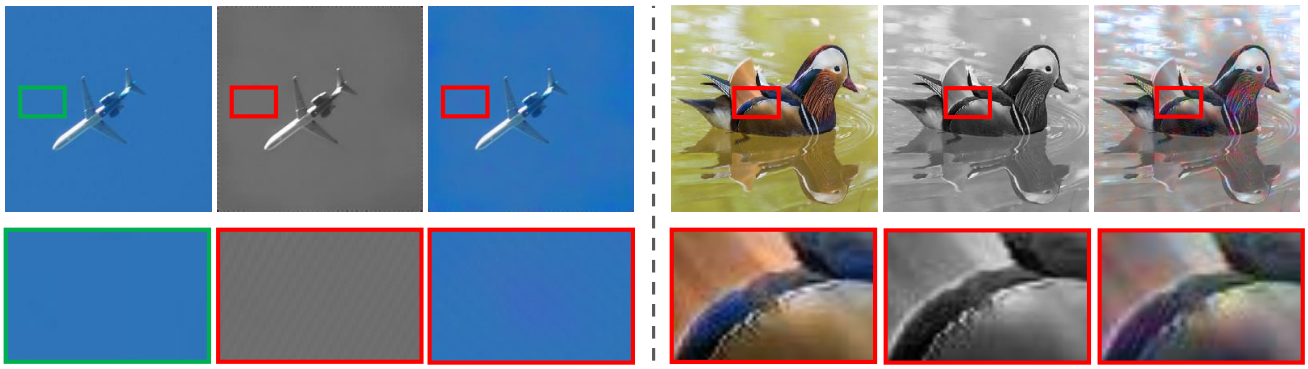


Fig. 2: Example results to illustrate the two key limitations in the baseline invertible grayscale method [10]. The left part is to show the artifacts in smooth regions while the right is to show the sensitivity to the JPEG compression of intermediate grayscale images.

pleasant results. However, the system proposed in [10] just treats texture regions and smooth regions in the same way.

The second limitation of [10] is that the grayscale images converted by [10] are very sensitive to JPEG compression, i.e., the hidden color information will be damaged and cannot be well recovered anymore after JPEG compression. However, the JPEG image format is widely used in real application scenarios, and users are likely to save the converted grayscale images in the JPEG format rather than lossless PNG format to save space. The main reason why the method [10] is not robust to JPEG compression is that the network does not see any compressed grayscale images during training by default. Therefore, in order to achieve JPEG robustness, JPEG compression should be incorporated into the training process. This is a non-trivial task because the real JPEG compression process is conducted in the DCT domain and uses some discrete sampling strategies, thus making it impossible to be differentiable.

Motivated by these two limitations, this paper proposes a JPEG robust invertible grayscale method. Following a similar framework as [10], the overall system still consists of one hiding sub-network and one restoring sub-network, but adversarial training and one JPEG simulator are newly introduced. Specifically, we first incorporate two auxiliary adversarial discriminator networks after these two sub-networks. Rather than explicitly telling the network where are the smooth regions containing unpleasant artifacts, we believe these two discriminators can easily identify the artifacts regions and help the system to produce visually pleasant grayscale and restored color images automatically.

To obtain JPEG robustness, though it is hard to use the real JPEG compression process directly in the end-to-end training, we insert one differentiable JPEG simulator layer between the hiding sub-network and restoring sub-network to simulate the compression process. This simulation is based on two observations: 1) The non-differentiability of JPEG compression comes from the intermediate quantization step for the frequency-domain coefficients. 2) Quantizing the coefficient is equivalent to limit the amount of information passed through specific frequency channels. Therefore in this simulation layer, we use a fixing mask to constrain that only low-frequency DCT coefficients can be passed. Thanks to the differentiability of this operation, this simulator can guide the learning of the hiding and restoring sub-networks by backpropagation to make them JPEG robust.

To train our system, a step-wise training strategy is adopted. Specifically, we first train a basic model of invertible grayscale following [10], then incorporate the aforementioned adversarial discriminator networks into the training and get a better model without visual artifacts. Finally, the JPEG simulator is utilized to achieve JPEG robustness. To demonstrate our superiority, extensive experiments have been conducted and show that our method can achieve better-restored color images for grayscale images with or without JPEG compression at the same time.

We further extend the proposed framework to other special types of intermediate grayscale images, i.e., simple edge maps and halftone images. Even though in this challenging case where rich texture and color information need to be hidden, experiments demonstrate the hiding sub-network and restoring sub-network can still collaborate well and produce visually pleasant results.

To summarize, our contributions are four-fold:

- We propose a JPEG robust invertible grayscale method and achieve much better results than our baseline [10].
- To avoid visual artifacts in smooth regions shown in [10], we incorporate two auxiliary adversarial discriminator networks to instruct the system to achieve more visually pleasing results.
- We leverage one JPEG simulator between the hiding sub-network and restoring sub-network during the training stage, and make our method more robust to real JPEG compression.
- Extensive experiments and analyses have been conducted. They not only demonstrate the superiority of our method but show powerful generalization ability of the proposed framework, which may inspire more innovative works in this field.

The rest of this paper is organized as follows. We review the related work in Section II, and the detailed technique

parts are elaborated in Section III. Then the training strategy is presented in Section V. In Section VI, comprehensive experiments and analysis are provided. Finally, the conclusion and future work are given in Section VII.

II. Related Work

A. Image Steganography

Formally, steganography is the process of concealing some types of messages (e.g., text, image, or video) within some types of covers (e.g., file, message, image, or video) in a way that the hidden message can be extracted after. Since invertible grayscale [10] is just to hide color information into the grayscale image, so it can be regarded as a special application of image steganography. In the past, a wide variety of steganography methods [11], [12], [13], [14] have been proposed in the literature. Most relevant to our work are methods for blind image steganography [15], [16], [17], where the message is encoded in an image and the decoder does not have access to the original cover image. As for visual quality influence, since Least-Significant Bit (LSB) based methods [18], [19] only modify the lowest-order bits of each image pixel depending on the bits of the secret message, it is very difficult to find the visual appearance change of the stego image which has embedded the target message. Very recently, rather than use pre-define low-order bits, Zhu *et al.* [20] uses a deep network to hide messages instead. However, they both have very limited hiding capacity, when hiding too much information, these methods will modify the picture a lot and cause many artifacts. For our problem, because 16 bits of color information should be hidden in each image pixel, these methods will totally fail. Even worse, to ensure perfectly recovering the hidden bits, many extra error-correction bits should also be included in these methods, which further increases the burden of the hiding system. Another difficulty of our task is that we need to recover high-fidelity cover images (grayscale structure) and messages (color information) at the same time.

B. Adversarial Networks

In the pioneering work [21], Goodfellow et al. propose the first Generative Adversarial Network (GAN) framework to generate realistic-looking images from random noise via adversarial learning. It is a generative deep model that pits two networks against one another. During training, the generator network **G** is trained to fool the discriminator network **D** which in turn tries to distinguish between the generated samples from **G** and the real samples. The key ingredient of this work is its proposed adversarial loss, which demonstrates its superpower in helping achieve visual realism in many image translation works [22], [23], [24], [25], [26].

Similarly, in this work, we try to leverage two patch-level discriminator networks to distinguish the generated grayscale images and the restored color images from the real grayscale/color images respectively, while the hiding sub-network and restoring sub-network try to generate realistic visually pleasant grayscale/restored color images to fool the discriminators. By jointly training them together, the gradient of discriminators can backpropagate to the two sub-networks and instruct them to produce much more visually pleasing results.

C. Color-to-grayscale Conversion

With the fast development of digital photography, most images captured by modern devices are color images. But considering the compatibility, cost or aesthetic issues, grayscale images are still widely used in some application scenarios. Color-to-grayscale conversion is a very classical research problem and has been extensively studied in the past decades. Naturally, it is a type of dimension reduction problem and often suffers from information loss. Common naive methods include extracting the lightness/luminance channel in the CIELab/YUV color space. However, they would diminish salient chromatic structures and lose important appearance features/contrast.

To better preserve the color contrast, many advanced color-to-grayscale methods have been proposed, which can be categorized into global and local methods respectively. In global methods, Gooch *et al.* [4] use chrominance and luminance differences to create grayscale target differences between nearby image pixels, then solve an optimization problem to get the final grayscale representation. Kuk *et al.* extends the idea of [4] by considering both the global and local contrasts. Rasche *et al.* [27] constructs a linear mapping from R^3 space to R^1 space that keeps the perceived distances between points in R^3 and that in R^1 as much as possible.

In local methods, different pixels are often processed differently and usually rely on local chrominance edges for enhancement. For example, Bala *et al.* [28] introduce high-frequency chrominance information into the luminance channel to preserve the distinction between adjacent colors. Neumann *et al.* [29] regard the color and luminance contrast as a gradient field and obtain the grayscale image via fast direct integration in that field. Smith *et al.* [30] use a two-step approach to first globally assign gray values and determine color ordering then locally enhance the grayscale to reproduce the original contrast. Different from these traditional color-to-grayscale conversion methods, we want the grayscale image to keep the color information and can roll back to the original color image with a decoder.

D. Image Colorization

Colorization aims to add meaningful and visually appealing colors to a grayscale image. Without explicit guidance, this is a highly ill-pose and multimodal problem. Previous colorization methods can be roughly categorized into example-based methods and learning-based methods. For the former type, user scribbles [31], [32], [33] or reference color images [34], [35], [36], [37] are provided as extra hints. These algorithms often first set the colors of some sparse seed pixels based on user scribbles or some correspondence matching methods, then propagate these colors to other unspecified pixels with the Markov Random Field model. The main drawbacks of such methods are intensive manual work or high reliance on a good reference.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

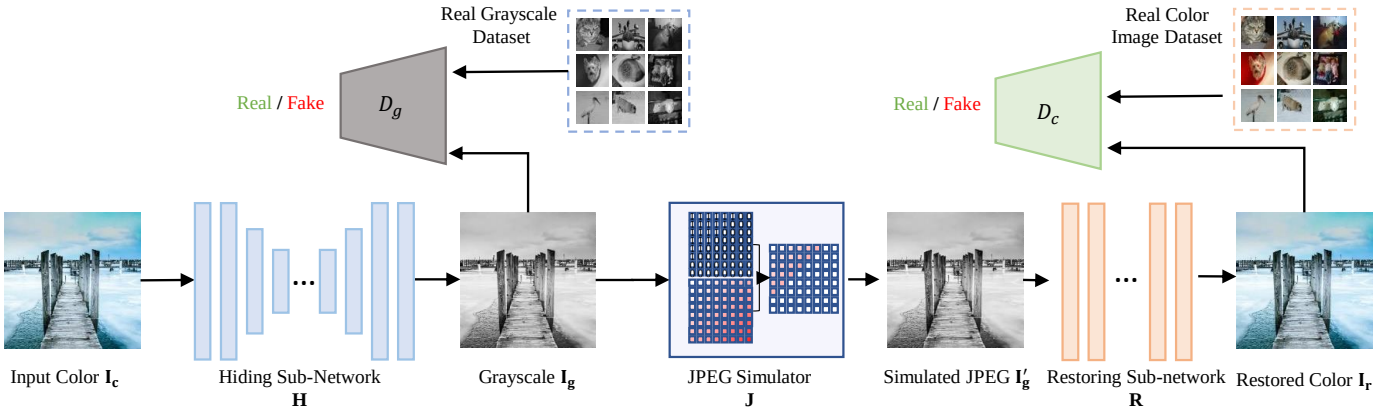


Fig. 3: The pipeline of our system. It consists of one hiding sub-network $\mathbf{H}$, one restoring sub-network $\mathbf{R}$, one JPEG simulator $\mathbf{J}$, and two discriminator networks $\mathbf{D_g}, \mathbf{D_c}$. $\mathbf{H}$ and $\mathbf{R}$ learn how to hide and restore color information respectively, while $\mathbf{D_g}, \mathbf{D_c}$ are used to judge whether the generated grayscale/color images are real enough and guide the learning of $\mathbf{H}, \mathbf{R}$. $\mathbf{J}$ simulate the JPEG compression process and make $\mathbf{H}, \mathbf{R}$ more JPEG robust.

Thanks to the great success in deep learning, many learning-based methods [38], [39], [40], [41] have been proposed recently. By leveraging large-scale datasets like ImageNet, these methods can learn how to colorize a grayscale image automatically. They either model this problem as a regression problem [39], [40] or a classification problem [38], [41]. The biggest advantage of such methods is no need for any extra user guidance, but their colorization results are often very conservative. Recently, some hybrid colorization methods [42], [43], [44], [45], [46], [47], [48], [49] are designed by combining the merits of the above example-based and learning-based methods. Though our restoring sub-network can invert the grayscale images back to the original color images, it is based on the color information pre-hidden in specially designed grayscale images from the hiding sub-network. Therefore, it does not suffer from the ambiguity problem in the classical colorization algorithms.

E. Reversible Color-to-gray Conversion

Reversible color-to-grayscale conversion aims at embedding the chromatic information of a full-color image into its grayscale version such that the original color image can be reconstructed in the future when necessary. There are many methods for reversible color-to-gray conversion. Conventional algorithms mainly focus on the quality of the reconstructed color image, which makes the intermediate grayscale image visually undesirable and suspicious[8]. To obtain stronger reversibility, many advanced methods[5], [9], [7], [6], [50] focused on designing specific encoding methods. Different from these methods, our method aims to ensure the great visual quality of intermediate grayscale images and strong robustness simultaneously.

III. JPEG Robust Invertible Grayscale

Given an input color image I_c, the goal of the baseline invertible grayscale method [10] is to generate a special type of grayscale image I_g that can be further inverted into a color image I_r. Ideally, I_r should be identical to I_c without any color information loss. To achieve this goal, one hiding sub-network $\mathbf{H}$ is used to hide the original color information of I_c in I_g, and another restoring sub-network $\mathbf{R}$ produces the final color image I_r only based on I_g, formally:

$$\begin{aligned} I_g &= \mathbf{H}(I_c) \\ I_r &= \mathbf{R}(I_g) = \mathbf{R}(\mathbf{H}(I_c)) \end{aligned} \tag{1}$$

Due to the inherent difficulty in hiding two-channel color information in a single-channel grayscale image, we observe that the intermediate grayscale images and restored color images from the baseline method [10] often contain some unpleasant artifacts, especially in smooth regions as shown in the left part of Figure 2. Intuitively, it is more difficult to hide information in smoothing regions than texture regions while ensuring unnoticeable, which is also common sense in traditional image steganography methods. To address this problem, we want to give some guidance to the hiding sub-network and restoring sub-network and let them learn better hiding/extracting strategies. However, designing explicit guidance rules by hand is not easy and may incur some bias. Motivated by the success of adversarial learning, we introduce two adversarial discriminator networks after the hiding sub-network and restoring sub-network respectively. By showing a large scale of real grayscale images and color images, we expect these discriminators can guide the learning of target sub-networks explicitly by gradient back-propagation.

In the original paper of [10], another limitation noted by Xia *et al.* is that their system is very sensitive to JPEG compression. In other words, if the intermediate grayscale image I_g is compressed by JPEG, the original invertibility will be destroyed and result in an awful restored color image I_r as shown in the right part of Figure 2. The underlying reason is that the baseline method [10] has not considered the JPEG compression process during training. Therefore, the very natural idea is to incorporate JPEG compression into the training. However, the indifferentiable sampling operation in real JPEG makes it a non-trivial task. To alleviate this problem, we resort to introducing one differentiable JPEG simulator

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

between the hiding sub-network and the restoring sub-network instead.

Combining the above two points, our whole system is shown in Figure 3. Specifically, it also consists of one hiding sub-network $\mathbf{H}$ and one restoring sub-network $\mathbf{R}$ following [10], but introduces two auxiliary adversarial discriminators $\mathbf{D_g}, \mathbf{D_c}$ and one JPEG simulator $\mathbf{J}$. Below we elaborate on the details of these parts, corresponding loss functions, and training strategy.

A. Network Structures

Hiding Sub-network H. In this task, we can regard our hiding sub-network as a special kind of encoder, which encodes the color information of the input image I_c into its corresponding grayscale image I_g that is invertible. Since this grayscale is the final product for visualization, we also require the encoded grayscale to be close to a general type of grayscale image that conforms to the input. We inherit the auto-encoder like the architecture of IG [10]. In the encoder part, one convolution layer first encodes the input image I_c to a feature map, then two enhancing residual blocks are used before feeding it into the following two down-sample blocks. Each down-sample block consists of two consecutive convolutional layers with stride 2 and 1 respectively. Given the downsampled feature maps, another four enhancing residual blocks are further used. Symmetrically in the decoder part, two upsample blocks first upsample the downsampled feature maps back to the original resolution. Each upsample block consists of one nearest neighbor upsampling layer and two convolutional layers. Then, the upsampled feature maps are also enhanced by two residual blocks. Finally, one simple convolutional layer is used to predict the final single-channel grayscale image I_g. Note that 3×3 kernel size is adopted in all the convolutional layers unless especially specified.

Restoring Sub-network R. Compared to hiding sub-network, the restoring sub-network $\mathbf{R}$ can be regarded as a color decoder that extracts the information hidden in the invertible grayscale. For the detailed network structure, it simply consists of one convolutional layer, eight residual blocks, and two convolutional layers. To constrain the value range of the output image, one tanh layer is added. Though $\mathbf{R}$ does not involve any downsample or upsample operation, we empirically find it is strong enough for information extraction.

Discriminator Network $\mathbf{D_g}, \mathbf{D_c}$. $\mathbf{D_g}, \mathbf{D_c}$ are introduced to guide the learning of $\mathbf{H}, \mathbf{R}$ as auxiliary networks which are only used during the training stage. Currently, there are two different types of discriminator networks that are widely used: global-based or patch-based. Given an input image I, the global-based discriminator will predict only one label that indicates I is real or fake, while the patch-based discriminator will predict the labels of all the fix-sized sliding patches in I to indicates each of them is real or fake.

In our motivation, we want the discriminator network to instruct $\mathbf{H}, \mathbf{R}$ to differentiate smooth regions and texture regions and put more effort into avoiding unpleasant artifacts in smooth regions. Patch-based discriminators (PatchGAN) are adopted for $\mathbf{D_g}, \mathbf{D_c}$ respectively. Another advantage of PatchGAN is that it can be applied to arbitrarily large images rather than fix-sized images in global-based discriminators. The detailed network structure of PatchGAN is very simple. It just consists of one convolutional layer at the head, several downsample convolutional layers with stride 2 to enlarge the receptive field of each patch, and two convolutional layers to get the final predictions for each patch.

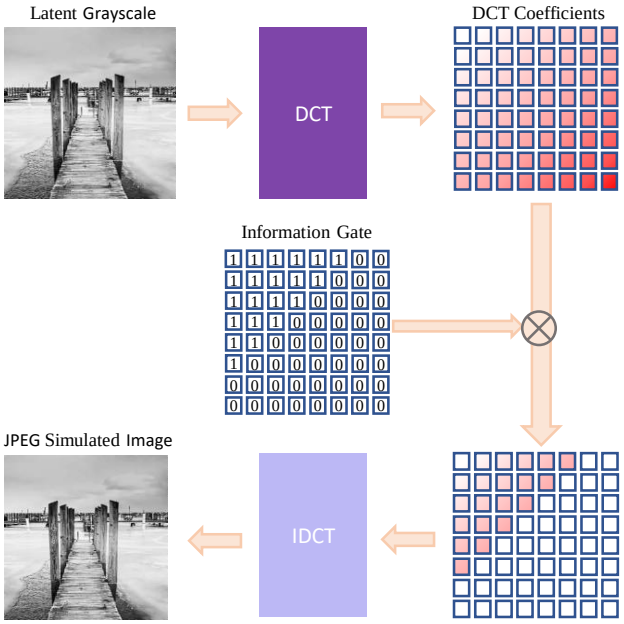


Fig. 4: The working principle of the JPEG simulator. To simulate the indifferentiable quantization process in real JPEG compression, an information gate mask is used to mask out the top-k high-frequency components in 8×8 DCT coefficients block where k is randomly picked from 32 to 64.

JPEG Simulator J. Since we want the synthesized invertible grayscale to be robust to JPEG compression and the deep network often highly relies on explicit supervised training, the JPEG compression process should be involved in the training process. Motivated by [20], we leverage a mask-based JPEG simulator to simulate the real JPEG compression algorithm. To better understand the motivation of the JPEG simulator $\mathbf{J}$, we first introduce the working principle of JPEG compression briefly. Specifically, it will divide the image into 8×8 regions, then computes a discrete cosine transformation (DCT) for each region, and the computed DCT coefficients represent different frequency components. Finally, these frequency-domain coefficients will be quantized for the latter Huffman-coding. Considering the whole process, its inherent non-differentiability is because of the intermediate quantization step, thus make gradient-based end-to-end learning impossible.

To alleviate this problem, the above quantization step is replaced by one JPEG gate mask which controls how many coefficients should be remained, because it has the same capability of limiting the amount of information flow theoretically. In the implementation, a stride 8 convolutional layer with kernel size 8×8 is used as the DCT transformation, and

each filter learns a DCT basis vector. Correspondingly, the output activations of this layer are just the computed DCT coefficients, which further pass the above JPEG gate mask. Finally, the masked activations are fed into another transpose convolutional layer which acts as inverse DCT and gets the JPEG simulated image. During training, the gate mask only keeps top-k low-frequency DCT coefficients. Compared to the implementation of [20] that adopts a fixed k, we randomly pick k from range $[32, 64]$ to make the learning robust to different compression levels.

B. Loss Functions

To train our network, the overall loss functions can be roughly categorized into four parts: invertible losses $\mathcal{L}^c_{inv}$ for final restored color images, conformity losses $\mathcal{L}^g_{con}$ for intermediate grayscale images, quantization loss ℓ_q, adversarial losses $\mathcal{L}_d$ for both grayscales and restored color images. Among them, grayscale conformity losses and quantization loss are inherited from the baseline method [10].

$$\mathcal{L} = \mathcal{L}^c_{inv} + \mathcal{L}^g_{con} + \ell_q + \mathcal{L}_d \tag{2}$$

1) Invertible Losses

To ensure the final restored color image I_r be identical to the original input color image I_c as much as possible, two kinds of loss functions are adopted: one basic reconstruction loss ℓ^c_r and one structure-preserving loss ℓ^c_s.

$$\mathcal{L}^c_{inv} = \lambda_1 * \ell^c_r + \lambda_2 * \ell^c_s \tag{3}$$

Reconstruction Loss. The basic reconstruction loss simply adopts the pixel level Mean Square Error to constrain the similarity, i.e.,

$$\ell^c_r = \frac{1}{N} * \|I_c - I_r\|^2 \tag{4}$$

Here N is the total pixel number of I_c. This loss can guarantee the hiding sub-network $\mathbf{H}$ and the restoring sub-network $\mathbf{R}$ can collaborate well to make I_r overall similar to I_c.

Structure Preserving Loss. As shown in Figure 2, some artifacts often appear in smooth regions of the final restored color image, thus incur some structure distortion compared to the original color image. To avoid such a structure distortion problem, we use the traditional structure similarity index loss (SSIM loss), which is more consistent with the human perception than the above absolute difference measurements. To put it formally, give a pair of images x, y, three different aspects of similarities are included according to human perception: luminance similarity $l(x, y)$, contrast similarity $c(x, y)$ and structure similarity $s(x, y)$. These similarities are mainly based on the summary statistics of relative measures including mean, variance, and covariance under sliding windows of size $\xi \times \xi$ with a step size of ξ along both horizontal and vertical directions. For each sliding window, each similarity function is computed as follows:

$$\begin{aligned} l(x, y) &= \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \\ c(x, y) &= \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \\ s(x, y) &= \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \end{aligned} \tag{5}$$

where μ, σ are the mean and standard deviation respectively. $C_1 = (K_1 L)^2$, $C_2 = (K_2 L)^2$ and $C_3 = \frac{1}{2}C_2$ are variables to stabilize the division with weak denominator, L is the dynamic range of the pixel-values, $K_1 = 0.01$ and $K_2 = 0.03$ are small constants. To enforce independence among those measures, the final SSIM index is constructed as the product of those metrics with exponential constant weights α, β, γ (equal to 1 by default). Replacing the x, y with I_c, I_r, the final loss function is:

$$\ell^c_s = l(I_c, I_r)^\alpha \cdot c(I_c, I_r)^\beta \cdot s(I_c, I_r)^\gamma \tag{6}$$

2) Conformity Losses Besides requiring the consistency of restored color images, the intermediate generated grayscale images should be visually pleasant and similar to one specific type of grayscale images (e.g., simple luminance channel or advanced color2gray [4]). Here, we roughly adopt the losses proposed in [10] but modify them a lot to make them general to different types of grayscale images. It consists of three parts: basic conformity loss ℓ^g_c, contrast loss ℓ^g_c, local structure loss ℓ^g_s.

$$\mathcal{L}^g_{con} = \lambda_3 * \ell^g_b + \lambda_4 * \ell^g_c + \lambda_5 * \ell^g_s \tag{7}$$

Basic Conformity Loss. This loss is to ensure the intermediate generated grayscale image I_g roughly conforms to one predefined type of grayscale image $f_{c\to g}(I_c)$ of the input color image. By contrast, Xia *et al.* only considers the special luminance channel based grayscale in [10]. Since the color information is designed to be hidden in I_g, so I_g should not be completely equal to $f_{c\to g}(I_c)$. Therefore, it is okay once the difference between I_g and $f_{c\to g}(I_c)$ is below a threshold of τ:

$$\ell^g_b = \frac{1}{N}\|\max(|I_g - f_{c\to g}(I_c)| - \tau, 0)\|_1 \tag{8}$$

Here, $|\cdot|$ is an element-wise absolute value operator, and $\|\cdot\|_1$ is the L1 norm function. Empirically, τ is set as 70 by default to allow a loose search space.

Contrast Loss. This loss is designed to preserve the contrast of the original color image I_c in the intermediate grayscale image I_g. In [10], Xia *et al.* find the perceptual loss [51] is a good metric to achieve this goal.

$$\ell^g_c = \frac{1}{N}\|VGG_k(I_g) - VGG_k(I_c)\|^2 \tag{9}$$

where $VGG_k(\cdot)$ denotes the VGG features extracted at layer k ("conv4_1" by default). Note that, because the original VGG network is trained for color images, I_g is repeated 3 times

along the channel dimension before feeding into the VGG network.

Local Structure Loss. Similarly, to make the local structure of the intermediate grayscale I_g conforms to that of the original color image I_c, a local variation-based structure loss is proposed in [10].

$$\ell_s^g = \frac{1}{N}\|Var(I_g) - Var(I_c)\|_1 \tag{10}$$

where $Var(\cdot)$ is a function that calculates the mean of local variation of an image. Ideally, with this loss, the original texture structure or local smoothness can be maintained in I_g.

3) Quantization Loss. To guarantee a great learning performance of $\mathcal{H}, \mathcal{R}$, they adopt floating-point precision by default like most common modern CNN models. For hiding or extracting color information, this is okay. But in real applications, the output grayscale image I_g is often saved in 8-bit unsigned integer precision. Without special guidance, many quantization errors will appear in the final restored color images. To alleviate this problem, Xia *et al.* [10] propose a quantization loss to encourage pixel values generated by $\mathbf{H}$ close to integers as much as possible, formally:

$$\ell_q = \lambda_6 * \|\min_{d=0}^{255}(\|I_g - M_d\|)\|_1 \tag{11}$$

where $\min(\cdot)$ is the element-wise minimum operator. M_d is a constant matrix with value d, whose size is same as I_g. Minimizing this loss is equivalent to making the pixel values of I_g to be integers as far as possible. It is shown in [10], this loss can suppress the quantization artifacts significantly.

4) Adversarial Losses. To remove the artifacts in smooth regions and produce more visually pleasant results, two auxiliary discriminator networks are introduced after $\mathbf{H}$ and $\mathbf{R}$ respectively. Specifically, the objective of discriminator $\mathbf{D_g}$ is to distinguish the generated grayscale image I_g from a real grayscale image, while the hiding sub-network $\mathbf{H}$ tries to generate high-quality grayscale image I_g to fool $\mathbf{D_g}$. During the training of $\mathbf{D_g}$, we use the real grayscale image dataset (denoted as $\mathcal{I}_g^r$) generated by one special color-to-grayscale conversion method f_{cg} as positive samples and use the generated grayscale images (denoted as $\mathcal{I}_g^f$) as negative samples. And the corresponding adversarial loss $\mathcal{L}_d^g$ is defined as:

$$\ell_d^g = \mathop{\mathbb{E}}_{x\in\mathcal{I}_g^r} log(D_g(x)) + \mathop{\mathbb{E}}_{y\in\mathcal{I}_g^f} log(1 - D_g(y))] \tag{12}$$

During training, the goal of $\mathbf{H}$ is to minimize the above objective function while $\mathbf{D_g}$ is to maximize it instead.

Similarly, for the discriminator $\mathbf{D_c}$, we denote the real color image dataset and the restored color image dataset as $\mathcal{I}_c^r$ and $\mathcal{I}_c^f$. And the final adversarial loss $\mathcal{L}_d^c$ is defined as:

$$\ell_d^c = \mathop{\mathbb{E}}_{x\in\mathcal{I}_c^r} log(D_g(x)) + \mathop{\mathbb{E}}_{y\in\mathcal{I}_c^f} log(1 - D_g(y))] \tag{13}$$

TABLE I: Quantitative comparisons of different training strategy combinations with PSNR, where PSNR-g and PSNR-c represent the PSNR value of intermediate grayscale and final restored color images respectively. Obviously, combining strategy-wise and mix training strategies can produce the best quantitative results.

Strategy	only Stage-wise	only Mix	Stage-wise + Mix
PSNR-g	32.73	32.12	36.00
PSNR-c	37.44	35.98	40.56

Because the restored image dataset $\mathcal{I}_c^f$ involves $\mathbf{H}, \mathbf{R}$ at the same time, this loss will encourage the collaboration of $\mathbf{H}, \mathbf{R}$ to produce more visually pleasant restored color images. So, the total adversarial losses $\mathcal{L}_d$ is the sum of ℓ_d^c and ℓ_d^g:

$$\mathcal{L}_d = \lambda_7 * \ell_d^g + \lambda_8 * \ell_d^c \tag{14}$$

In the following experiments, we will demonstrate these two adversarial losses are both very helpful in training.

C. Training Strategy

To train our system, we combine two important training strategies: step-wise training and mix training. Figure 5 and Table I are the qualitative and quantitative comparison results to show the importance of stage-wise training and mix training. Below are the motivations and details of these two strategies.

Step-wise Training. Synthesizing JPEG robust invertible grayscale is a challenging task because we need to guarantee the visual quality of the intermediate grayscale images and final color images, and JPEG robustness at the same time. In fact, these three objectives are a little contradictory, thus making straightforward end-to-end training very difficult. To alleviate the training problem, we train our model in three steps by default in our experiment. In the first step, we train a basic model which only consists of the hiding sub-network $\mathbf{H}$ and restoring sub-network $\mathbf{R}$. This model is to implement the basic invertible grayscale that can invert back to color version like [10]. Then in the second step, we add the two auxiliary discriminators $\mathbf{D}_g, \mathbf{D}_c$ to make the above learned $\mathbf{H}$ and $\mathbf{R}$ collaborate better so that the images generated by them cannot be distinguished by $\mathbf{D}_g, \mathbf{D}_c$. Finally, in the third step, we incorporate the JPEG simulator between $\mathbf{H}$ and $\mathbf{R}$ to simulate the real JPEG compression procedure. Combining the above three steps, the training process is much easier and able to generate visually better results.

Mix training. As described before, we want the JPEG simulator can simulate the real JPEG compression procedure well and make our system JPEG robust. However, we find always using this simulator will make the model biased to simulated JPEG images, and perform worse on real clean images. Besides, we also want our model to see some real JPEG images rather than only the simulated ones to avoid training bias. Therefore, we feed different types of training images in the system. Specifically, at each training step, one batch consists of 8 images of three different types: 2 real JPEG images, 4 simulated JPEG images, and 2 clean images. Since

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

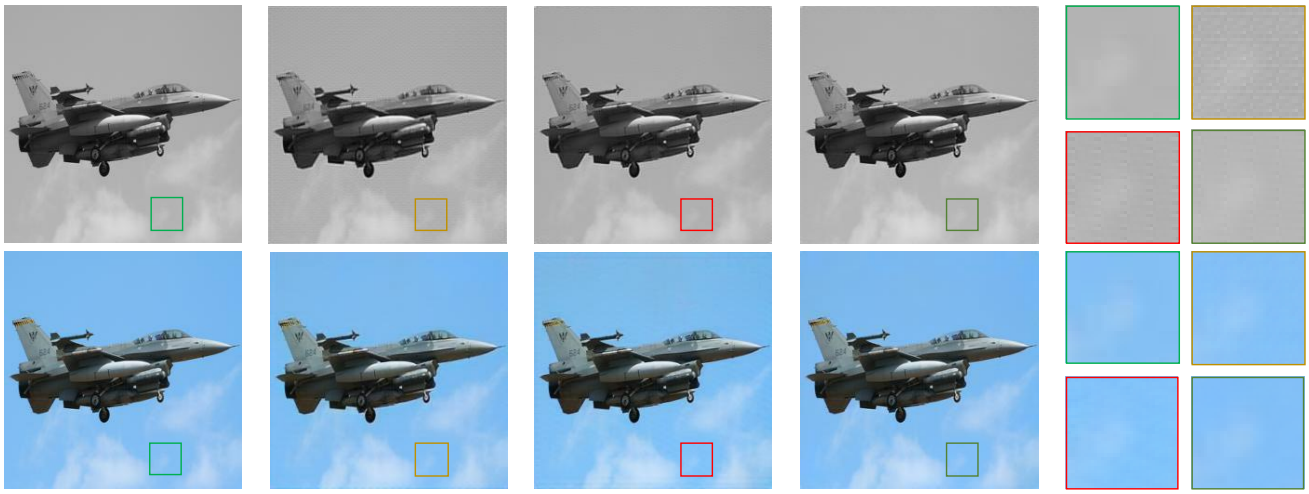


Fig. 5: Visual comparison examples of different training strategies. Obviously, by combining both stage-wise training and mix training, our method can achieve both much visually better intermediate grayscale images (top row) and final restored color images (bottom row).

TABLE II: Descriptions of Loss Functions

Losses	Function	Weight
Basic reconstruction loss ℓ_r^c	To ensure the final restored color images be identical to the original color images	$\lambda_1 : 3$
Structure-preserving loss ℓ_s^c	To eliminate artifacts in the smooth regions	$\lambda_2 : 0.1$
Basic conformity loss ℓ_b^g	To make the intermediate grayscales conform to one pre-defined type of grayscale images	$\lambda_3 : 1$
Contrast loss ℓ_c^g	To make the contrast of intermediate grayscales conform to original color images	$\lambda_4 : 1e-7$
Local structure loss ℓ_s^g	To make the local structure of intermediate grayscales conform to original color images	$\lambda_5 : 0.5$
Quantization loss ℓ_q	Forcing pixel values of the intermediate grayscales to be integers as far as possible	$\lambda_6 : 1$
Adversarial grayscale loss ℓ_d^g	Adversarial losses on grayscales for adversarial training	$\lambda_7 : 0.01$
Adversarial color loss ℓ_d^c	Adversarial losses on color images for adversarial training	$\lambda_8 : 0.01$

the real JPEG process is not differentiable, these JPEG images will break the gradient flow between $\mathbf{H}$ and $\mathbf{R}$. So they can only guide $\mathbf{R}$ directly but influence $\mathbf{H}$ indirectly by $\mathbf{R}$.

IV. Experiments

A. Implementation Details

Our network is trained on 16000 images randomly sampled from the PASCAL-VOC2012 dataset, and each image is scaled to 256×256 in the training phase. We train the network with batch size as 8 for $400k$ iterations. Following the aforementioned step-wise training strategy, our model is trained with three steps. For the hiding sub-network and restoring sub-network in all these three steps, the Adam optimizer with a polynomial learning rate decay strategy is used. Specifically, the initial learning rate of the first step is $1e-4$ and decayed to $1e-6$ in the first 200k iterations. The learning rate of the second and third step is initialized to be $1e-5$ and decayed into $1e-7$ in $100k$ iterations respectively. For the discriminator networks D_c, D_g, we also adopt the Adam optimization method with the initial learning rate of $2e-4$. Based on previous work, we set our loss weights as: : $\lambda_1 = 3, \lambda_2 = 0.1, \lambda_3 = 1, \lambda_4 = 1e-7, \lambda_5 = 0.5, \lambda_6 = 1, \lambda_7 = 0.01, \lambda_8 = 0.01$ in the below experiments by default. The weights of existing methods' losses are basically inherited from [10], others are set for balancing gradients. For better understanding, we enumerate these losses in Table II

TABLE III: Quantitative PSNR comparison results with the baseline method [10]. It can be seen that our method can not only achieve much better intermediate grayscale images but restore better final color images both from clean and JPEG compressed grayscale images.

Methods	Grayscale	Restored Color	JPEG Restored color
Baseline [10]	33.76	40.16	25.88
Ours	**36.00**	**40.56**	**33.13**

B. Experiments Results

To demonstrate our superiority, we compare our method with the baseline method [10] both quantitatively and qualitatively.

Quantitative Evaluation. As mentioned before, the objective of our system can be categorized into three aspects: 1) the intermediate grayscale images should overall conform to the basic grayscale images defined by $f_{c\to g}$ without noticeable artifacts, 2) the final restored color images should be identical to the original color images, 3) the system should be robust to JPEG compression. To quantitatively measure these ingredients, PSNR is used as the default evaluation metric here. Note that for the evaluation of JPEG robustness, the intermediate grayscale images will be first compressed before being fed into the restoring sub-network.

As shown in Table III, our method achieves higher PSNR

TABLE IV: Importance of structure preserving loss. Our method can achieve better performance with structure preserving loss on PSNR.

methods	Grayscale	Restored Color	JPEG-95 Restored color
Without SSIM	35.21	40.21	32.33
With SSIM	36.00	40.56	33.13

than the baseline method [10] on both the intermediate and final results. Specifically, for the case where the intermediate grayscale images are saved without JPEG compression, our method can generate visually better intermediate grayscale images and final restored images simultaneously. This means that the hiding sub-network $\mathbf{H}$ and the restoring sub-network $\mathbf{R}$ learn to collaborate very well. Even $\mathbf{H}$ hides color information in an unnoticeable way, $\mathbf{R}$ is still able to extract it out. For the case when the intermediate grayscale images are compressed by JPEG, the performance of the baseline [10] will degrade a lot. By contrast, our method can still achieve a reasonable PSNR value of 33.13.

Qualitative Evaluation. As shown in the Fig.6, our method can achieve much better restored color images than [10] when the intermediate grayscale images are JPEG compressed. Here "bs-*" means the results of the baseline method [10], "*-clean" and "*-jpeg" are the restored results for grayscale images that are not JPEG compressed and JPEG compressed respectively.

C. More Discussions

In this section, we will first conduct some ablation studies to justify the importance of our design, then give more extension experiments to show the properties and generalization abilities of our method.

Importance of Adversarial Losses. As described in the introduction part, the baseline method [10] often suffers from some artifacts in smooth regions, which motivates us to incorporate the discriminators into our network. To demonstrate the importance of newly added adversarial losses, we provide some detailed cases in Figure 7 whose artifacts are significantly suppressed by incorporating adversarial losses when compared to the baseline [10].

Importance of Structure Preserving Loss. For the final restored color image I_r, only a pixel-level mean square loss is used in the baseline method [10]. Though it can roughly guarantee I_r to be overall similar to the original input color image, it does not consider the structure conformity explicitly. However, as shown in Figure 6, when the intermediate image is compressed by JPEG, the original structure cannot be preserved well in the final restored color image, which is especially worse for smooth regions. Motivated by this, a new SSIM based structure-preserving loss is introduced in our method. To show its effectiveness, we conduct two control experiments with/without SSIM loss. As shown in Table IV, incorporating SSIM loss can help get better results for both clean and JPEG compressed restored images.

Difference with CycleGAN. Our method looks like a cyclic conversion between color and grayscale images, and seems to be solvable using CycleGAN[24]. However, CycleGAN only guarantees that the output images conform to the corresponding image classes (either grayscale or color). Due to the unsupervised nature of CycleGAN, there is no guarantee that the generated grayscale conforms to the corresponding input color image as required in our case. The key ingredient of CycleGAN is cycle loss which requires cycle supervision in the training process. In our method, given an original color image, the hiding sub-network learns how to hide the color/texture information into a special type of invertible grayscale image so that the restoring sub-network can recover the original color/texture back from it. In other words, there is no cycle supervision in our framework.

To further detail the difference between CycleGAN and our work, we conduct a specific experiment in Fig.8. As shown in the left of Fig.8, the CycleGAN cannot guarantee the restored color information conforms to the corresponding input color information at all. As shown in the right of Fig.8, our restoring sub-network cannot generate a color image from a normal grayscale because no color information is hidden in a normal grayscale, while CycleGAN can generate a proper color image by guessing.

Difference with Colorization Methods. The core of our method is that we can accurately recover a grayscale back to its original color version. However, recent colorization methods can also colorize a grayscale to a color image. Different from colorization methods, our method focuses on the hiding and extracting of color information. To further show the difference, we also adopt experiments for comparison.

As shown in Fig.9 , our method can accurately invert the grayscale back to its color version by extracting color information while colorization methods colorize the grayscale by guessing.

Fixed or Random k in JPEG Simulator. In our method, we introduce a JPEG simulator to simulate the real JPEG compression procedure. Analogy to the real JPEG compression quality, we can control the compression degree in the JPEG simulator by changing the number of DCT coefficients k that will be kept. Here, DCT coefficients sorted by zigzag represent information from low to high frequencies. By experiments, we find dropping too many DCT coefficients (i.e., small k) makes the model difficult to converge and hurts the restored image quality of clean grayscale images with JPEG compression. On the other hand, if we keep too many DCT coefficients (i.e., large k), the learned model cannot obtain strong robustness to real JPEG compression. To solve this problem, we randomly sample k from a range of [32-64] rather than using a fix k like [20]. As the baseline, we also conduct two control experiments with a fix k as 32 and 48 respectively. We utilize LPIPS to evaluate perceptual quality.[52] LPIPS evaluates the distance between two image patches. A higher score means a larger difference, while a lower score means a larger similarity.

As shown in Table V and Figure 10, our training strategy with random k in JPEG Simulator can achieve better perfor-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

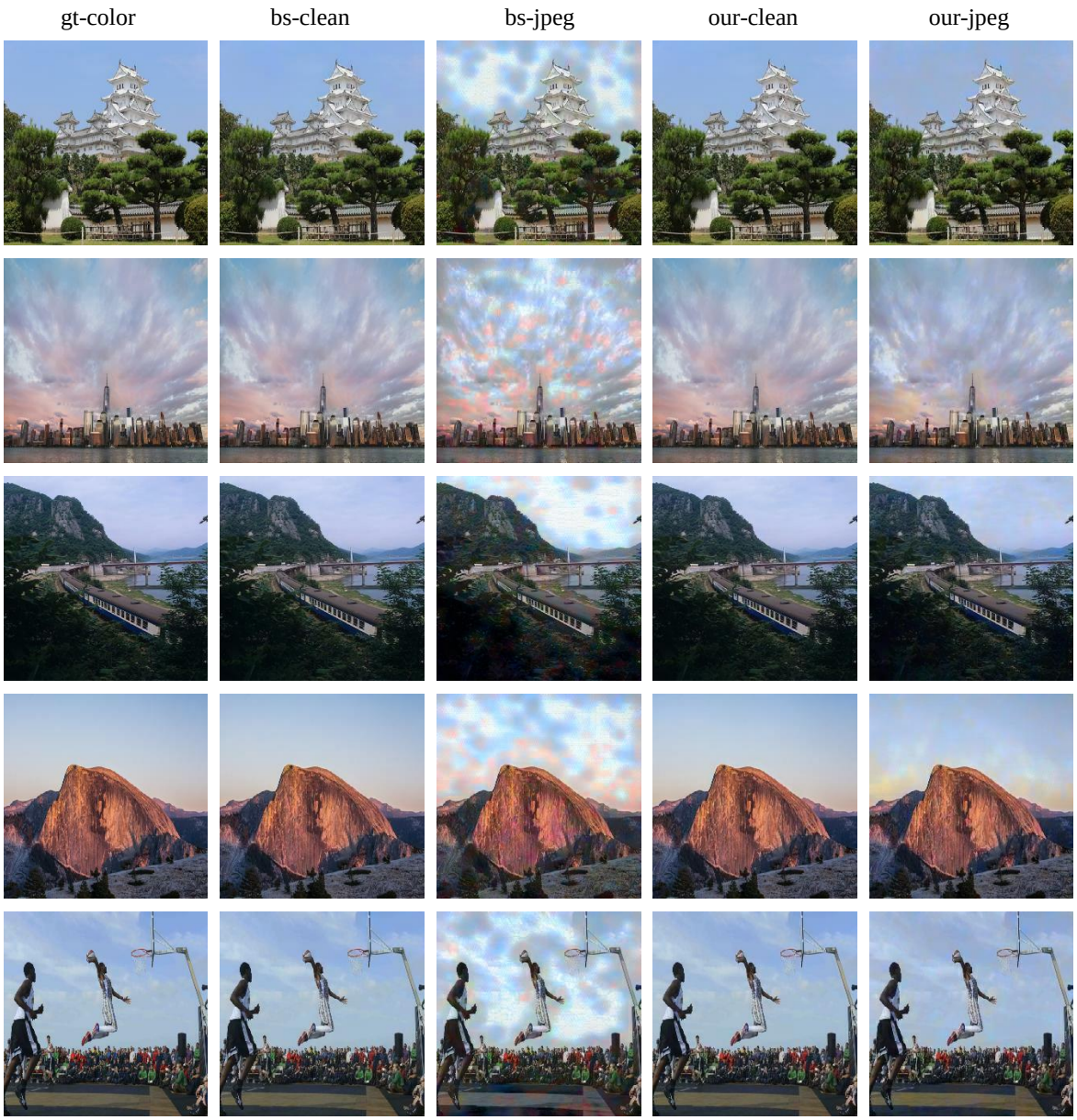


Fig. 6: Visual comparisons of JPEG-95 robustness to the baseline method [10]. Obviously, our method can achieve much better restored color images than [10] when the intermediate grayscale images are JPEG compressed. Here "bs-*" means the results of the baseline method [10], "*-clean" and "*-jpeg" are the restored results for grayscale images that are not JPEG compressed and JPEG compressed respectively.

TABLE V: PSNR Comparisons of fixed or random k in JPEG simulator.

methods	Grayscale	Restored Color	JPEG-95 Restored color
Fixed 32	33.99	32.75	29.54
Fixed 48	35.93	38.64	29.58
Ours (32-64)	**36.00**	**40.56**	**33.13**

mance than the fixed ones. There are several reasons for this result. On the one hand, with random k during training, the model can learn robustness to real JPEG compression with different compression qualities. On the other hand, we find adopting a fixed k specific will make the model overfit this specific k easily.

Robustness to Gaussian Noise. In real scenarios, the intermediate grayscale images might suffer from some degradation. To evaluate the robustness of our method, we deliberately add some Gaussian noises into the intermediate grayscale images and try to recover their color back. Two examples are given in Figure 11. It can be seen that our method shows great robustness, and the restored color images are still visually pleasing.

Robustness to Different JPEG Compression Qualities. As shown in Table VI, with a lower JPEG quality quality, the intermediate grayscale images will be severely compressed, and more hidden color information will be damaged, thus

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics

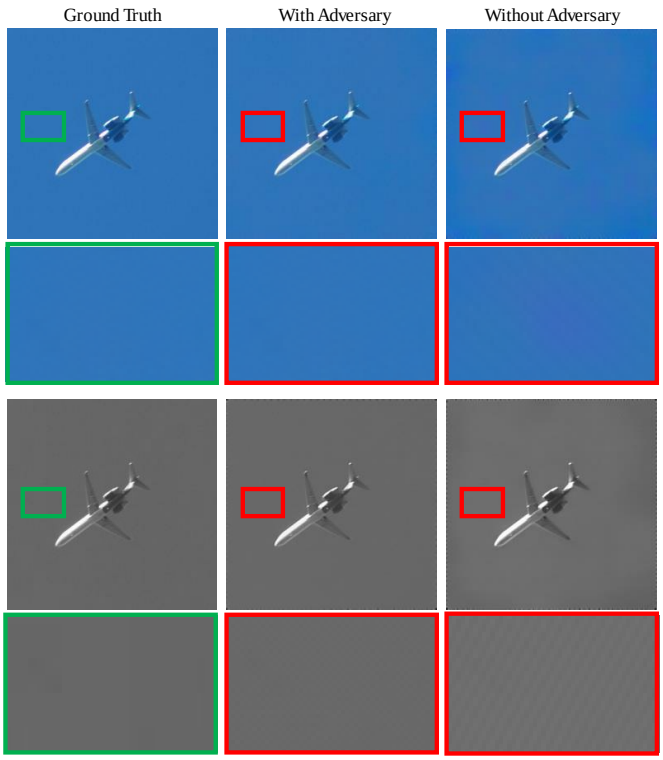


Fig. 7: Examples to show the importance of the newly added adversarial losses, which significantly remove the artifacts in the smooth regions.

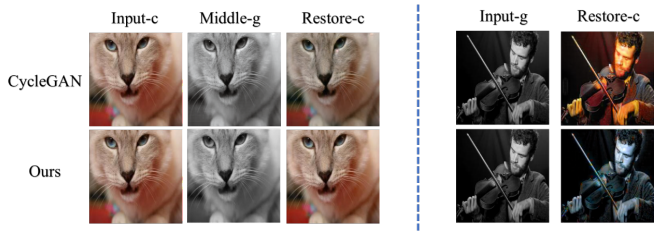


Fig. 8: The Input-c is a normal color image, and Input-g is a normal grayscale image. Middle-g is built by our hiding sub-network $\mathbf{H}$ and the color-to-gray generator of CycleGAN, $\mathbf{G_A}$. Restore-c is built by our restoring sub-network $\mathbf{R}$ and gray-to-color generator of CycleGAN, $\mathbf{G_B}$.

causing worse final restored color images. It indicates that our method cannot handle very severe JPEG compression either. However, by incorporating the JPEG simulator during training, our method consistently outperforms the baseline method [10] by a large margin. To provide more objective results, we draw two plots to demonstrate our advantages on PSNR and SSIM[53] in Fig.12a and Fig.12b.

Robustness to Webp Compression Methods. In real scenarios, the intermediate grayscale images might suffer from other compression methods, such as Webp compression. To evaluate the scalability of our method, we take additional experiments on Webp compression. We compress intermediate grayscale images by Webp and try to recover their color back. As shown in Fig.12c and Fig.12d. , the experiment results show that our method can resist Webp compression.

Fig. 9: Examples to show the difference of our method and general colorization methods. Our method can accurately invert the grayscale back to its color version while colorization methods cannot reconstruct accurate color information.

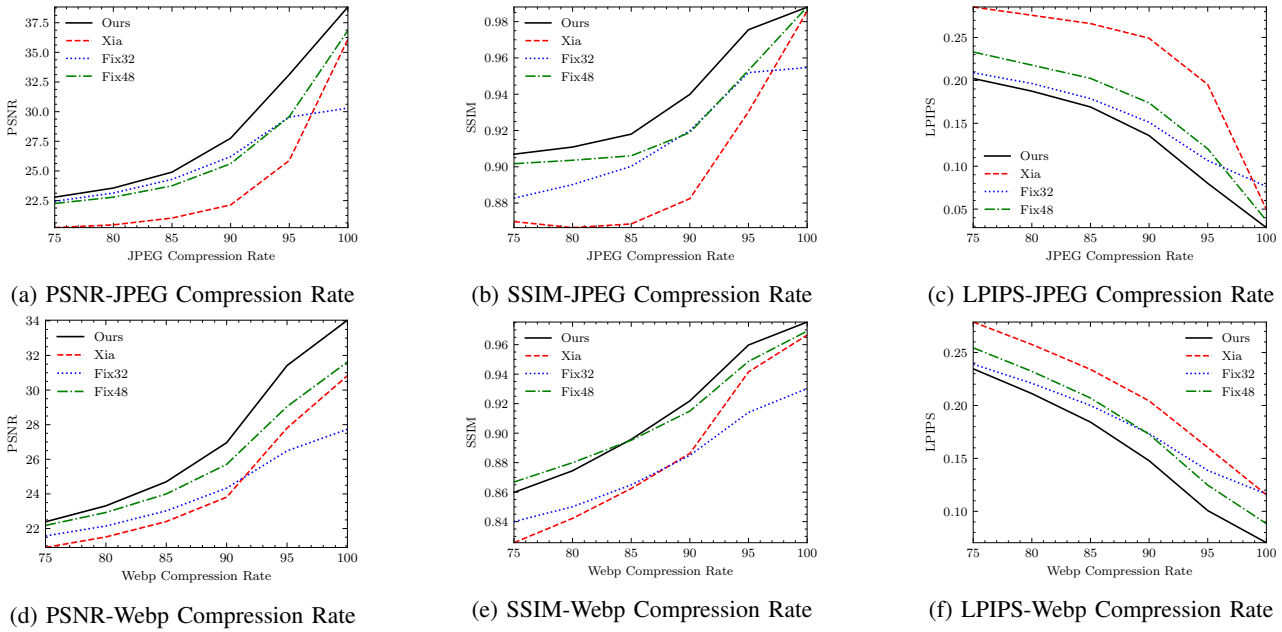
Extension to Different Types of Grayscale. In this paper, by default we adopt the vanilla grayscale images as the intermediate images. However, conformity loss is not always required. We tried to cancel the conformity loss and invertible ability stay. As shown in Fig.13 , our network can stay invertibility if we cancel the gray conformity loss. However, the visual quality of intermediate results will be terrible. With the constraint, our method can provide undistinguishable intermediate results while keeping the same invertibility.

The previous experiment indicates to us that the proposed framework is very general and may be applied to other types of intermediate images. So we try to define $f_{c\to g}$ as other extremely difficult types of images, i.e., the edge map of the original color image and the halftone images.

In the first case, the intermediate images need to contain both the simple color information and the complex texture information simultaneously, which is very challenging. This task can be regarded as a special type of edge2image translation problem [22], where texture/color information is already hidden in the input edge. We test our model on the CelebA dataset [54] and use the Canny algorithm to pre-generate the edges. During training, to ensure the intermediate generated image conforms to the edge map, we increase the weight of $\mathcal{L}_{con}^{g}$ in Equation 2. More detail, we increased the weight of λ_3 by 1000% and set τ to 10 in Equation 8. In Figure 14, two examples are provided to show the effectiveness of this special application. Even though texture information and color information are highly suppressed in this special type of edge map, our restoring sub-network can still generate visually plausible results.

In the second case, we use the same configuration for halftone images. Halftone images are binary images that served as analog representations and are widely used in digital image printing. Since only dots varying either in size or in spacing are used, a large portion of image information is lost in halftone images, making it extremely difficult to recover the original shape and textures. However, as shown in Figure 15, our method can still synthesize plausible intermediate

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics



(a) PSNR-JPEG Compression Rate (b) SSIM-JPEG Compression Rate (c) LPIPS-JPEG Compression Rate

(d) PSNR-Webp Compression Rate (e) SSIM-Webp Compression Rate (f) LPIPS-Webp Compression Rate

Fig. 10: Robustness comparison on fixed or random k strategies.

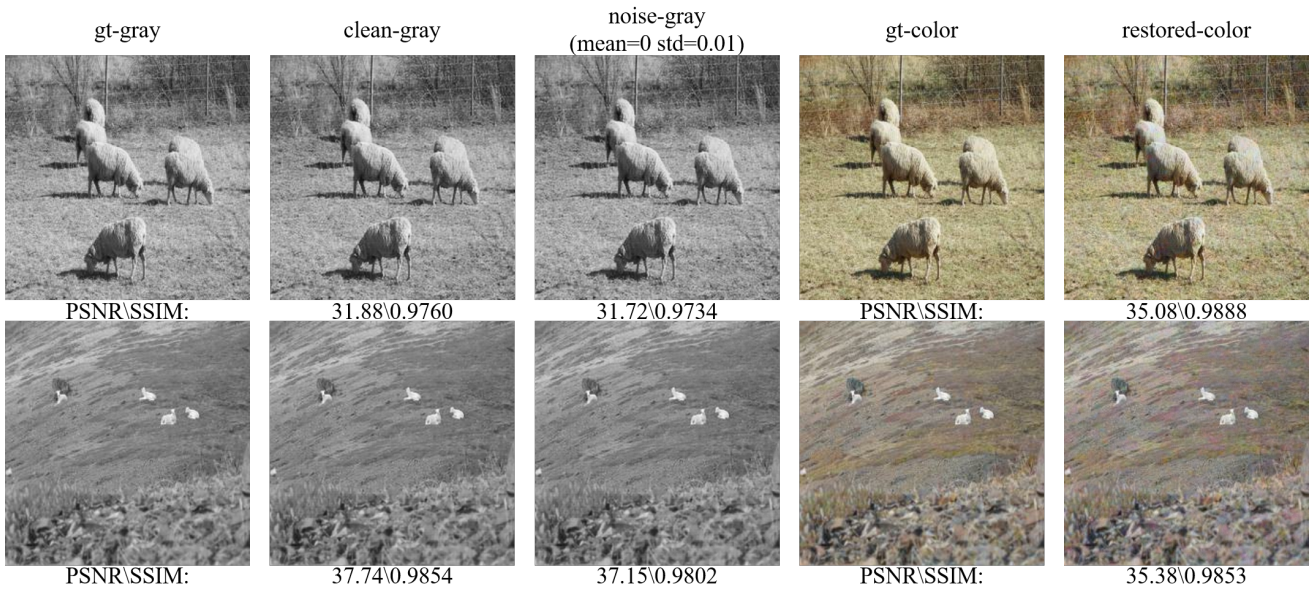
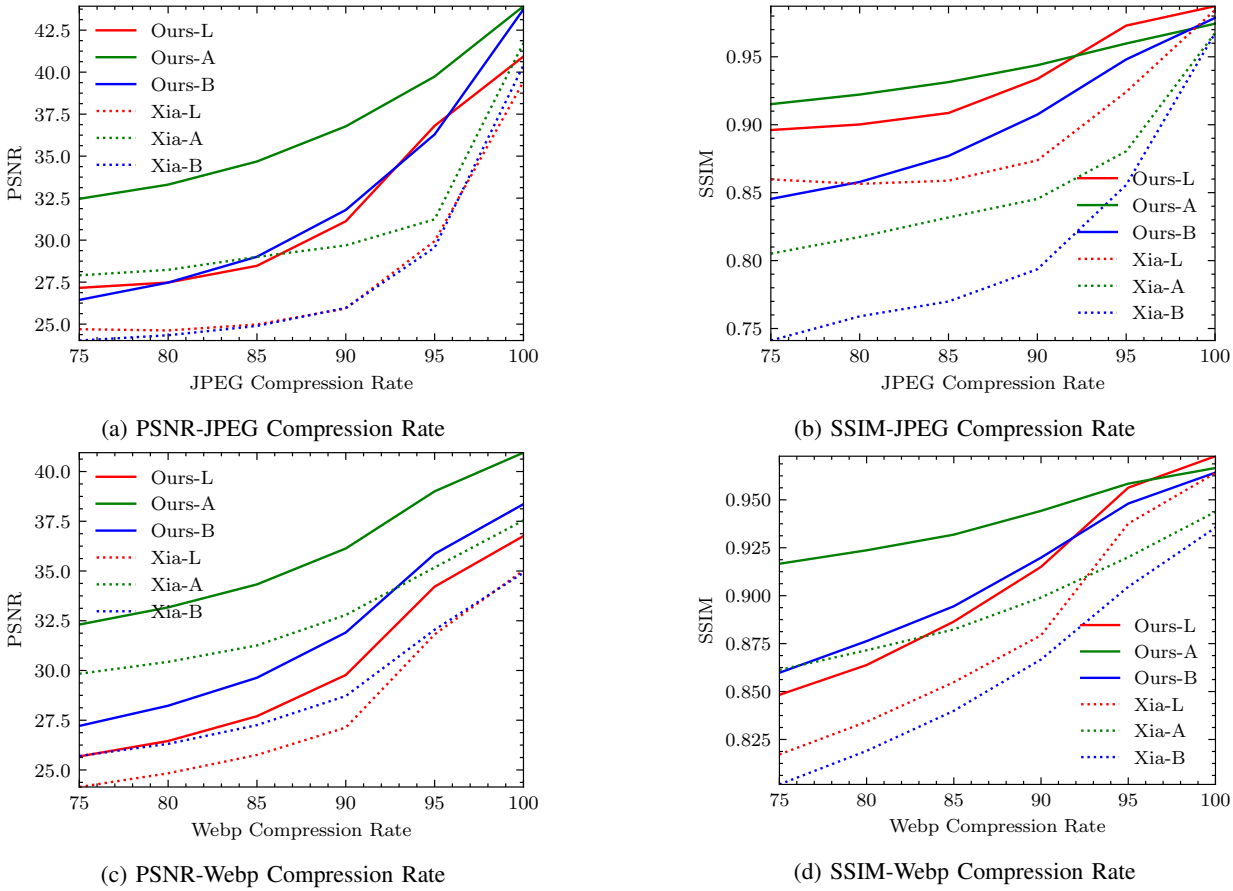


Fig. 11: Examples to show the robustness of our method for the cases where the intermediate images are degraded with Gaussian noises. It shows that our restoring sub-network can learn to restore hidden information from degraded intermediate images well.

TABLE VI: Robustness comparison to different JPEG compression qualities with PSNR. It shows that the final restored color images will become worse when the intermediate grayscale images are more JPEG compressed, but our method outperforms the baseline method [10] consistently.

methods	JPEG-75	JPEG-80	JPEG-85	JPEG-90	JPEG-95	JPEG-100
Ours	22.79	23.56	24.90	27.76	33.13	38.83
Xia [10]	20.22	20.45	21.02	22.13	25.88	36.10

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2021.3088531, IEEE Transactions on Visualization and Computer Graphics



(a) PSNR-JPEG Compression Rate

(b) SSIM-JPEG Compression Rate

(c) PSNR-Webp Compression Rate

(d) SSIM-Webp Compression Rate

Fig. 12: Robustness comparison to JPEG and Webp. L/A/B represent the corresponding color channel in LAB space.

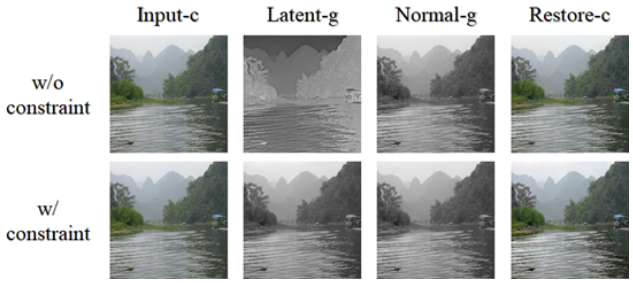


Fig. 13: The Input-c is a normal color image and Normal-g is a grayscale image generated by the Input-c with existing color-to-gray method. Latent-g is synthesized by our hiding sub-network. Restore-c is built by our restoring sub-network.

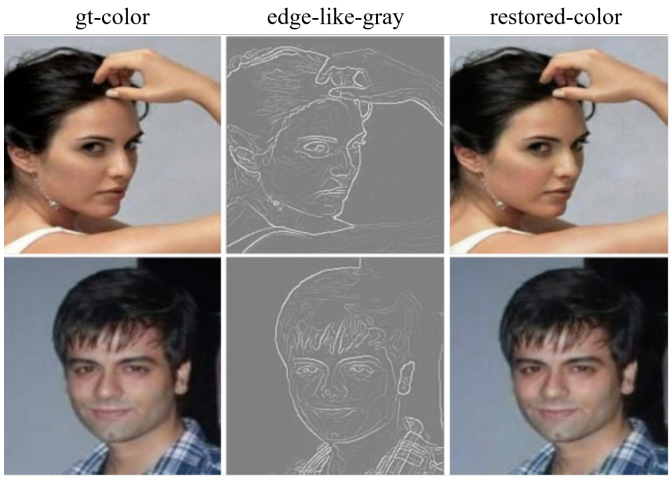


Fig. 14: Extension to the case where the intermediate images are edge-like images. It shows that our hiding sub-network can learn how to suppress the texture information and color information into the edge maps well so that the restoring sub-network can get visually plausible reconstruction results.

halftone-like images and recover their fidelity afterward.

V. Conclusion

In this paper, we propose a JPEG robust invertible grayscale system. This system consists of two sub-networks: hiding sub-network and restoring sub-network. Given an original color image, the hiding sub-network learns how to hide the color/texture information into a special type of invertible grayscale image so that the restoring sub-network can recover the original color/texture back from it. We improve the performance of this system from two aspects: 1) Two auxiliary adversarial discriminators are leveraged to avoid the visual artifacts in smooth regions. 2) JPEG simulator is incorporated to make the system robust to real JPEG compression. Extensive experiments demonstrate our superior performance

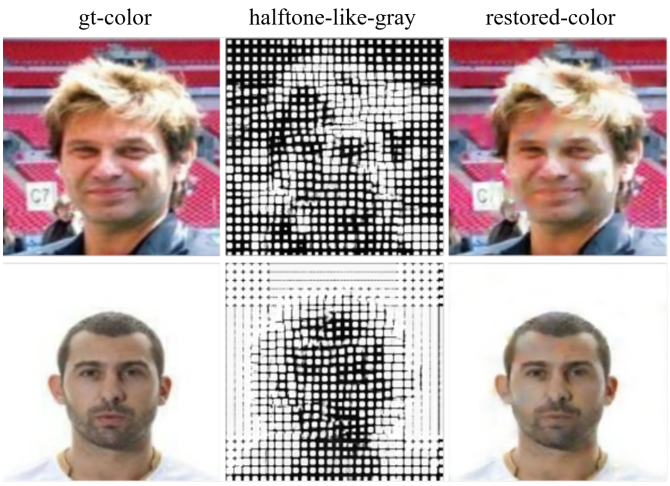


Fig. 15: Extension to the case where the intermediate images are halftone-like images. Despite the super inner difficulty, our method can still work quite well.

over the baseline method [10]. We further extend the proposed framework to other types of intermediate grayscale images, i.e., edge maps and halftone images. Though it is challenging to hide texture and color information simultaneously in the simple edge map without noticeable artifacts, the proposed framework is still able to recover the original color images and generate visually pleasant results.

REFERENCES

[1] M. D. Fairchild, *Color appearance models*. John Wiley & Sons, 2013.

[2] C. Lu, L. Xu, and J. Jia, "Contrast preserving decolorization," in *2012 IEEE International Conference on Computational Photography (ICCP)*. IEEE, 2012, pp. 1–7.

[3] J. G. Kuk, J. H. Ahn, and N. I. Cho, "A color to grayscale conversion considering local and global contrast," in *Asian Conference on Computer Vision*. Springer, 2010, pp. 513–524.

[4] A. A. Gooch, S. C. Olsen, J. Tumblin, and B. Gooch, "Color2gray: salience-preserving color removal," in *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 634–639.

[5] M. Cui, J. Hu, A. Razdan, and P. Wonka, "Color-to-gray conversion using isomap," *VC*, vol. 26, no. 11, pp. 1349–1360, November 2010.

[6] T. Horiuchi, X. Wen, and K. Hirai, "Reversible color-to-gray mapping with resistance to jpeg encoding," in *2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, 2018, pp. 13–16.

[7] T. Horiuchi, F. Nohara, and S. Tominaga, "Accurate reversible color-to-gray mapping algorithm without distortion conditions," *PRL*, vol. 31, no. 15, pp. 2405–2414, November 2010.

[8] F. Nohara, T. Horiuchi, and S. Tominaga, "An accurate algorithm for color to gray and back," in *ICIP*, 2009, pp. 485–488.

[9] R. de Queiroz, "Reversible color-to-gray mapping using subband domain texturization," *PRL*, vol. 31, no. 4, pp. 269–276, March 2010.

[10] M. Xia, X. Liu, and T.-T. Wong, "Invertible grayscale," in *SIGGRAPH Asia 2018 Technical Papers*. ACM, 2018, p. 246.

[11] W. Luo, F. Huang, and J. Huang, "Edge adaptive image steganography based on lsb matching revisited," *IEEE Transactions on information forensics and security*, vol. 5, no. 2, pp. 201–214, 2010.

[12] L. M. Marvel, C. G. Boncelet, and C. T. Retter, "Spread spectrum image steganography," *IEEE Transactions on image processing*, vol. 8, no. 8, pp. 1075–1083, 1999.

[13] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Transactions on information forensics and security*, vol. 8, no. 3, pp. 553–562, 2013.

[14] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Processing*, vol. 94, pp. 118–127, 2014.

[15] M. Ramkumar and A. N. Akansu, "Self-noise suppression schemes in blind image steganography," in *Multimedia Systems and Applications II*, vol. 3845. International Society for Optics and Photonics, 1999, pp. 55–66.

[16] K. C. Widadi, P. H. Ainianta, and C. C. Wah, "Blind steganography using direct sequence/frequency hopping spread spectrum technique," in *2005 5th International Conference on Information Communications & Signal Processing*. IEEE, 2005, pp. 1125–1129.

[17] F. Alturki and R. Mersereau, "Secure blind image steganographic technique using discrete fourier transformation," in *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, vol. 2. IEEE, 2001, pp. 542–545.

[18] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proceedings of 1st International Conference on Image Processing*, vol. 2. IEEE, 1994, pp. 86–90.

[19] R. B. Wolfgang and E. J. Delp, "A watermark for digital images," in *Proceedings of 3rd IEEE International Conference on Image Processing*, vol. 3. IEEE, 1996, pp. 219–222.

[20] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[22] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[23] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *Advances in Neural Information Processing Systems*, 2017.

[24] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[25] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[26] Z. Tan, D. Chen, Q. Chu, M. Chai, J. Liao, M. He, L. Yuan, G. Hua, and N. Yu, "Efficient semantic image synthesis via class-adaptive normalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[27] K. Rasche, R. Geist, and J. Westall, "Detail preserving reproduction of color images for monochromats and dichromats," *IEEE Computer Graphics and Applications*, vol. 25, no. 3, pp. 22–30, 2005.

[28] R. Bala and R. Eschbach, "Spatial color-to-grayscale transform preserving chrominance edge information," in *Color and Imaging Conference*, vol. 2004, no. 1. Society for Imaging Science and Technology, 2004, pp. 82–86.

[29] L. Neumann, M. Čadík, and A. Nemcsics, "An efficient perception-based adaptive color to gray transformation," in *Proceedings of the Third Eurographics conference on Computational Aesthetics in Graphics, Visualization and Imaging*. Eurographics Association, 2007, pp. 73–80.

[30] K. Smith, P.-E. Landes, J. Thollot, and K. Myszkowski, "Apparent greyscale: A simple and fast conversion to perceptually accurate images and video," in *Computer Graphics Forum*, vol. 27, no. 2. Wiley Online Library, 2008, pp. 193–200.

[31] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *ACM transactions on graphics (tog)*, vol. 23, no. 3. ACM, 2004, pp. 689–694.

[32] Y. Qu, T.-T. Wong, and P.-A. Heng, "Manga colorization," in *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3. ACM, 2006, pp. 1214–1220.

[33] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, "Natural image colorization," in *Proceedings of the 18th Eurographics conference on Rendering Techniques*. Eurographics Association, 2007, pp. 309–320.

[34] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," in *ACM transactions on graphics (TOG)*, vol. 21, no. 3. ACM, 2002, pp. 277–280.

[35] R. Ironi, D. Cohen-Or, and D. Lischinski, "Colorization by example." in *Rendering Techniques*. Citeseer, 2005, pp. 201–210.

[36] A. Bugeau, V.-T. Ta, and N. Papadakis, "Variational exemplar-based image colorization," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 298–307, 2013.

[37] R. K. Gupta, A. Y.-S. Chia, D. Rajan, E. S. Ng, and H. Zhiyong, "Image colorization using similar images," in *Proceedings of the 20th ACM international conference on Multimedia*. ACM, 2012, pp. 369–378.

[38] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European conference on computer vision*. Springer, 2016, pp. 649–666.

[39] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 415–423.

[40] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *European Conference on Computer Vision*. Springer, 2016, pp. 577–593.

[41] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 110, 2016.

[42] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros, "Real-time user-guided image colorization with learned deep priors," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 119, 2017.

[43] M. He, D. Chen, J. Liao, P. V. Sander, and L. Yuan, "Deep exemplar-based colorization," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–16, 2018.

[44] T. Z. F. Fang, T. Wang and G. Zhang, "A superpixel-based variational model for image colorization," *IEEE Transactions on Visualization and Computer Graphics*, 2019.

[45] Y. G. J. Zhu and H. Ma, "A data-driven approach for furniture and indoor scene colorization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 9, pp. 2473–2486, 2018.

[46] F. Fang, T. Wang, T. Zeng, and G. Zhang, "A superpixel-based variational model for image colorization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 10, pp. 2931–2943, 2020.

[47] Y. Xiao, J. Wu, J. Zhang, P. Zhou, Y. Zheng, C. S. Leung, and L. Kavan, "Interactive deep colorization and its application for image compression," *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2020.

[48] J.-W. Su, H.-K. Chu, and J.-B. Huang, "Instance-aware image colorization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[49] P. Vitoria, L. Raad, and C. Ballester, "Chromagan: Adversarial picture colorization with semantic class distribution," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 2445–2454.

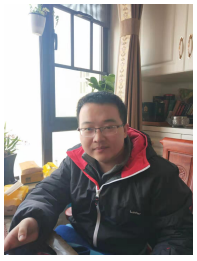
[50] J. Lin, T. Horiuchi, K. Hirai, and S. Tominaga, "Color image recovery system from printed gray image," in *2014 Southwest Symposium on Image Analysis and Interpretation*, 2014, pp. 41–44.

[51] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711.

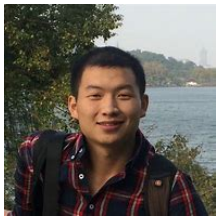
[52] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[53] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

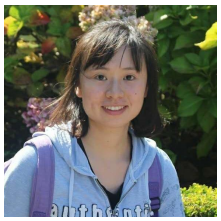
[54] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.



Kunlin Liu received his B.S. degree in 2018 from University of Science and Technology in China. He is currently pursuing the Ph.D. degree in electronic engineering in University of Science and Technology of China. His research interests include multimedia security and multimedia manipulation and AI security.



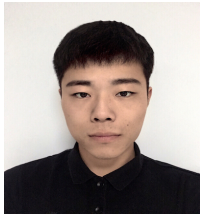
Dongdong Chen is currently a senior researcher at Microsoft Research. Before that, he received a Ph.D. degree from the University of Science and Technology of China (USTC) under the joint PhD program between MSRA and USTC in 2019. His research interests include image generation, style transfer, AI security and general representation learning.



Jing Liao is an Assistant Professor with the Department of Computer Science, City University of Hong Kong (CityU) since Sep 2018. Prior to that, she was a Researcher at Visual Computing Group, Microsoft Research Asia from 2015 to 2018. She received the B.Eng. degree from HuaZhong University of Science and Technology and dual Ph.D. degrees from Zhejiang University and Hong Kong UST. Her primary research interests fall in the fields of Computer Graphics,Computer Vision, Image/Video Processing, Digital Art and Computational Photography.

Weiming Zhang received his M.S. degree and Ph.D. degree in 2002 and 2005 respectively from the Zhengzhou Information Science and Technology Institute, P.R. China. Currently, he is a professor with the School of Information Science and Technology, University of Science and Technology of China. His research interests include information hiding and multimedia security.

Hang Zhou received his B.S. degree in 2015 from Shanghai University (SHU) and a Ph.D. degree in 2020 from the University of Science and Technology of China (USTC). Currently, he is a postdoctoral researcher at Simon Fraser University. His research interests include computer graphics, multimedia security and deep learning.



Jie Zhang is currently a Ph.D. student with the University of Science and Technology of China (USTC). He received the B.S. degree in 2017 from China University of Geosciences, Beijing. His primary research interests include IP protection of deep models, media watermarking and AI security.

Wenbo Zhou received his B.S. degree in 2014 from Nanjing University of Aeronautics and Astronautics, China , and Ph.D. degree in 2019 from University of Science and Technology of China , where he is currently postdoctoral researcher. His research interests include information hiding and AI security.

Nenghai Yu received his B.S. degree in 1987 from Nanjing University of Posts and Telecommunications, an M.E. degree in 1992 from Tsinghua University and a Ph.D. degree in 2004 from the University of Science and Technology of China, where he is currently a professor. His research interests include multimedia security,multimedia information retrieval, video processing and information hiding.